IOWA STATE UNIVERSITY
Digital Repository

Retrospective Theses and Dissertations

Iowa State University Capstones, Theses and Dissertations

2006

A new message authentication code based on the non-associativity of quasigroups

Kristen Ann Meyer
Iowa State University

Follow this and additional works at: https://lib.dr.iastate.edu/rtd

 Part of the Mathematics Commons

Recommended Citation

Meyer, Kristen Ann, "A new message authentication code based on the non-associativity of quasigroups " (2006). *Retrospective Theses and Dissertations*. 1480.
https://lib.dr.iastate.edu/rtd/1480

This Dissertation is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Retrospective Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.

A new message authentication code based on the non-associativity of quasigroups

by

Kristen Ann Meyer

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Mathematics

Program of Study Committee:
Clifford Bergman, Major Professor
Jennifer Davidson
Roger Maddux
Jonathan Smith
Sung-Yell Song

Iowa State University

Ames, Iowa

2006

Copyright © Kristen Ann Meyer, 2006. All rights reserved.

UMI Number: 3218982

UMI Microform 3218982

Copyright 2006 by ProQuest Information and Learning Company.
All rights reserved. This microform edition is protected against unauthorized copying under Title 17, United States Code.

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

Graduate College
Iowa State University

NOTE:
Electronic theses
will not contain
the signed thesis
approval page
here.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

A quasigroup is a set of elements with one binary operation whose multiplication table forms a Latin square. Because quasigroups are not required to be associative, multiplying a string of elements together in different orders can produce different results. A message authentication code, or a MAC, is a cryptographical tool used to verify the authenticity of a message. In this dissertation, we create a new message authentication code called QMAC whose security is based on this non-associativity. In order to obtain security against forgeries, a highly non-associative quasigroup of large order must be used. Methods for efficiently creating and representing such quasigroups are also discussed.

CHAPTER 1. INTRODUCTION

Keyed hash functions, or message authentication codes (MACs), are a widely studied and used cryptographic tool. As the name suggests, MACs are most often used to authenticate a message. A MAC can be used both to ensure that a message has not been changed in transit and to verify the sender of a message. The key part of a message authentication code is the "secret" input to the MAC. By definition, the output of a MAC (called a check digit or an authentication tag) must be easy to compute with knowledge of the secret key, but difficult to compute without knowledge of this key. In addition, in order for a MAC to be practical, the key must require a reasonable amount of storage space, and the number of potential keys must be large enough to prevent attacks on the MAC.

MACs can be created in a variety of ways. Many MACs are based on some other cryptographic function, such as a hash function or a block cipher. The security of these MACs is then derived from the security of the hash function or the block cipher. However, MACs can also be developed which are not derived from any other cryptographic process, but instead are based on some other algebraic structure. In this case, the security of the MAC comes from properties of the underlying algebraic structure. In (4), Denes and Keedwell developed a MAC based on Latin squares.

A natural companion to a Latin square is the quasigroup whose multiplication table is the Latin square. Therefore, it would seem that a MAC could also be created based on quasigroups. Because of the non-associativity of quasigroups, there is a natural choice for the "secret" input to the MAC—the order in which the quasigroup elements are multiplied together to create the authentication tag. In addition, because there are an enormous number of non-associative quasigroups of large order, quasigroups will work well for this type of application. In order for

a MAC based on a quasigroup of large order to be easily computable, efficient ways to store the quasigroup and perform the quasigroup multiplication are needed.

This paper will examine various methods for creating a message authentication code based on a quasigroup. Chapter 2 formally defines quasigroups and MACs and discusses some of their properties. In Chapter 3, we will describe Denes and Keedwell's MAC and then detail how to create a new MAC (which we call QMAC) that utilizes the non-associativity of the chosen quasigroup. We will also examine how to implement QMAC and look at keyspace and security concerns. Chapter 4 will describe several methods for efficiently creating quasigroups of large order. Finally, in Chapter 5 we will explore how to measure "how non-associative" a quasigroup is and will look at several methods for producing highly non-associative quasigroups.

CHAPTER 2. BASIC CONCEPTS AND DEFINITIONS

2.1 Quasigroups and Latin Squares

Definition 1. A **quasigroup** $(Q, \circ)$ is a set Q of elements along with a binary operation $\circ$ with the following properties:

1. For all $a, b \in Q$, $a \circ b \in Q$ (that is, Q is *closed under* $\circ$).

2. For all $a, b \in Q$, there exist unique $x, y \in Q$ so that $x \circ a = b$ and $a \circ y = b$ (that is, $(Q, \circ)$ has *unique solubility of equations*).

Notice that a quasigroup is not required to be associative, commutative, or to have an identity element. If there is no identity element, then it doesn't make sense to talk about the inverse of an element. As in Examples 1 and 2, finite quasigroups are often defined by their multiplication table, or Cayley table.

Because of the unique solubility of equations, each element will appear exactly once in each row and exactly once in each column of the multiplication table of $(Q, \circ)$. That is, each row and column is a permutation of the elements of Q. This implies that if $|Q| = n$, then the interior of the Cayley table for $(Q, \circ)$ forms an n by n Latin square. (An n by n Latin square is made up of n distinct elements, each of which appears exactly once in each row and exactly once in each column.)

Example 1. Let $Q = \mathbb{Z}_6 = \{0, 1, 2, 3, 4, 5\}$ and let $x \circ y := x + y \mod 6$. Then $(Q, \circ)$ is simply addition modulo 6 on $\mathbb{Z}_6$ and the Cayley table for $(Q, \circ)$ is shown in Figure 2.1. Notice that $(Q, \circ)$ is a quasigroup because its interior is a 6 by 6 Latin square. In this case, $(Q, \circ)$ is also a group.

$Q =$

$\circ$	0	1	2	3	4	5
0	0	1	2	3	4	5
1	1	2	3	4	5	0
2	2	3	4	5	0	1
3	3	4	5	0	1	2
4	4	5	0	1	2	3
5	5	0	1	2	3	4

Figure 2.1 The quasigroup $(Q, \circ)$ for Example 1.

It follows from Definition 1 that every group is a quasigroup. However, there are also quasigroups which are not groups, as shown in Example 2.

Example 2. Let $Q = \mathbb{Z}_6$ and let $\circ$ be as shown in Figure 2.2. Then $(Q, \circ)$ is a quasigroup because the interior of its Cayley table is a Latin square. Notice that $(Q, \circ)$ is neither associative nor commutative. In addition, $(Q, \circ)$ does not have an identity element. Therefore, $(Q, \circ)$ is clearly not a group.

$Q =$

$\circ$	0	1	2	3	4	5
0	3	1	4	5	2	0
1	0	4	3	2	5	1
2	1	5	0	4	3	2
3	4	3	2	1	0	5
4	5	2	1	0	4	3
5	2	0	5	3	1	4

Figure 2.2 The quasigroup $(Q, \circ)$ for Example 2.

2.2 Hash Functions and Message Authentication Codes

According to (7), two types of hash functions can be discussed: unkeyed hash functions (or simply hash functions) and keyed hash functions (or message authentication codes). Strictly speaking, a hash function can be defined using only the properties in Definition 2.

Definition 2. A **hash function** is a function h with the following two properties:

1. For a binary string x of arbitrary finite length, $h(x)$ is a binary string of fixed length n (*compression*).

2. Given an input x, $h(x)$ is easy to compute (*ease of computation*).

However, it is desirable for an unkeyed hash function to have more requirements than just these two properties. If a hash function is used for cryptographic purposes, it is required to have one or more of these three properties:

1. Given an output y, it is computationally infeasible to find an input x so that $h(x) = y$ (*preimage resistance* or *one-way function*).

2. Given an input x, it is computationally infeasible to find another input $x' \neq x$ so that $h(x) = h(x')$ (*2nd preimage collision resistance*).

3. It is computationally infeasible to find distinct input values x, x' so that $h(x) = h(x')$ (*collision resistance*).

The other type of hash function is a keyed hash function, or a Message Authentication Code (MAC). A MAC is defined as follows:

Definition 3. A **message authentication code**, or **MAC**, is a function h_k, where k is a secret key, with the following properties:

1. Given a value k and an input x, $h_k(x)$ is easy to compute (*ease of computation*). The value of $h_k(x)$ is usually called the **MAC-value of x**.

2. For an arbitrary-length input x, $h_k(x)$ is of fixed length n. (*compression*).

3. Given a fixed number of input-output pairs $(x_i, h_k(x_i))$ for $i = 1, \ldots, m$ and any other input $x \notin \{x_1, \ldots, x_m\}$ it is computationally infeasible to compute $h_k(x)$ without knowledge of k (*computation-resistance*).

Notice what Definition 3 does and does not require. We only require that a MAC be easy to compute with knowledge of the key and difficult to compute without knowledge of the key.

Because of this, it is possible that with knowledge of the key, it is very easy to find collisions for the MAC. This situation does not contradict the definition of a MAC.

As mentioned in Chapter 1, MACs are used often in cryptography to create a check digit (i.e. an authentication tag) or to authenticate a message. If Bob wants to make sure that the message he receives was really sent by Alice and that this message hasn't been altered or corrupted in transit, he can accomplish this by using a MAC. Alice and Bob simply need to decide on a MAC and exchange the secret key. Alice then computes the MAC-value associated with the message prior to sending it and appends this value to the end of the message. When Bob receives the message, he checks that the appended authentication tag is indeed the correct MAC-value and therefore verifies that the message has come from Alice (or from someone else with knowledge of the secret key) and that the message has not been changed in transit. If Eve, who does not have knowledge of the secret key, wants to either substitute her own message or alter the message, she will be unable to do so because of the computation resistance property of the MAC. In fact, even if Eve has seen a fixed number of previous messages sent from Alice to Bob with the authentication tags attached, she will still be unable to change the current message or substitute a new one because of the computation resistance property of the MAC. This does require that Alice and Bob change the value of k when a large number of messages have been sent, so as to not give any information away to Eve about the value of k or about how to compute authentication tags.

CHAPTER 3. AUTHENTICATION SCHEMES USING QUASIGROUPS

3.1 Denes and Keedwell Scheme

Recall that as discussed in Section 2.2, message authentication codes (or MACs) are used to authenticate messages. If Alice and Bob are the only two people who share a secret key for a MAC, they can use the MAC and the secret key to create a check digit (also known as a MAC-value or an authentication tag) for a message. This check digit is then appended on to the end of the message. When Bob receives a message from Alice (or vice versa), Bob will use his knowledge of the key to compute the check digit and verify the message. If the check digit that Bob computes matches the check digit appended onto the message, then Bob can be sure that the message came from Alice, because she is the only other person with knowledge of the secret key (and therefore, the only person with knowledge of how to compute the check digit). If the check digits do not match, then Bob knows either that the message is not from Alice or that it was changed in transit. Either way, Bob knows not to trust the contents of the message.

Therefore, in order to create a MAC, a function $h_k(x)$ is needed where h_k has the properties described in Definition 3. In particular, $h_k(x)$ should be easy to compute with knowledge of k and difficult to compute without knowledge of k. MACs have been created from a variety of mathematical structures which use many different pieces of information for the secret key. One classic example of a MAC is a hash-based MAC, called HMAC.

Definition 4. Let h be a hash function (as defined in Definition 2) and let k be a key. Create a MAC (called **HMAC**) by defining

$$HMAC(x) = h(k\|p_1\|h(k\|p_2\|x))$$

where p_1 and p_2 are distinct strings used to pad k to a full block (that is, make the input to HMAC the appropriate length) and $\|$ denotes concatenation.

Because h is a hash function, it is a one-way function according to Definition 2. Therefore, HMAC(x) will be difficult to compute without knowledge of k, because both inputs to h (that is, $k\|p_2\|x$ and $k\|p_1\|h(k\|p_2\|x)$) will be unknown.

In (4), Denes and Keedwell suggest using a quasigroup $(Q, \circ)$ and its binary operation to create a MAC. Their scheme is outlined below.

Let $(Q, \circ)$ be a quasigroup, where $|Q| = q$. Assume that $\mathbf{m} = m_1, m_2, \ldots, m_n$ is a message over $(Q, \circ)$ and assume that we wish to create a signature consisting of $b_0, b_1, \ldots, b_{s-1}$ for $\mathbf{m}$, where $b_i \in Q$ for $i = 0, \ldots, s-1$. Choose t so that $n = st$. We will separate $\mathbf{m}$ into s mutually disjoint subsets S_i, for $i = 0, \ldots, s-1$ where each S_i consists of exactly t elements of the message. (If the message length n is not an exact multiple of t, so that we have $n = (s-1) * t + r$ where $0 < r < t$, then create the subsets so that the last subset S_{s-1} contains r message elements instead of t elements.) Assume that $S_i = \{m_{i_1}, m_{i_2}, \ldots, m_{i_t}\}$. Then calculate b_i by

$$b_i = [(\{(m_{i_1} \circ m_{i_2}) \circ m_{i_3}\} \circ m_{i_4}) \circ \cdots] \circ m_{i_t}$$

The message and signature are then sent as $m_1, \ldots, m_n, b_0, \ldots, b_{s-1}$.

As with any other MAC, the objective of this scheme is to authenticate a message $m_1, \ldots, m_n$ by computing $b_0, \ldots, b_{s-1}$ and appending this to the end of the message. The expectation is that without knowledge of the key (in this case, how the sets S_i are created), an attacker will not be able to compute $b_0, \ldots, b_{s-1}$ correctly. Therefore, a message which has the correct MAC value (that is, the correct values of $b_0, \ldots, b_{s-1}$) must have been sent by someone with knowledge of the key, and so the message can be authenticated.

Because the parenthesis scheme is the same for computing every value of b_i, the security of the scheme lies in how the sets S_i are created. Denes and Keedwell suggest the following method for this creation.

Define a Latin square L which will be used to create S_i. If s many subsets S_i are to be created, then L should consist of the elements $\{0, 1, \ldots, s-1\}$. Notice that L will contain

s^2 elements, so the positions in L can be labeled from 1 to s^2, beginning with the upper left corner, traveling across the first row, returning to the leftmost element in the second row, and so on. Then each element from $\{0, 1, \ldots, s-1\}$ will have exactly s numbered positions associated with it. If $i \in \{0, 1, \ldots, s-1\}$ has associated positions $i_1, i_2, \ldots, i_s$, then the subset S_i will consist of message elements $m_{i_1}, m_{i_2}, \ldots, m_{i_s}$.

This method of creating a signature $b_0, b_1, \ldots, b_{s-1}$ can be thought of as a MAC. Denote this MAC by h_k, where $h_k(m_{i_1}, m_{i_2}, \ldots, m_{i_t}) = b_i$. This implies that the private information is the method of dividing the message $\mathbf{m}$ into sets $S_0, S_1, \ldots, S_{s-1}$. Because the Latin square L was used to create these sets, the key for this MAC is L.

In this method, we are working with both a quasigroup $(Q, \circ)$ and a Latin square L. In order to reduce the storage space needed, Denes and Keedwell suggest using the same structure for both $(Q, \circ)$ and L. Notice, however, that this implies that $(Q, \circ)$ cannot be made public.

Example 3. Assume that we would like to create a signature using the method above for the message 1033210322001120 using the Latin square L given in Figure 3.1. Then $n = 16$ and we choose $s = 4$, implying that $t = 4$. Subset S_0 will consist of the 2nd, 8th, 11th, and 13th message elements, because 0 appears in positions 2, 8, 11, and 13 of L. S_1 will consist of the 1st, 6th, 12th, and 15th message elements, because 1 appears in positions 1, 6, 12, and 15 of L. S_2 and S_3 can be similarly constructed.

$$L = \begin{array}{cccc} 1 & 0 & 3 & 2 \\ 3 & 1 & 2 & 0 \\ 2 & 3 & 0 & 1 \\ 0 & 2 & 1 & 3 \end{array} \qquad Q = \begin{array}{c|cccc} \circ & 0 & 1 & 2 & 3 \\ \hline 0 & 1 & 0 & 3 & 2 \\ 1 & 3 & 1 & 2 & 0 \\ 2 & 2 & 3 & 0 & 1 \\ 3 & 0 & 2 & 1 & 3 \end{array}$$

Figure 3.1 The Latin square L and its associated quasigroup $(Q, \circ)$.

If we choose to use the quasigroup $(Q, \circ)$ created by using L as its multiplication table as suggested above, then the signature is as follows:

$$S_0 = 0, 3, 0, 1 \qquad b_0 = [(0 \circ 3) \circ 0] \circ 1 = (2 \circ 0) \circ 1 = 2 \circ 1 = 3$$

$$S_1 = 1,1,0,2 \qquad b_1 = [(1 \circ 1) \circ 0] \circ 2 = (1 \circ 0) \circ 2 = 3 \circ 2 = 1$$

$$S_2 = 3,0,2,1 \qquad b_2 = [(3 \circ 0) \circ 2] \circ 1 = (0 \circ 2) \circ 1 = 3 \circ 1 = 2$$

$$S_3 = 3,2,2,0 \qquad b_3 = [(3 \circ 2) \circ 2] \circ 0 = (1 \circ 2) \circ 0 = 2 \circ 0 = 2$$

Therefore, the message and signature are sent as 10332103220011203122.

As with virtually any MAC, there are both positive and negative aspects of Denes and Keedwell's scheme. One positive aspect deals with security. Although Denes and Keedwell do not discuss the security of this scheme in (4), the security is analyzed in (2). Dawson, Donovan, and Offer conclude that if the product is obtained by multiplying consecutive digits of the message together, then it is possible for an eavesdropper to forge a signature. This corresponds to simply letting $S_i = \{m_{i(s-1)+1}, m_{i(s-1)+2}, \ldots, m_{is}\}$ for $i = 0, \ldots, s-1$. However, if a Latin square is used to divide the message up into the sets S_i, then Dawson, Donovan, and Offer conclude that this method is secure against the attack developed in their paper. Other than (2), there have been no other papers published exploring the security of Denes and Keedwell's authentication scheme, so it appears that currently, the scheme is considered secure.

In addition to the apparent security of this method, another positive aspect can be seen in the necessary computations that need to be performed. If the Latin square L (and hence the corresponding quasigroup $(Q, \circ)$) are exchanged ahead of time, the sets S_i can be "precomputed". That is, even though the message elements aren't yet known (because the message hasn't been sent yet), it can be determined that S_i will consist of the message elements in positions $i_1, i_2, \ldots, i_t$. When the message is sent, the actual contents of S_i can be quickly computed. The only other computation which cannot be performed until the message is sent or received is the multiplication of the elements of S_i to calculate b_i. However, this is a relatively fast process because it only involves quasigroup multiplications.

There are also negative aspects of this MAC. It appears that the most problematic aspect is the fact that the key is a quasigroup, possibly one which is quite large (depending on the block size chosen for the message). If Alice and Bob wish to use this MAC to authenticate

messages, they would need to find a way to efficiently represent the quasigroup $(Q, \circ)$ and also find a way to exchange $(Q, \circ)$ securely. In addition, if Alice and Bob want to create an authentication tag of s digits, then $(Q, \circ)$ would need to consist of s^2 many elements, making $(Q, \circ)$ quite large to store.

Another negative feature of Denes and Keedwell's scheme is that it does not quite satisfy the requirements for a MAC. In Definition 3, it is required that a MAC produce a fixed-length output. However, the output from this scheme is $b_0, b_1, \ldots, b_{s-1}$, which depends on the value of s.

One final negative aspect of this MAC comes from the fact that properties of the quasigroup $(Q, \circ)$ are not being utilized. This authentication scheme would work if we used a group G instead of a quasigroup $(Q, \circ)$, especially since the parentheses scheme used to create b_i remains fixed for any set S_i and any quasigroup $(Q, \circ)$. That is, b_i is created by multiplying all the elements in S_i together from left to right. Because $(Q, \circ)$ is a quasigroup, $(Q, \circ)$ could be chosen so that it is non-associative. However, by multiplying from left to right, this non-associativity is not beng exploited. Therefore, a new MAC (which we call QMAC) can be created which relies on the non-associativity of $(Q, \circ)$ for its security. QMAC is discussed in Section 3.2.

3.2 Creation of QMAC

Denes and Keedwell's message authentication scheme (or MAC) discussed in Section 3.1 can be modified to create a MAC (called QMAC) which utilizes the non-associative properties of quasigroups. In QMAC, the elements of the message will again be thought of as elements of some quasigroup $(Q, \circ)$. Unlike the scheme in (4), $(Q, \circ)$ can be made public. The secret key for QMAC will be the order in which the message elements are multiplied together to create the MAC-value. If the quasigroup $(Q, \circ)$ is chosen to be non-associative, then the order in which message elements are multiplied will change the check digit that is created. Essentially, this allows us to have a parentheses scheme as the secret key.

Definition 5. Let $(Q, \circ)$ be a quasigroup.

1. By a **parenthesis scheme** on $x_1, x_2, \ldots, x_t$, we simply mean a choice of parenthesizing $x_1 \circ x_2 \circ \cdots \circ x_t$ so that it forms a well-defined expression in the non-associative operation $\circ$.

2. A **QMAC key of length** t is a parenthesis scheme on $x_1, \ldots, x_t$ along with a constant $c \in Q$.

The authentication tag for a message $\mathbf{m}$ is then computed by multiplying the message elements together in the order specified by the key, except that every innermost multiplication $(x_i \circ x_{i+1})$ is replaced by $(x_i \circ c) \circ x_{i+1}$.

Example 4. Let $Q = \{0, 1, \ldots, 7\}$ and let the Cayley table for $(Q, \circ)$ be given in Figure 3.2.

$Q =$

$\circ$	0	1	2	3	4	5	6	7
0	5	3	2	1	7	6	0	4
1	0	6	5	4	2	1	3	7
2	2	0	7	6	4	3	5	1
3	7	5	4	3	1	0	2	6
4	6	4	3	2	0	7	1	5
5	3	1	0	7	5	4	6	2
6	4	2	1	0	6	5	7	3
7	1	7	6	5	3	2	4	0

Figure 3.2 The quasigroup $(Q, \circ)$ for Example 4.

