MAC Constructions: Security Bounds and Distinguishing Attacks

by

Avradip Mandal

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Combinatorics & Optimization

Waterloo, Ontario, Canada, 2007

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

We provide a simple and improved security analysis of **PMAC**, a Parallelizable MAC (Message Authentication Code) defined over arbitrary messages. A similar kind of result was shown by Bellare, Pietrzak and Rogaway at Crypto 2005, where they have provided an improved bound for **CBC** (Cipher Block Chaining) MAC, which was introduced by Bellare, Killan and Rogaway at Crypto 1994. Our analysis idea is much more simpler to understand and is borrowed from the work by Nandi for proving *Indistinguishability* at Indocrypt 2005 and work by Bernstein. It shows that the advantage for any distinguishing attack for n–bit PMAC based on a random function is bounded by $\mathrm{O}(\frac{\sigma q}{2^n})$, where σ is the total number of blocks in all q queries made by the attacker. In the original paper by Black and Rogaway at Eurocrypt 2002 where PMAC was introduced, the bound is $\mathrm{O}(\frac{\sigma^2}{2^n})$.

We also compute the collision probability of CBC MAC for suitably chosen messages. We show that the probability is $\Omega(\ell q^2/N)$ where ℓ is the number of message blocks, N is the size of the domain and q is the total number of queries. For random oracles the probability is $O(q^2/N)$. This improved collision probability will help us to have an efficient distinguishing attack and MAC-forgery attack. We also show that the collision probability for PMAC is $\Omega(q^2/N)$ (strictly greater than the birthday bound). We have used a purely combinatorial approach to obtain this bound. Similar analysis can be made for other CBC MAC extensions like XCBC, TMAC and OMAC.

Acknowledgements

I would like to thank my supervisor Dr.Andris Ambainis who helped me in every aspect during my masters degree. This work is done with Dr.Mridul Nandi. I would like thank him for helping me writing the thesis, and for introducing me to this exciting area of research. I would also like to thank Dr.Alfred Menezes and Dr.Douglas Stinson who agreed to become reader of this thesis and gave me their valuable comments.

Contents

1 Introduction **1**

1.1 Cryptography 1
1.2 The objectives of cryptography 2
1.3 Attacks 3
1.4 Provable security 5
1.5 Message Authentication Codes 6
1.6 Our work and previous results 8
1.7 Chapter outline 9

2 Cryptography basics **11**

2.1 Message Authentication Codes (MAC) and its security notions 11
2.2 Distinguishing attacks 14
2.2.1 Different notions of distances and their cryptographic significance . 14
2.2.2 Distinguisher of families of functions or random functions 18
2.2.3 A note on uniform random functions 21

3 A note on graph theory **22**

3.1 Directed graphs 22
3.2 Function graphs 24

4 An attack on CBC-MAC **27**

5 An attack on PMAC **36**

6 Improved security analysis of PMAC **40**
6.1 Definition of PMAC 40
6.2 Improved security analysis of PMAC 42

7 Conclusion **49**

Bibliography **51**

List of Figures

1.1 Message Authentication Code 7

2.1 CBC MAC 13
2.2 PMAC with $\ell = 2$ 14

3.1 Unicycle function graph 24
3.2 Tree and unicycle function graphs 26

4.1 Possible function graphs for estimating $|\mathcal{F}_{i,j,k}|$ 31
4.2 Possible function graphs for estimating $|\mathcal{F}_{i,j,k,m}|$ 35

6.1 PMAC 42

Chapter 1

Introduction

1.1 Cryptography

Cryptography is the science of securing communications. The fundamental and classical task of cryptography is to provide confidentiality by encryption methods. The message to be transmitted – it can be text, numerical data, an executable program or any other kind of information – is called plaintext. Alice encrypts the plaintext m and obtains ciphertext c. The ciphertext c is transmitted to Bob. Bob turns the ciphertext back into the plaintext by decryption. To decrypt, Bob needs some secret information, a secret decryption key. The adversary Eve still may intercept the ciphertext. However, the encryption should guarantee secrecy and prevent Eve from deriving any information about the plaintext from the observed ciphertext.

Encryption is very old. For example, the Caesar's shift cipher, where each plaintext character is replaced by the character 3 to the right modulo 26, (i.e. a is replaced by d, b by e,..., y by b, and z by c) is more than 2000 years ago. Every encryption method provides an encryption algorithm E and a decryption algorithm D. In symmetric-key cryptography both algorithms depend on the same secret key k. Caesar's cipher is an example of symmetric-key encryption scheme where the key is the offset 3. The Data Encryption Standard (DES) is another example of symmetric-key encryption.

In 1976 W. Diffie and M.E. Hellman published their famous paper, *New directions in Cryptography* [17]. There they introduced the revolutionary concept of public-key cryp-

tography. They also provided a solution to the long standing problem of key exchange and pointed the way to digital signatures. *Public-key encryption* methods are asymmetric. Each recipient of messages has his personal key $k = (pk, sk)$, consisting of two parts: pk is the encryption key and is made public, sk is the decryption key and is kept secret. If Alice wants to send a message m to Bob, she encrypts m by use of Bob's publicly known encryption key pk. Bob decrypts the ciphertext by use of his decryption key sk, which is known only to him.

Mathematically speaking public-key encryption is a so-called *one-way function* with a *trapdoor*. Anyone can easily encrypt a plaintext using the public key pk, but the other direction is difficult. It is practically impossible to deduce the plaintext from the ciphertext, without knowing the secret key sk (which is called the trapdoor information).

Public-key encryption methods require more complex computations and are less efficient than classical symmetric methods. Thus symmetric methods are used for the encryption of large amounts of data. Before applying symmetric encryption, Alice and Bob have to agree on a key. To keep this key secret, they need a secure communication channel. It is common practice to use public-key encryption for this purpose.

1.2 The objectives of cryptography

Providing confidentiality is not the only objective of cryptography. Cryptography is also used to provide solutions to other problems.

1. **Data integrity**: The receiver of a message should be able to check whether the message was modified during transmission, either accidentally or deliberately. No one should be able to substitute a false message for the original message, or for part of it.

2. **Authentication**: The receiver of a message should be able to verify its origin. No one should be able to send a message to Bob and pretend to be Alice (*data origin authentication*). When initiating a communication, Alice and Bob should be able to identify each other (*entity authentication*).

3. **Non-repudiation**: The sender should not be able to later deny that she sent a message.

If messages are written on paper, the medium – paper – provides security against manipulation. Handwritten personal signatures are intended to guarantee authentication and non-repudiation. If electronic media are used, the medium itself provides no security at all, since it is easy to replace some bytes in a message during its transmission over a computer network, and it is particularly easy if the network is publicly accessible, like the internet.

So, while encryption has a long history, the need for techniques providing data integrity and authentication resulted from the rapidly increasing significance of electronic communication.

There are symmetric as well as public-key methods to ensure the integrity of messages. *Digital signatures* require public-key methods. As with classical handwritten signatures, they are intended to provide authentication and non-repudiation. Note that non-repudiation is an indispensable feature if digital signatures are used to sign contracts. Digital signatures depend on the secret key of the signer – they can generated only by him. On the other hand, anyone can check whether a signature is valid, by a publicly known verification algorithm *Verify*, which depends on the public key of the signer. It is common not to sign the message itself, but to apply a *cryptographic hash function* first and then sign the hash value. Digital Signatures depend on the message. Different messages generate different signatures. So they can also be used to provide message authentication.

The symmetric-key method to ensure integrity of messages is achieved by *Message Authentication Codes* (MAC), which we will discuss in Section 1.5 in more detail.

1.3 Attacks

The primary goal of cryptography is to keep the plaintext secret from eavesdroppers trying to get some information about the plaintext. As discussed before, adversaries may also be active and try to modify the message. Then cryptography is expected to guarantee the integrity of messages. Adversaries are assumed to have complete access to the communication channel.

Cryptanalysis is the science of studying attacks against cryptographic schemes. Successful attacks may, for example, recover the plaintext (or parts of the plaintext) from the ciphertext, substitute parts of the original message, or forge digital signatures. Cryptography and cryptanalysis are often subsumed by the more general term cryptology.

A fundamental assumption in cryptanalysis was first stated by A. Kerkhoff in the nineteenth century, and is usually referred to as Kerkhoff's principle. It states that the adversary knows all the details of the cryptosystem, including algorithms and their implementation. According to this principle, the security of a cryptosystem must be entirely based on the secret keys.

Attacks on the secrecy of an encryption scheme try to recover plaintexts from ciphertexts, or even more drastically the secret key. In the following we only consider a passive attacker Eve, who does not try to modify the messages. However the attacker has access to plaintexts and ciphertexts, and she may have control over choosing plaintexts and/or ciphertexts. Of course she does not have access to the secret key. The possible attacks depend on the actual resources of Eve. They are usually classified as follows:

1. **Ciphertext-only attack**: Eve has the ability to obtain ciphertexts. This is likely to be the case in any encryption scenario. Even if Eve cannot perform other more sophisticated attacks, one must assume that she can get access to the encrypted messages. An encryption method that cannot resist a ciphertext-only attack is completely insecure.

2. **Known-plaintext attack**: Eve has the ability to obtain plaintext-ciphertext pairs. Using the information from these pairs, she attempts to decrypt a ciphertext for which she does not have the plaintext.

3. **Chosen-plaintext attack**: Eve has the ability to obtain ciphertexts for plaintexts of her choosing. Then she attempts to decrypt a ciphertext for which she does not have the plaintext. Here she has access to the encrypting device only once. This means after she starts analysis, she cannot access the encrypting device any more.

4. **Adaptively-chosen-plaintext attack**: This is the same as the previous attack, except now Eve may do some analysis on the plaintext-ciphertext pairs, and subsequently, get more pairs. She may switch between gathering pairs and performing the

analysis as often as she likes. This means that she has either lengthy access to the encrypting device or can somehow make repeated use of it.

5. **Chosen- and adaptively-chosen ciphertext attack**: These two attacks are similar to the above plaintext attacks. Eve can choose ciphertexts and gets the corresponding plaintexts. She has access to the decryption device.