Assume that we want to create a check digit for the message $\mathbf{m} = 146277$. If we choose the key to be the parenthesis scheme $(m_1 \circ m_2) \circ (((m_3 \circ m_4) \circ m_5) \circ m_6)$ along with the constant $c = 3$, then our authentication tag is

$$\begin{aligned} h_k(\mathbf{m}) &= ((1 \circ 3) \circ 4) \circ ((((6 \circ 3) \circ 2) \circ 7) \circ 7) \\ &= (4 \circ 4) \circ (((0 \circ 2) \circ 7) \circ 7) \\ &= 0 \circ ((2 \circ 7) \circ 7) \\ &= 0 \circ (1 \circ 7) \\ &= 0 \circ 7 \\ &= 4 \end{aligned}$$

and so the message and the authentication tag would be sent as 1462774.

The number of possible parentheses schemes will be discussed in Section 3.3.2, and it can be seen that unless $(Q, \circ)$ is very large, there are more parentheses schemes than quasigroup elements. Therefore, the pigeonhole principle says that there will be collisions (that is, two different parentheses schemes which produce the same authentication tag). However, it appears to be difficult to find one of these collisions without knowledge of the key, so QMAC still satisfies the properties required by Definition 3.

3.3 Implementation of QMAC

We showed how to calculate $h_k(m_1, m_2, \ldots, m_t)$ in Example 4, where t is the number of inputs that the function h_k takes. This implies that if our message $\mathbf{m}$ consists of exactly t elements, we can calculate $h_k(\mathbf{m})$. However, this situation is extremely unlikely to occur in practice, because messages are usually quite long and vary in length. Therefore, we need to consider methods of creating an authentication tag for a message which consists of more than t elements. Several possible methods will be discussed in Section 3.3.1. For ease of notation, let $h_k(m_1, \ldots, m_t)$ denote the multiplication of $m_1, \ldots, m_t$ in the order specified by k and let $H(\mathbf{m})$ denote the authentication tag created for the message $\mathbf{m}$.

In addition to the above concern, we also need to consider issues regarding the key. Because a QMAC key is a parenthesis scheme on t elements along with a single quasigroup element, we need to have some way of representing the key. The size of the keyspace also needs to be examined. Notice that the size of the keyspace depends on t and on $|Q|$. These topics associated with the key will be discussed in Section 3.3.2.

Finally, the security of this MAC will be examined in Section 3.3.3.

3.3.1 Implementing QMAC on a longer message

Before discussing how to compute $H(\mathbf{m})$, notice that there is an important difference between QMAC and the MAC created by Denes and Keedwell. Recall that the security of Denes and Keedwell's MAC is derived from the way the sets S_i are created. This means that

in order to maintain the MAC's security, the method for breaking the message up into blocks needs to be kept secret. However, in QMAC, the security depends on the order in which the message elements are multiplied together. Therefore, we don't need to be at all concerned about how the message is broken up into blocks. We can break the message up into blocks in any way that we choose, and we can make this method of creating blocks public.

There are several different methods that can be used to define $H_k(\mathbf{m})$ from h_k. The description of each method follows, along with an analysis of its space and time complexity. In each case, assume that h_k takes t message elements as input.

Method 1: This method is similar to the way that many "traditional" hash functions are implemented, by treating h_k as a compression function and beginning with a fixed initial value (IV) which is an element of $(Q, \circ)$. The MAC value computed in the previous step is used as one of the inputs to h_k, which is being computed in the current step. For this method, assume that $\mathbf{m}$ consists of $(t-1) \cdot l$ quasigroup elements for some positive integer l. That is, assume $\mathbf{m} = m_1, m_2, \ldots, m_{(t-1)l}$, where $m_i \in Q$ for $i = 1, \ldots, (t-1)l$. If necessary, Alice can pad the message $\mathbf{m}$ in order to make its length a multiple of $t-1$. That is, Alice can add data to the end of the message to make it the correct length. This data could either be random or it could contain information about the message, such as its length and blocksize. This method is described by

$$\begin{aligned} x_0 &= IV \in Q \\ x_i &= h_k\left(x_{i-1}, m_{(i-1)(t-1)+1}, \ldots, m_{i(t-1)}\right) \quad \text{for } i = 1, \ldots, l \\ H_k(\mathbf{m}) &= x_l \end{aligned}$$

Storage: Notice that the hash value x_i is only used to compute the value of x_{i+1}. This means that when x_{i+1} is computed, there is no longer any need to store x_i, so x_{i+1} can replace x_i in the storage queue. Therefore, in addition to the storage needed for the message, this method requires that only one additional quasigroup element be stored.

Time: In order to analyze the time complexity of each method, we first need to determine

how many quasigroup multiplications will be performed. In Method 1, because $i = 1, \ldots, l$, we need to compute h_k l-many times. Notice, however, that we do not know the exact number of quasigroup multiplications each computation of h_k requires because each innermost product in k requires one additional multiplication. Since the number of variables in the parenthesis scheme is t, we can certainly bound the total number of quasigroup multiplications by $2tl$. Thus the time complexity is linear in the size of the message.

Method 2: For this method, assume that $|\mathbf{m}| = t \cdot l$, again padding $\mathbf{m}$ if necessary. This method is similar to Method 1. It is described by

$$\begin{aligned} x_0 &= IV \in Q \\ x_i &= x_{i-1} \circ h_k\left(m_{(i-1)t+1}, \ldots, m_{it}\right) \quad \text{for } i = 1, \ldots, l \\ H_k(\mathbf{m}) &= x_l. \end{aligned}$$

Storage: As in Method 1, the hash value x_i is only used to compute the value of x_{i+1}. When x_{i+1} is computed, there is no longer any need to store x_i, so x_{i+1} can replace x_i in the storage queue. Therefore, in addition to the storage needed for the message, this method also requires that only one additional quasigroup element be stored.

Time: Again, we need to compute h_k l-many times. As in Method 1, we can bound the total number of quasigroup multiplications by $2tl$. Thus the time complexity is again linear in the length of the message. So Method 2 requires the same amount of storage and approximately the same amount of time as Method 1.

Method 3: For this method, we arrange the message elements as the levels in a complete, balanced t-ary tree. Each internal node of the tree is obtained by applying h_k to its t descendants. The final value of $H_k(\mathbf{m})$ is the value at the root node. A pictorial representation for Method 3 when $t = 3$ can be seen in Figure 3.3. For ease of notation, let $m_1' = h_k(m_1, m_2, m_3)$, $m_2' = h_k(m_4, m_5, m_6)$, and so on.

This method can be implemented using a single push-down stack. The stack accepts ordered pairs (q, j) in which q is a quasigroup element and j is a natural number corresponding to the

$$\underbrace{\underbrace{m_1, m_2, m_3}_{m'_1}\ \underbrace{m_4, m_5, m_6}_{m'_2}\ \underbrace{m_7, m_8, m_9}_{m'_3}}_{m''_1}\ \underbrace{\underbrace{m_{10}, m_{11}, m_{12}}_{m'_4}\ \underbrace{m_{13}, m_{14}, m_{15}}_{m'_5}\ \underbrace{m_{16}, m_{17}, m_{18}}_{m'_6}}_{m''_2}$$

etc.

Figure 3.3 A pictorial representation of Method 3.

level in the tree at which q appeares. Begin with the stack empty and proceed according to the following algorithm.

1. For $i = 1$ to n do
2. Push $(m_i, 0)$ onto stack
3. If top t elements of stack all have same level j, then
4. Pop t elements from stack
5. Apply h_k to obtain value q
6. Push $(q, j + 1)$ onto stack
7. Goto (3)
8. Fi
9. Od

It is most natural to apply this algorithm to a message of length $n = t^l$. In that case, upon completion of the main loop, the stack will contain a single element (q, l) and the value of $H_k(\mathbf{m})$ will be q.

Storage: It is easy to see that at any point in the algorithm, the level components (that is, the second component in (q, j)) increase as we go down the stack. In addition, there are never more than $t - 1$ entries of any level on the stack at once. It therefore follows that the maximum stack size is $(t - 1)l$ entries, which is logarithmic in the length of the message.

Time: The number of applications of h_k is equal to the number of internal nodes in the tree, which is $1 + t + t^2 + \cdots + t^{l-1}$. Since each application of h_k requires approximately t quasigroup multiplications, the total time involved is $t \cdot \left(\frac{t^l-1}{t-1}\right) \approx t^l - 1$.

As we have seen, all three approaches run in linear time. However, unlike Method 3, Methods 1 and 2 require only a constant amount of auxiliary storage. Method 3, on the other hand, does not use an initial value and may pipeline better. It is not clear whether or not these advantages for Method 3 outweigh the increased storage requirements or the added complexity of the algorithm.

3.3.2 Keyspace Concerns

Recall that the key for QMAC is an order for performing multiplication on a string of t quasigroup elements (represented by a parenthesis scheme) along with a quasigroup element c. We need to describe how to represent this key, discuss the size of the keyspace, and determine how large t needs to be in order to have an acceptably large keyspace.

Representing the Key. The way that we choose to represent the key involves changing the parentheses order into a string of numbers. For example, if $t = 7$, consider the following grouping of parentheses:

$$((m_1 \underset{\uparrow 1}{\circ} m_2) \underset{\uparrow 2}{\circ} m_3) \underset{\uparrow 3}{\circ} ((m_4 \underset{\uparrow 4}{\circ} (m_5 \underset{\uparrow 5}{\circ} m_6)) \underset{\uparrow 6}{\circ} m_7)$$

We could represent this parentheses scheme as 1, 2, 5, 4, 6, 3, because the first multiplication performed is 1, followed by 2, then multiplication 5 is performed, and so on. Notice that there are multiple ways of representing this parentheses scheme. A few are:

1, 2, 5, 4, 6, 3

1, 5, 2, 4, 6, 3

5, 4, 6, 1, 2, 3

Because a QMAC key involves both a parenthesis scheme and an additional quasigroup element c, the QMAC key corresponding to the parenthesis scheme detailed above could be represented

as $1, 2, 5, 4, 6, 3, c$. In this case, the innermost products are $m_1 \circ m_2$ and $m_5 \circ m_6$. Therefore, h_k would be computed as follows.

$$h_k(m_1, \ldots, m_7) = (((m_1 \circ c) \circ m_2) \circ m_3) \circ ((m_4 \circ ((m_5 \circ c) \circ m_6)) \circ m_7).$$

Storage Needed for the Key. As discussed in Section 3.3.1, there are $t-1$ quasigroup operations that are performed when h_k is computed. This means that the key will be a permutation of the set $\{1, \ldots, t-1\}$, along with an additional quasigroup element c. Therefore, the space required to store the key is $(t-1) \lg (t-1) + \lg |Q|$, which is approximately linear in the number of inputs to h_k and logarithmic in the size of $(Q, \circ)$.

Size of the Keyspace. At first glance, because the keys are simply permutations of $\{1, \ldots, t-1\}$ along with a quasigroup element c, it seems as though the keyspace should have size $(t-1)! \cdot |Q|$. However, because there are some keys which have more than one representation (see the previous discussion), the size of the keyspace is actually smaller. Notice that it is true that the keyspace contains $(t-1)! \cdot |Q|$ valid keys. However, if two of these keys produce the same parentheses scheme, as soon as an attacker finds one of these keys the security of QMAC will be compromised. It doesn't matter whether the attacker finds the representation of the key that was actually used or not—what matters is that he has found the parentheses scheme that QMAC uses, and hence has found the secret key.

To determine the number of unique keys in the keyspace, we need to know the number of ways to uniquely parenthesize a string of t elements. This is equal to C_{t-1}, where C_n is the n^{th} Catalan number. So $|\mathcal{K}| = C_{t-1} \cdot |Q|$. There are several equivalent formulas for C_n. The formula given in (13) is

$$\begin{aligned} C_n &= \frac{1}{n+1}\binom{2n}{n} \\ &= \frac{(2n)!}{n!(n+1)!} \end{aligned}$$

It turns out that $C_{68} < 2^{128} < C_{69}$, so if a keyspace containing 2^{128} elements is desired, h_k would need to take at most 70 quasigroup elements as input (that is, we would need $t \leq 70$). Because the quasigroup $(Q, \circ)$ will probably be quite large, h_k could take significantly fewer

quasigroup elements as input and still have a keyspace containing 2^{128} elements. If $|Q| = 2^8$, h_k would need to take at most 66 quasigroup elements as input. If $|Q| = 2^{16}$, we would need at most 62 inputs to h_k. Choosing $(Q, \circ)$ so that $|Q| = 2^{32}$ requires that h_k only take 52 quasigroup elements as input.

Asymptotically, $C_n \sim \frac{4^n}{n^{3/2}\sqrt{\pi}}$. Notice that $C_n = \Omega(2^n)$, because

$$\begin{aligned} \log_2 C_n &= n \log_2 4 - \frac{3}{2} \log_2 n - \log_2 \sqrt{\pi} \\ &= 2n - \frac{3}{2} \log_2 n - .82575 \\ &> 2n - n - .82575 \qquad (3.1) \\ &= n - .82575 \\ &\sim n \end{aligned}$$

where (3.1) holds because $\log_2 n < \frac{2}{3}n$ for $n \geq 3$. This implies that $C_n \sim 2^n$ or $C_n = \Omega(2^n)$. Therefore, the size of the keyspace increases exponentially in the length of the key.

3.3.3 Security Concerns.

Before we can consider the security of QMAC, we need to understand what it means for a MAC to be secure. Assume that Alice and Bob are sending messages back and forth and assume that Eve is an attacker. Eve's goal is to disrupt the communication between Alice or Bob, which she can do in several ways. Eve could intercept a message from Alice to Bob (or vice versa), alter the message, and then re-send it. She could also create her own message and send it to Alice, pretending that the message came from Bob (or vice versa). If Alice and Bob are using a MAC to provide authentication for their messages, then either attack by Eve requires that she be able to compute an authentication tag for a message. This is known as forging an authentication tag.

If Eve was somehow able to obtain knowledge of the key k, she could clearly forge an authentication tag for any message she wanted. The only secret information contained in the computation of the authentication tag is the key. All the specifications for the MAC, the

message $\mathbf{m}$, and the quasigroup $(Q, \circ)$ are made public. Therefore, if Eve has the key, she can compute authentication tags in the same way as Alice and Bob.

However, it is also possible that Eve might be able to forge authentication tags even without knowledge of the key k. If she sees enough message/authentication pairs pass between Alice and Bob, she may be able to gain information about how to compute the authentication tag without knowledge of the key. Therefore, we need to make sure that a MAC is secure against this possibility as well.

Before discussing the types of forgeries that Eve can compute, we need to talk about what sort of abilities Eve has. That is, we would like to know what type of attack Eve is capable of. There are several types of forgeries, or attacks, that Eve can attempt. In each of the attacks described in this section, we assume that Eve has access to an oracle which models the behavior of h_k for some fixed (but arbitrary) value of k. That is, if Eve inputs x_i, the oracle responds with $h_k(x_i)$. Basically, this implies that for a message x_i, Eve is allowed to see the authentication tag associated with this message. The type of attack that Eve can mount depends on how much control she has over the values of x_i. There are several different kinds of attacks that Eve could attempt.

- A **known-text attack** is an attack where Eve has a certain number of message-authentication tag pairs $(x_i, h_k(x_i))$, but she has no control over what these pairs are. That is, Eve is allowed to use the oracle to compute $h_k(x_i)$, but she is not allowed to choose the values of x_i.

- A **chosen-text attack** is an attack where Eve has a certain number of message-authentication tag pairs $(x_i, h_k(x_i))$, and she is allowed to choose the values of x_i. However, she must choose all of the values of x_i before she starts using the oracle to compute $h_k(x_i)$.

- An **adaptive chosen-text attack** is an attack where Eve has a certain number of message-authentication tag pairs $(x_i, h_k(x_i))$, and she is allowed to choose the value of x_i based on the results of previous queries to the oracle. That is, Eve chooses x_1 and uses the oracle to compute $h_k(x_1)$. Based on this result, she chooses x_2 and uses the oracle

to compute $h_k(x_2)$. This process continues for as many calls as Eve is allowed to make to the oracle.

Clearly, an adaptive chosen-text attack gives Eve the most power. She can analyze previous message-authentication tag pairs and then use this information to try to guess the value of the key or to try to forge an authentication tag. However, notice that this situation may not always be allowed by the oracle. Perhaps the oracle takes a list of inputs and produces a list of outputs, and Eve is only allowed to input one list of messages to the oracle. This situation would only allow for a chosen-text attack, not an adaptive chosen-text attack.

In (9), Preneel requires that a MAC needs to be able to withstand an adaptive chosen-text attack regardless of whether or not such an attack is possible. This requirement ensures that a MAC will be secure against the "worst" kind of attack—that is, the type of attack in which the attacker can get the most information about how a MAC-value is computed from a message. Notice that if a MAC is secure against an adaptive chosen-text attack, it will also be secure against a chosen-text attack and a known-text attack. Therefore, if QMAC is secure against an adaptive chosen-text attack, it achieves the highest level of security desired.

The security of QMAC rests on the non-associativity of the quasigroup $(Q, \circ)$. Clearly, if $(Q, \circ)$ is associative, then for a block $m_1, \ldots, m_t$, every key k will produce exactly the same value of $h_k(m_1, \ldots, m_t)$. Therefore, in order to make QMAC more secure, we would like $(Q, \circ)$ to be a "highly non-associative" quasigroup. Methods for measuring "how non-associative" a quasigroup is and for constructing such quasigroups are discussed in Chapter 5. In addition, it appears that the more structure $(Q, \circ)$ has (for example, having an identity element or being commutative), the more the security of QMAC is in question. It is possible that an attacker may be able to use the structure present in $(Q, \circ)$ to assist in forging a MAC-value without knowledge of the key. Also, Q needs to be a large quasigroup in order to reduce the probability of an attacker finding a collision by randomly guessing a value of $h_k(m_1, \ldots, m_t)$. In addition, the size of the keyspace increases linearly as the size of $(Q, \circ)$ increases. Beyond these two relationships, it is not entirely clear what impact the size of $(Q, \circ)$ has on the security of QMAC. Practical methods for constructing large quasigroups are discussed in Chapter 4.

Many of the MACs that have been discussed in the literature are based on other cryptographic primitives, such as block ciphers or hash functions. A proof of security (if one is given at all) usually rests on the assumption that the underlying primitive is secure. Since QMAC is not based on a familiar primitive, it is unclear how to even formulate basic questions about its security and how to determine whether it is secure against an adaptive chosen-text attack.

It would seem more natural to define h_k without the inclusion of the constant $c \in Q$ (see Definition 5). However, it turns out that there is a chosen-text attack on that scheme, as we now show. Suppose our key k is simply a parenthesis scheme on $x_1, \dots x_t$. According to Definition 3, Eve is allowed to ask the oracle for the values of $h_k(m_1, \dots, m_t)$ for several inputs of her choosing.

Eve first chooses any message and asks for $h_k(m_1, \dots, m_t) = z$. Then for each value of $j = 1, 2, \dots, t-1$, she chooses elements m'_j, m'_{j+1} so that $m_j \neq m'_j$ and $m'_j \circ m'_{j+1} = m_j \circ m_{j+1}$. Notice that this is always possible by the second condition of Definition 1. Eve then requests the value

$$w_j = h_k(\dots, m_{j-1}, m'_j, m'_{j+1}, m_{j+2}, \dots).$$

If $(x_i \circ x_{i+1})$ is an innermost product of k, then $w_i = z$. Once such an i has been found, Eve is able to compute the values of $h_k(\dots, m_{i-1}, u, v, m_{i+2}, \dots) = z$ for any value of u by choosing v so that $u \circ v = m_i \circ m_{i+1}$. It may not be possible for Eve to forge authentication tags for any meaningful messages, but the fact that she can forge authentication tags violates the security requirements for a MAC.

The inclusion of the constant c in Definition 5 serves to hide the position of the innermost products. Notice that if the quasigroup contains an identity element, 1, then the value of $h_k(1, 1, 1, \dots, 1)$ will return some non-associated power of c. This does not seem to expose the value of c, but it is probably prudent to avoid quasigroups with an identity element.

Besides this technique, we currently know of no other attacks on QMAC. Note that this does not imply that QMAC is secure. It simply implies that at this time, no successful attacks against QMAC have been found. The best approach to determining whether QMAC is secure appears to be to design potential attacks and determine if they will succeed. If an attack is

successful, then the specifications of QMAC will need to be changed in order to guard against that attack.

CHAPTER 4. METHODS OF CONSTRUCTING LARGE QUASIGROUPS

In order to be able to practically use QMAC, we need to look at the underlying quasigroup. Recall that $(Q, \circ)$ is not part of the secret key, so we do not need to discuss how Alice and Bob can exchange $(Q, \circ)$. However, we do need to consider what properties $(Q, \circ)$ needs to have. There are two key properties that are required: that $(Q, \circ)$ is "large" and that it is "highly non-associative". Chapter 4 will deal with methods for creating and storing large quasigroups, and Chapter 5 will discuss issues associated with the non-associativity of the quasigroups.

In order to partially ensure QMAC's security, we need to work with quasigroups of large order. If $(Q, \circ)$ is small, it would be relatively easy for an attacker to find a collision for h_k, even without the key, because there would only be a small number of authentication tags available. Even if Eve does not know the key, she could still choose a random authentication tag for a message and have a good chance that it matches the actual authentication tag, thus succeeding in compromising the security of QMAC.

Because we will be working with quasigroups of large order, it is computationally infeasible (both in terms of space and in terms of time) to store the entire quasigroup and perform multiplication by a table lookup. This implies that we need to find a more efficient method for representing quasigroups of large order and performing multiplication within these quasigroups. In this chapter, methods will be discussed which address this problem.

4.1 Isotopies

One common way of creating quasigroups is through isotopies. The following definition is taken from (3).

Definition 6. Let $(Q_1, \circ)$ and $(Q_2, \star)$ be two quasigroups. Q_1 and Q_2 are **isotopic** if there are bijections $f, g, h : Q_1 \to Q_2$ so that $f(x \circ y) = g(x) \star h(y)$ for all $x, y \in Q_1$. The ordered triple (f, g, h) is called an **isotopy**.

Notice that an isotopy can be used to create a quasigroup $(Q, \star)$ from another quasigroup $(Q, \circ)$ by defining

$$x \star y = f^{-1}(g(x) \circ h(y))$$

for $x, y \in Q$.

Claim. If $(Q, \star)$ is created from $(Q, \circ)$ as described above, then $(Q, \star)$ is a quasigroup.

Proof. Clearly, Q is closed under $\star$. Therefore, we just need to show that $(Q, \star)$ has unique solubility of equations. This can be done by showing that $L_a(x) = a \star x$ and $R_a(x) = x \star a$ are permutations, for all $a \in Q$.

First, show that $L_a(x)$ is one-to-one for some arbitrary $a \in Q$. Assume that $L_a(x) = L_a(y)$ and show that $x = y$. Notice that because f is a bijection, f^{-1} is also a bijection. In addition, h is a bijection, so we have

$$\begin{aligned} L_a(x) &= L_a(y) \\ a \star x &= a \star y \\ f^{-1}(g(a) \circ h(x)) &= f^{-1}(g(a) \circ h(y)) \\ g(a) \circ h(x) &= g(a) \circ h(y) \\ h(x) &= h(y) \\ x &= y \end{aligned}$$

Next, show that $L_a(x)$ is onto. Let $y \in Q$. We need to find $x \in Q$ so that $L_a(x) = y$. Because $(Q, \circ)$ is a quasigroup, there is some $z \in Q$ so that $g(a) \circ z = f(y)$. Because h is a

bijection, h is onto, so there is some $x \in Q$ so that $h(x) = z$, or $x = h^{-1}(z)$. Then we have

$$\begin{aligned} L_a(x) &= a \star x \\ &= f^{-1}\left(g(a) \circ h(x)\right) \\ &= f^{-1}\left(g(a) \circ h(h^{-1}(z))\right) \\ &= f^{-1}\left(g(a) \circ z\right) \\ &= f^{-1}(f(y)) \\ &= y \end{aligned}$$

Hence $L_a(x)$ is a permutation on $(Q, \star)$. By symmetry, we can also see that $R_a(x)$ is also a permutation on $(Q, \star)$. This implies that $(Q, \star)$ is a quasigroup.

□

Example 5. Let $(Q, \circ) = (\mathbb{Z}_4, +)$ where $\mathbb{Z}_4 = \{0, 1, 2, 3\}$ and addition is computed modulo 4, so that the Cayley table for $(Q, \circ)$ is given in Figure 4.1. Let the isotopy (f, g, h) be as shown in Figure 4.2.

$Q =$

$\circ$	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

Figure 4.1 The Cayley table for $(Q, \circ)$

x	$f(x)$	$g(x)$	$h(x)$
0	1	2	3
1	0	3	1
2	3	1	0
3	2	0	2

Figure 4.2 The isotopy (f, g, h)

Then the quasigroup $(Q, \star)$ is produced and its Cayley table is given in Figure 4.3. Notice that although $(Q, \circ)$ is a group, $(Q, \star)$ is neither associative nor commutative and has no

identity element.

$$Q = \begin{array}{c|cccc} \star & 0 & 1 & 2 & 3 \\ \hline 0 & 0 & 2 & 3 & 1 \\ 1 & 3 & 1 & 2 & 0 \\ 2 & 1 & 3 & 0 & 2 \\ 3 & 2 & 0 & 1 & 3 \end{array}$$

Figure 4.3 The Cayley table for $(Q, \star)$

Normally when we are working with isotopies, we will choose f to be the identity map, so that a quasigroup operation is defined by

$$x \star y = g(x) \circ h(y).$$

4.1.1 Affine Isotopies

If the isotopies that we choose have special properties, then the quasigroup which is created by the isotopy will also have special properties. In particular, $(Q, \circ)$ has special properties if our isotopies are linear or affine maps.

Definition 7. Let $(Q, +)$ be a group and let $f : Q \to Q$. f is a **linear map** if $f(x + y) = f(x) + f(y)$ for all $x, y \in Q$. f is an **affine map** if $f(x + y) = f(x) + f(y) - f(0)$, where $0 \in Q$ denotes the identity element under $+$.

Notice that an affine map requires both an identity element and inverse elements, so we must start with a group in order to have the concept of an affine map well-defined. Therefore, in this section, we will always create a quasigroup $(Q, \circ)$ from a group $(Q, +)$.

Example 6. Let $Q = \mathbb{Z}_8 = \{0, 1, \ldots, 7\}$ and let the group operation be addition modulo 8. Then we can create a quasigroup $(Q, \circ)$ from $(\mathbb{Z}_8, +)$ by isotopy, with $f(x) = 3x + 4$ and

$g(x) = 5x + 1$. Notice that f and g are affine, because

$$\begin{aligned} f(x+y) &= 3(x+y)+4 \\ &= 3x+3y+4 \\ &= (3x+4)+(3x+4)-4 \\ &= f(x)+f(y)-f(0) \end{aligned}$$

and

$$\begin{aligned} g(x+y) &= 5(x+y)+1 \\ &= 5x+5y+1 \\ &= (5x+1)+(5y+1)-1 \\ &= g(x)+g(y)-g(0) \end{aligned}$$

Since we are creating $(Q, \circ)$ by isotopy, we will define

$$x \circ y = f(x) + g(y)$$

for $x, y \in \mathbb{Z}_8$. Then $(Q, \circ)$ is shown in Figure 4.4, where $Q = \mathbb{Z}_8$.

$Q =$

$\circ$	0	1	2	3	4	5	6	7
0	5	2	7	4	1	6	3	0
1	0	5	2	7	4	1	6	3
2	3	0	5	2	7	4	1	6
3	6	3	0	5	2	7	4	1
4	1	6	3	0	5	2	7	4
5	4	1	6	3	0	5	2	7
6	7	4	1	6	3	0	5	2
7	2	7	4	1	6	3	0	5

Figure 4.4 The Cayley table for $(\mathbb{Z}_8, \circ)$

Notice that $(Q, \circ)$ is not associative, because $(0 \circ 2) \circ 4 = 7 \circ 4 = 6$ but $0 \circ (2 \circ 4) = 0 \circ 7 = 0$. It is not commutative, because $0 \circ 1 = 2$ but $1 \circ 0 = 0$. There is no identity element and hence no inverses. However, $(Q, \circ)$ is still quite structured—the elements in each row always

appear in the same order, and the elements in each column are also arranged in the same order. In addition, each row is a copy of the previous row, shifted one space to the right. Each column is a copy of the previous column, shifted one space down. These properties hold for any quasigroup created by affine isotopies and will be formally stated in Properties 1 and 2. However, these properties are easier to see in a quasigroup created from $(Q, +)$, where Q is a group of exponent 2 (that is, for any $x \in Q$, $x + x = 0$, where $0 \in Q$ denotes the group identity element). Therefore, consider the following example.

Example 7. Let $Q = \mathbb{Z}_2^3 = \{\langle x_2, x_1, x_0\rangle : x_i \in \mathbb{Z}_2 \text{ for } i = 0, 1, 2\}$ and let the group operation on Q be exclusive-or, denoted by $\oplus$. This implies that $(Q, \oplus)$ has exponent 2, because $x \oplus x = \langle 0, 0, 0\rangle$ for all $x \in Q$. Then we can create a quasigroup $(Q, \circ)$ from $(\mathbb{Z}_2^3, \oplus)$ by choosing two affine maps f and g, where $f, g : \mathbb{Z}_2^3 \to \mathbb{Z}_2^3$ are bijections. Let

$$f(\langle x_2, x_1, x_0\rangle) = \langle x_1, x_0, x_2\rangle$$

and let

$$g(\langle x_2, x_1, x_0) = \langle x_2 \oplus x_1, x_0, x_2 \oplus 1\rangle$$

Then f is affine, because

$$\begin{aligned} f(\langle x_2, x_1, x_0\rangle \oplus \langle y_2, y_1, y_0\rangle) &= f(\langle x_2 \oplus y_2, x_1, \oplus y_1, x_0 \oplus y_0\rangle) \\ &= \langle x_1 \oplus y_1, x_0 \oplus y_0, x_2 \oplus y_2\rangle \\ &= \langle x_1, x_0, x_2\rangle \oplus \langle y_1, y_0, y_2\rangle \\ &= f(\langle x_2, x_1, x_0\rangle) \oplus f(\langle y_2, y_1, y_0\rangle) \oplus f(\langle 0, 0, 0\rangle) \end{aligned}$$

g is also affine, because

$$\begin{aligned} g(\langle x_2, x_1, x_0\rangle \oplus \langle y_2, y_1, y_0\rangle) &= g(\langle x_2 \oplus y_2, x_1 \oplus y_1, x_0 \oplus y_0\rangle) \\ &= \langle x_2 \oplus y_2 \oplus x_1 \oplus y_1, x_0 \oplus y_0, x_2 \oplus y_2 \oplus 1\rangle \\ &= \langle x_2 \oplus x_1, x_0, x_2 \oplus 1\rangle \oplus \langle y_2 \oplus y_1, y_0, y_2 \oplus 1\rangle \oplus \langle 0, 0, 1\rangle \\ &= g(\langle x_2, x_1, x_0\rangle) \oplus g(\langle y_2, y_1, y_0\rangle) \oplus g(\langle 0, 0, 0\rangle) \end{aligned}$$

For ease of notation, we will represent the group element $\langle x_2, x_1, x_0 \rangle$ by the integer x to which $\langle x_2, x_1, x_0 \rangle$ corresponds if x is written in binary. (For example, we would denote the string $\langle 0, 1, 0 \rangle$ by the integer 2.) If we define $x \circ y = f(x) \oplus g(y)$, then $(Q, \circ)$ is shown in Figure 4.5.

$Q =$

$\circ$	0	1	2	3	4	5	6	7
0	1	3	5	7	0	2	4	6
1	3	1	7	5	2	0	6	4
2	5	7	1	3	4	6	0	2
3	7	5	3	1	6	4	2	0
4	0	2	4	6	1	3	5	7
5	2	0	6	4	3	1	7	5
6	4	6	0	2	5	7	1	3
7	6	4	2	0	7	5	3	1

Figure 4.5 The Cayley table for $\left(\mathbb{Z}_2^3, \circ\right)$

Notice that $(Q, \circ)$ created in Example 7 is not associative, because $0 \circ (2 \circ 4) = 0 \circ 4 = 0$ but $(0 \circ 2) \circ 4 = 5 \circ 4 = 3$. It is commutative, but this property does not need to hold in general for a quasigroup created in this way. There is no identity element and hence no inverses. However, there are definite "pairing" properties present. For example, the elements 1 and 3 always appear next to each other in rows and columns, so we say they are "paired". Similarly, 5 and 7 are paired, and so on. These element pairing properties in rows and columns are defined more generally in Definitions 8 and 9.

Definition 8. Let $(Q, \circ)$ be a quasigroup and let $a, b, k, x, y \in Q$. Assume that:

- a appears in row x, column y.
- b appears in row x, column $y + k$.

Then elements a and b are **row paired over** k if whenever a appears in row z, column w, b appears in row z, column $w + k$, for any $z, w \in Q$.

Definition 9. Let $(Q, \circ)$ be a quasigroup and let $a, b, k, x, y \in Q$. Assume that:

- a appears in row x, column y.

- b appears in row $x + k$, column y.

Then elements a and b are **column paired over** k if whenever a appears in row z, column w, b appears in row $z + k$, column w, for any $z, w \in Q$.

In any quasigroup $(Q, \circ)$ created from $(Q, +)$ by affine isotopies, these pairing properties always hold. That is, for any $a, b, k, x, y \in Q$, if a appears in row x, column y and b appears in row x, column $y + k$, then a and b are row paired over k. Similarly, if a appears in row x, column y and b appears in row $x + k$, column y, then a and b are column paired over k. This pairing property is made more precise in Property 1.

Property 1. Let $(Q, \circ)$ be a quasigroup created from the group $(Q, +)$ by

$$x \circ y = f(x) + g(y) \text{ for } x, y \in Q$$

where $f, g : Q \to Q$ are bijective affine maps. Then

$$x \circ y = z \circ w \Rightarrow x \circ (y + k) = z \circ (w + k) \tag{4.1}$$

and

$$x \circ y = z \circ w \Rightarrow (x + k) \circ y = (z + k) \circ w \tag{4.2}$$

for any $x, y, z, w, k \in Q$.

Proof. Only (4.1) will be proved, the proof of (4.2) is similar. We have

$$\begin{gathered} x \circ y = z \circ w \\ f(x) + g(y) = f(z) + g(w) \\ f(x) + g(y) + g(k) - g(0) = f(z) + g(w) + g(k) - g(0) \\ f(x) + g(y + k) = f(z) + g(w + k) \\ x \circ (y + k) = z \circ (w + k) \end{gathered}$$

□

Another "pairing" property that can be seen in $(\mathbb{Z}_2^3, \circ)$ in Figure 4.5 is the pairing of entire rows and columns. For example, notice that row 0 read from left to right and row 7 read from right to left are exactly the same sequence of elements from $\mathbb{Z}_2^3$. This holds true for every row in $(\mathbb{Z}_2^3, \circ)$—each row is the reversal of another row in $(\mathbb{Z}_2^3, \circ)$. Another type of row pairing is the reversal of elements in columns for two given rows. For example, row 0 and row 1 have every pair of elements reversed. (That is, row 0 has 1 3 5 7 $\cdots$ and row 1 has 3 1 5 7 $\cdots$.) Each row in $(\mathbb{Z}_2^3, \circ)$ is paired with another row in this way. The same pairing properties hold for columns—for example, columns 0 and 7 are paired in the first sense, and columns 0 and 1 are paired in the second sense. These row and column pairing properties are defined more generally in Definitions 10 and 11.

Definition 10. Let $(Q, \circ)$ be a quasigroup and let $k, x, y \in Q$. Rows x and y are **paired over** k if for any $a, z \in Q$, whenever a appears in row x, column z, a also appears in row y, column $z + k$.

Definition 11. Let $(Q, \circ)$ be a quasigroup and let $k, x, y \in Q$. Columns x and y are **paired over** k if for any $a, z \in Q$, whenever a appears in row z, column x, a also appears in row $z + k$, column y.

There are two natural questions that exist. First, given a quasigroup $(Q, \circ)$ and a row or column x, is there a row or column y with which x is paired over k? Second, assuming that row or column x has such a pair, what is the value of y? If our quasigroup is created by using affine isotopies, the answer to the first question is yes and the second question is answered in Property 2.

Property 2. Let $(Q, \circ)$ be a quasigroup created from the group $(Q, +)$ by

$$x \circ y = f(x) + g(y) \text{ for } x, y \in Q$$

where $f, g : Q \to Q$ are bijective affine maps. Then rows x and y are paired over k if and only if $x \circ 0 = y \circ k$, where $0 \in Q$ denotes the group identity element. That is, for any $z \in Q$,

$$x \circ z = y \circ (z + k) \Leftrightarrow x \circ 0 = y \circ k \tag{4.3}$$

Notice that for $x, k \in Q$, y is guaranteed to exist because of unique solubility of equations in a quasigroup.

Similarly, columns x and y are paired over k if and only if $0 \circ x = k \circ y$. That is, for any $z \in Q$,

$$z \circ x = (z + k) \circ y \Leftrightarrow 0 \circ x = k \circ y \tag{4.4}$$

Again, notice that y is guaranteed to exist.

Proof. Only (4.3) will be proved; the proof of (4.4) is similar.

$\Rightarrow$ Assume that $x \circ z = y \circ (z + k)$. Then we have

$$\begin{aligned} f(x) + g(z) &= f(y) + g(z + k) \\ f(x) + g(z) &= f(y) + g(z) + g(k) - g(0) \\ f(x) &= f(y) + g(k) - g(0) \\ f(x) + g(0) &= f(y) + g(k) \\ x \circ 0 &= y \circ k \end{aligned}$$

$\Leftarrow$ Assume that $x \circ 0 = y \circ k$. Then we have

$$\begin{aligned} f(x) + g(0) &= f(y) + g(k) \\ f(x) &= f(y) + g(k) - g(0) \\ f(x) + g(z) &= f(y) + g(z) + g(k) - g(0) \\ f(x) + g(z) &= f(y) + g(z + k) \\ x \circ z &= y \circ (z + k) \end{aligned}$$

□

4.1.2 Non-affine Isotopies

It is also possible to create a quasigroup using isotopies that are not affine maps. In this case, none of the properties discussed in Section 4.1.1 are required to hold, as shown in Example 8.

x	0	1	2	3	4	5	6	7
$f(x)$	1	4	6	3	2	7	0	5
$g(x)$	4	2	1	0	6	5	7	3

Figure 4.6 Non-affine isotopies f and g

Example 8. Let $Q = \mathbb{Z}_8$ as defined in Example 6 and let the group operation on Q be addition modulo 8. Let $f, g : Q \to Q$ be as in Figure 4.6.

Notice that both f and g are non-affine bijections, because

$$f(1+3) = f(4) = 2 \text{ but } f(1) + f(3) - f(0) = 4 + 3 - 1 = 6$$

and

$$g(2+4) = g(6) = 7 \text{ but } g(2) + g(4) - g(0) = 1 + 6 - 4 = 3.$$

If a quasigroup $(Q, \circ)$ is created by isotopy from f and g (so $x \circ y = f(x) + g(y)$), then $(Q, \circ)$ is shown in Figure 4.7.

$Q =$

$\circ$	0	1	2	3	4	5	6	7
0	5	3	2	1	7	6	0	4
1	0	6	5	4	2	1	3	7
2	2	0	7	6	4	3	5	1
3	7	5	4	3	1	0	2	6
4	6	4	3	2	0	7	1	5
5	3	1	0	7	5	4	6	2
6	4	2	1	0	6	5	7	3
7	1	7	6	5	3	2	4	0

Figure 4.7 The Cayley table for $(Q, \circ)$ created from a non-affine isotopy

Notice that $(Q, \circ)$ does not have any of the pairing properties defined in Section 4.1.1. For example, 3 and 2 are adjacent to each other in row 0, but this does not happen in row 1. The elements 5 and 0 are adjacent to each other in column 0, but not in column 1. A similar analysis shows that none of the pairing properties hold.

However, even though none of these pairing properties hold, there are still other properties present in a quasigroup $(Q, \circ)$ created by isotopy from an abelian group. One such property

is described in Property 3.

Property 3. Let $(Q, \circ)$ be a quasigroup created from an abelian group $(Q, +)$ by

$$x \circ y = f(x) + g(y) \text{ for } x, y \in Q$$

where $f, g : Q \to Q$ are bijections. Then

$$\left.\begin{array}{l} a \circ c = x \\ a \circ d = x + z \\ b \circ c = y \end{array}\right\} \implies b \circ d = y + z$$

Proof. By the definition of $x \circ y$, we can see that the following equations are true.

$$\begin{array}{lll} f(a) + g(c) = x & & \\ f(a) + g(d) = x + z & \Rightarrow & g(d) = x + z - f(a) \\ f(b) + g(c) = y & \Rightarrow & f(b) = y - g(c) \end{array}$$

Therefore,

$$\begin{aligned} b \circ d &= f(b) + g(d) \\ &= y - g(c) + x + z - f(a) \\ &= y + z + x - f(a) - g(c) \\ &= y + z, \text{ as desired.} \end{aligned}$$

□

Example 9. Let $Q = \mathbb{Z}_{16}$ where addition is computed modulo 16, and let $f, g : Q \to Q$ be given in Figure 4.8.

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$f(x)$	1	6	9	15	0	7	8	14	3	4	11	13	2	5	10	12
$g(x)$	3	15	5	4	2	11	12	13	9	6	0	14	7	10	1	8

Figure 4.8 Isotopies f and g

Then the quasigroup $(Q, \circ)$ created from f and g by isotopy is shown in Figure 4.9. We can see that the relationship described in Property 3 is present in $(Q, \circ)$, as required. For example,

$Q =$

$\circ$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	4	0	6	5	3	12	13	14	10	7	1	15	8	11	2	9
1	9	5	11	10	8	1	2	3	15	12	6	4	13	0	7	14
2	12	8	14	13	11	4	5	6	2	15	9	7	0	3	10	1
3	2	14	4	3	1	10	11	12	8	5	15	13	6	9	0	7
4	3	15	5	4	2	11	12	13	9	6	0	14	7	10	1	8
5	10	6	12	11	9	2	3	4	0	13	7	5	14	1	8	15
6	11	7	13	12	10	3	4	5	1	14	8	6	15	2	9	0
7	1	13	3	2	0	9	10	11	7	4	14	12	5	8	15	6
8	6	2	8	7	5	14	15	0	12	9	3	1	10	13	4	11
9	7	3	9	8	6	15	0	1	13	10	4	2	11	14	5	12
10	14	10	0	15	13	6	7	8	4	1	11	9	2	5	12	3
11	0	12	2	1	15	8	9	10	6	3	13	11	4	7	14	5
12	5	1	7	6	4	13	14	15	11	8	2	0	9	12	3	10
13	8	4	10	9	7	0	1	2	14	11	5	3	12	15	6	13
14	13	9	15	14	12	5	6	7	3	0	10	8	1	4	11	2
15	15	11	1	0	14	7	8	9	5	2	12	10	3	6	13	4

Figure 4.9 The Cayley table for $(\mathbb{Z}_{16}, \circ)$ created by isotopy from $(\mathbb{Z}_{16}, +)$

notice that $5 \circ 4 = 9$, $5 \circ 8 = 0$, and $15 \circ 4 = 14$. In the notation of Property 3, this indicates that $x = 9$, $z = 7$, and $y = 14$. Therefore, we must also have $15 \circ 8 = 14 + 7 = 5$.