1.4 Provable security

It is desirable to design cryptosystems that are provably secure. Provably secure means mathematical proofs show that the cryptosystem resists certain types of attacks. Pioneering work in this field was done by C.E. Shannon. In his information theory, he developed a measurement for the amount of information associated with a message and the notion of perfect secrecy. A *perfectly secret* cipher perfectly resists all ciphertext-only attacks. An adversary gets no information whatsoever about the plaintext, even if his resources in computational power and time are unlimited. *Vernam's one-time pad* which encrypts a message m by XORing it bitwise with a truly random bit string, is the most famous perfectly secret cipher. It even resists all the passive attacks mentioned. This can be mathematically proven by Shannon's theory. Unfortunately Vernam's one-time pad and all perfectly secret ciphers are usually impractical. It is not practical in most situations to generate and handle truly random bit sequences of sufficient length as required for perfect secrecy.

More recent approaches to provable security therefore abandon the ideal of perfect secrecy and the unrealistic assumption of unbounded computing power of adversary. Only attacks that might be *feasible* are taken into account. Feasible means that the attacks can be performed by an *efficient algorithm*. Certainly attacker algorithms with non-polynomial running times are not efficient. Conversely algorithms with polynomial running times are often considered efficient ones. If the attacker uses probabilistic algorithms then average running times are taken into account.

The security of a public-key cryptosystem is based on the hardness of some computational problem. For example, the secret keys of an RSA scheme could be easily deduced if computing the factors of a large integer was possible. However, it is believed that factoring

large integers is infeasible. There are no mathematical proofs for the hardness of the computational problems used in public-key systems. Therefore, security proofs for public-key methods are always conditional. They depend on the validity of underlying assumptions.

1.5 Message Authentication Codes

A message authentication code (MAC) function computes a MAC from a message and a secret key. If the originator and the receiver share knowledge of that secret key, the receiver can calculate the same function of the message and secret key and see if it matches the MAC accompanying the message. If the MAC matches, then the receiver knows, within the strength of the MAC function and key, that somebody with possession of the secret key produced the MAC. Of course, every receiver that can verify the MAC needs to know this secret key. Thus all the holders of that secret key can create valid MACs even if they should only receive and verify these codes. So a MAC is a symmetric-key method to ensure data integrity and authenticity.

A difficulty with MAC authentication in a system with multiple originators and receivers is that we must choose between two strategies, both of which have problems:

1. We could have a different secret for every pair of entities. This method is logistically difficult because the number of keys increases with the square of the number of entities and the keys must be securely distributed. If the system includes E number of entities, we should have $E(E-1)/2$ secret keys.

2. Share one secret among all the entities. This technique is relatively insecure. The more entities that have a secret, the more likely the secret is to be compromised due to loss, subversion, or betrayal. This technique also means the same secret will be used many times; the more exposures of the uses of a secret, the easier an adversary may find it to break that secret analytically. In addition, with this strategy any of the entities can forge messages from any of the other entities and a recipient will be unable to detect this fraud based on the MAC.

MACs are usually implemented through keyed hash functions. Usually a MAC is a public algorithm with a secret compression function. In other words the secret key deter-

mines which compression function we should use among a family of functions. The family can be a family of *random functions* or a family of *random permutations*.

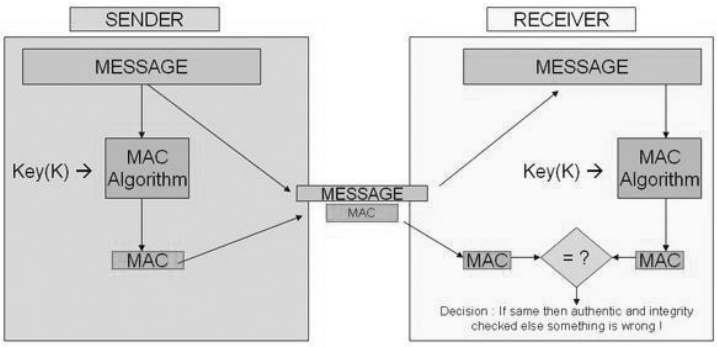

Figure 1.1: Message Authentication Code

There are various MAC algorithms that are used in practice. A few of them are as follows.

1. CBC MAC : *Cipher Block Chaining* (**CBC**) MACs are implemented by passing the data through a block cipher and serially XORing the output with next block of data (Figure 2.1).

2. DAC : The *Data Authentication Algorithm* (**DAA**) is a former U.S. government standard for producing cryptographic message authentication codes. According to the standard, a code produced by the DAA is called a *Data Authentication Code* (**DAC**). The algorithm is not considered secure by today's standards. The DAA is equivalent to CBC-MAC, with DES as the underlying cipher, truncated to between 24 and 56 bits (inclusive).

3. UMAC : A *message authentication code based on universal hashing* (**UMAC**), is a type of message authentication code (MAC) calculated choosing a hash function from a class of hash functions according to some secret (random) process and applying it to the message. The resulting digest or fingerprint is then encrypted to hide the identity of the

hash function used. As with any MAC, it may be used to simultaneously verify both the data integrity and the authenticity of a message. A UMAC has provable cryptographic strength and is usually a lot less computationally intensive than other MACs.

4. HMAC : A *keyed-hash message authentication code* (**HMAC**), is a type of message authentication code (MAC) calculated using a cryptographic hash function in combination with a secret key. As with any MAC, it may be used to simultaneously verify both the data integrity and the authenticity of a message. Any iterative cryptographic hash function, such as MD5 or SHA-1, may be used in the calculation of an HMAC; the resulting MAC algorithm is termed HMAC-MD5 or HMAC-SHA-1 accordingly. The cryptographic strength of the HMAC depends upon the cryptographic strength of the underlying hash function, on the size and quality of the key and the size of the hash output length in bits.

5. CMAC : *Cipher-based MAC* (**CMAC**) is a block cipher-based message authentication code algorithm, it may be used to provide assurance of the authenticity and, hence, the integrity of binary data. This mode of operation fixes security deficiencies of CBC-MAC (CBC-MAC is secure only for fixed-length messages).

The core of the CMAC algorithm is a variation of CBC-MAC that Black and Rogaway proposed and analyzed under the name XCBC [5]. The XCBC algorithm efficiently addresses the security deficiencies of CBC-MAC. Iwata and Kurosawa proposed an improvement of XCBC and named the resulting algorithm One-Key CBC-MAC (OMAC) in [9]. They later submitted OMAC1, a refinement of OMAC, and additional security analysis. The OMAC1 variation efficiently reduces the key size of XCBC. CMAC is equivalent to OMAC1.

6. PMAC : *Parallelizable MAC* (**PMAC**) was introduced by J. Black and P. Rogaway [6]. All the incoming data blocks are passed through block ciphers parallelly, essentially reducing the processing time (Figure 6.1).

1.6 Our work and previous results

PMAC, a Parallelizable MAC [6] was introduced by J. Black and P. Rogaway, who showed that the security of PMAC can be upper bounded by $O(\frac{\sigma^2}{N})$ against chosen plain-text at-

tacker. In loose terms this means that if the attacker makes q queries, each with length $\ell_i n$, $i = 1, \ldots, q$, then PMAC is secure against any attacker (even computationally unbounded ones) with $\sigma^2 = O(N)$, where $N = 2^n$, and $\sigma = \Sigma_{i=1}^{q} \ell_i$. Here n is the block length of PMAC. If all the queries have the same length ℓn, then the attacker cannot succeed with $O(\frac{\sqrt{N}}{\ell})$ queries.

In our work, we have improved this upper bound. We got an upper bound of security as $O(\frac{q\sigma}{N})$ for PMAC based on random functions. This means the attacker cannot succeed even with $O(\sqrt{\frac{N}{\ell}})$ queries, where each query is of length ℓn. A similar kind of result was shown by Bellare, Pietrzak and Rogaway [2] at Crypto 2005, where they have provided improved bounds for *Cipher Block Chaining* (**CBC**) MACs [3]. Our analysis idea is much simpler to understand and is borrowed from the work of Bernstein [4] and Nandi [13].

Our next work gives us attack algorithms for CBC-MAC and PMAC based on random functions. We calculate the *collision* probability for CBC-MAC and PMAC for a set of chosen plaintexts. Then we show that this collision probability leads to distinguishing attacks in both cases. In the case of CBC-MAC a collision also readily leads us to a chosen-plaintext attack or *forgery* attack. By collision we mean among the queries made by the attacker, at least two of them generate the same MAC value.

In more detail, in the case of CBC-MAC we showed for a chosen set of q messages of length ℓn, the collision probability is $\Omega(\frac{\ell q^2}{N})$, where $N = 2^n$. In the case of random oracles we know by the birthday bound that the collision probability is bounded by $O(\frac{q^2}{N})$. This improved probability leads to a distinguishing attack. Similarly in the case of PMAC we showed for a chosen set of q messages of length ℓn, the collision probability is $\Omega(\frac{q^2}{N})$, where $N = 2^n$. Though asymptotically the lower bound for the PMAC collision probability is the same as the upper bound for the birthday attack, we show that the PMAC collision probability is strictly greater than the birthday bound. In fact we show the probability difference can be bounded by $\Omega(\frac{q^2}{N})$. This also leads to a distinguishing attack.

1.7 Chapter outline

The chapters are organized as follows.

1. Introduction : A brief introduction to cryptography and our work.

2. Cryptography basics : Here we introduce Message Authentication Code (MACs) and their security notions. Then we discuss random functions, different notions of distances between random variables, and why these are significant with respect to distinguishing attacks.

3. A note on graph theory : We state some graph theoretical preliminaries which later become significant for the analysis of our attack on CBC-MAC.

4. An attack on CBC-MAC : We provide a detailed analysis of our attack on CBC-MAC.

5. An attack on PMAC : We provide a detailed analysis of our attack on PMAC.

6. Improved security analysis of PMAC : We state our modified definition of PMAC and provide its security analysis.

7. Conclusion : We discuss the significance of our work and propose some relevant open research areas.

Chapter 2

Cryptography basics

2.1 Message Authentication Codes (MAC) and its security notions

Definition of MAC

A MAC is a family of functions $\{F_k\}_{k\in\mathcal{K}}$ where $F_k : \mathcal{M} \to T$, $\mathcal{M}$ is the message space, T is the tag space, and $k \in \mathcal{K}$ is a secret key chosen uniformly from a key space. If $t = F_k(M)$ then t is called the *tag* of the message M. In this paper, we consider $T = \{0,1\}^n$ with a group addition $+$ and identity element $\mathbf{0}$, and $\mathcal{M} = \{0,1\}^{\leq L} \stackrel{\Delta}{=} \cup_{i\leq L} \{0,1\}^i$ for a sufficiently large integer L and a fixed integer n. A reasonable choice of parameters are $n = 128$ and $L = 2^{64}$.