If we use the quasigroup $(Q, \circ)$ created by isotopy from $(\mathbb{Z}_2^k, \oplus)$, then Property 3 can be restated in a simpler form. Because $\oplus$ is its own inverse operation, we can move elements from one side of an equation to the other without needing to negate the element. This gives us the following property for $(\mathbb{Z}_2^k, \oplus)$.

Property 4. Let $(Q, \circ)$ be a quasigroup created from the group $(\mathbb{Z}_2^k, \oplus)$ by

$$x \circ y = f(x) \oplus g(y) \text{ for } x, y \in Q$$

where $f, g : \mathbb{Z}_2^k \to \mathbb{Z}_2^k$ are bijections. Then

$$a \circ c = b \circ d \Leftrightarrow a \circ d = b \circ c \text{ for } a, b, c, d \in Q.$$

Proof. $\Rightarrow$ Assume $a \circ c = b \circ d$. Then we have

$$f(a) \oplus g(c) = f(b) \oplus g(d)$$

$$f(a) \oplus g(d) = f(b) \oplus g(c)$$

$$a \circ d = b \circ c$$

The reverse implication follows immediately. □

Example 10. Let $f, g : \mathbb{Z}_2^4 \to \mathbb{Z}_2^4$ be the same as in Example 9, where elements of $\mathbb{Z}_2^4$ are represented as integers for ease of notation. The quasigroup $(Q, \circ)$ created from f and g by isotopy is shown in Figure 4.10.

$Q =$

$\circ$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	2	14	4	5	3	10	13	12	8	7	1	15	6	11	0	9
1	5	9	3	2	4	13	10	11	15	0	6	8	1	12	7	14
2	10	6	12	13	11	2	5	4	0	15	9	7	14	3	8	1
3	12	0	10	11	13	4	3	2	6	9	15	1	8	5	14	7
4	3	15	5	4	2	11	12	13	9	6	0	14	7	10	1	8
5	4	8	2	3	5	12	11	10	14	1	7	9	0	13	6	15
6	11	7	13	12	10	3	4	5	1	14	8	6	15	2	9	0
7	13	1	11	10	12	5	2	3	7	8	14	0	9	4	15	6
8	0	12	6	7	1	8	15	14	10	5	3	13	4	9	2	11
9	7	11	1	0	6	15	8	9	13	2	4	10	3	14	5	12
10	8	4	14	15	9	0	7	6	2	13	11	5	12	1	10	3
11	14	2	8	9	15	6	1	0	4	11	13	3	10	7	12	5
12	1	13	7	6	0	9	14	15	11	4	2	12	5	8	3	10
13	6	10	0	1	7	14	9	8	12	3	5	11	2	15	4	13
14	9	5	15	14	8	1	6	7	3	12	10	4	13	0	11	2
15	15	3	9	8	14	7	0	1	5	10	12	2	11	6	13	4

Figure 4.10 The Cayley table for $(\mathbb{Z}_2^4, \circ)$ created from non-affine isotopies

Notice that both f and g are non-affine bijections, because

$$f(2 \oplus 9) = f(11) = 13 \text{ but } f(2) \oplus f(9) \oplus f(0) = 9 \oplus 4 \oplus 1 = 12$$

and

$$g(2 \oplus 9) = g(11) = 14 \text{ but } g(2) \oplus g(9) \oplus g(0) = 5 \oplus 6 \oplus 3 = 0.$$

However, as shown in Figure 4.10, $(\mathbb{Z}_2^4, \circ)$ has the pairing described in Property 4. For example, notice that $5 \circ 4 = 5$, $5 \circ 8 = 14$, and $15 \circ 4 = 14$. Therefore, we must also have $15 \circ 8 = 5$.

It is not clear whether these pairing properties pose security risks for QMAC. Because the secret key is the parenthesis scheme and the quasigroup is public information in QMAC, Eve does not need to utilize structure present in $(Q, \circ)$ to try and generate the quasigroup. However, this structure may give Eve a foothold for forging authentication tags without knowledge of

the key. Therefore, although we have no explicit reason for doing so, it may be wise to avoid quasigroups generated by affine isotopies and by isotopy from abelian groups. It appears that creating a quasigroups from a non-abelian group by isotopy may be a viable option, but more research is needed on this topic.

As mentioned at the beginning of this chapter, quasigroups of large order are required in order to create a secure MAC. If isotopies are used to create such a quasigroup, the only information that needs to be stored are the maps f and g, along with the group $(Q, +)$ that is used to generate the quasigroup $(Q, \circ)$. This is a significant improvement over storing the entire quasigroup and performing multiplication by a table lookup. However, if we could create a quasigroup using only one map instead of two, we would need even less storage space. Using only one map would also improve the time considerations, because the image of $x \in Q$ would only need to be computed under one map instead of under two. Section 4.2 develops this idea.

4.2 Complete Mappings

One way that a quasigroup can be created from a group using only one map uses a special kind of mapping called a complete mapping.

Definition 12. Let $(G, +)$ be a group and let $i : G \to G$ denote the identity map on G. $\theta : G \to G$ is a **complete mapping** if

- θ is a bijection and
- $i - \theta$ is a bijection, where $(i - \theta)(x) = x - \theta(x)$.

Example 11. Let $(G, +) = (\mathbb{Z}_9, +)$, where $\mathbb{Z}_9 = \{0, 1, \ldots, 8\}$ and addition is performed modulo 9. Then $\theta(x) = 5x + 3$ is a complete mapping as seen in Figure 4.11, because both θ and $i - \theta$ are bijections.

Before using complete maps to generate quasigroups, it would be helpful to know whether or not a group G has a complete map. This is a subject that has been extensively studied. A few key results are presented in Section 4.2.1.

x	0	1	2	3	4	5	6	7	8
$\theta(x)$	3	8	4	0	5	1	6	2	7
$x - \theta(x)$	6	2	7	3	8	4	0	5	1

Figure 4.11 A complete mapping on $\mathbb{Z}_9$

4.2.1 Admissible Groups

Definition 13. Let G be a group. G is **admissible** if there is a complete map $\theta : G \to G$.

As mentioned previously, QMAC requires using a large quasigroup. In addition, because much of cryptography deals with binary strings, it would be convenient to create a quasigroup from a group G of order $2m$. Unfortunately, many of these groups are inadmissible. We first prove in Proposition 1 that if a group G has a subgroup of odd order and index 2, then G is not admissible.

Proposition 1. *Let G be a group with $|G| = 2m$, where m is odd. If G has a subgroup of order m, then G is not admissible.*

Proof. Let $N \leq G$ be the subgroup of order m. Consider the cosets N and $G - N$ of G. For ease of notation, call these cosets C_1 and C_2. Notice that $|C_1| = |C_2| = m$. We first show that if both $\theta(x)$ and $(i - \theta)(x)$ are permutations, then exactly half of the elements of G must be mapped back to their original coset by θ. Because $(i - \theta)(x) = x - \theta(x)$ is a permutation, we must have $x - \theta(x) \in C_1$ exactly half the time and $x - \theta(x) \in C_2$ exactly half the time. Because C_1 is a subgroup of G, in order to have $x - \theta(x) \in C_1$, we need x and $\theta(x)$ to be in the same coset. Hence we need θ to map exactly half of the elements of G back to their original coset. That is, we need θ to map x back to its original coset exactly m times.

Now we show that if $n = 2m$, where m is odd, then it is impossible for exactly half the elements of G to be mapped back to their original coset by θ. Define the following quantities:

$$k = \text{number of elements from } C_1 \text{ mapped back into } C_1 \text{ by } \theta$$
$$j = \text{number of elements from } C_2 \text{ mapped back into } C_2 \text{ by } \theta$$

Because exactly half of the elements of G must be mapped back into their original coset, we must have $k + j = m$. In addition, observe that

$$m - k = \text{number of elements from } C_1 \text{ mapped into } C_2 \text{ by } \theta$$
$$m - j = \text{number of elements from } C_2 \text{ mapped into } C_1 \text{ by } \theta$$

However, because k elements have already been mapped into C_1, there must be $m - k$ spots left for elements. But we know that there are $m - j$ elements that are also mapped into C_1, so therefore we must have $m - k = m - j$, or $k = j$. However, we also have $k + j = m$; hence $2k = m$. But m was odd, so this is a contradiction. Therefore, there is no map $\theta : G \to G$ so that both θ and $i - \theta$ are permutations when $|G| = 2m$ where m is odd and G has a subgroup of order m. □

The next proposition is a slight modification of a result presented in (8). Paige proves that if $(G, +)$ is an abelian group with exactly one element of order 2, then there is no mapping $\theta : G \to G$ so that both θ and $i+\theta$ are bijections. In addition, the converse is also proved, which enables us to easily determine whether or not a finite abelian group is admissible. Because our definition of admissible requires θ and $i - \theta$ to be bijections, small changes are made in the proof. The following notation is used:

- G is a finite abelian group where $|G| = n$. G is written multiplicatively.
- $\theta : G \to G$ is a 1-1 mapping.
- $\eta : G \to G$ is defined by $\eta(x) := x\theta(x)^{-1}$.
- The order of η (that is, the number of distinct elements $\eta(x)$, for $x \in G$), is $O(\eta)$.

Notice that G is admissible if and only if η is 1-1.

Proposition 2. *If G has exactly one element of order 2, then G is not admissible, but there is some $\theta : G \to G$ so that $O(\eta) = n - 1$.*
If G does not have exactly one element of order 2, then G is admissible.

Proof. We need several lemmas and corollaries in order to prove Proposition 2.

Lemma 1. *If G has exactly one element of order 2 (call this element x), then $\prod_{g\in G} g = x$. If G does not have exactly one element of order 2, then $\prod_{g\in G} g = 1$.*

Proof. For ease of notation, let $G = \{1, g_1, g_2, \ldots, g_{n-1}\}$. Consider the following cases.

Case 1: G contains no elements of order 2. Then for all $i = 1, \ldots, n-1$, $g_i \neq g_i^{-1}$. That is, each non-identity element in G has a unique inverse element where the element and its inverse are not equal. Because G is abelian, this implies

$$\prod_{g\in G} g = 1 \cdot g_1 \cdot g_2 \cdots g_n = 1 \cdot g_1 \cdot g_1^{-1} \cdots g_i \cdot g_i^{-1} = 1$$

and hence $\prod_{g\in G} g = 1$.

Case 2: G contains exactly one element of order 2. Without loss of generality, assume that this element is g_1. Then $g_1 = g_1^{-1}$ and for all $i = 2, \ldots, n-1$, we have $g_i \neq g_i^{-1}$. This implies

$$\prod_{g\in G} g = g_1 \cdot 1 \cdot g_2 \cdot g_2^{-1} \cdots g_i \cdot g_i^{-1} = g_1 \cdot 1 = g_1$$

and hence $\prod_{g\in G} g = g_1$.

Case 3: G contains more than one element of order 2. Let $B = \{g \in G : |g| = 2\}$. Decompose G into a direct product of subgroups as follows:

$$G \cong A_1 \times A_2 \times \cdots \times A_k \times A,$$

where $A_i = \langle a_i \rangle$ for $i = 1, \ldots, k$, $|a_i| = 2^{m_i}$ for $m_i \in \mathbb{N}$, and $2 \nmid |A|$. This implies that each A_i has a unique element of order 2. Call this element b_i. Because G has more than one element of order 2, we know that $k > 1$.

Let $g = \langle g_1, g_2, \ldots, g_k, h\rangle \in A_1 \times \cdots \times A_k \times A$ have order 2. Then $g_i^2 = 1$ for $i = 1, \ldots, k$ and $h^2 = 1$. Because $2 \nmid |A|$, $2 \nmid |h|$ and so $h = 1$. Because $g_1^2 = 1$, we know that $g_i \in \{b_i, 1\}$ for $i = 1, \ldots, k$. Therefore, $g = \langle b_1^{\alpha_1}, b_2^{\alpha_2}, \ldots, b_k^{\alpha_k}, 1\rangle$, where $\alpha_i \in \{0, 1\}$ for $i = 1, \ldots, k$. Notice that we cannot have $\alpha_1 = \alpha_2 = \cdots = \alpha_k = 0$ because then $g = \langle 1, 1, \ldots, 1\rangle$, which does not have order 2.

In order to evaluate $\prod_{g\in B} g$, we can evaluate the product of all the elements of order 2 from $A_1 \times \cdots \times A_k \times A$. From above, all these elements must have the form $\langle b_1^{\alpha_1}, \ldots, b_k^{\alpha_k}, 1\rangle$

where $\alpha = \langle \alpha_1, \ldots, \alpha_k \rangle \in \{0,1\}^k - \langle 0, \ldots, 0 \rangle$. Therefore, we need to evaluate the product over all possible values of α. That is,

$$\prod_{g \in B} g \cong \prod_{\alpha \in \{0,1\}^k - \langle 0,\ldots,0 \rangle} \langle b_1^{\alpha_1}, \ldots, b_k^{\alpha_k}, 1 \rangle.$$

Notice that b_i will appear in the i^{th} component of this product only when $\alpha_i = 1$. This happens in exactly 2^{k-1} values of α, so b_i will appear in the i^{th} component of the product 2^{k-1} times. Because $k > 1$, b_i will appear in the i^{th} component of the product an even number of times. Since $|b_i| = 2$, the i^{th} component of the product is 1.

Therefore,

$$\prod_{g \in B} g \cong \prod_{\alpha} \langle b_i^{\alpha_1}, \ldots, b_k^{\alpha_k}, 1 \rangle = \langle 1, \ldots, 1, 1 \rangle$$

and so $\prod_{g \in B} g = 1$.

Finally, because each element in $G - B$ has a unique inverse element where the element and its inverse are not equal, we can easily see that $\prod_{g \in G-B} g = 1$. Hence

$$\prod_{g \in G} g = \prod_{g \in B} g \prod_{g \in G-B} g = 1 \cdot 1 = 1.$$

□

Lemma 2. *If $O(\eta) = n$, then* $\prod_{g \in G} g = 1$.

Proof. Assume that there is a 1-1 map $\theta : G \to G$ so that $O(\eta) = n$. Let $G = \{x_1, \ldots, x_n\}$. Then we have the following:

$$\begin{aligned} \prod_{i=1}^{n} \eta(x_i) &= \prod_{i=1}^{n} x_i \theta(x_i)^{-1} \\ &= \prod_{i=1}^{n} x_i \prod_{x=1}^{n} \theta(x_i)^{-1} \end{aligned}$$

Because θ and η are 1-1 maps,

$$\prod_{i=1}^{n} \theta(x_i)^{-1} = \prod_{i=1}^{n} \eta(x_i) = \prod_{i=1}^{n} x_i$$

Let $p = \prod_{i=1}^{n} x_i$. Then

$$p \cdot p = p \Rightarrow p = 1$$

as desired. □

Corollary 1. *If* $\prod_{g \in G} g \neq 1$, *then* $O(\eta) < n$ *for all* $\theta : G \to G$.

Corollary 2. *If* G *has exactly one element of order 2, then* G *is not admissible.*

Proof. By Lemma 2, we know that if G has exactly one element of order 2, then $\prod_{g \in G} g \neq 1$. By Corollary 1, if $\prod_{g \in G} g \neq 1$, then $O(\eta) < n$ for all $\theta : G \to G$. Therefore, there is no 1-1 map θ so that η is also 1-1. Hence G is not admissible. □

Notice that this proves half of the main proposition, so now we just need to show that if G does not have exactly one element of order 2, then G is admissible.

Lemma 3. *If* $O(\eta) \leq n - 2$ *for some 1-1 map* $\theta : G \to G$, *then there is another 1-1 map* $\theta' : G \to G$ *so that* $O(\eta') > O(\eta)$.

Proof. Let θ be a mapping where $O(\eta) = r \leq n - 2$. Let $\{\eta(x_i) | i = 1, \ldots, r\}$ be the r distinct images of η in G.

Case (1): There are $h, k > r$ so that $x_h \theta(x_k)^{-1}$ is not in the range of η. Then define θ' by

$$\theta'(x_h) = \theta(x_k) \tag{4.5}$$

$$\theta'(x_k) = \theta(x_h)$$

$$\theta'(x_i) = \theta(x_i) \text{ for } i \neq h, k \tag{4.6}$$

We need to show that $O(\eta') > O(\eta)$. First, notice that

$$\eta'(x_i) = x_i \theta'(x_i)^{-1} = x_i \theta(x_i)^{-1} = \eta(x_i) \text{ for } i = 1, \ldots, r, \text{ by (4.6).}$$

Therefore, $O(\eta') \geq O(\eta)$. In addition,

$$\eta'(x_h) = x_h \theta'(x_h)^{-1} = x_h \theta(x_k)^{-1} \text{ by (4.5)}$$

which is not in the range of η. Therefore, $O(\eta') > O(\eta)$.

Case (2): For all $h, k > r$, $x_h\theta(x_k)^{-1} = \eta(x_i)$ for some $i \in \{1, \ldots, r\}$. Because $O(\eta) = r$, we must have $\eta(x_{r+1}) = \eta(x_i)$ for some $i \in \{1, \ldots, r\}$. Without loss of generality, assume

$$\eta(x_{r+1}) = \eta(x_1). \tag{4.7}$$

Subcase (a): $x_1\theta(x_{r+2})^{-1}$ is not in the range of η. Then define

$$\theta'(x_1) = \theta(x_{r+2}) \tag{4.8}$$

$$\theta'(x_{r+2}) = \theta(x_1)$$

$$\theta'(x_i) = \theta(x_i) \text{ for } i \neq 1, r+2 \tag{4.9}$$

Then

$$\eta'(x_i) = x_i\theta'(x_i)^{-1} = x_i\theta(x_i)^{-1} = \eta(x_i) \text{ for } i = 2, \ldots, r \text{ by (4.9)}.$$

In addition,

$$\eta'(x_{r+1}) = x_{r+1}\theta'(x_{r+1})^{-1} = x_{r+1}\theta(x_{r+1})^{-1} = \eta(x_{r+1}) = \eta(x_1)$$

by (4.9) and (4.7), and because $\eta(x_1) \neq \eta(x_i)$ for $i \in \{2, \ldots, r\}$, $O(\eta') \geq O(\eta)$. Finally,

$$\eta'(x_1) = x_1\theta'(x_1)^{-1} = x_1\theta(x_{r+2})^{-1} \text{ by (4.8)}$$

which is not in the range of η. Therefore, $O(\eta') > O(\eta)$.

Subcase (b): $x_1\theta(x_{r+2})^{-1}$ is in the range of η. That is, there is some $i \in \{1, \ldots, r\}$ so that $x_1\theta(x_{r+2})^{-1} = \eta(x_i)$. Because θ^{-1} is a permutation,

$$x_1\theta(x_{r+2})^{-1} \neq x_1\theta(x_1)^{-1} = \eta(x_1).$$

Therefore, without loss of generality, assume

$$x_1\theta(x_{r+2})^{-1} = \eta(x_2). \tag{4.10}$$

Subcase (b_i): $x_2\theta(x_1)^{-1}$ is not in the range of η. Then define

$$\theta'(x_1) = \theta(x_{r+2}) \tag{4.11}$$

$$\theta'(x_2) = \theta(x_1) \tag{4.12}$$

$$\theta'(x_{r+2}) = \theta(x_2)$$

$$\theta'(x_i) = \theta(x_i) \text{ for } i \neq 1, 2, r+2 \tag{4.13}$$

Then

$$\eta'(x_i) = x_i\theta'(x_i)^{-1} = x_i\theta(x_i)^{-1} = \eta(x_i) \text{ for } i = 3, \ldots, r \text{ by (4.13).}$$

In addition,

$$\eta'(x_{r+1}) = \eta(x_{r+1}) = x_{r+1}\theta'(x_{r+1})^{-1} = x_{r+1}\theta(x_{r+1})^{-1} = \eta(x_1) \text{ by (4.13) and (4.7)}$$

and

$$\eta'(x_1) = x_1\theta'(x_1)^{-1} = x_1\theta(x_{r+2})^{-1} = \eta(x_2) \text{ by (4.11) and (4.10).}$$

Therefore, $O(\eta') \geq O(\eta)$. Finally,

$$\eta'(x_2) = x_2\theta'(x_2)^{-1} = x_2\theta(x_1)^{-1} \text{ by (4.12)}$$

which is not in the range of η. Therefore, $O(\eta') > O(\eta)$.

Subcase (b_{ii}): $x_2\theta(x_1)^{-1} = \eta(x_i)$ for some $i \in \{1, \ldots, r\}$. Notice that $x_2\theta(x_1)^{-1} \neq \eta(x_1)$ and $x_2\theta(x_1)^{-1} \neq \eta(x_2)$, so assume without loss of generality that

$$x_2\theta(x_1)^{-1} = \eta(x_3).$$

Continue considering cases. If $x_3\theta(x_2)^{-1}$ is not in the range of η, then we can show that $O(\eta') > O(\eta)$ by an argument similar to the one in case b_i. If $x_3\theta(x_2)^{-1}$ is in the range of η, we can show that $x_3\theta(x_2)^{-1} \neq \eta(x_i)$ for $i = 1, 2, 3$. Therefore, without loss of generality assume that $x_3\theta(x_2)^{-1} = \eta(x_4)$. Continue in the same way until we have

$$\eta(x_{r+1}) = \eta(x_1)$$

$$x_1\theta(x_{r+2})^{-1} = \eta(x_2) \tag{4.14}$$

$$x_{i+1}\theta(x_i)^{-1} = \eta(x_{i+2}) \text{ for } i = 1, \ldots, k, \text{ where } k+2 \leq r. \tag{4.15}$$

Claim. We now want to show that

$$\eta(x_1)\theta(x_{r+2})^{-1} = \eta(x_{i+1})\theta(x_i)^{-1}$$

for $i = 1, \ldots, k+1$, where $k+2 \leq r$.