Security Notions of MAC

There are two popular security notions for Message Authentication Code, namely security against *distinguishing attack* and security against *forgery attack*. The distinguishing attack is a weaker attack than forgery. In other words, if a construction is secure against distinguishing attacks then it is also secure against forgery attacks with at least the same security level. Thus, we mainly analyze the distinguishing attack security for PMAC.

1. Distinguishing Attack : Let Adversary $\mathcal{A}^O$ be an oracle algorithm where

- $O = F_k$, chosen uniformly from $\mathcal{F} = \{F_k : \mathcal{M} \to T;\ k \in \mathcal{K}\}$ (k is uniform on $\mathcal{K}$) or
- $O = F$, chosen uniformly from $\mathbf{Func}(\mathcal{M}, T) \stackrel{\Delta}{=} \{F;\ \ F : \mathcal{M} \to T\}$ (or **Func** only).

Remark 2.1. A *random function* is a probability distribution on $\mathbf{Func}(\mathcal{M}, T)$. If the distribution is uniform then we say that it is a *uniform random function*. Note that the uniform distribution on $\mathcal{K}$ induces a probability distribution on **Func**.

The adversary can make at most q queries to the oracle O adaptively consisting of at most σ many blocks and runs in time at most t. Finally, it returns either 1 or 0. The *advantage for distinguishing attack* is computed as follows :

$$\mathbf{Adv}_{\mathcal{F},\ \mathbf{Func}}(\mathcal{A}) \stackrel{\Delta}{=} \left|\ \mathbf{Pr}[\mathcal{A}^{\mathcal{F}} = 1] - \mathbf{Pr}[\mathcal{A}^{\mathbf{Func}} = 1]\ \right|,$$

$$\mathbf{Adv}_{\mathcal{F},\mathbf{Func}}(q, \sigma, t) \stackrel{\Delta}{=} \mathbf{max}_{\mathcal{A}}\ \mathbf{Adv}_{\mathcal{F},\ \mathbf{Func}}(\mathcal{A} : q, \sigma, t),$$

where the maximum is taken over all distinguishers $\mathcal{A}$ with runtime at most t making at most q queries consisting of at most σ many blocks. For simplicity, we also denote $\mathbf{Adv}_{\mathcal{F}}(\mathcal{A})$ and $\mathbf{Adv}_{\mathcal{F}}(q, \sigma, t)$ in place of $\mathbf{Adv}_{\mathcal{F},\ \mathbf{Func}}(\mathcal{A})$ and $\mathbf{Adv}_{\mathcal{F},\mathbf{Func}}(q, \sigma, t)$ respectively.

The definition of *block* is given later in Section 6.1. Intuitively, a padded message is divided in n-bit components which are called blocks.

If the advantage is high then the attacker $\mathcal{A}$ can distinguish the uniform random functions from functions in $\mathcal{F}$ with high probability. If it is negligible, we sometimes say that the family $\mathcal{F}$ is a pseudorandom function family.

2. MAC-forgery : In case of a MAC-forgery attack, an attacker makes successive queries M_i's to the oracle F_k (where k is secret and chosen uniformly from $\mathcal{K}$) and obtains responses $F_k(M_i)$'s. Let $(M_1, t_1 = F_k(M_1)), \cdots, (M_q, t_q = F_k(M_q))$ be all the *query-responses*. If the attacker can return a pair (M, t) such that $(M, t) \neq (M_i, t_i)$ for all i and t is a valid tag (i.e., $t = F_k(M)$) then we say that the attacker forges successfully. The probability of forging successfully a message-tag pair is the advantage for a MAC-forgery attack.

If one can forge a message (say (M, t)) using this forgery attacker, then one can also launch a distinguishing attack (same as the forgery attacker except at the end it will submit the query M and will check whether the response is t or not). Thus a forgery attacker is much stronger than a distinguishing attacker. Equivalently, security against distinguishing attack is a stronger notion than security against forgery attack.

Examples of MACs

In this section we will briefly describe CBC-MAC [3] and PMAC [6]. Later, in Chapters 4 and 5 we will study attacks on them. Let f be a function on a group $(D, +)$ (i.e, from $(D, +)$ to $(D, +)$) where $|D| = N$.

1. **CBC-MAC :** For a fixed $\ell \geq 1$, define the iterated functions recursively as follow :

   $$f^+(x_1, \cdots, x_\ell) :=: f^+_\ell(x_1, \cdots, x_\ell) := f(f^+_{\ell-1}(x_1, \cdots, x_{\ell-1}) + x_\ell),$$

   where $x_i \in D$, $f^+_0() :=: f^+_0(\lambda) := 0$ and λ is the empty string. We denote $f^+(x_1, \cdots, x_\ell)$ by $\mathbf{CBC}^f(x_1, \cdots, x_\ell)$.

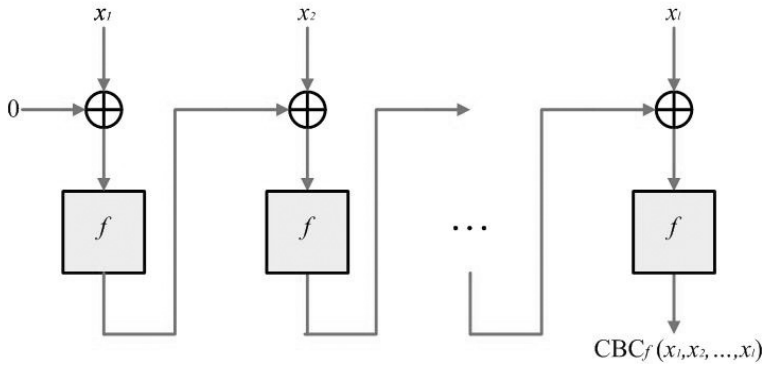

Figure 2.1: CBC MAC

2. **PMAC :** We consider a simpler PMAC definition for fixed-length messages. For a general definition of PMAC (for any size message input) see [6]. Let $(x_1, \cdots, x_{\ell-1}, x_\ell) \in D^\ell$ where each $x_i \in D$ and $\ell > 1$. Compute $w = \sum_{i=1}^{\ell-1} f(x_i + c_i \cdot f(0))$ where $c_1, \cdots, c_{\ell-1}$ are known constants from D. The output of PMAC is $f(w + x_\ell)$. For $\ell = 2$ (Figure 2.2), the value of PMAC at (x_1, x_2) is

   $$\mathbf{PMAC}(x_1, x_2) = f(x_2 + f(x_1 + c_1 f(0))).$$

   We will consider $\ell = 2$ in our PMAC attack algorithm and will calculate the collision probability for suitably chosen messages in Chapter 5.

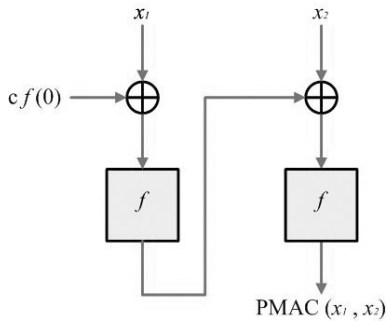

Figure 2.2: PMAC with $\ell = 2$

2.2 Distinguishing attacks

2.2.1 Different notions of distances and their cryptographic significance

(1) Statistical Distance :

Let X and Y be two random variables taking values on a finite set S. We define the *statistical distance* between X and Y by

$$\mathrm{d}_{\mathrm{stat}}(X, Y) := \max_{T \subset S} \left|\Pr[X \in T] - \Pr[Y \in T]\right|.$$

Note that

$$\Pr[X \in T] - \Pr[Y \in T] = \Pr[Y \notin T] - \Pr[X \notin T],$$

and hence

$$\mathrm{d}_{\mathrm{stat}}(X, Y) = \max_{T \subset S}(\Pr[X \in T] - \Pr[Y \in T]).$$

The statistical distance measures the distance between the distribution of the random variables. In fact, it is really a *metric* or *distance function* on the set of all distributions on S. It measures how close their distributions are. For identically distributed random

variables X and Y, $\mathrm{d}_{\mathrm{stat}}(X,Y) = 0$ and if the random variables are disjoint[1] then the statistical distance is one. In all other cases it lies between zero and one. Now we provide an equivalent definition of statistical distance and study some standard examples.

Lemma 2.2. $\mathrm{d}_{\mathrm{stat}}(X,Y) = \Pr[X \in T_0] - \Pr[Y \in T_0] = \frac{1}{2} \times \sum_{a \in S} \big|\Pr[X = a] - \Pr[Y = a]\big|$, *where* $T_0 = \{a \in S : \Pr[X = a] \geq \Pr[Y = a]\}$.

Proof. For T_0 as defined above, it is easy to see that

$$\sum_{a \in S} \big|\Pr[X = a] - \Pr[Y = a]\big| = 2 \times \big(\Pr[X \in T_0] - \Pr[Y \in T_0]\big).$$

For any $T \subset S$, $\quad 2 \times (\Pr[X \in T] - \Pr[Y \in T])$

$$= \textstyle\sum_{a \in T} \big(\Pr[X = a] - \Pr[Y = a]\big) - \sum_{a \notin T} \big(\Pr[X = a] - \Pr[Y = a]\big)$$
$$\leq \textstyle\sum_{a \in S} |\Pr[X = a] - \Pr[Y = a]|. \qquad \square$$

Example 2.3. Let X and Y be uniformly distributed on S and $T \subset S$ respectively. Then by Lemma 2.2,

$$\mathrm{d}_{\mathrm{stat}}(X,Y) = \frac{1}{2} \times \left((\frac{1}{|T|} - \frac{1}{|S|}) \times |T| + \frac{|S| - |T|}{|S|} \right) = 1 - \frac{|T|}{|S|}.$$

Thus, if the size of T is very close to the size of S, then the statistical distance is also very close to zero. On the other hand, if the size of T is negligible compare to that of S, then the statistical distance is close to one.