Proof. This will be shown by induction on i.

Base Case: $i = 1$.

$$\begin{aligned}
\eta(x_1)\theta(x_{r+2})^{-1} &= x_1\theta(x_1)^{-1}\theta(x_{r+2})^{-1} \\
&= \theta(x_1)^{-1}x_1\theta(x_{r+2})^{-1} \\
&= \theta(x_1)^{-1}\eta(x_2) \text{ by (4.14)} \\
&= \eta(x_{1+1})\theta(x_1)^{-1}.
\end{aligned}$$

Induction Step: Assume

$$\eta(x_1)\theta(x_{r+2})^{-1} = \eta(x_{i+1})\theta(x_i)^{-1} \tag{4.16}$$

for all $i \leq j$, for some $j < k+1$ and show that

$$\eta(x_1)\theta(x_{r+2})^{-1} = \eta(x_{i+2})\theta(x_{i+1})^{-1}.$$

Recall that we already know

$$x_{i+1}\theta(x_i)^{-1} = \eta(x_{i+2}) \text{ for } i = 1, \ldots, k \text{ by (4.15)}$$

so in particular,

$$x_{j+1}\theta(x_i)^{-1} = \eta(x_{i+2}). \tag{4.17}$$

Therefore,

$$\begin{aligned}
\eta(x_1)\theta(x_{r+2})^{-1} &= \eta(x_{i+1})\theta(x_i)^{-1} \text{ by (4.16)} \\
&= x_{i+1}\theta(x_{i+1})^{-1}\theta(x_i)^{-1} \\
&= \theta(x_{i+1})^{-1}\eta(x_{i+2}) \text{ by (4.17)}
\end{aligned}$$

and so we have

$$\eta(x_1)\theta(x_{r+2})^{-1} = \eta(x_{i+1})\theta(x_i)^{-1} \tag{4.18}$$

for $i = 1, \ldots, k+1$, where $k+2 \leq r$. □

Next, we need to show that $x_{k+2}\theta(x_{k+1})^{-1} \neq \eta(x_i)$ for $i = 1, \ldots, k+2$. Assume not. Then there is some $i \in \{1, \ldots, k+2\}$ so that

$$x_{k+2}\theta(x_{k+1})^{-1} = \eta(x_i). \tag{4.19}$$

This implies that for this value of i,

$$\begin{aligned}
\eta(x_i)\theta(x_{k+2})^{-1} &= x_{k+2}\theta(x_{k+1})^{-1}\theta(x_{k+2})^{-1} \text{ by (4.19)} \\
&= \theta(x_{k+1})^{-1}\eta(x_{k+2}) \\
&= \theta(x_{i-1})^{-1}\eta(x_i) \text{ by (4.18).}
\end{aligned}$$

This implies that $\theta(x_{k+2})^{-1} = \theta(x_{i-1})^{-1}$. But θ^{-1} is a permutation, so this is a contradiction because $i \leq k+1$, so $i - 1 \neq k+2$. Therefore,

$$x_{k+2}\theta(x_{k+1})^{-1} \neq \eta(x_i) \text{ for } i = 1, \ldots, k+2.$$

Consider the following subcases:

Subcase(a): $x_{k+2}\theta(x_{k+1})^{-1}$ is not in the range of η. Then define

$$\theta'(x_1) = \theta(x_{r+2}) \tag{4.20}$$

$$\theta'(x_{i+1}) = \theta(x_i) \text{ for } i = 1, \ldots, k+1 \tag{4.21}$$

$$\theta'(x_{r+2}) = \theta(x_{k+2})$$

$$\theta'(x_i) = \theta(x_i) \text{ for } i = k+3, \ldots, r, r+1, r+3, \ldots, n. \tag{4.22}$$

Then we have the following:

$\eta'(x_i) = x_i\theta'(x_i)^{-1} = x_i\theta(x_i)^{-1} = \eta(x_i)$ for $i = k+3, \ldots, r$ by (4.22)

$\eta'(x_{i+1}) = x_{i+1}\theta'(x_{i+1})^{-1} = x_{i+1}\theta(x_i)^{-1} = \eta(x_{i+2})$ for $i = 1, \ldots, k$ by (4.21) and (4.15)

$\eta'(x_1) = x_1\theta'(x_1)^{-1} = x_1\theta(x_{r+2})^{-1} = \eta(x_2)$ by (4.20) and (4.14)

$\eta'(x_{r+1}) = x_{r+1}\theta'(x_{r+1})^{-1} = x_{r+1}\theta(x_{r+1})^{-1} = \eta(x_{r+1}) = \eta(x_1)$ by (4.22) and (4.7).

Therefore, $O(\eta') \geq O(\eta)$. Finally,

$$\eta'(x_{k+2}) = x_{k+2}\theta'(x_{k+2})^{-1} = x_{k+2}\theta(x_{k+1})^{-1} \text{ by (4.21)}$$

which is not in the range of η. Therefore, $O(\eta') > O(\eta)$.

Subcase (b): $x_{k+2}\theta(x_{k+1})^{-1} = \eta(x_i)$ for some $i \in \{1, \ldots, r\}$. We already know that $x_{k+2}\theta(x_{k+1})^{-1} \neq \eta(x_i)$ for $i = 1, \ldots, k+2$. Without loss of generality, assume $x_{k+2}\theta(x_{k+1})^{-1} = \eta(x_{k+3})$.

Continue in the same fashion, breaking down into subcases. Eventually, because r is finite, we will reach a point where $x_j\theta(x_{j-1})^{-1}$ is not in the range of η and there is no "alternative" subcase. This will imply that $O(\eta') > O(\eta)$, as desired. □

Finally, we are ready to prove the main proposition. We already know that if G has exactly one element of order 2, then G is not admissible by Corollary 2. Therefore, all that is left to show is that if G does not have exactly one element of order 2, then G is admissible.

By Lemma 3, possibly applied multiple times, we know that there is some $\theta : G \to G$ so that $O(\eta) \geq n - 1$. We need to show that $O(\eta) = n$.

Let $G = \{x_1, \ldots, x_n\}$ and let $\{\eta(x_i) : i = 1, \ldots, n-1\}$ be the range elements of η which we know are distinct. Let z be the "leftover" element of G. If we can show that $\eta(x_n) = z$, then we will have shown that η is 1-1 and hence that G is admissible. Consider the following:

$$\prod_{i=1}^{n-1} x_i\theta(x_i)^{-1} = \prod_{i=1}^{n-1} \eta(x_i)$$

$$x_n^{-1}\theta(x_n)\prod_{i=1}^{n} x_i\theta(x_i)^{-1} = z^{-1}z\prod_{i=1}^{n-1} \eta(x_i)$$

$$x_n^{-1}\theta(x_n)\prod_{i=1}^{n} x_i\prod_{i=1}^{n} \theta(x_i)^{-1} = z^{-1}\left(\prod_{i=1}^{n-1} \eta(x_i)\right)z$$

Notice that $\prod_{i=1}^{n} x_i = \prod_{i=1}^{n} \theta(x_i)^{-1} = \left(\prod_{i=1}^{n-1} \eta(x_i)\right)z$. Call this product p. Because G does not

have exactly one element of order 2, $p = 1$. Therefore, we have

$$x_n^{-1}\theta(x_n)p^2 = z^{-1}p$$

$$x_n^{-1}\theta(x_n) = z^{-1}$$

$$x_n\theta(x_n)^{-1} = z$$

$$\eta(x_n) = z$$

Hence η is a permutation, and G is admissible. □

Because $\mathbb{Z}_{2m}$ is abelian and has exactly one element of order 2 for any integer m, Proposition 2 tells us that $\mathbb{Z}_{2m}$ is not admissible. However, any finite abelian group of odd order is admissible because it has no elements of order 2 (since the order of an element must divide the order of the group). Also, if $|G| > 2$ and G has exponent 2, then G is admissible because every non-identity element in G has order 2. Therefore, the group $(\mathbb{Z}_2^k, \oplus)$, where

$$\mathbb{Z}_2^k = \{\langle x_{k-1}, x_{k-2}, \ldots, x_1, x_0\rangle : x_i \in \mathbb{Z}_2 \text{ for } i = 0, \ldots, k-1\}$$

and $\oplus$ denotes exclusive-or, is admissible when $k > 1$.

Recall that the quasigroups we create will be used for cryptographic purposes. This implies that the message will need to be written as a sequence of elements from the quasigroup. Because messages are often written as binary strings, it is most convenient for our purposes to create a quasigroup from the group $(\mathbb{Z}_2^k, \oplus)$. In addition, there are many complete mappings on $(\mathbb{Z}_2^k, \oplus)$.

Now that we have a method of determining whether a group is admissible, we need to look at how to create a quasigroup from an admissible group and a complete mapping.

4.2.2 Creating Quasigroups Using Complete Maps

In (10), Sade suggests creating a quasigroup $(Q, \circ)$ from an admissible group $(Q, +)$ and a complete mapping θ by defining

$$x \circ y = \theta(x - y) + y$$

for $x, y \in Q$.

Claim. If $x \circ y$ is defined as above, then $(Q, \circ)$ is a quasigroup.

Proof. Clearly, Q is closed under $\circ$. Therefore, we just need to show that $(Q, \circ)$ has unique solubility of equations. This can be done by showing that $L_a(x) = a \circ x$ and $R_a(x) = x \circ a$ are permutations, for all $a \in Q$.

First, show that $L_a(x)$ is one-to-one for some arbitrary $a \in Q$. Assume that $L_a(x) = L_a(y)$ and show that $x = y$. Notice that because $(i - \theta)(x)$ is a permutation, then

$$x - \theta(x) = y - \theta(y) \quad \Rightarrow \quad x = y \tag{4.23}$$

Therefore, we have

$$\begin{aligned}
L_a(x) &= L_a(y) \\
a \circ x &= a \circ y \\
\theta(a - x) + x &= \theta(a - y) + y \\
\theta(a - x) + x - a &= \theta(a - y) + y - a \\
(a - x) - \theta(a - x) &= (a - y) - \theta(a - y) \quad \text{by (4.23)} \\
a - x &= a - y \\
x &= y
\end{aligned}$$

Next, show that $L_a(x)$ is onto. Let $y \in Q$. We need to find $x \in Q$ so that $L_a(x) = y$. Because $i - \theta$ is a permutation and because G is closed under subtraction, there is some $n \in Q$ so that $n - \theta(n) = a - y$. Notice that this implies $\theta(n) = y - a + n$. Choose $x = -n + a$. Then

$$\begin{aligned}
L_a(x) &= a \circ x \\
&= \theta(a - x) + x \\
&= \theta(a - (-n + a)) - n + a \\
&= \theta(n) - n + a \\
&= y - a + n - n + a \\
&= y
\end{aligned}$$

Hence $L_a(x)$ is a permutation on $(Q, \circ)$. Now we need to show that $R_a(x)$ is also a permutation on Q.

Again, first show that $R_a(x)$ is one-to-one. Assume that $R_a(x) = R_a(y)$ and show that $x = y$.

$$R_a(x) = R_a(y)$$
$$x \circ a = y \circ a$$
$$\theta(x - a) + a = \theta(y - a) + a$$
$$\theta(x - a) = \theta(y - a)$$
$$x - a = y - a$$
$$x = y$$

Finally, show that $R_a(x)$ is onto. Let $y \in Q$. We need to find $x \in Q$ so that $R_a(x) = y$. Because θ is a permutation and because G is closed under subtraction, there is some $n \in Q$ so that $\theta(n) = y - a$. Choose $x = n + a$. Then

$$\begin{aligned} R_a(x) &= R_a(n + a) \\ &= (n + a) \circ a \\ &= \theta(n + a - a) + a \\ &= \theta(n) + a \\ &= y - a + a \\ &= a \end{aligned}$$

Hence R_a is a permutation and therefore $(Q, \circ)$ is a quasigroup. □

Example 12. Let Q be the additive group of five elements. That is, $Q = \mathbb{Z}_5 = \{0, 1, 2, 3, 4\}$ where $x+y$ is computed modulo 5 for $x, y \in Q$. By Proposition 2, Q is admissible because it is a finite abelian group which does not have exactly one element of order 2 (in particular, it has no elements of order 2). Let $\theta : Q \to Q$ be given by $\theta(x) = 2x$. Then $(i-\theta)(x) = x-2x = -x = 4x$ and so θ is a complete mapping. If we create $(Q, \circ)$ by $x \circ y = \theta(x - y) + y$, then the Cayley table for $(Q, \circ)$ is shown in Figure 4.12. Notice that although $(Q, +)$ is a group, $(Q, \circ)$ is not associative nor commutative and has no identity element.

$Q =$

$\circ$	0	1	2	3	4
0	0	4	3	2	1
1	2	1	0	4	3
2	4	3	2	1	0
3	1	0	4	3	2
4	3	2	1	0	4

Figure 4.12 The Cayley table for $(Q, \circ)$

If we choose $(Q, +) = (\mathbb{Z}_2^k, \oplus)$, then a quasigroup operation is defined by

$$x \circ y = \theta(x \oplus y) \oplus y.$$

Example 13. Let $(Q, +) = (\mathbb{Z}_2^3, \oplus)$ and let

$$\theta(\langle x_2, x_1, x_0 \rangle) = \begin{cases} \langle x_1, x_0 \oplus 1, x_2 \rangle & \text{if } x_2 = 0; \\ \langle x_1 \oplus 1, x_0 \oplus 1, x_2 \rangle & \text{if } x_2 = 1. \end{cases}$$

Then θ and $i \oplus \theta$ are permutations as shown in Figure 4.13. If the elements of $(\mathbb{Z}_2^3, \oplus)$ are represented as integers corresponding to binary strings for ease of notation, then θ and $i - \theta$ are shown in Figure 4.14 and the quasigroup created by $x \circ y = \theta(x \oplus y) \oplus y$ is shown in Figure 4.15. As in Example 12, note that although $(\mathbb{Z}_2^3, \oplus)$ is an abelian group, $(\mathbb{Z}_2^3, \circ)$ is neither commutative nor associative and has no identity element.

x	$\theta(x)$	$(i \oplus \theta)(x)$
$\langle 0,0,0 \rangle$	$\langle 0,1,0 \rangle$	$\langle 0,1,0 \rangle$
$\langle 0,0,1 \rangle$	$\langle 0,0,0 \rangle$	$\langle 0,0,1 \rangle$
$\langle 0,1,0 \rangle$	$\langle 1,1,0 \rangle$	$\langle 1,0,0 \rangle$
$\langle 0,1,1 \rangle$	$\langle 1,0,0 \rangle$	$\langle 1,1,1 \rangle$
$\langle 1,0,0 \rangle$	$\langle 1,1,1 \rangle$	$\langle 0,1,1 \rangle$
$\langle 1,0,1 \rangle$	$\langle 1,0,1 \rangle$	$\langle 0,0,0 \rangle$
$\langle 1,1,0 \rangle$	$\langle 0,1,1 \rangle$	$\langle 1,0,1 \rangle$
$\langle 1,1,1 \rangle$	$\langle 0,0,1 \rangle$	$\langle 1,1,0 \rangle$

Figure 4.13 θ and $i \oplus \theta$ on $\mathbb{Z}_2^3$

x	0	1	2	3	4	5	6	7
$\theta(x)$	2	0	6	4	7	5	3	1
$(i \oplus \theta)(x)$	2	1	4	7	3	0	5	6

Figure 4.14 θ and $i \oplus \theta$ on integers corresponding to elements of $\mathbb{Z}_2^3$

$Q =$

$\circ$	0	1	2	3	4	5	6	7
0	2	1	4	7	3	0	5	6
1	0	3	6	5	1	2	7	4
2	6	5	0	3	7	4	1	2
3	4	7	2	1	5	6	3	0
4	7	4	1	2	6	5	0	3
5	5	6	3	0	4	7	1	2
6	3	0	5	6	2	1	4	7
7	1	2	7	4	0	3	6	5

Figure 4.15 The quasigroup $(Q, \circ)$ created using the complete map θ

4.2.2.1 Affine Complete Mappings

As with isotopies, we could choose our complete mapping to have special properties. If we choose θ to be an affine complete map and create $(Q, \circ)$ by $x \circ y = \theta(x - y) + y$, then $(Q, \circ)$ is again isotopic to $(Q, +)$. That is, creating a quasigroup by using an affine complete map is simply a special case of creating a quasigroup by isotopy. In this case, the isotopy is given by $f(x) = \theta(x)$ and $g(x) = \theta(-x) + \theta(0) + x$, because

$$\begin{aligned} x \circ y &= \theta(x - y) + y \\ &= \theta(x + (-y)) + y \\ &= \theta(x) + \theta(-y) - \theta(0) + y \text{ because } \theta \text{ is affine.} \\ &= f(x) + g(y) \end{aligned}$$

Because both θ and $i - \theta$ are bijections, f and g are bijections, so this is indeed an isotopy. In addition, because θ is an affine map, f and g are also affine maps. This implies that the pairing properties discussed in Section 4.1.1 will be present in $(Q, \circ)$.

If $(Q, \circ)$ is a quasigroup created from a group $(Q, \oplus)$ of exponent 2 by an affine complete mapping θ (so that $x \circ y = \theta(x \oplus y) \oplus y$), then Property 2 can be restated in a simpler form.

Property 5. Let $(Q, \circ)$ be a quasigroup created from the group $(Q, \oplus)$ of exponent 2 by

$$x \circ y = \theta(x \oplus y) \oplus y \text{ for } x, y \in Q$$

where $\theta : Q \to Q$ is an affine complete mapping. Then rows x and y are paired over k (in the sense of Definition 10) if and only if $x \circ k = \theta(y)$. That is, for any $z \in Q$,

$$x \circ z = y \circ (z \oplus k) \Leftrightarrow x \circ k = \theta(y) \tag{4.24}$$

Notice that for $x, k \in Q$, y is guaranteed to exist because of unique solubility of equations in a quasigroup.

Similarly, columns x and y are paired over k (in the sense of Definition 11) if and only if $k \circ x = (i \oplus \theta)(y)$. That is, for any $z \in Q$,

$$z \circ x = (z \oplus k) \circ y \Leftrightarrow k \circ x = y \oplus \theta(y) \tag{4.25}$$

Again, notice that y is guaranteed to exist.

Proof. Only (4.24) will be proved; the proof of (4.25) is similar.

$\Rightarrow$ Assume that $x \circ z = y \circ (z \oplus k)$. Let $0 \in Q$ denote the group identity element. Then we have

$$\begin{aligned}
\theta(x \oplus z) &= \theta(y \oplus (z \oplus k)) \oplus (z \oplus k) \\
\theta(x) \oplus \theta(z) \oplus \theta(0) \oplus z &= \theta(y) \oplus \theta(z \oplus k) \oplus \theta(0) \oplus z \oplus k \\
\theta(x) \oplus \theta(z) &= \theta(y) \oplus \theta(z) \oplus \theta(k) \oplus \theta(0) \oplus k \\
\theta(x) \oplus \theta(k) \oplus \theta(0) \oplus k &= \theta(y) \\
\theta(x \oplus k) \oplus k &= \theta(y) \\
x \circ k &= y
\end{aligned}$$

$\Leftarrow$ Assume that $x \circ k = \theta(y)$. Then we have

$$\begin{aligned}
\theta(x \oplus k) \oplus k &= \theta(y) \\
\theta(x) \oplus \theta(k) \oplus \theta(0) \oplus k &= \theta(y) \\
\theta(x) \oplus \theta(z) \oplus \theta(0) \oplus z &= \theta(y) \oplus \theta(k) \oplus k \oplus \theta(z) \oplus z \\
\theta(x \oplus z) \oplus z &= \theta(y) \oplus \theta(k \oplus z) \oplus \theta(0) \oplus k \oplus z \\
\theta(x \oplus z) \oplus z &= \theta(y \oplus (z \oplus k)) \oplus z \oplus k \\
x \circ z &= y \circ (z \oplus k)
\end{aligned}$$

□

Example 14. Let $Q = \mathbb{Z}_2^4$ and define $\theta : Q \to Q$ by

$$\theta(\langle x_3, x_2, x_1, x_0 \rangle) = \langle x_3 \oplus x_2, x_1, x_0, x_3 \rangle.$$

That is, θ is defined by rotating the bits of $x \in Q$ one bit to the left and exclusive or-ing the "new" first component of x with the "old" first component of x. In order to prove that $(Q, \circ)$ is a quasigroup, we need to show that θ is a complete map. We also need to show that θ is affine in order to prove that the pairing properties of Sections 4.1.1 and 4.1.2 are present in $(Q, \circ)$.

Claim. θ is an affine map.

Proof. In order to show that θ is affine, we need to show that $\theta(x \oplus y) = \theta(x) \oplus \theta(y) \oplus \theta(0)$ for all $x, y \in Q$. Notice that $\theta(0) = 0$. θ is affine because

$$\begin{aligned}
\theta(\langle x_3, x_2, x_1, x_0 \rangle \oplus \langle y_3, y_2, y_1, y_0 \rangle) &= \theta(\langle x_3 \oplus y_3, x_2 \oplus y_2, x_1 \oplus y_1, x_0 \oplus y_0 \rangle) \\
&= \langle x_3 \oplus y_3 \oplus x_2 \oplus y_2, x_1 \oplus y_1, x_0 \oplus y_0, x_3 \oplus y_3 \rangle \\
&= \langle x_3 \oplus x_2, x_1, x_0, x_3 \rangle \oplus \langle y_3 \oplus y_2, y_1, y_0, y_3 \rangle \\
&= \theta(x) \oplus \theta(y) \oplus \theta(0).
\end{aligned}$$

□

To see that θ is a complete map, notice that Figure 4.16 demonstrates that both θ and $i \oplus \theta$ are bijections. This can also be seen in Figure 4.17, which shows how θ and $i \oplus \theta$ act on the integer representations of the elements of $(Q, \circ)$. The quasigroup $(Q, \circ)$ created from θ by $x \circ y = \theta(x \oplus y) \oplus y$ is shown in Figure 4.18. Again, notice that all the pairing properties discussed in Sections 4.1.1 and 4.1.2 are present in $(Q, \circ)$.

x	$\theta(x)$	$x \oplus \theta(x)$
$\langle 0,0,0,0 \rangle$	$\langle 0,0,0,0 \rangle$	$\langle 0,0,0,0 \rangle$
$\langle 0,0,0,1 \rangle$	$\langle 0,0,1,0 \rangle$	$\langle 0,0,1,1 \rangle$
$\langle 0,0,1,0 \rangle$	$\langle 0,1,0,0 \rangle$	$\langle 0,1,1,0 \rangle$
$\langle 0,0,1,1 \rangle$	$\langle 0,1,1,0 \rangle$	$\langle 0,1,0,1 \rangle$
$\langle 0,1,0,0 \rangle$	$\langle 1,0,0,0 \rangle$	$\langle 1,1,1,0 \rangle$
$\langle 0,1,0,1 \rangle$	$\langle 1,0,1,0 \rangle$	$\langle 1,1,1,1 \rangle$
$\langle 0,1,1,0 \rangle$	$\langle 1,1,1,0 \rangle$	$\langle 1,0,1,0 \rangle$
$\langle 0,1,1,1 \rangle$	$\langle 1,1,1,0 \rangle$	$\langle 1,0,0,1 \rangle$
$\langle 1,0,0,0 \rangle$	$\langle 1,0,0,1 \rangle$	$\langle 0,0,0,1 \rangle$
$\langle 1,0,0,1 \rangle$	$\langle 1,0,1,1 \rangle$	$\langle 0,0,1,0 \rangle$
$\langle 1,0,1,0 \rangle$	$\langle 1,1,0,1 \rangle$	$\langle 0,1,1,1 \rangle$
$\langle 1,0,1,1 \rangle$	$\langle 1,1,1,1 \rangle$	$\langle 0,1,0,0 \rangle$
$\langle 1,1,1,0 \rangle$	$\langle 0,0,0,1 \rangle$	$\langle 1,1,0,1 \rangle$
$\langle 1,1,0,1 \rangle$	$\langle 0,0,1,1 \rangle$	$\langle 1,1,1,0 \rangle$
$\langle 1,1,1,0 \rangle$	$\langle 0,1,0,1 \rangle$	$\langle 1,0,1,1 \rangle$
$\langle 1,1,1,1 \rangle$	$\langle 0,1,1,1 \rangle$	$\langle 1,0,0,0 \rangle$

Figure 4.16 θ and $i \oplus \theta$ on $\mathbb{Z}_2^4$

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\theta(x)$	0	2	4	6	8	10	12	14	9	11	13	15	1	3	5	7
$x \oplus \theta(x)$	0	3	6	5	12	15	10	9	1	2	7	4	13	14	11	8

Figure 4.17 θ and $i \oplus \theta$ on integers corresponding to elements of $\mathbb{Z}_2^4$

Because QMAC requires that we work with large quasigroups, Example 14 is interesting, but not particularly useful. However, this method could be useful if it could be extended to define $\theta : \mathbb{Z}_2^k \to \mathbb{Z}_2^k$. It turns out that we can indeed extend this method by defining $\theta : \mathbb{Z}_2^k \to \mathbb{Z}_2^k$ by

$$\theta(\langle x_{k-1}, x_{k-2}, \ldots, x_1, x_0 \rangle) = \langle x_{k-1} \oplus x_{k-2}, x_{k-3}, \ldots, x_0, x_{k-1} \rangle.$$

$Q =$

$\circ$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	3	6	5	12	15	10	9	1	2	7	4	13	14	11	8
1	2	1	4	7	14	13	8	11	3	0	5	6	15	12	9	10
2	4	7	2	1	8	11	14	13	5	6	3	0	9	10	15	12
3	6	5	0	3	10	9	12	15	7	4	1	2	11	8	13	14
4	8	11	14	13	4	7	2	1	9	10	15	12	5	6	3	0
5	10	9	12	15	6	5	0	3	11	8	13	14	7	4	1	2
6	12	15	10	9	0	3	6	5	13	14	11	8	1	2	7	4
7	14	13	8	11	2	1	4	7	15	12	9	10	3	0	5	6
8	9	10	15	12	5	6	3	0	8	11	14	13	4	7	2	1
9	11	8	13	14	7	4	1	2	10	9	12	15	6	5	0	3
10	13	14	11	8	1	2	7	4	12	15	10	9	0	3	6	5
11	15	12	9	10	3	0	5	6	14	13	8	11	2	1	4	7
12	1	2	7	4	13	14	11	8	0	3	6	5	12	15	10	9
13	3	0	5	6	15	12	9	10	2	1	4	7	14	13	8	11
14	5	6	3	0	9	10	15	12	4	7	2	1	8	11	14	13
15	7	4	1	2	11	8	13	14	6	5	0	3	10	9	12	15

Figure 4.18 The quasigroup $(Q, \circ)$ created using the map θ.

The proof in Example 14 can be easily modified to show that $\theta : \mathbb{Z}_2^k \to \mathbb{Z}_2^k$ is also affine. However, the "proof" in Example 14 that θ was a complete map relied on an exhaustive listing of the images of θ and $i \oplus \theta$ on $\mathbb{Z}_2^4$. Clearly, this is not possible on $\mathbb{Z}_2^k$. Therefore, we need to show that θ is a complete map in another way.

Claim. $\theta : \mathbb{Z}_2^k \to \mathbb{Z}_2^k$ is a complete map (that is, both θ and $i \oplus \theta$ are permutations).

Proof. Notice that because we are working in a finite group, we only need to show that both functions are one-to-one.

To show that θ is one-to-one, assume that $\theta(x) = \theta(y)$ and show that $x = y$. Because $\theta(x) = \theta(y)$, we have

$$\langle x_{k-1} \oplus x_{k-2}, x_{k-3}, \ldots, x_0, x_{k-1} \rangle = \langle y_{k-1} \oplus y_{k-2}, y_{k-3}, \ldots, y_0, y_{k-1} \rangle.$$

This shows that $x_i = y_i$ for $i = 0, \ldots, k-3, k-1$. In addition, because $x_{k-1} \oplus x_{k-2} = y_{k-1} \oplus y_{k-2}$ and $x_{k-1} = y_{k-1}$, we must also have $x_{k-2} = y_{k-2}$. Therefore, $x = y$ and so θ is a bijection.

To show that $i \oplus \theta$ is one-to-one, assume that $x \oplus \theta(x) = y \oplus \theta(y)$ and show that $x = y$. Notice that

$$x \oplus \theta(x) = \langle x_{k-1}, x_{k-2}, \ldots, x_1, x_0 \rangle \oplus \langle x_{k-1} \oplus x_{k-2}, x_{k-3}, \ldots, x_0, x_{k-1} \rangle$$

and

$$y \oplus \theta(y) = \langle y_{k-1}, y_{k-2}, \ldots, y_1, y_0 \rangle \oplus \langle y_{k-1} \oplus y_{k-2}, y_{k-3}, \ldots, y_0, y_{k-1} \rangle.$$

Therefore,

$$\langle x_{k-2}, x_{k-2} \oplus x_{k-3}, \ldots, x_1 \oplus x_0, x_0 \oplus x_{k-1} \rangle = \langle y_{k-2}, y_{k-2} \oplus y_{k-3}, \ldots, y_1 \oplus y_0, y_0 \oplus y_{k-1} \rangle.$$

By looking at the most significant bit, we can see that $x_{k-2} = y_{k-2}$, and by equating the rest of the bits, we can clearly see that $x_i = y_i$ for $i = 0, \ldots, k-1$. Therefore, $x = y$ and so $i \oplus \theta$ is also a bijection. This implies that θ is a complete map, as desired. □

Notice that the quasigroup created in Example 14 is idempotent. That is, for all $x \in Q$, $x \circ x = x$. This is also true for the extended mapping θ on $\mathbb{Z}_2^k$ because

$$\begin{aligned} x \circ x &= \theta(x \oplus x) \oplus x \\ &= \theta(0) \oplus x \\ &= 0 \oplus x \\ &= x \end{aligned}$$

It seems as though this property may compromise the security of a keyed hash function, so we would like to change θ so that the quasigroup created is not idempotent. It is actually quite easy to define another mapping $\psi : Q \to Q$ based on θ so that ψ is a complete affine map but the quasigroup created is not idempotent. We define ψ by

$$\psi(\langle x_{k-1}, x_{k-2}, \ldots, x_1, x_0 \rangle) = \langle x_{k-1} \oplus x_{k-2}, x_{k-3}, \ldots, x_0, x_{k-1} \oplus 1 \rangle.$$

It can be shown that ψ is an affine complete map in the same way that these properties were shown for θ. If $\psi : \mathbb{Z}_2^4 \to \mathbb{Z}_2^4$, then the action of ψ on the group elements is shown in Figure 4.19 and on the binary representations of the group elements in Figure 4.20. The quasigroup $(Q, \star)$ created is shown in Figure 4.21.

x	$\psi(x)$	$x \oplus \psi(x)$
$\langle 0,0,0,0\rangle$	$\langle 0,0,0,1\rangle$	$\langle 0,0,0,1\rangle$
$\langle 0,0,0,1\rangle$	$\langle 0,0,1,1\rangle$	$\langle 0,0,1,0\rangle$
$\langle 0,0,1,0\rangle$	$\langle 0,1,0,1\rangle$	$\langle 0,1,1,1\rangle$
$\langle 0,0,1,1\rangle$	$\langle 0,1,1,1\rangle$	$\langle 0,1,0,0\rangle$
$\langle 0,1,0,0\rangle$	$\langle 1,0,0,1\rangle$	$\langle 1,1,0,1\rangle$
$\langle 0,1,0,1\rangle$	$\langle 1,0,1,1\rangle$	$\langle 1,1,1,0\rangle$
$\langle 0,1,1,0\rangle$	$\langle 1,1,0,1\rangle$	$\langle 1,0,1,1\rangle$
$\langle 0,1,1,1\rangle$	$\langle 1,1,1,1\rangle$	$\langle 1,0,0,0\rangle$
$\langle 1,0,0,0\rangle$	$\langle 1,0,0,0\rangle$	$\langle 0,0,0,0\rangle$
$\langle 1,0,0,1\rangle$	$\langle 1,0,1,0\rangle$	$\langle 0,0,1,1\rangle$
$\langle 1,0,1,0\rangle$	$\langle 1,1,1,0\rangle$	$\langle 0,1,1,0\rangle$
$\langle 1,0,1,1\rangle$	$\langle 1,1,1,0\rangle$	$\langle 0,1,0,1\rangle$
$\langle 1,1,1,0\rangle$	$\langle 0,0,0,0\rangle$	$\langle 1,1,1,0\rangle$
$\langle 1,1,0,1\rangle$	$\langle 0,0,1,0\rangle$	$\langle 1,1,1,1\rangle$
$\langle 1,1,1,0\rangle$	$\langle 0,1,0,0\rangle$	$\langle 1,0,1,0\rangle$
$\langle 1,1,1,1\rangle$	$\langle 0,1,1,0\rangle$	$\langle 1,0,0,1\rangle$

Figure 4.19 ψ and $i \oplus \psi$ on $\mathbb{Z}_2^4$

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\psi(x)$	1	3	5	7	9	11	13	15	8	10	12	14	0	2	4	6
$x \oplus \psi(x)$	1	2	7	4	13	14	11	8	0	3	6	5	12	15	10	9

Figure 4.20 ψ and $i \oplus \psi$ on integers corresponding to elements of $\mathbb{Z}_2^4$

$Q =$

$\star$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	2	7	4	13	14	11	8	0	3	6	5	12	15	10	9
1	3	0	5	6	15	12	9	10	2	1	4	7	14	13	8	11
2	5	6	3	0	9	10	15	12	4	7	2	1	8	11	14	13
3	7	4	1	2	11	8	13	14	6	5	0	3	10	9	12	15
4	9	10	15	12	5	6	3	0	8	11	14	13	4	7	2	1
5	11	8	13	14	7	4	1	2	10	9	12	15	6	5	0	3
6	13	14	11	8	1	2	7	4	12	15	10	9	0	3	6	5
7	15	12	9	10	3	0	5	6	14	13	8	11	2	1	4	7
8	8	11	14	13	4	7	2	1	9	10	15	12	5	6	3	0
9	10	9	12	15	6	5	0	3	11	8	13	14	7	4	1	2
10	12	15	10	9	0	3	6	5	13	14	11	8	1	2	7	4
11	14	13	8	11	2	1	4	7	15	12	9	10	3	0	5	6
12	0	3	6	5	12	15	10	9	1	2	7	4	13	14	11	8
13	2	1	4	7	14	13	8	11	3	0	5	6	15	12	9	10
14	4	7	2	1	8	11	14	13	5	6	3	0	9	10	15	12
15	6	5	0	3	10	9	12	15	7	4	1	2	11	8	13	14

Figure 4.21 The quasigroup $(Q, \star)$ created using an affine complete map.

4.2.2.2 Non-affine Complete Mappings

There are also complete mappings θ which are not affine. Because the map is not affine, the pairing properties discussed in Sections 4.1.1 and 4.1.2 may not be present. This creates a quasigroup which is less structured. Using quasigroups with less structure make QMAC more secure. In this section, we will look at two examples of quasigroups created with non-affine complete mappings. Notice that the quasigroup in Example 15 still has the element pairing property described in Property 1 (with $k = 1$), but not Properties 2 or 4. The quasigroup in Example 16 has none of these properties.

Example 15. Let $\theta : \mathbb{Z}_2^4 \to \mathbb{Z}_2^4$ be as shown in Figure 4.22. Then the action of θ on the binary representations of the group elements is shown in Figure 4.23 and the quasigroup $(Q, \circ)$ created is shown in Figure 4.24, where $x \circ y = \theta(x \oplus y) \oplus y$.

x	$\theta(x)$	$x \oplus \theta(x)$
$\langle 0,0,0,0\rangle$	$\langle 1,1,1,0\rangle$	$\langle 1,1,1,0\rangle$
$\langle 0,0,0,1\rangle$	$\langle 1,0,1,0\rangle$	$\langle 1,0,1,1\rangle$
$\langle 0,0,1,0\rangle$	$\langle 1,0,0,0\rangle$	$\langle 1,0,1,0\rangle$
$\langle 0,0,1,1\rangle$	$\langle 1,1,1,0\rangle$	$\langle 1,1,1,1\rangle$
$\langle 0,1,0,0\rangle$	$\langle 0,1,0,0\rangle$	$\langle 0,0,0,0\rangle$
$\langle 0,1,0,1\rangle$	$\langle 0,0,0,0\rangle$	$\langle 0,1,0,1\rangle$
$\langle 0,1,1,0\rangle$	$\langle 0,0,0,1\rangle$	$\langle 0,1,1,1\rangle$
$\langle 0,1,1,1\rangle$	$\langle 0,1,0,1\rangle$	$\langle 0,0,1,0\rangle$
$\langle 1,0,0,0\rangle$	$\langle 1,0,1,1\rangle$	$\langle 0,0,1,1\rangle$
$\langle 1,0,0,1\rangle$	$\langle 1,1,1,1\rangle$	$\langle 0,1,1,0\rangle$
$\langle 1,0,1,0\rangle$	$\langle 0,0,1,1\rangle$	$\langle 1,0,0,1\rangle$
$\langle 1,0,1,1\rangle$	$\langle 0,1,1,1\rangle$	$\langle 1,1,1,0\rangle$
$\langle 1,1,1,0\rangle$	$\langle 1,1,0,1\rangle$	$\langle 0,0,0,1\rangle$
$\langle 1,1,0,1\rangle$	$\langle 1,0,0,1\rangle$	$\langle 0,1,0,0\rangle$
$\langle 1,1,1,0\rangle$	$\langle 0,1,1,0\rangle$	$\langle 1,0,0,0\rangle$
$\langle 1,1,1,1\rangle$	$\langle 0,0,1,0\rangle$	$\langle 1,1,0,1\rangle$

Figure 4.22 A non-affine complete map θ and $i \oplus \theta$ on $\mathbb{Z}_2^4$

First, notice that θ is not affine, because

$$\theta(2 \oplus 4) = \theta(6) = 1 \text{ but } \theta(2) \oplus \theta(4) \oplus \theta(0) = 8 \oplus 4 \oplus 14 = 2.$$

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\theta(x)$	14	10	8	12	4	0	1	5	11	15	3	7	13	9	6	2
$x \oplus \theta(x)$	14	11	10	15	0	5	7	2	3	6	9	12	1	4	8	13

Figure 4.23 θ and $i \oplus \theta$ on integers corresponding to elements of $\mathbb{Z}_2^4$

$Q =$

$\circ$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	11	10	15	0	5	7	2	3	6	9	12	1	4	8	13
1	10	15	14	11	4	1	3	6	7	2	13	8	5	0	12	9
2	8	13	12	9	5	0	2	7	11	14	1	4	10	15	3	6
3	12	9	8	13	1	4	6	3	15	10	5	0	14	11	7	2
4	4	1	3	6	10	15	14	11	5	0	12	9	7	2	13	8
5	0	5	7	2	14	11	10	15	1	4	8	13	3	6	9	12
6	1	4	6	3	12	9	8	13	14	11	7	2	15	10	5	0
7	5	0	2	7	8	13	12	9	10	15	3	6	11	14	1	4
8	11	14	1	4	9	12	0	5	6	3	2	7	8	13	15	10
9	15	10	5	0	13	8	4	1	2	7	6	3	12	9	11	14
10	3	6	9	12	2	7	11	14	0	5	4	1	13	8	10	15
11	7	2	13	8	6	3	15	10	4	1	0	5	9	12	14	11
12	13	8	4	1	15	10	5	0	12	9	11	14	2	7	6	3
13	9	12	0	5	11	14	1	4	8	13	15	10	6	3	2	7
14	6	3	15	10	7	2	13	8	9	12	14	11	4	1	0	5
15	2	7	11	14	3	6	9	12	13	8	10	15	0	5	4	1

Figure 4.24 The quasigroup $(Q, \circ)$ created using the non-affine complete map θ.

Therefore, Property 2 does not hold because the proof of this property required that θ was affine. In addition, Property 4 does not hold because

$$5 \circ 7 = 10 \circ 15 \text{ but } 5 \circ 15 \neq 10 \circ 7.$$

However, θ does have the property that

$$\theta(x \oplus 1) = \theta(x) \oplus \theta(1) \oplus \theta(0)$$

for all $x \in \mathbb{Z}_2^4$. This is the only condition that we needed to prove Property 1, with $k = 1$, so the element pairing property still holds even though θ is not affine.

Example 16. Let $\theta : \mathbb{Z}_2^4 \to \mathbb{Z}_2^4$ be as shown in Figure 4.25. Then the action of θ on the binary representations of the group elements is shown in Figure 4.26 and the quasigroup $(Q, \circ)$ created is shown in Figure 4.27, where $x \circ y = \theta(x \oplus y) \oplus y$.

x	$\theta(x)$	$x \oplus \theta(x)$
$\langle 0,0,0,0\rangle$	$\langle 1,1,1,0\rangle$	$\langle 1,1,1,0\rangle$
$\langle 0,0,0,1\rangle$	$\langle 0,1,1,0\rangle$	$\langle 0,1,1,1\rangle$
$\langle 0,0,1,0\rangle$	$\langle 0,0,1,1\rangle$	$\langle 0,0,0,1\rangle$
$\langle 0,0,1,1\rangle$	$\langle 1,1,1,0\rangle$	$\langle 1,1,0,1\rangle$
$\langle 0,1,0,0\rangle$	$\langle 0,0,1,0\rangle$	$\langle 0,1,1,0\rangle$
$\langle 0,1,0,1\rangle$	$\langle 1,1,0,1\rangle$	$\langle 1,0,0,0\rangle$
$\langle 0,1,1,0\rangle$	$\langle 0,1,0,1\rangle$	$\langle 0,0,1,1\rangle$
$\langle 0,1,1,1\rangle$	$\langle 1,0,0,1\rangle$	$\langle 1,1,1,0\rangle$
$\langle 1,0,0,0\rangle$	$\langle 1,0,0,0\rangle$	$\langle 0,0,0,0\rangle$
$\langle 1,0,0,1\rangle$	$\langle 1,0,1,1\rangle$	$\langle 0,0,1,0\rangle$
$\langle 1,0,1,0\rangle$	$\langle 1,1,1,1\rangle$	$\langle 0,1,0,1\rangle$
$\langle 1,0,1,1\rangle$	$\langle 0,0,0,1\rangle$	$\langle 1,0,1,0\rangle$
$\langle 1,1,1,0\rangle$	$\langle 0,1,1,1\rangle$	$\langle 1,0,1,1\rangle$
$\langle 1,1,0,1\rangle$	$\langle 0,1,0,0\rangle$	$\langle 1,0,0,1\rangle$
$\langle 1,1,1,0\rangle$	$\langle 1,0,1,0\rangle$	$\langle 0,1,0,0\rangle$
$\langle 1,1,1,1\rangle$	$\langle 0,0,0,0\rangle$	$\langle 1,1,1,1\rangle$

Figure 4.25 A non-affine complete map θ and $i \oplus \theta$ on $\mathbb{Z}_2^4$

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\theta(x)$	12	6	3	14	2	13	5	9	8	11	15	1	7	4	10	0
$x \oplus \theta(x)$	12	7	1	13	6	8	3	14	0	2	5	10	11	9	4	15

Figure 4.26 θ and $i \oplus \theta$ on integers corresponding to elements of $\mathbb{Z}_2^4$

$Q =$

$\circ$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	12	7	1	13	6	8	3	14	0	2	5	10	11	9	4	15
1	6	13	12	0	9	7	15	2	3	1	11	4	8	10	14	5
2	3	15	14	5	1	12	4	10	7	8	2	0	6	13	9	11
3	14	3	4	15	13	0	11	5	9	6	1	3	12	7	10	8
4	2	12	7	10	8	3	5	9	15	13	0	11	4	6	1	14
5	13	3	11	6	2	9	8	4	12	14	10	1	7	5	15	0
6	5	8	0	14	7	11	10	1	2	9	13	15	3	12	6	4
7	9	4	15	1	10	6	0	11	8	3	14	12	13	2	5	7
8	8	10	13	2	3	1	12	7	4	15	9	5	14	0	11	6
9	11	9	3	12	0	2	6	13	14	5	4	8	1	15	7	10
10	15	0	10	8	14	5	1	3	11	7	6	13	8	4	12	2
11	1	14	9	11	4	15	2	0	6	10	12	7	5	8	3	13
12	7	5	8	3	12	14	9	6	10	4	15	2	0	11	13	1
13	4	6	2	9	15	13	7	8	5	11	3	14	10	1	0	12
14	10	1	5	7	11	4	14	12	13	0	8	6	15	3	2	9
15	0	11	6	4	5	10	13	15	1	12	7	9	2	14	8	3

Figure 4.27 The quasigroup $(Q, \circ)$ created using the non-affine complete map θ.