Example 2.4. Let $S = \mathrm{Func}(G,G)$ where $\mathrm{Func}(H,G)$ denotes the set of all functions from H to G. Let $T = \mathrm{Func}^{\mathrm{inj}}(G,G)$ be the subset containing all injective functions (or permutations, since the domain and range are identical). We say u (or v) is a *uniform random function* (or *uniform random injective function*) if it is uniformly distributed on S (or T respectively). Thus from Example 2.3 we know that

$$\mathrm{d}_{\mathrm{stat}}(u,v) = 1 - \frac{N!}{N^N}$$

which is very close to one for large N, where $|G| = N$.

[1] X and Y are said to be disjoint if there exists a subset T such that $\Pr[X \in T] = 1$ and $\Pr[Y \in T] = 0$

Example 2.5. Given any distinct and fixed $x_1, \cdots, x_k \in G$, let the *k-sampling output* of u be $(u(x_1),\cdots,u(x_k))$ and denoted as $u[k](x_1, \cdots, x_k)$. Let $X = (u(x_1), \cdots, u(x_k))$ and $Y = (v(x_1),\cdots,v(x_k))$. Here u and v are as in Example 2.4. Then we can see that X is uniformly distributed on $S = G^k$ and Y is uniformly distributed on $T = G[k] := \{(y_1, \cdots, y_k) \in G^k : y_i\text{'s are distinct}\}$ and hence (again by Example 2.3)

$$\mathrm{d}_{\mathrm{stat}}(X, Y) = 1 - \frac{N(N-1)\cdots(N-k+1)}{N^k} \approx 1 - \exp^{-k(k-1)/2N}.$$

Here we note that if $k \ll \sqrt{N}$ then the statistical distance is very close to zero.

Next we present two results which will help to give an upper bound on the statistical distance of two distributions. If the probability of the event $\{X = a\}$ is not small compared to that of $\{Y = a\}$ for all choices of a (or on a set with high probability) then the statistical distance is also small. More precisely, we have the following two lemmas.

Lemma 2.6. *Let X and Y be two random variables taking values on S, and let $\epsilon > 0$. If*

$$\Pr[X = a] \geq (1 - \epsilon) \times \Pr[Y = a]$$

for all $a \in S$, or if

$$\Pr[X = a] \leq (1 + \epsilon) \times \Pr[Y = a]$$

for all $a \in S$, then $\mathrm{d}_{\mathrm{stat}}(X, Y) \leq \epsilon$.

Proof. For any subset $T \subset S$,

$$\Pr[X \in T] \geq (1 - \epsilon) \times \Pr[Y \in T]$$

since $\Pr[X = a] \geq (1 - \epsilon) \times \Pr[Y = a]$ for all $a \in S$. So,

$$\Pr[Y \in T] - \Pr[X \in T] \leq \epsilon \times \Pr[Y \in T] \leq \epsilon$$

Thus, $\mathrm{d}_{\mathrm{stat}}(X, Y) \leq \epsilon$. The proof of the other case is similar. □

Lemma 2.7. *Let X and Y be two random variables taking values on S. Let $T \subset S$ be a subset such that*

$$\Pr[Y \notin T] \leq \epsilon_2 \text{ and}$$

$$\Pr[X = a] \geq (1 - \epsilon_1) \times \Pr[Y = a]$$

for all $a \in T$. Then $\mathrm{d}_{\mathrm{stat}}(X, Y) \leq \epsilon_1 + \epsilon_2$.

Proof. Let T_1 be the set

$$T_1 = \{a \in S : \Pr[Y = a] \geq \Pr[X = a]\}.$$

$$\begin{aligned}
\text{Now, } \mathrm{d}_{\mathrm{stat}}(X, Y) &= \Pr[Y \in T_1] - \Pr[X \in T_1] \\
&= \Pr[Y \in T_1 \cap T] - \Pr[X \in T_1 \cap T] + \Pr[Y \in T_1 \backslash T] - \Pr[X \in T_1 \backslash T] \\
&\leq \epsilon_1 \Pr[Y \in T_1 \cap T] + \Pr[Y \in T_1 \backslash T] \\
&\leq \epsilon_1 + \Pr[Y \notin T] \\
&\leq \epsilon_1 + \epsilon_2
\end{aligned}$$

□

(2) Computational Distance

The statistical distance is also popularly known as information-theoretic distance. In cryptography, there is another notion of distance, known as *computational distance*. Let $\mathcal{A}(\cdot)$ be a *probabilistic algorithm* which runs with an input $a \in S$ and giving outputs 0 or 1. Define the $\mathcal{A}$-*distance* between X and Y as follows:

$$\mathrm{d}^{\mathcal{A}}(X, Y) = \big|\Pr[\mathcal{A}(X) = 1] - \Pr[\mathcal{A}(Y) = 1]\big|.$$

Here, $\mathcal{A}(X)$ means the distribution of output of $\mathcal{A}(z)$ where z follows the distribution of X. Similarly for $\mathcal{A}(Y)$. As $\mathcal{A}$ is a probabilistic algorithm it can use a string r chosen from some set $\mathcal{R}$ with a distribution which is *independent* of X and Y. So we consider $\mathcal{A}$ as having two inputs $r \in \mathcal{R}$ and $z \in S$. We next state a fact which shows a relationship between statistical and computational distances.

Lemma 2.8. *For any* $\mathcal{A}$, $\mathrm{d}^{\mathcal{A}}(X, Y) \leq \mathrm{d}_{\mathrm{stat}}(X, Y)$. *Conversely, there exists an algorithm* $\mathcal{A}_0$ *such that* $\mathrm{d}^{\mathcal{A}_0}(X, Y) = \mathrm{d}_{\mathrm{stat}}(X, Y)$.

Proof. The output of $\mathcal{A}$ is completely determined by a pair (r, z), where r is the random string chosen from $\mathcal{R}$ and z is the input. Let $S_{r_0} = \{a \in S : \mathcal{A}(r_0, a) = 1\}$. Then

$$\begin{aligned}
\mathrm{d}^{\mathcal{A}}(X, Y) &= \big|\Pr[\mathcal{A}(r, X) = 1] - \Pr[\mathcal{A}(r, Y) = 1]\big| \\
&= \big|\textstyle\sum_{r_0 \in \mathcal{R}} \Pr[r = r_0]\big(\Pr[\mathcal{A}(r_0, X) = 1 \mid r = r_0] - \Pr[\mathcal{A}(r_0, Y) = 1 \mid r = r_0]\big)\big| \\
&= \big|\textstyle\sum_{r_0 \in \mathcal{R}} \Pr[r = r_0]\big(\Pr[\mathcal{A}(r_0, X) = 1] - \Pr[\mathcal{A}(r_0, Y) = 1]\big)\big|
\end{aligned}$$

$$= \left| \sum_{r_0 \in \mathcal{R}} \Pr[r = r_0] \big(\Pr[X \in S_{r_0}] - \Pr[Y \in S_{r_0}] \big) \right|$$
$$\leq \mathrm{d}_{\mathrm{stat}}(X, Y).$$

The equality holds if $S_{r_0} = T_0$ as in Lemma 2.2. Thus, on input z, $\mathcal{A}_0$ computes the probabilities $\Pr[X = z]$ and $\Pr[Y = z]$, and outputs 1 if $\Pr[X = z] \geq \Pr[Y = z]$, otherwise 0. Hence $\mathrm{d}^{\mathcal{A}_0}(X, Y) = \mathrm{d}_{\mathrm{stat}}(X, Y)$. □

In Lemma 2.8 note that algorithm $\mathcal{A}_0$ is not necessarily efficient and does not necessarily use any random strings. One can consider only deterministic algorithm in the situation when computational power is unbounded. Intuitively, one can perform the computation for all random choices and then choose the random string which yields best performance. We will show in the next section that we can ignore the random string while we distinguish two classes of functions by using unbounded computation.

2.2.2 Distinguisher of families of functions or random functions

In this section we examine how a distinguisher can behave. We also show how the advantage of the distinguisher can be obtained by computing the statistical distance between two possible *views* of the distinguisher.

By a random function we mean some distribution on the set $\mathrm{Func}(H, G)$, the set of all functions from H to G. In Example 2.4, we defined two families of random functions, namely uniform random function and uniform random injective function. In cryptography, these families are used as ideal candidates for some primitives. Now we follow the notations used in Examples 2.4 and 2.5. Let f be a random function. For each $\mathbf{x} = (x_1, \cdots, x_k) \in H[k]$, $f[k](\mathbf{x}) = (f(x_1), \cdots, f(x_k))$ follows the distribution induced by the distribution of f. More precisely, for any $\mathbf{y} = (y_1, \cdots, y_k) \in G^k$,

$$\Pr[f[k](\mathbf{x}) = \mathbf{y}] = \sum_{f_0 \in I} \Pr[f = f_0], \quad \text{where} \quad I := \{f \in \mathrm{Func}(H, G) : f[k](\mathbf{x}) = \mathbf{y}\}.$$

Let f and g be two random functions. Let $\mathcal{D}$ be a distinguisher that has a function oracle which is either f or g. The distinguisher behaves as follows.

1. First it chooses a random string r from $\mathcal{R}$.
2. Based on r it makes query $x_1 := x_1(r) \in H$ and obtains $y_1 \in G$.