As in Example 15, notice that θ is not affine, because

$$\theta(1 \oplus 2) = \theta(3) = 14 \text{ but } \theta(1) \oplus \theta(2) \oplus \theta(0) = 6 \oplus 3 \oplus 12 = 9.$$

This again shows that Property 2 does not hold. Property 4 does not hold because

$$3 \circ 4 = 14 \circ 8 \text{ but } 3 \circ 8 \neq 14 \circ 4.$$

In addition, for all $x \in Q$, there is some $k \in Q$ so that

$$\theta(x \oplus k) \neq \theta(x) \oplus \theta(k) \oplus \theta(0).$$

Therefore, there is no pairing of elements for any $k \in Q$ and Property 1 also does not hold.

4.3 T-functions

Another way of creating a quasigroup uses a special type of function defined by Klimov and Shamir in (6). Recall that $\prod_{i=1}^{k} \mathbb{Z}_2^n$ denotes the set of ordered k-tuples whose components are binary strings of length n. For ease of notation, if $f : \prod_{i=1}^{k} \mathbb{Z}_2^n \to \prod_{i=1}^{l} \mathbb{Z}_2^n$ and $x = \langle x_{k-1}, x_{k-2}, \ldots, x_1, x_0 \rangle$, let $[x]_{\star,i}$ denote the i^{th} bit of $x_\star$ and let $[f(x)]_{\star,i}$ denote the i^{th} bit of component $\star$ of $f(x)$. If $l = 1$ (that is, $f : \prod_{i=1}^{k} \mathbb{Z}_2^n \to \mathbb{Z}_2^n$), then $f(x)$ has only one component, so denote its i^{th} bit by $[f(x)]_i$.

Definition 14. Let $f : \prod_{i=1}^{k} \mathbb{Z}_2^n \to \prod_{i=1}^{l} \mathbb{Z}_2^n$. f is a **T-function** if for all $x \in \prod_{i=1}^{k} \mathbb{Z}_2^n$, $[f(x)]_{\star,i}$ depends only on the rightmost i bits of each component of x, for any $\star \in \{0, \ldots, k-1\}$.

Example 17. The addition function $f : \prod_{i=1}^{2} \mathbb{Z}_2^n \to \mathbb{Z}_2^n$ given by $f(x, y) = x + y \mod 2^n$ is a T-function. $[f(x, y)]_0$ (that is, the rightmost bit of $f(x, y)$) depends only on the rightmost bits of x and y. This is true because $[f(x, y)]_0 = [x]_0 + [y]_0$. The second bit of $f(x, y)$ depends on the second bits of x and y, plus the carry from the first bits of x and y. That is, $[f(x, y)]_1 = [x]_1 + [y]_1 + \alpha_0$, where α_0 is the carry from adding the rightmost bits of x and y. Therefore, α_0 only depends on $[x]_0$ and $[y]_0$ and so $[f(x, y)]_1$ only depends on $[x]_1, [x]_0, [y]_1$, and $[y]_0$. We can continue in the following way (making the result more precise by using induction if desired) to see that $f(x, y)$ is a T-function.

Notice that any function that uses a combination of addition, subtraction, and multiplication modulo 2^n and Boolean operations (such as and, or, bitwise exclusive-or) will be a T-function. This can easily be seen by using an argument similar to the one in Example 17. In addition, composition of two T-functions will also be a T-function. Because we will be using T-functions to create a quasigroup, it is convenient to choose $k = l$, so that f takes as input and produces as output k many binary strings of length n.

In order to use a T-function f to define a quasigroup operation, we will see that f needs to be a permutation. Therefore, we need to know which T-functions are invertible. The following result from (6) answers this question.

Claim. Let $v : \mathbb{Z}_2^n \to \mathbb{Z}_2^n$ be a T-function. Then $f(x) = c + x + 2v(x) \mod 2^n$ is invertible, where $c \in \mathbb{Z}_{2^n}$. (Notice that $f(x)$ is also a T-function.)

Proof. Because $\mathbb{Z}_2^n$ is finite, we only need to show that $f(x)$ is 1-1. Assume that $f(x) = f(y)$ for $x, y \in \mathbb{Z}_2^n$ and show that $x = y$. Proceed by induction on i, where i represents a bit of x and y. Let $x = \langle x_{n-1}, x_{n-2}, \ldots, x_1, x_0 \rangle$ and $y = \langle y_{n-1}, y_{n-2}, \ldots, y_1, y_0 \rangle$, so i ranges from 0 to $k - 1$.

Base Case: $i = 0$. We need to show that $[x]_0 = [y]_0$. Notice that $[f(x)]_0 = [f(y)]_0$ because we assumed that $f(x) = f(y)$. Therefore, we have

$$\begin{aligned}
[f(x)]_0 &= [c + x + 2v(x)]_0 \\
&= [c]_0 \oplus [x]_0 \oplus [2v(x)]_0 \\
&= [c]_0 \oplus [x]_0 \oplus [v(x)]_0 \oplus [v(x)]_0 \\
&= [c]_0 \oplus [x]_0
\end{aligned}$$

and

$$\begin{aligned}
[f(y)]_0 &= [c + y + 2v(y)]_0 \\
&= [c]_0 \oplus [y]_0 \oplus [2v(y)]_0 \\
&= [c]_0 \oplus [y]_0 \oplus [v(y)]_0 \oplus [v(y)]_0 \\
&= [c]_0 \oplus [y]_0
\end{aligned}$$

Hence $[c]_0 \oplus [x]_0 = [c]_0 \oplus [y]_0$ and so $[x]_0 = [y]_0$.

Induction Step Assume that $[x]_j = [y]_j$ for $j = 1, \ldots, i-1, i < n$ and show that $[x]_i = [y]_i$. We can see that

$$[f(x)]_i = [c + x + 2v(c)]_i = [c]_i \oplus [x]_i \oplus [2v(x)]_i \oplus \alpha(x)_i$$

where $\alpha(x)_i$ is the carry from the previous bits of $f(x)$ modulo 2. That is, $\alpha(x)_i$ depends only on $[f(x)]_j$ for $j = 1, \ldots, i-1$. However, because bitwise multiplication by 2 simply shifts bits to the left,

$$[f(x)]_i = [c]_i \oplus [x]_i \oplus [v(x)]_{i-1} \oplus \alpha(x)_i.$$

Similarly,

$$[f(y)]_i = [c]_i \oplus [y]_i \oplus [v(y)]_{i-1} \oplus \alpha(y)_i.$$

Because v is a T-function, $[v(x)]_{i-1}$ depends only on $[x]_j$ for $j = 0, \ldots, i-1$. Similarly, $[v(y)]_{i-1}$ depends only on $[y]_j$ for $j = 0, \ldots, i-1$. Therefore, by our induction hypothesis, we know that $[v(x)]_{i-1} = [v(y)]_{i-1}$. In addition, $\alpha(x)_i$ depends only on $[x]_j$ for $j = 0, \ldots, i-1$ and $\alpha(y)_i$ depends only on $[y]_j$ for $j = 0, \ldots, i-1$. Our induction hypothesis again implies that $\alpha(x)_i = \alpha(y)_i$. Finally, because $f(x) = f(y)$, we know that $[f(x)]_i = [f(y)]_i$ and so

$$[c]_i \oplus [x]_i \oplus [v(x)]_{i-1} \oplus \alpha(x)_i = [c]_i \oplus [y]_i \oplus [v(y)]_{i-1} \oplus \alpha(y)_i$$

$$[x]_i \oplus [v(x)]_{i-1} \oplus \alpha(x)_i = [c]_i \oplus [v(y)]_{i-1} \oplus \alpha(y)_i$$

$$[x]_i = [y]_i$$

Therefore, $x = y$ and so $f(x) = c + x + 2v(x)$ is a T-function if $v(x)$ is a T-function. □

The converse of this proposition is also true, but the converse is not needed for this paper, so it will not be proved.

We can now define a quasigroup operation based on a T-function. Let $Q = \mathbb{Z}_2^n$ and let $v : Q \times Q \to Q$ be a T-function. Create a quasigroup $(Q, \circ)$ by defining

$$x \circ y = c + (x + y) + 2v(x, y) \mod 2^n$$

where $c \in Q$.

Claim. If $x \circ y$ is defined as above, then $(Q, \circ)$ is a quasigroup.

Proof. Because Q is finite, we only need to show that $L_a(x) = a \circ x$ and $R_a(x) = x \circ a$ are invertible for all $a \in Q$. First, consider $L_a(x)$ for some arbitrary $a \in Q$.

$$\begin{aligned} L_a(x) = a \circ x &= c + (a + x) + 2v(a, x) \\ &= (c + a) + x + 2v(a, x) \end{aligned}$$

Notice that if $v(y, z)$ is a T-function of two variables and y is fixed, then we can think of $v(y, z)$ as a T-function of one variable. Therefore, $L_a(x)$ has the form of an invertible T-function and hence $L_a(x)$ is invertible. A similar argument shows that $R_a(x)$ is also invertible. Therefore, $(Q, \circ)$ is a quasigroup. □

Example 18. Let $v : \mathbb{Z}_2^3 \times \mathbb{Z}_2^3 \to \mathbb{Z}_2^3$ be given by

$$v(x, y) = x^2 y + 3(x \vee y)$$

where addition and multiplication are computed mod $2^3 = 8$ and $\vee$ represents Boolean or. Let $c = \langle 1, 0, 1 \rangle \in \mathbb{Z}_2^3$. As discussed above, define

$$x \circ y = c + (x + y) + 2v(x, y) = 5 + x + y + x^2 y + 3(x \vee y).$$

Then the quasigroup in Figure 4.28 is created, where the binary representatives of the quasigroup elements are used for ease of notation.

$Q =$

$\circ$	0	1	2	3	4	5	6	7
0	5	4	3	2	1	0	7	6
1	4	7	6	1	0	3	2	5
2	3	2	5	4	7	6	1	0
3	2	5	0	3	6	1	4	7
4	1	0	7	6	5	4	3	2
5	0	3	2	5	4	7	6	1
6	7	6	1	0	3	2	5	4
7	6	1	4	7	2	5	0	3

Figure 4.28 The quasigroup $(Q, \circ)$ created using a T-function $v(x, y)$.

One possible disadvantage of using a T-function to create a quasigroup can be seen in the resulting structure present in the quasigroup. In the quasigroup from Example 18, notice that the entries in each row and each column will alternate between even and odd numbers. That is, if $x \circ y$ is even, then $x \circ (y+1)$ and $(x+1) \circ y$ will be odd and vice versa. It can be easily seen that this property holds in any quasigroup created from a T-function. Notice that both x and y are either even or odd. Clearly, if x is even, then $x+1$ is odd and vice versa. Recall that

$$x \circ y = c + x + y + 2v(x, y)$$

and

$$x \circ (y+1) = c + x + y + 1 + 2v(x, y+1).$$

Because $2v(\cdot, \cdot)$ is always even, the parity of $x \circ (y+1)$ will be different than the parity of $x \circ y$. Therefore, the entries in each row will alternate between even and odd elements. A similar argument can be used to show that column entries also alternate.

Chapter 4 assures us that there are efficient methods of creating and storing large quasigroups. Using isotopies and complete mappings are two of these methods. In addition, if we choose affine isotopies or an affine complete mapping, the quasigroup created has special properties. Now that we know it is practical to work with large quasigroups, Chapter 5 will explore the other key property required of quasigroups used for QMAC—that these quasigroups are "highly non-associative".

CHAPTER 5. NON-ASSOCIATIVITY OF QUASIGROUPS

As discussed in Chapter 4, there are various methods which are quite efficient for creating and storing large quasigroups. However, in addition to requiring $(Q, \circ)$ to be large in order to ensure the security of QMAC, we also need $(Q, \circ)$ to be "highly non-associative". Recall that the security of QMAC depends on different keys (that is, different parentheses schemes) producing different authentication tags. Therefore, if there are a significant number of keys which "collapse" to create the same authentication tag for a given message, the security of QMAC will be compromised. The worst-case scenario occurs when **every** key produces the same authentication tag for a given message. This will occur if $(Q, \circ)$ is an associative quasigroup, i.e. a group. Therefore, in order to avoid as much collapsing of keys as possible, we would like $(Q, \circ)$ to be "as non-associative" as possible.

There are two questions which need to be answered. First, for a given quasigroup $(Q, \circ)$, how can we determine "how non-associative" it is? One answer to this question will be presented in Section 5.1. And second, what kind of methods can be used to create large, highly non-associative quasigroups? Chapter 4 gives a variety of efficient methods for constructing large quasigroups. However, some of these methods (such as affine isotopies) create quasigroups which have a significant amount of structure, which may create security issues for QMAC. Section 5.2 will examine methods for creating large, highly non-associative quasigroups with very little structure.

5.1 Measuring the Non-Associativity of Quasigroups

In order to measure "how non-associative" a quasigroup $(Q, \circ)$ is, we will translate the method used to measure the non-commutativity of groups into a method which measures the

non-associativity of quasigroups. We need to consider a special group associated with a group G.

Definition 15. Let $(G, \cdot)$ be a group and let L_x, R_x denote left and right multiplication by $x \in G$, respectively. The **multiplication group of** $(G, \cdot)$ is the subgroup of $\mathrm{Sym}(G)$ generated by all left and right multiplications of G. That is, $\mathrm{Mult}(G) := \langle L_x, R_x : x \in G \rangle_{\mathrm{Sym}(G)}$.

If $(G, \cdot)$ is an abelian group, then $L_x = R_x$ for all $x \in G$. This implies that when $\mathrm{Mult}(G)$ is generated, a certain amount of "collapsing" occurs, and so $\mathrm{Mult}(G)$ is a smaller group. Therefore, if $(G, \cdot)$ is "more non-commutative", then $\mathrm{Mult}(G)$ will be a larger group. Hence we can use the size of $\mathrm{Mult}(G)$ to measure "how non-commutative" a group $(G, \cdot)$ is.

Now consider a quasigroup $(Q, \circ)$. We can define $\mathrm{Mult}(Q)$ in the same way that we defined $\mathrm{Mult}(G)$ for a group $(G, \cdot)$. If $(Q, \circ)$ is associative, then $L_x = R_y$ for all $x, y \in Q$. This implies that $\mathrm{Mult}(Q)$ will be a smaller group. Therefore, a larger multiplication group implies that $(Q, \circ)$ is "more non-associative".

Notice that $\mathrm{Mult}(Q)$ is a subgroup of $\mathrm{Sym}(Q)$, so $|\mathrm{Mult}(Q)| \leq |Q|!$. This means that the "most non-associative" quasigroups will have $|\mathrm{Mult}(Q)| = |Q|!$, or $\mathrm{Mult}(Q) = \mathrm{Sym}(Q)$. This discussion leads to the following definition.

Definition 16. Let $(Q, \circ)$ be a quasigroup. Q is **highly non-associative** if $\mathrm{Mult}(Q) = \mathrm{Sym}(Q)$.

Before determining whether using quasigroups with this type of non-associativity provides some measure of security for QMAC, we need to determine if it is even possible to use these type of quasigroups. If there are very few large highly non-associative quasigroups or if they are difficult to create using some kind of efficient method, then we will need to devise a more practical way to measure non-associativity. Because the quasigroups used in QMAC are public knowledge and are not part of Alice and Bob's secret key, a quasigroup can be used multiple times and a new quasigroup does not need to be chosen every time a new message is sent. However, it does not seem wise to use the same quasigroup for every message sent between Alice and Bob, because Eve may be able to study the structure of the quasigroup and gain some information about the parenthesis scheme being used as the key. Therefore, although

we do not need as many quasigroups as messages, we still need a fair number of large highly non-associative quasigroups in order to make QMAC more secure. The next result from (12) assures us that there are a large number of these quasigroups.

Definition 17. Let $\mathcal{P}$ be a property that a finite quasigroup may or may not possess. Let $p(n)$ denote the number of Latin squares of order n that are associated with quasigroups having property $\mathcal{P}$. Let $l(n)$ denote the total number of Latin squares on $\{1, 2, \ldots, n\}$. Then **almost all finite quasigroups have property** $\mathcal{P}$ if $\lim_{n\to\infty} \frac{p(n)}{l(n)} = 1$.

Proposition 3. *Almost all finite quasigroups are highly non-associative.*

The other criterion for using these types of quasigroups is that they are easy to create and store (as discussed in Chapter 4). It turns out that this is also the case. Several methods for creating such quasigroups are discussed in the next section.

5.2 Creating Highly Non-Associative Quasigroups

Before discussing how to create large, highly non-associative quasigroups, it is important to note that it is easy to create a highly non-associative quasigroup using affine isotopies. This result is made more precise in Proposition 4.

Proposition 4. *Let $(Q, +)$ be a cyclic group and let $g : Q \to Q$ be a transposition. Then $(Q, \circ)$ given by $x \circ y = x + g(y)$ is a highly non-associative quasigroup (that is, Mult(Q) = Sym(Q)).*

Proof. Because $(Q, \circ)$ is generated from isotopy from $f = id$ and g, both f and g appear in Mult(Q). Because g is a transposition, it is a permutation of order 2.

In addition, L_x and R_x are in Mult(Q) for all $x \in Q$. In particular, $L_a \in$ Mult(Q) for $a \in Q$ where $g(a) = a$. This implies that L_a is a cycle of length $n = |Q|$. Together, g and L_a generate the entire symmetric group of size n. Therefore, Sym(Q) = Mult(Q) and so $(Q, \circ)$ is highly non-associative. □

Example 19. Let $(Q, +) = (\mathbb{Z}_8, +)$ and let g interchange the elements 0 and 1. Then the Cayley table for $(Q, \circ)$ created by $x \circ y = x + g(y)$ is shown in Figure 5.1. Notice that because of Proposition 4, $|\text{Mult}(Q)| = 8!$, but $(Q, \circ)$ is clearly a highly structured quasigroup.

$Q =$

$\circ$	0	1	2	3	4	5	6	7
0	1	0	2	3	4	5	6	7
1	2	1	3	4	5	6	7	0
2	3	2	4	5	6	7	0	1
3	4	3	5	6	7	0	1	2
4	5	4	6	7	0	1	2	3
5	6	5	7	0	1	2	3	4
6	7	6	0	1	2	3	4	5
7	0	7	1	2	3	4	5	6

Figure 5.1 A structured highly non-associative quasigroup.

In Sections 4.1.1 and 4.1.2, we saw that quasigroups created from abelian groups by affine isotopies have a significant amount of structure. While it is also true that any quasigroup created by isotopy from an abelian group has a certain amount of structure, this structure is amplified if the isotopies are affine maps. Therefore, in addition to requiring that our quasigroups have full multiplication groups, we need to create them using methods other than affine isotopies.

5.2.1 Isotopies and Feistel Networks

The basic idea of using isotopies to create large quasigroups still appears to have a certain amount of promise, as long as we use non-linear bijections for our isotopies. It is easy to create a table-driven non-linear bijection, which would meet our criteria as long as we didn't care about the size of our quasigroups. However, since we need to create large quasigroups (containing on the order of 2^{16} elements), it is not practical to store two table-driven bijections. In general, using elementary operations, it is much more difficult to create a non-linear function which is not table-driven. To solve this problem, we can turn to the work which has been done in creating non-linear functions for cryptographic purposes.

One commonly used technique for creating a non-linear cryptographic function is a Feistel network. A Feistel network takes any function and transforms it into a bijection. (See (11) for an in-depth treatment of Feistel networks.) Methods for creating quasigroups using Feistel networks are discussed in Sections 5.2.1.1 and 5.2.1.2.

5.2.1.1 Isotopies Using Two Feistel Network Functions

A Feistel network utilizes an auxiliary function called an S-box to create a bijection as described in Definition 18.

Definition 18. Let k be a positive integer and let $f : \mathbb{Z}_2^k \to \mathbb{Z}_2^k$ be any function. Create $F : \mathbb{Z}_2^{2k} \to \mathbb{Z}_2^{2k}$ by defining

$$F(l, r) := (r, l \oplus f(r))$$

where (l, r) denotes the element of $\mathbb{Z}_2^{2k}$ created by concatenating l and r and $\oplus$ denotes bitwise exclusive-or.

Notice that F is clearly a bijection because it has an inverse function, which is given by $G(l, r) := (r \oplus f(l), l)$. In addition, if we start with a non-linear function f, F will usually also be a non-linear function. Therefore, it is possible to start with a function on $\mathbb{Z}_2^k$ and create a non-linear bijection on $\mathbb{Z}_2^{2k}$ which is not table-driven. In addition, even if f is table-driven, we will need to store a much smaller table because f acts on bitstrings of length k instead of bitstrings of length $2k$.

We can use Feistel networks to create two isotopies F and G, and then use these two isotopies to create a quasigroup, as shown in Example 20.

Example 20. Let f and g be functions from $\mathbb{Z}_2^3$ to itself as shown in Figure 5.2, where the elements of $\mathbb{Z}_2^3$ are represented as integers for ease of notation. Create F and G using Feistel networks, and use F and G as isotopies to create $(Q, \circ)$. (That is, $Q = \mathbb{Z}_2^6$, so $|Q| = 2^6$, and $x \circ y = F(x) \oplus G(y)$ for $x, y \in Q$.) Then the quasigroup $(Q, \circ)$ does not have a full multiplication group, but $\dfrac{|\mathrm{Sym}(Q)|}{|\mathrm{Mult}(Q)|} = 2$. $(Q, \circ)$ is too large to display here.

x	0	1	2	3	4	5	6	7
$f(x)$	3	0	1	6	5	4	2	7
$g(x)$	6	4	5	2	3	0	7	1

Figure 5.2 Functions f and g used in a Feistel network.

After creating a number of quasigroups using two functions generated by Feistel networks and examining their multiplication groups, we found that a quasigroup with a full multiplication group was never generated. A significant number of the quasigroups had Mult(Q) = Alt(Q) as in Example 20, but none had Mult(Q) = Sym(Q). This fact can be explained by the following proposition from (5).

Proposition 5. *Let $f : \mathbb{Z}_2^k \to \mathbb{Z}_2^k$ and let $F : \mathbb{Z}_2^{2k} \to \mathbb{Z}_2^{2k}$ be created from f by a Feistel network. Then F is an even permutation.*

Proof. We will prove Proposition 5 by decomposing F into two functions σ and θ which map from $\mathbb{Z}_2^{2k}$ to itself and showing that both functions are even permutations when $k > 1$. Let σ and θ be given by

$$\sigma(x, y) := (x \oplus f(y), y)$$
$$\theta(x, y) := (y, x)$$

Then clearly $F = \theta \circ \sigma$. In addition, the order of both σ and θ is 2, so σ and θ written in cycle notation consist entirely of disjoint transpositions.

Claim. If $k > 1$, then θ is an even permutation.

Proof. Notice that $\theta(x, x) = (x, x)$ and $\theta(x, y) = (y, x)$. Therefore, if $x \neq y$, θ interchanges x and y. There are 2^k elements of the form (x, x) in $\mathbb{Z}_2^{2k}$, so θ fixes 2^k elements and interchanges $2^{2k} - 2^k$ elements. Therefore, θ consists of $\frac{1}{2}(2^{2k} - 2^k)$ disjoint transpositions. But $\frac{1}{2}(2^{2k} - 2^k) = 2^{k-1}(2^k - 1)$ and because $k > 1$, 2^{k-1} is even. Hence θ is made up of an even number of transpositions and so θ is an even permutation. □

Claim. If $k > 1$, then σ is an even permutation.

Proof. Let $\langle 0, \ldots, 0\rangle \in \mathbb{Z}_2^k$ and let n be the size of the preimage of $\langle 0, \ldots, 0\rangle$ under f. (That is, $n = |\{z \in \mathbb{Z}_2^k : f(z) = \langle 0, \ldots, 0\rangle\}|$). If $n = 2^k$, then $f(z) = \langle 0, \ldots, 0\rangle$ for all $z \in \mathbb{Z}_2^k$. This implies that for all $(x, y) \in \mathbb{Z}_2^{2k}$ we have

$$\sigma(x, y) = (x \oplus f(y), y) = (x, y).$$

So σ is the identity permutation and hence is even.

If $n < 2^k$, then there is some $y \in \mathbb{Z}_2^k$ so that $f(y) \neq \langle 0, \ldots, 0 \rangle$. Because $f(y) \neq \langle 0, \ldots, 0 \rangle$, we can see that $(x, y) \neq \sigma(x, y)$. However, we do have the following:

$$\sigma(x, y) = (x \oplus f(y), y)$$

$$\sigma(x \oplus f(y), y) = (x \oplus f(y) \oplus f(y), y) = (x, y)$$

Therefore, σ fixes exactly n elements of $\mathbb{Z}_2^{2k}$ and interchanges the rest in pairs. In order to give a convenient formula for how many elements σ interchanges, notice that σ interchanges (x, y) in $\mathbb{Z}_2^{2k}$ as long as $f(y) \neq \langle 0, \ldots, 0 \rangle$. There are 2^k choices for x and $2^k - n$ choices for y, so σ interchanges $2^k(2^k - n)$ elements of $\mathbb{Z}_2^{2k}$. Therefore, σ consists of $\frac{1}{2} \cdot 2^k(2^k - n) = 2^{k-1}(2^k - n)$ disjoint transpositions. Because $k > 1$, 2^{k-1} is even. Hence σ is made up of an even number of transpositions and so σ is an even permutation □

Finally, because f is the composition of two even permutations, f is also an even permutation. □

Proposition 5 tells us that it is impossible to use two isotopies created using Feistel networks to generate a highly non-associative quasigroup. However, because Feistel networks are a promising way to create non-linear functions, we would like to avoid abandoning this method completely.

5.2.1.2 Isotopies Using One Feistel Network Function

We can still use the idea of isotopies and Feistel networks to create quasigroups with a large multiplication group. However, in order to make it possible to have $\mathrm{Mult}(Q) = \mathrm{Sym}(Q)$, we will need to require that one of the isotopies is an odd permutation. Depending on the isotopies chosen, this method may not produce a quasigroup with a full multiplication group every time. However, Proposition 3 implies (and computational results verify) that highly non-associative quasigroups are relatively easy to generate in this way. One such quasigroup is shown in Example 21.

Example 21. Let $f : \mathbb{Z}_2^2 \to \mathbb{Z}_2^2$ and $g : \mathbb{Z}_2^4 \to \mathbb{Z}_2^4$ be as shown in Figure 5.3, where the elements of $\mathbb{Z}_2^2$ and $\mathbb{Z}_2^4$ are represented as integers for ease of notation. Notice that g is a cycle of length 16, so g is an odd permutation. Create $F : \mathbb{Z}_2^4 \to \mathbb{Z}_2^4$ from f by a Feistel network and use F and g as isotopies to create $(Q, \circ)$ as shown in Figure 5.4. (That is, $Q = \mathbb{Z}_2^4$ and $x \circ y = F(x) \oplus g(y)$ for $x, y \in Q$.) Then $(Q, \circ)$ has a full multiplication group.

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$f(x)$	2	1	3	0												
$g(x)$	3	15	5	4	2	11	12	13	9	6	0	14	7	10	1	8

Figure 5.3 Functions f (used in a Feistel network) and g (an odd permutation).

$\circ$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	13	7	6	0	9	14	15	11	4	2	12	5	8	3	10
1	6	10	0	1	7	14	9	8	12	3	5	11	2	15	4	13
2	8	4	14	15	9	0	7	6	2	13	11	5	12	1	10	3
3	15	3	9	8	14	7	0	1	5	10	12	2	11	6	13	4
4	0	12	6	7	1	8	15	14	10	5	3	13	4	9	2	11
5	7	11	1	0	6	15	8	9	13	2	4	10	3	14	5	12
6	9	5	15	14	8	1	6	7	3	12	10	4	13	0	11	2
7	14	2	8	9	15	6	1	0	4	11	13	3	10	7	12	5
8	3	15	5	4	2	11	12	13	9	6	0	14	7	10	1	8
9	4	8	2	3	5	12	11	10	14	1	7	9	0	13	6	15
10	10	6	12	13	11	2	5	4	0	15	9	7	14	3	8	1
11	13	1	11	10	12	5	2	3	7	8	14	0	9	4	15	6
12	2	14	4	5	3	10	13	12	8	7	1	15	6	11	0	9
13	5	9	3	2	4	13	10	11	15	0	6	8	1	12	7	14
14	11	7	13	12	10	3	4	5	1	14	8	6	15	2	9	0
15	12	0	10	11	13	4	3	2	6	9	15	1	8	5	14	7

Figure 5.4 The quasigroup $(Q, \circ)$ created using a Feistel network and an odd permutation.

Notice that because $(Q, \circ)$ was created from $(\mathbb{Z}_2^4, \oplus)$ by isotopy, it necessarily has the pairing of elements described in Property 4 of Section 4.1.2. However, none of the pairing properties from Section 4.1.1 are present in $(Q, \circ)$. Therefore, although $(Q, \circ)$ has some structure, the

choices of f and g ensure that it does not have as much structure as a quasigroup created by affine isotopies.

As mentioned above, Feistel networks appear to be a promising way to create non-linear functions, resulting in a quasigroup with a full multiplication group. In addition, only f and g need to be stored in order to create $(Q, \circ)$, and performing multiplication in $(Q, \circ)$ does not require a table look-up. It appears that it would be worthwhile to explore using a Feistel network to create $(Q, \circ)$ from a non-abelian group $(Q, \star)$ by isotopy. Perhaps this method will again result in a quasigroup with a full multiplication group, but with significantly less structure than a quasigroup created by isotopy from an abelian group.

5.2.2 Isotopies and Linear Feedback Shift Registers

As mentioned previously, we can use a Feistel network and an odd permutation as isotopies to create a quasigroup with a full multiplication group. We proposed one method for creating this odd permutation in Section 5.2.1.2. In this section, we would like to use another cryptographic tool to create a non-linear odd permutation.

One common problem in cryptography involves producing pseudorandom sequences of bitstrings. The idea of a linear feedback shift register (LFSR) is often used to produce such a sequence. However, LFSRs can also be used to create functions whose input and output are bitstrings of a specified length. The basic idea of a LFSR is as follows: the output is obtained by dropping the least significant input bit, shifting the remaining input bits and using the input bits, along with the coefficients of a specified polynomial of degree k over $\mathbb{Z}_2$, to create a new most significant output bit. A more precise definition of a LFSR is given in Definition 19.

Definition 19. Let $q(x) = c_0 + c_1x + c_2x^2 + \cdots + c_{k-1}x^{k-1} + x^n$ be a polynomial over $\mathbb{Z}_2[x]$

where $c_0 \neq 0$. Define f and f' as follows:

$$\begin{aligned} f' : \mathbb{Z}_2^k &\to \mathbb{Z}_2 \\ \langle b_0, b_1, \ldots, b_{k-1} \rangle &\mapsto b_0 c_0 \oplus b_1 c_1 \oplus \cdots \oplus b_{k-1} c_{k-1} \\ f : \mathbb{Z}_2^k &\to \mathbb{Z}_2^k \\ \langle x_0, x_1 \ldots x_{k-1} \rangle &\mapsto \langle x_1, x_2, \ldots, x_{k-1}, f'(\langle x_0, \ldots, x_{k-1} \rangle) \rangle \end{aligned}$$

Then f is a **linear feedback shift function**.

If f is written as a permutation, then the length of its cycles depends on the choice of $q(x)$. If $q(x)$ is a polynomial without certain properties, then the cycles of f will be of varying length. However, if $q(x)$ is one of the following type of polynomials, then we can say something more specific about the cycle decomposition of f.

Definition 20. Let $q(x)$ be a monic polynomial of degree k over the field $\mathbb{Z}_2$.

- $q(x)$ is **irreducible** if $q(x)$ cannot be factored nontrivially over $\mathbb{Z}_2[x]$.
- $q(x)$ is **primitive** if $q(x)$ is irreducible and $q(x) \nmid x^n - 1$ for all $n < 2^k - 1$.

Before discussing the cycle decomposition of f, notice that $\langle 0, \ldots, 0 \rangle$ is always in a cycle by itself, because

$$\begin{aligned} f'(\langle 0, \ldots, 0 \rangle) &= 0, \text{ so} \\ f(\langle 0, \ldots, 0 \rangle) &= \langle 0, 0, \ldots, 0, f'(\langle 0, \ldots, 0 \rangle) \rangle \\ &= \langle 0, \ldots, 0 \rangle \end{aligned}$$

If $q(x)$ is irreducible, then each of the cycles (except the cycle containing $\langle 0, \ldots, 0 \rangle$) will have length t, where t is the order of some root α of $q(x)$ in the multiplicative group $\mathbb{F}_{2^k}^*$. Because the order of an element must divide the order of the group, we can see that $t \mid 2^k - 1$ and therefore t is odd. So the image of f consists of $\frac{2^k - 1}{t}$ cycles of odd length t and one cycle of length 1.

If $q(x)$ is primitive, we are saying precisely that each of its roots has order $2^k - 1$ in the group $\mathbb{F}^*_{2^k}$. Therefore, there is the cycle of length 1 containing $\langle 0, \ldots, 0\rangle$, and then all the other elements are contained in another cycle (of length $2^k - 1$).

Also, note that f is clearly a linear function (as the name suggests), because

$$\begin{aligned} f'(x \oplus y) &= (x_0 \oplus y_0) \cdot c_0 \oplus \cdots \oplus (x_{k-1} \oplus y_{k-1}) \cdots_{k-1} \\ &= (x_0 \cdot c_0 \oplus \cdots \oplus x_{k-1} \cdot c_{k-1}) \oplus (y_0 \cdot c_0 \oplus \cdots \oplus y_{k-1} \cdot c_{k-1} \\ &= f'(x) \oplus f'(y), \text{ so} \\ f(x \oplus y) &= \langle x_1 \oplus y_1, \ldots, x_{k-1} \oplus y_{k-1}, f'(x \oplus y)\rangle \\ &= f(x) \oplus f(y) \end{aligned}$$

Based on the discussion above, we would like to modify f in order to solve two problems. We would like $\langle 0, \ldots, 0\rangle$ to not be in a cycle by itself (so $f(\langle 0, \ldots, 0\rangle) \neq \langle 0, \ldots, 0\rangle$) and we would like f to be a nonlinear function. However, we would like to preserve the basic premise of creating f by shifting input bits and creating one new output bit. Let g denote the function created by modifying f.

In order to insert $\langle 0, \ldots, 0\rangle$ into a cycle but still produce an output by shifting bits, notice that we will need to have $g(\langle 1, 0, \ldots, 0\rangle) = \langle 0, \ldots, 0\rangle$. Let $\overline{x_i}$ denote the bitwise complement of x_i (that is, $\overline{0} = 1$ and $\overline{1} = 0$) and define g as follows:

$$\begin{aligned} g' &: \mathbb{Z}_2^k \to \mathbb{Z}_2 \\ &\langle b_0, \ldots, b_{k-1}\rangle \mapsto f'(\langle b_0, \ldots, b_{k-1}\rangle) \oplus \overline{b_1} \cdots \overline{b_{k-1}} \\ g &: \mathbb{Z}_2^k \to \mathbb{Z}_2^k \\ &\langle x_0, \ldots, x_{k-1}\rangle \mapsto \langle x_1, \ldots, x_{k-1}, g'(\langle x_0, \ldots, x_{k-1}\rangle)\rangle \end{aligned}$$

Notice that the images of f and g differ only in the placement of $\langle 0, \ldots, 0\rangle$ in their cycle decompositions. In the image of f, $\langle 0, \ldots, 0\rangle$ is a cycle of length 1. In the image of g, $\langle 0, \ldots, 0\rangle$ is inserted into the cycle containing $\langle 1, 0, \ldots, 0\rangle$ and all other cycles of f are left unchanged.

If $q(x)$ is primitive, then this modified function g produces a **De Bruijn sequence**. We would finally like to show that if $q(x)$ is irreducible or primitive, then g is an odd permutation.

If $q(x)$ is irreducible, then all but one of the cycles have t many elements (where t is odd) and one cycle (the one where $\langle 0, \ldots, 0 \rangle$ was inserted) contains $t + 1$ elements. Therefore, g is made up of a number of even cycles and one odd cycle, so g is an odd permutation. If $q(x)$ is primitive, it is necessarily irreducible and hence the g is again an odd permutation.

Notice that in order to create g, it is only necessary to store the coefficients of the polynomial $q(x)$. Therefore, we can produce an odd non-linear permutation from a bitstring of length k. This implies that this modification to functions produced using LFSRs gives us another method for creating odd permutations. The function g can be used in conjunction with another bijection F created using a Feistel network as isotopies to produce a quasigroup $(Q, \circ)$. It appears that $(Q, \circ)$ will have a large multiplication group and not much structure. $(Q, \circ)$ would probably have even less structure if it was created by isotopy from a non-abelian group. However, this conjecture has not yet been verified with computational results.

5.2.3 Complete Mappings

Recall that in Section 4.2, a proof was given of Paige's result concerning which finite abelian groups are admissible. The method used in the proof involves beginning with any function (not necessarily a bijection) and constructing a complete mapping from this function. This method is completely deterministic. Therefore, it could be coded and used to produce complete mappings and hence quasigroups. It appears that this construction will usually produce a non-linear complete mapping. Therefore, this may be another promising method for creating a large, highly non-associative quasigroup with very little structure. Again, this conjecture needs to be verified with computational results.

CHAPTER 6. CONCLUSIONS

The original idea of creating a message authentication code based on the properties of Latin squares and quasigroups can be attributed to Denes and Keedwell in (4). In this dissertation, a new message authentication code, QMAC, is developed whose security is based on the non-associativity of quasigroups. Recall that in QMAC, only the multiplication order of the quasigroup elements is secret. In particular, the quasigroup $(Q, \circ)$ can be made public. Because of this design, QMAC appears to be an improvement on Denes and Keedwell's MAC. Several methods for implementing QMAC were discussed and their time and space requirements were analyzed. We also addressed issues related to the keyspace and the security of QMAC.

In order for QMAC to be useful, it is necessary for the underlying quasigroup to have certain properties. In particular, $(Q, \circ)$ must be large and highly non-associative. We explored various methods for efficiently creating and storing large quasigroups—isotopies, complete mappings, and T-functions. Isotopies and complete maps appear to be the most promising methods, especially if non-affine functions are used. In order to determine "how non-associative" a quasigroup was, the size of its multiplication group was considered. If the multiplication group was large, we said that the quasigroup was highly non-associative. We then looked at methods for creating highly non-associative quasigroups, including non-linear isotopies and complete mappings. One particularly promising method for creating highly non-associative quasigroups involves using a Feistel network to produce a non-linear isotopy.

There are still questions which need to be answered about QMAC and the methods discussed in this dissertation. The most pressing issue involves further study regarding the security of QMAC. Currently, we only know that QMAC is secure against the attack given in

Section 3.3.3. Other attacks still need to be developed. QMAC either needs to be demonstrated secure against this attack or the specifications of QMAC need to be changed in order to make it secure. In particular, we need to examine whether structure present in $(Q, \circ)$ poses a security risk to QMAC. If so, then other methods need to be developed for creating the quasigroup used in QMAC.

As discussed in Chapters 4 and 5, there are several methods for developing quasigroups without much structure which appear to have potential. These include creating $(Q, \circ)$ by non-linear isotopy from a non-abelian group and creating $(Q, \circ)$ using a non-linear complete mapping. Both these methods need to be explored further.

Finally, we would like to examine other useful methods for determining "how non-associative" a quasigroup is.

BIBLIOGRAPHY

[1] O. Chein, H. Pflugfelder, and J. D. H. Smith, *Quasigroups and loops: theory and applications*, Heldermann Verlag, Berlin, 1990.

[2] E. Dawson, D. Donovan, and A. Offer, *Quasigroups, isotopisms, and authentication schemes*, Australasian Journal of Combinatorics, 13 (1996), p. 75.

[3] J. Denes and A. Keedwell, *Latin squares and their applications*, Academic Press, Inc., Budapest, 1974.

[4] ——, *A new authentication scheme based on latin squares*, Discrete Mathematics, 106/107 (1992), p. 157.

[5] S. Even and O. Goldreich, *Des-like functions can generate the alternating group*, IEEE Transactions on Information Theory, 29 (1983), p. 863.

[6] A. Klimov and A. Shamir, *A new class of invertible mappings*, Lecture Notes in Computer Science, 2523 (2002), p. 470.

[7] A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of applied cryptography*, CRC Press, Boca Raton, FL, 1997.

[8] L. J. Paige, *A note on finite abelian groups*, Bulletin of the American Mathematical Society, 53 (1947), p. 590.

[9] B. Preneel, *The state of cryptographic hash functions*, Lectures on Data Security, Lecture Notes in Computer Science, 1561 (1999), p. 158.

[10] A. Sade, *Quasigroupes automorphes par le groupe cyclique*, Canadian Journal of Mathematics, 9 (1957), p. 321.

[11] B. Schneier, *Applied cryptography*, John Wiley & Sons, Inc., New York, NY, 1996.

[12] J. D. H. Smith, *An introduction to quasigroups and their representations*, CRC Press, to appear.

[13] J. H. van Lint and R. M. Wilson, *A course in combinatorics*, Cambridge University Press, New York, NY, 1992.

ACKNOWLEDGEMENTS

There were many times when I thought I would never see the end of this road. Now that it actually lies before me, there are so many people I need to thank for helping me to get here.

Thanks to Dr. Cliff Bergman for all his guidance, reassurance, and support through the research and writing of this dissertation. His insights and encouragement helped to keep me on track and showed me that I actually could enjoy doing research. Thanks also to my committee members for their assistance and contributions to this work: Drs. Jennifer Davidson, Roger Maddux, Jonathan D.H. Smith, and Sung Yell Song.

An enormous "thank you" is also due to my family and friends. Thanks so much for all of your support and encouragement. If you hadn't been there for me to lean on and complain to, you wouldn't be reading this. I am so very blessed that God has placed all of you in my life.