3. Then it makes queries $x_2 = x_2(r, y_1) \in H$ and obtains $y_2 \in G$ and so on.

Even if x_2 depends on x_1, it is a function of r and y_1 since x_1 is a function of r only. Thus, x_i is a function of $(r, y_1, \cdots, y_{i-1})$. We say that these functions $x_1, x_2, \cdots$ are *query functions* (or $\mathbf{x} = (x_1, \cdots, x_k)$ is a k-query function) and the tuple $(y_1, \cdots, y_k) \in G^k$ is the *conditional view* of the distinguisher (conditioned on the random string r) where k is the number of queries. Note that the output of $\mathcal{D}$ is completely determined by the chosen random string r and the conditional view $(y_1, \cdots, y_k)$. We define the advantage of $\mathcal{D}$ to distinguish between f and g as

$$\mathrm{Adv}_{f,g}(\mathcal{D}) = |\Pr[\mathcal{D}^f = 1] - \Pr[\mathcal{D}^g = 1]|.$$

Define

$$\mathrm{d}_{f,g}(k) = \max_{\mathcal{D}} \mathrm{Adv}_{f,g}(\mathcal{D}),$$

where the maximum is taken over all oracle algorithms $\mathcal{D}$ which make at most k queries. This denotes the maximum distinguishing advantage for two random functions f and g where the attacker is making at most k queries. Note that there is no restriction on the computational resources of $\mathcal{D}$. We can think of $\mathcal{D}$ as a tuple of functions $(x_1, \cdots, x_k, \mathcal{A})$ where the x_i's are query functions and $\mathcal{A}$ is the final output function which takes $(r, y_1, \cdots, y_k)$ as input. The tuple $(y_1, \cdots, y_k)$ depends on the random string r and query functions (x_i's). Denote this view without the random string $(y_1, \cdots, y_k)$ by $f[k]_{r,x_1,\cdots,x_k}$ or $g[k]_{r,x_1,\cdots,x_k}$ (in short, $f[k]_{r,\mathbf{x}}$ or $g[k]_{r,\mathbf{x}}$) for the random function f and g respectively. So basically, $\mathcal{A}$ is distinguishing two families of random variables $\{f[k]_{r,x_1,\cdots,x_k}\}_{r\in\mathcal{R}}$ and $\{g[k]_{r,x_1,\cdots,x_k}\}_{r\in\mathcal{R}}$. Thus,

$$\begin{aligned}\mathrm{Adv}_{f,g}(\mathcal{D}) &= \Big|\sum_{r\in\mathcal{R}} \Pr[\mathcal{A}(r, f[k]_{r,\mathbf{x}}) = 1] \times \Pr[r] - \sum_{r\in\mathcal{R}} \Pr[\mathcal{A}(r, g[k]_{r,\mathbf{x}}) = 1] \times \Pr[r]\Big| \\ &= \sum_{r\in\mathcal{R}} \Pr[r] \times \mathrm{d}^{\mathcal{A}}(f[k]_{r,\mathbf{x}}, g[k]_{r,\mathbf{x}}) \\ &\leq \sum_{r\in\mathcal{R}} \Pr[r] \times \mathrm{d}_{\mathrm{stat}}(f[k]_{r,\mathbf{x}}, g[k]_{r,\mathbf{x}}).\end{aligned}$$

Hence, given any probabilistic distinguisher $\mathcal{D} = (x_1, \cdots, x_k, \mathcal{A})$ one can define a deterministic distinguisher $\mathcal{D}_0 = (x_1, \cdots, x_k, \mathcal{A}_0)$ such that $\mathrm{Adv}_{f,g}(\mathcal{D}) \leq \mathrm{Adv}_{f,g}(\mathcal{D}_0)$. Here, $\mathcal{D}_0$

chooses a random string r_0 with probability one (i.e., a deterministic algorithm) such that $\mathrm{d}_{\mathrm{stat}}(f[k]_{r,\mathbf{x}},\, g[k]_{r,\mathbf{x}}) = \max_{r\in\mathcal{R}} \mathrm{d}_{\mathrm{stat}}(f[k]_{r,\mathbf{x}}, g[k]_{r,\mathbf{x}})$ and $\mathcal{A}_0$ behaves as in Lemma 2.8.

Now we state two assumptions that will be used for the remainder of this thesis.

Assumption 1 (Distinguishers are deterministic) We assume that all distinguishing algorithms are deterministic. Thus, x_1 is a constant and x_i is a function of $(y_1, \cdots, y_{i-1})$ for $i \geq 2$.

Assumption 2 (Query functions are distinct) To avoid overly-complex notation we use the same notation x_i to denote the function as well as the output of the function. We will assume that all outputs of x_i's (or x_i as a functional value) are distinct (otherwise one can restrict to the set of distinct values of x_i).

Now we use the notation $f[k]_{x_1,\cdots,x_k}$ instead of $f[k]_{r,x_1,\cdots,x_k}$ to denote the view of the distinguisher. We can write that

$$d_{f,g}(k) = \max_{\mathbf{x}} \mathrm{d}_{\mathrm{stat}}(f[k]_{\mathbf{x}}, g[k]_{\mathbf{x}}),$$

where the maximum is taken over all k-query functions $\mathbf{x} = (x_1, \cdots, x_k)$. Thus, to obtain an upper bound on $d_{f,g}(k)$, it would be enough to bound $\mathrm{d}_{\mathrm{stat}}(f[k]_{\mathbf{x}}, g[k]_{\mathbf{x}})$ for each k-query function $\mathbf{x}$. The following theorem, due to D. J. Bernstein [4], shows how one can obtain this.

Theorem 2.9. *If* $\Pr[f[k](\mathbf{a}) = \mathbf{y}] \geq (1-\epsilon) \times \Pr[g[k](\mathbf{a}) = \mathbf{y}]$ *for each* $\mathbf{a} \in H[k]$ *and* $\mathbf{y} \in G^k$*, then for any* k*-query function* $\mathbf{x} = (x_1, \cdots, x_k)$*,* $\mathrm{d}_{\mathrm{stat}}(f[k]_{\mathbf{x}},\, g[k]_{\mathbf{x}}) \leq \epsilon$ *and hence* $\mathrm{d}_{f,g}(k) \leq \epsilon$*.*

Proof. $\Pr[f[k]_{x_1,\cdots,x_k} = (y_1, \cdots, y_k)]$

$= \Pr[f[k](a_1, \cdots, a_k) = (y_1, \cdots, y_k)]$ ($(a_1, \cdots, a_k)$ is uniquely determined by $(y_1, \cdots, y_k)$)

$\geq (1-\epsilon) \times \Pr[g[k](a_1, \cdots, a_k) = (y_1, \cdots, y_k)]$

$= (1-\epsilon) \times \Pr[g[k]_{x_1,\cdots,x_k} = (y_1, \cdots, y_k)]$, for all $(y_1, \cdots, y_k) \in G^k$.

The Theorem now follows from Lemma 2.6. □

Remark 2.10. The above theorem implies for any adversary with oracle access to f or g, the distinguishing advantage is at most ϵ.

2.2.3 A note on uniform random functions

Let $\mathcal{F}_G$ denote the set of all functions from G to G. Let f be a function randomly chosen from $\mathcal{F}_G$ with uniform distribution. Then we call $f : G \to G$ a uniform random function. Now we have the following straightforward but important properties of f.

1. For all $x, y \in G$, $\mathbf{Pr}[f(x) = y] = \frac{1}{|G|}$.
2. For all $x, y, x_i \in G, \text{such that } x \neq x_i \text{for all } i \in \{1, \ldots, k\}$, $\mathbf{Pr}[f(x) = y | f(x_1) = y_1, \ldots, f(x_k) = y_k] = \frac{1}{|G|}$.

In other words the second property tells us that knowing the function on some values does not help us guessing the output on a new value.

Chapter 3

A note on graph theory

In this chapter we collect some definitions and notions from graph theory that we will use in subsequent chapters.

3.1 Directed graphs

Let $G = (D, E)$ be a directed graph where $E \subset D \times D$ and $|D| = N$. Denote the number of edges by $e(G)$. The *degree* of a vertex v (denoted as $\mathbf{deg}(v)$) is the sum of *out-degree* (or $\mathbf{deg_{out}}(v)$) and *in-degree* (or $\mathbf{deg_{in}}(v)$) of the vertex where

$$\mathbf{deg_{out}}(v) = |\{u : (v, u) \in E\}| \text{ and } \mathbf{deg_{in}}(v) = |\{u : (u, v) \in E\}|.$$

Define $V(G) = \{v : \mathbf{deg}(v) > 0\}$. Denote the number of vertices with positive degree by $r(G)$ i.e., $|V(G)| = r(G)$. We denote $\Delta(G) = r(G) - e(G)$. This is an important parameter of a graph. The undirected graph that corresponds to a directed graph is the graph obtained by considering all directed edges as undirected. This undirected graph may contain a *self loop* and at most two *parallel edges*. For a connected undirected graph G, G is a *tree* if and only if $\Delta(G) = 1$. Also for a connected undirected graph G', G' contains exactly one cycle if and only if $\Delta(G') = 0$; these graphs are called *unicycle graphs*.

In this thesis we are interested in the following families of directed graphs:

1. A *straight line path* of length k is a directed graph $G = (D, E)$ where $E = \{(x_0, x_1), (x_1, x_2),$

$\cdots, (x_{k-1}, x_k)\}$ and $x_0, x_1, \cdots, x_k$ are distinct. Here x_0 is the *source node* and x_k is the *end point* of the straight line path, and $\Delta(G) = 1$.

2. A *cycle* is a directed graph $G = (D, E)$ where $E = \{(x_0, x_1), (x_1, x_2), \cdots, (x_{k-1}, x_k = x_0)\}$ and $x_0, x_1, \cdots, x_{k-1}$ are distinct. Here k can be 1, in which case the cycle consists of a single self loop (x_0, x_0) (note that $x_1 = x_0$). If G is a cycle, then $\Delta(G) = 0$.

3. An *s-unicycle* is a directed graph $G = (D, E)$ where E is union of a cycle C and $s_1 (\leq s)$ distinct straight line paths $P_1, \cdots, P_{s_1}$ whose end points are vertices of the cycle C. The paths P_i are not necessarily disjoint. Each straight line path contributes at most one node in $V(G)$ with in-degree zero. Thus there are at most s nodes in $V(G)$ with in-degree zero. If G is an s-unicycle, then $\Delta(G) = 0$.

Let $G_1 = (D, E_1)$ and $G_2 = (D, E_2)$ be two directed graphs. A function $\alpha : D \to D$ is an *isomorphism* from G_1 to G_2 if α is bijection and $(x, y) \in E_1$ if and only if $(\alpha(x), \alpha(y)) \in E_2$. If such a function exists we write $G_1 \cong G_2$ and say that G_1 and G_2 are isomorphic. Moreover for $A \subset D$, if α is the identity on A then α is called an *A-isomorphism* and we write $G_1 \cong_A G_2$. G_1 and G_2 are said to be *A-isomorphic*. If there is no such A-isomorphism then G_1 and G_2 are *non A-isomorphic*. The notion of A-isomorphism is not standard but we need this notion in this thesis.

Example 3.1. In Figure 3.1 all graphs are isomorphic but G_1 and G_2 are not A-isomorphic where $A = \{1, 2, 4, 5\}$. Clearly, G_1 and G_3 are A-isomorphic, where $A = \{1, 2, 4, 5\}$.

Lemma 3.2. *The number of directed graphs isomorphic to a given graph $G = (D, E)$ is at most $N(N-1)\cdots(N-r+1)$ (which is less than N^r), where $r = |V(G)|$. If $A \subset V(G)$ has size s, then the number of graphs A-isomorphic to G is at most $(N-s)(N-s+1)\cdots(N-r+1)$ (which is less than N^{r-s}).*

Proof. Let $G' = (D, E')$ be an isomorphic copy of G, and let $\alpha : G \to G'$ be an isomorphism. G' is completely determined by $V(G')$ and α (α determines uniquely the edge set E'). Now we can choose $V(G') \subset D$ in $\binom{N}{r}$ ways. For each choice there are at most $r!$ isomorphisms. Thus we have at most $\binom{N}{r} \times r!$ isomorphic copies of G. Similarly, we can prove for the second part. Note that if G' is an A-isomorphic copy of G and $A \subset V(G)$ then $A \subset V(G')$. Thus, we can choose $V(G')$ in $\binom{N}{r-s}$ ways and for each choice there are at most $(r-s)!$ isomorphisms. □

3.2 Function graphs

A directed graph $G = (D, E)$ is called a *function graph* if $(x, y_1), (x, y_2) \in E$ implies that $y_1 = y_2$. A partial function f on D can be uniquely characterized by a function graph and vice versa by the following rule:

$$f(x) = y \text{ if and only if } (x, y) \in E.$$

Define the domain of a function graph G as

$$\mathbf{Dom}(G) = \{v : \mathbf{deg_{out}}(v) > 0\}.$$

Since it is a function graph, $\mathbf{Dom}(G) = \{v : \mathbf{deg_{out}}(v) = 1\}$. Moreover $\mathbf{Dom}(G)$ is the domain of the corresponding partial function. Clearly, $|\mathbf{Dom}(G)| = e(G)$ (map an edge $(v, w) \in E$ to $v \in \mathbf{Dom}(G)$).

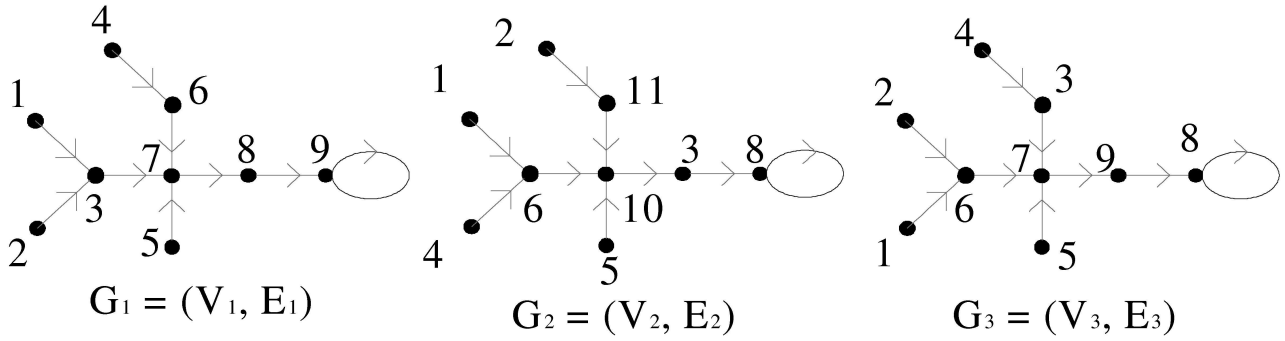

Figure 3.1: Unicycle function graph

Example 3.3. The graphs in Figure 3.1 are function graphs since there are no nodes with out-degree greater than one. The partial function corresponding to G_1 is $f(1) = 3, f(2) = 3, f(3) = 7, f(4) = 6, f(5) = 7, f(6) = 7, f(7) = 8, f(8) = 9, f(9) = 9$. The domain of the graph G_1 is $\mathbf{Dom}(G_1) = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

A function f is called *compatible* with a function graph $G = (D, E)$ if $f(x) = y$ whenever $(x, y) \in E$. Note that for a given function graph G there are N^{N-e} compatible functions $f : D \to D$ with G where $e = |e(G)| = |\mathbf{Dom}(G)|$, namely

$$f(x) = \begin{cases} y, & \text{if } (x, y) \in E \text{ (in other words } x \in \mathbf{Dom}(G)), \\ *, & \text{otherwise.} \end{cases} \tag{3.1}$$

Here " * " means that the function is defined arbitrarily. Now we know that, given an s-set $A \subset \mathbf{Dom}(G)$, there are at most N^{r-s} many A-isomorphic graphs where $r = |V(G)|$. Thus,

$$|\mathcal{F}_G = \{f : D \to D : f \text{ is compatible with some } G' \cong_A G\}| \leq N^{N-s+\Delta},$$

where $\Delta = \Delta(G)$.

Now we consider a special class of function graphs called (ℓ, A)-*iterated function graphs* where $A \subset D$ is an s-set. A function graph G is called an (ℓ, A)-iterated graph (or function graph) if there exists a function f such that the domain of G is

$$\mathbf{Dom}(G) = \{y : f^{(i)}(x) = y, x \in A,\ 0 \leq i \leq \ell - 1\}.$$

We denote by $G_{\ell,A}[f]$ the (ℓ, A)-iterated function graph for the function f (this is unambiguous since the graph is completely determined the tuple (ℓ, A, f)). For $A = \{x\}$, we sometime write $G_{\ell,x}[f]$. Thus $G_{\ell,A}[f]$ is the union (not necessarily disjoint) of $G_{\ell,x}[f]$, where x ranges over all elements in A. Now $G_{\ell,x}[f]$ can be one of the following :

1. $f^{(0)}(x) = x, f^{(1)}(x) = f(x), \cdots, f^{(\ell)}(x)$ are distinct. In this case the graph is a straight line path.

2. $f^{(0)}(x) = x = w_0, f^{(1)}(x) = f(x) = w_1, \cdots, w_\ell = f^{(\ell)}(x)$ are not distinct; that is $w_0, \cdots, w_i$ are distinct for some $i < \ell$ and $w_{i+1} = w_j$, for some $j \leq i$. This is a 1-unicycle (or sometimes called ρ straight line path as the structure looks like the letter ρ).

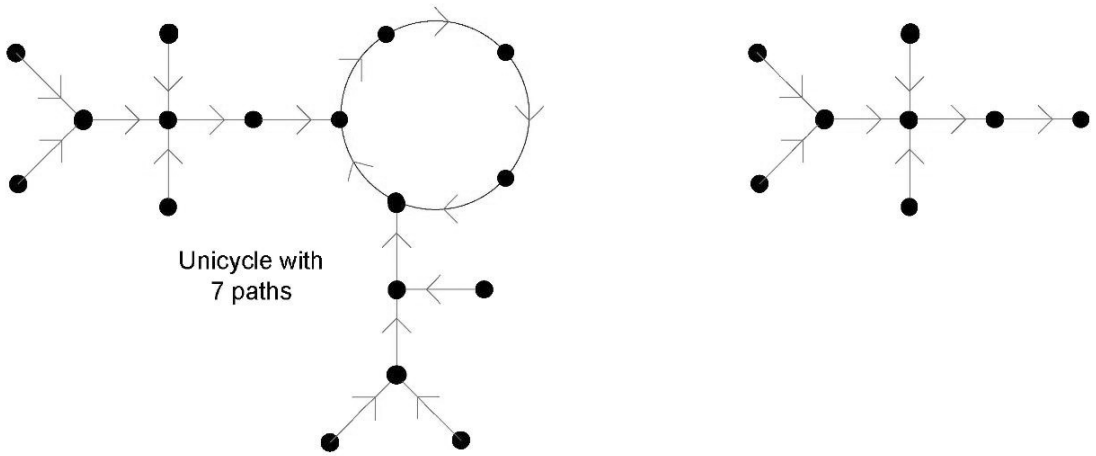

Figure 3.2: Tree and unicycle function graphs

Theorem 3.4. *$G_{\ell,A}[f]$ is the union of unicycles and straight line paths. More specifically, it is the disjoint union of trees and unicycle paths where the number of nodes in $V(G_{\ell,A}[f])$ with zero out-degree is at most $s = |A|$.*

Proof. We skip the proof as it is straightforward and needs some more notation. One can see the idea of the proof by examining different examples such as in Figure 3.2. □

Chapter 4

An attack on CBC-MAC

The only known attack on CBC-MAC is based on collisions. Suppose we know a collision for CBC-MAC, that is $X_1, X_2 \in D^*$ such that $\mathbf{CBC}(X_1) = \mathbf{CBC}(X_2)$. Then we know that $\mathbf{CBC}(X_1, x) = \mathbf{CBC}(X_2, x)$ for any $x \in D$. Thus we have the following forgery attack. Make query (X_1, x) and obtain the response t. Then $((X_2, x), t)$ is a valid message-tag pair. For distinguishing attack one can use this valid message-tag pair to distinguish. This forgery attack can also be mounted on PMAC and many other MACs including XCBC [5], TMAC [11] and OMAC [9].

In this chapter we will explore how efficiently (in terms of the number of queries) one can obtain a collision for CBC-MAC. We will estimate the collision probability more closely for suitably chosen messages. Similar techniques can be used for PMAC and other CBC-like constructions.

Let $X_i = (x_i, 0, \cdots, 0) \in D^\ell$ be ℓ-tuples such that the x_i's for $1 \leq i \leq q$, are pairwise distinct. Clearly, $\mathbf{CBC}^f(x_i, 0, \cdots, 0) = f^{(\ell)}(x_i)$, where the function $f^{(i)}(x)$ is defined as follows for $i \geq 0$:

$$f^{(0)}(x) = x \text{ and } f^{(i)}(x) = \overbrace{f \circ \cdots \circ f}^{i \text{ times}}(x) \text{ for } i \geq 1.$$

That is,

$$f^{(i)}(x) = f(f^{(i-1)}(x)).$$

We want to find a lower bound for the collision probability, i.e.,

$$\begin{aligned}
&\mathbf{Pr}_f[\mathbf{CBC}^f(X_i) = \mathbf{CBC}^f(X_j) \text{ for some } i \neq j] \\
&= \mathbf{Pr}_f[f^{(\ell)}(x_i) = f^{(\ell)}(x_j) \text{ for some } i \neq j] \\
&= \mathbf{Pr}_f[\bigcup_{1 \leq i < j \leq q} C_{i,j}] \\
&\geq \sum_{i<j} \mathbf{Pr}_f[C_{i,j}] - 3 \sum_{i<j<k} \mathbf{Pr}_f[C_{i,j,k}] - \frac{1}{2} \sum_{\substack{i<j,k<m \\ \{i,j\} \cap \{k,m\} = \emptyset}} \mathbf{Pr}_f[C_{i,j} \cap C_{k,m}] \qquad (4.1)
\end{aligned}$$

where $C_{i,j}$ denotes the event that $f^{(\ell)}(x_i) = f^{(\ell)}(x_j)$ and $C_{i,j,k}$ denotes the event that $f^{(\ell)}(x_i) = f^{(\ell)}(x_j) = f^{(\ell)}(x_k)$. The last inequality follows from Principle of Inclusion and Exclusion. Now for any event E,

$$\mathbf{Pr}_f[E] = \frac{|\{f : D \to D | E \text{ is true } \}|}{N^N},$$

since N^N is the total number of functions $f : D \to D$. Thus, this is the probability that a uniform random function on D (or a function chosen uniformly from the set of all functions) satisfies the event E. To have an estimate in Equation 4.1, we are interested in computing the following bounds.

1. $|\mathcal{F}_{i,j,k}| \leq 2\ell^2 N^{N-2} + 6\ell^6 N^{N-3}$, where

   $$\mathcal{F}_{i,j,k} = \{f : f^{(\ell)}(x_i) = f^{(\ell)}(x_j) = f^{(\ell)}(x_k)\}$$

   and x_i, x_j, x_k are distinct.

2. $|\mathcal{F}_{i,j,k,m}| \leq N^{N-2}\ell^2 + N^{N-3}(6\ell^4 + 4\ell^5) + 28\ell^8 N^{N-4}$, where

   $$\mathcal{F}_{i,j,k,m} = \{f : f^{(\ell)}(x_i) = f^{(\ell)}(x_j), f^{(\ell)}(x_k) = f^{(\ell)}(x_m)\}$$

   and x_i, x_j, x_k, x_m are distinct.

3. $|\mathcal{F}_{i,j}| \geq \ell N^{N-1} \exp(-\frac{4\ell^2}{N})$, where

   $$\mathcal{F}_{i,j} = \{f : f^{(\ell)}(x_i) = f^{(\ell)}(x_j)\},$$

   x_i and x_j are distinct and, $1 \leq \ell \leq \frac{N}{4} + \frac{1}{2}$.

To prove these three bounds we will use the properties of directed graphs that were introduced in Chapter 3. If $y = f(x)$ then (x, y) will be considered as an edge of a graph. The events $C_{i,j}$, $C_{i,j,k}$, $C_{i,j} \cap C_{k,m}$ can then be translated into a counting problem in graph theory, more precisely to compute the number of non-isomorphic graphs in a special class. We will describe these problems in more detail below. Now, we have the following main theorem of this chapter (by using Equation 4.1 and the three bounds listed above).

Theorem 4.1. *The collision probability,* $\mathbf{Pr}_f[\mathbf{CBC}^f(X_i) = \mathbf{CBC}^f(X_j)$ *for some* $i \neq j]$, *is at least*

$$\Delta := \binom{q}{2}\frac{\ell}{N}exp\left(-\frac{4\ell^2}{N}\right) - 3\binom{q}{3}\left(\frac{2\ell^2}{N^2} + \frac{6\ell^6}{N^3}\right) - \frac{1}{2}\binom{q}{2}\binom{q-2}{2}\left(\frac{28\ell^8}{N^4} + \frac{6\ell^4 + 4\ell^5}{N^3} + \frac{\ell^2}{N^2}\right)$$

For large N, *the above expression is* $\Omega(\frac{q^2\ell}{N}) - c(q, \ell, N)$ *when* $\frac{q^2\ell}{N} < \frac{8}{3}$ *and* $\ell = o(N^{\frac{1}{3}})$. *Also,* $c(q, \ell, N)$ *goes to zero when* $q^2\ell = O(N), \ell = o(N^{\frac{1}{3}})$ *and* N *is large.*

Remark 4.2. Bellare [2] proved that CBC-MAC based on random permutations is secure and the advantage is bounded by $O(\ell q^2/N)$ provided that $\ell = o(N^{1/3})$. Here we show that there is an attack on CBC based on random functions with advantage $\Omega(\ell q^2/N)$. The idea behind our attack cannot be easily extended to CBC-MAC based on random permutations. These observations indicate that CBC-MAC based on random permutations is more secure than that based on random functions.

Lower Bound of Probability of Collision Events

We first present the main ideas used to prove the aforementioned bounds.

Suppose that we want to find an upper bound on $\mathbf{Pr}[E]$, where E is some event related to uniform random functions and the probability is computed with respect to the uniform random functions. (Recall that each function $f \in \mathbf{Func}(D, D)$ has equal probability that is $\frac{1}{N^N}$.) To do so, we count the number (or give an upper bound) of functions $f \in \mathbf{Func}(D, D)$ such that the event E is true. Let $\mathcal{F} = \{f : E \text{ is true}\}$. Then $\mathbf{Pr}[E] = \frac{|\mathcal{F}|}{N^N}$. We then proceed as follows.

1. Associate each function $f \in \mathcal{F}$ to a function graph $G \in \mathcal{G}$ such that f is compatible with G. Here $\mathcal{G}$ is some classes of function graphs.

2. Now for a suitable choice of s-set A, partition $\mathcal{G}$ by A-isomorphism. That is, $\mathcal{G} = \bigsqcup_{i=1}^{L} \mathcal{G}_i$ where all elements within $\mathcal{G}_i$ are A-isomorphic and graphs belonging to two distinct classes are non-A-isomorphic. Let $\Delta_i = \Delta(G)$ where $G \in \mathcal{G}_i$.

3. Now we can derive an upper bound as follows. We know by Lemma 3.2 that $|\mathcal{F}_G| \leq N^{N-s+\Delta_i}$ for $G \in \mathcal{G}_i$. We denote $\mathcal{F}_G$ by $\mathcal{F}_{\mathcal{G}_i}$. Note that this is well defined. Since $\mathcal{F} \subseteq \bigsqcup_{i=1}^{L} \mathcal{F}_{\mathcal{G}_i}$, we have $|\mathcal{F}| \leq \sum_{i=1}^{L} N^{N-s+\Delta_i}$. If we know that there are L_i many classes whose Δ value is $i \geq 0$, then

$$|\mathcal{F}| \leq \sum_{i \geq 0} L_i \cdot N^{N-s+i}. \tag{4.2}$$

4. Consequently it suffices to suitably associate a function to a function graph, choose a suitable set A, and compute L_i for all possible values of i.

Upper Bound on $|\mathcal{F}_{i,j,k}|$

First consider $\mathcal{F}_{i,j,k}$. We define $A = \{x_i, x_j, x_k\}$ and associate each $f \in \mathcal{F}_{i,j,k}$ to the function graph $G_{\ell,A}[f] \in \mathcal{G}$ where $\mathcal{G} = \{G_{\ell,A}[f] : f \in \mathcal{F}_{i,j,k}\}$. Note that $G \in \mathcal{G}$ is either a tree (in which case $\Delta = 1$) or a 3-unicycle (in which case $\Delta = 0$). Now we have to count L_i, the number of non-A-isomorphic graphs for $\Delta \in \{0, 1\}$. As we can see in Figure 4.1 there are two possibilities.

1. $\Delta = 1$: G is a tree. Let x be the collision node, that is $f^{(\ell)}(x_i) = f^{(\ell)}(x_j) = f^{(\ell)}(x_k) = x$. Each tree is determined by the point of intersection of the x_j path and the x_i path (ℓ choices), and the point of intersection of the x_k path and union of the x_i and x_j paths (at most 2ℓ choices because there are at most 2ℓ edges in the union of x_i and x_j paths). Thus, $L_1 \leq \ell \cdot 2\ell = 2\ell^2$.

2. $\Delta = 0$: G is a 3-unicycle graph. Each 3-unicycle graph is determined by the length of the cycle, the distance from x_i to the cycle, the location of the point of intersection of x_j path and union of x_i path and the cycle, and the location of the point of

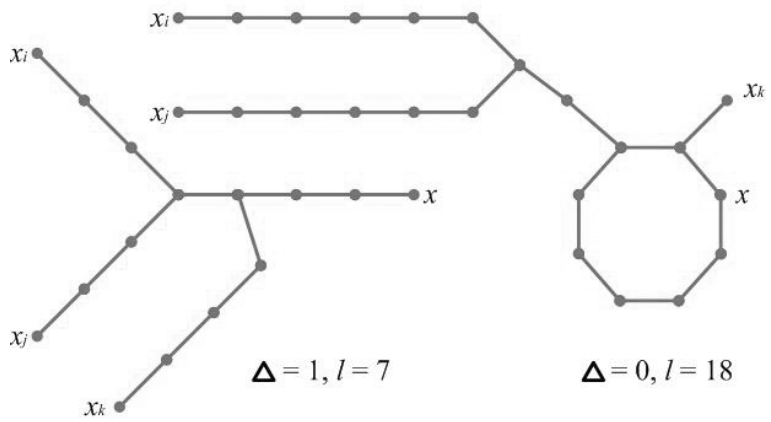

Figure 4.1: Possible function graphs for estimating $|\mathcal{F}_{i,j,k}|$

intersection of x_k path and union of x_i path, x_j path and the cycle. There are at most ℓ choices for each dependency, except for the location of the point of intersection (for x_k) which has at most 2ℓ choices. Thus $L_0 \leq \ell \cdot \ell \cdot (2\ell \cdot \ell) \cdot (3\ell \cdot \ell) = 6\ell^6$.

Thus, $|\mathcal{F}_{i,j,k}| \leq 2\ell^2 N^{N-2} + 6\ell^6 N^{N-3}$.

Upper Bound on $|\mathcal{F}_{i,j,k,m}|$

We define $A = \{x_i, x_j, x_k, x_m\}$ and associate each $f \in \mathcal{F}_{i,j,k,m}$ to the function graph $G_{\ell,A}[f] \in \mathcal{G}$ where $\mathcal{G} = \{G_{\ell,A}[f] : f \in \mathcal{F}_{i,j,k}\}$. As we can see in Figure 4.2, there are various possibilities for $G \in \mathcal{G}$ depending upon the value of $\Delta \in \{0, 1, 2\}$.

1. $\Delta = 2$: G is the disjoint union of two trees consisting of two paths each. Let $f^{(\ell)}(x_i) = f^{(\ell)}(x_j) = x$ and $f^{(\ell)}(x_k) = f^{(\ell)}(x_m) = y$. Each tree is determined by the point of intersection of the x_j and x_i paths (ℓ choices) and the point of intersection of the x_k and x_m paths (ℓ choices). Thus, $L_2 \leq \ell^2$.

2. $\Delta = 1$: G is either a tree consisting of four paths or the union of a 2-unicycle and a tree consisting of two paths. In this case $L_1 \leq \ell \cdot 2\ell \cdot \ell \cdot 3\ell + 2 \cdot \ell \cdot \ell \cdot \ell \cdot \ell \cdot 2\ell = 6\ell^4 + 4\ell^5$.

(When G is union of a 2-unicycle and a tree consisting of two paths, there are 2 choices whether x_i, x_j is in the tree or not, ℓ choices for the tree, ℓ choices for the cycle length, ℓ choices for the distance of x_k from the cycle, ℓ choices for the distance of x_m from the cycle, 2ℓ choices for the point of intersection of x_k and x_m path.)

3. $\Delta = 0$: G is either a 4-unicycle graph or the union of two disjoint 2-unicycle graphs. Hence $L_0 \leq 24\ell^8 + 4\ell^8 = 28\ell^8$.

Thus, $|\mathcal{F}_{i,j,k,m}| \leq N^{N-2}\ell^2 + N^{N-3}(6\ell^4 + 4\ell^5) + 28\ell^8 N^{N-4}$.

Lower Bound on $|\mathcal{F}_{i,j}|$

Let $\mathcal{F}_{i,j} = \{f : f^{(\ell)}(x_i) = f^{(\ell)}(x_j)\}$ for distinct $x_i, x_j \in D$, and let

$$\mathcal{F}_{i,j}^k = \{f \in \mathcal{F}_{i,j} : f^{(k)}(x_i) = f^{(k)}(x_j) \text{ and } f^{(k_1)}(x_i), f^{(k_2)}(x_j) \text{ are distinct } 0 \leq k_1, k_2 \leq k-1\}$$

where $1 \leq k \leq \ell$. That is $\mathcal{F}_{i,j}^k$ is the set of functions where all intermediate outputs are distinct before the kth round and at the kth round we have a collision. Clearly $\mathcal{F}_{i,j}^k$'s are disjoint sets and $\mathcal{F}_{i,j} \supseteq \bigsqcup_{k=1}^{\ell} \mathcal{F}_{i,j}^k$. So, $|\mathcal{F}_{i,j}| \geq \sum_{k=1}^{\ell} |\mathcal{F}_{i,j}^k|$. The following lemma is a straightforward counting argument.

Lemma 4.3. *For $2 \leq k \leq \ell$, $|\mathcal{F}_{i,j}^k| = (N-2)(N-3)...(N-2k+1)N^{N-2k+1}$ and $|\mathcal{F}_{i,j}^1| = N^{N-1}$.*

Proof. If the collision is at the first round then $f(x_i) = f(x_j)$ can take N values. And the remainder of the $(N-2)$ points in the domain of f can also take N values each. Hence

$$|\mathcal{F}_{i,j}^1| = N \times N^{N-2} = N^{N-1}.$$

Now for the counting of $\mathcal{F}_{i,j}^k$, with a similar approach $(f(x_i), f(x_j))$ can take $(N-2)(N-3)$ values (as $f(x_i), f(x_j)$ are distinct from x_i, x_j), $(f^{(2)}(x_i), f^{(2)}(x_j))$ can take $(N-4)(N-5)$ values, and so on. Finally $(f^{(k-1)}(x_i), f^{(k-1)}(x_j))$ can take $(N-2k+2)(N-2k+1)$ values, and $f^{(k)}(x_i) = f^{(k)}(x_j)$ can take N values. The remainder of the $(N-2k)$ points in the domain of f can take N values each. Hence all together we have

$$|\mathcal{F}_{i,j}^k| = (N-2)(N-3)...(N-2k+1)N^{N-2k+1}.$$

□

Lemma 4.4. *For* $1 \leq \ell \leq \frac{N}{4} + \frac{1}{2}$, $\mathbf{Pr}_f[C_{i,j}] \geq \frac{\ell}{N} \exp(-\frac{4\ell^2}{N})$.

Proof. From Lemma 4.3 we have

$$|\mathcal{F}| \geq N^{N-1}(1 + \sum_{i=2}^{\ell}(1 - \frac{2}{N})(1 - \frac{3}{N})...(1 - \frac{2k-1}{N})).$$

Hence

$$\mathbf{Pr}_f[C_{i,j}] \geq \left(1 + \sum_{k=2}^{\ell}(1 - \frac{2}{N})(1 - \frac{3}{N})...(1 - \frac{2k-1}{N})\right)\frac{1}{N}$$

Replacing each term in the sum by $\prod_{k=1}^{2\ell-1}(1 - \frac{k}{N})$ we get,

$$\mathbf{Pr}_f[C_{i,j}] \geq \frac{\ell}{N}\prod_{k=1}^{2\ell-1}(1 - \frac{k}{N})$$

$$\geq \frac{\ell}{N}\exp(-\frac{4\ell^2}{N}).$$

In the last step we have used the inequality $1 - x \geq \exp(-2x)$ which is true for $0 \leq x \leq 0.5$ (and hence we need that $\ell \leq N/4 + 1/2$). □

We finally can prove Theorem 4.1.

Theorem 4.1. *The collision probability,* $\mathbf{Pr}_f[\mathbf{CBC}^f(X_i) = \mathbf{CBC}^f(X_j)$ *for some* $i \neq j]$, *is at least*

$$\Delta := \binom{q}{2}\frac{\ell}{N}exp\left(-\frac{4\ell^2}{N}\right) - 3\binom{q}{3}\left(\frac{2\ell^2}{N^2} + \frac{6\ell^6}{N^3}\right) - \frac{1}{2}\binom{q}{2}\binom{q-2}{2}\left(\frac{28\ell^8}{N^4} + \frac{6\ell^4 + 4\ell^5}{N^3} + \frac{\ell^2}{N^2}\right).$$

For large N, *the above expression is* $\Omega(\frac{q^2\ell}{N}) - c(q, \ell, N)$ *when* $\frac{q^2\ell}{N} < \frac{8}{3}$ *and* $\ell = o(N^{\frac{1}{3}})$. *Also,* $c(q, \ell, N)$ *goes to zero when* $q^2\ell = O(N), \ell = o(N^{\frac{1}{3}})$ *and* N *is large.*

Proof. Equation 4.1 and the upper bounds on $|\mathcal{F}_{i,j,k}|$, $|\mathcal{F}_{i,j,k,m}|$ and Lemma 4.4 readily give the expression for Δ. We know $\binom{q}{2} \geq \frac{q^2}{3}$, $\binom{q}{3} \leq \frac{q^3}{6}$ and $\binom{q}{2}\binom{q-2}{2} \leq \frac{q^4}{4}$. Hence we have

$$\Delta \geq \frac{q^2\ell}{3N}\exp\left(\frac{-4\ell^2}{N}\right) - \frac{q^3}{2}\left(\frac{2\ell^2}{N^2} + \frac{6\ell^6}{N^3}\right) - \frac{q^4}{8}\left(\frac{28\ell^8}{N^4} + \frac{6\ell^4 + 4\ell^5}{N^3} + \frac{\ell^2}{N^2}\right).$$

Putting $\alpha = \frac{q^2\ell}{N}$ and

$$c(q,\ell,N) = \frac{1}{2}\left(\frac{q^2\ell}{N}\right)^{1.5}\left(\frac{2\ell^{0.5}}{N^{0.5}} + \frac{6\ell^{4.5}}{N^{1.5}}\right) + \frac{1}{8}\left(\frac{q^2\ell}{N}\right)^{2}\left(\frac{28\ell^6}{N^2} + \frac{6\ell^2+4\ell^3}{N}\right),$$

we get, $\Delta \geq \alpha(\frac{1}{3} - \frac{\alpha}{8}) - c(q,\ell,N)$ when $\ell = o(N^{\frac{1}{3}})$ and N is large. So $\Delta \geq d\alpha - c(q,\ell,N)$, when $\alpha \leq \frac{8}{3} - d$. Hence

$$\Delta \geq \Omega(\frac{q^2\ell}{N}) - c(q,\ell,N),$$

as long as $\frac{q^2\ell}{N} < \frac{8}{3}$. Also $c(q,\ell,N)$ goes to zero when $q^2\ell = O(N), \ell = o(N^{\frac{1}{3}})$ and N is large. □

Exact computation of collision probability

We can use Theorem 4.1 to have a distinguishing attack and MAC-forgery. Here we will see how the bound in Theorem 4.1 is practically meaningful. We will compute the collision probability numerically for suitable choices of ℓ and q. This calculation is important as sometimes the constant can make a real difference.

Example 4.5. MAC forgery when $n = 64$.

Taking $q = c_1 N^{\frac{1}{3}}$ and $\ell = c_2 N^{\frac{1}{3}}$, $\alpha = c_1^2 c_2$ we get,

$$\Delta \approx \frac{\alpha}{2} - \frac{\alpha^2}{8} - (3\alpha^{\frac{3}{2}}c_2^{\frac{9}{2}} + \frac{7}{2}\alpha^2 c_2^6 + \frac{1}{2}\alpha^2 c_2^3).$$

So for small c_2, $\Delta \approx \frac{\alpha}{2} - \frac{\alpha^2}{8}$ To maximize Δ we take $\alpha = 2$, and we get $\Delta \approx 0.5$.
Hence taking $q = \sqrt{20} \cdot 2^{\frac{64}{3}}$, $\ell = 0.1 \times 2^{\frac{64}{3}}$, we get $\Delta = 0.499$. □

Example 4.6. MAC forgery when $n = 128$.

Taking $q = \sqrt{20} \cdot 2^{\frac{128}{3}}$, $\ell = 0.1 \cdot 2^{\frac{128}{3}}$, we get $\Delta = 0.499$. □

To conclude here we have shown one attack on CBC-MAC (finding a collision pair), with queries $X_1, \cdots, X_q$ where X_i is an ℓ-tuple $(x_i, 0, \cdots, 0)$, $x_i \in D$ for $1 \leq i \leq q$. The success probability of the attack is at least Δ.

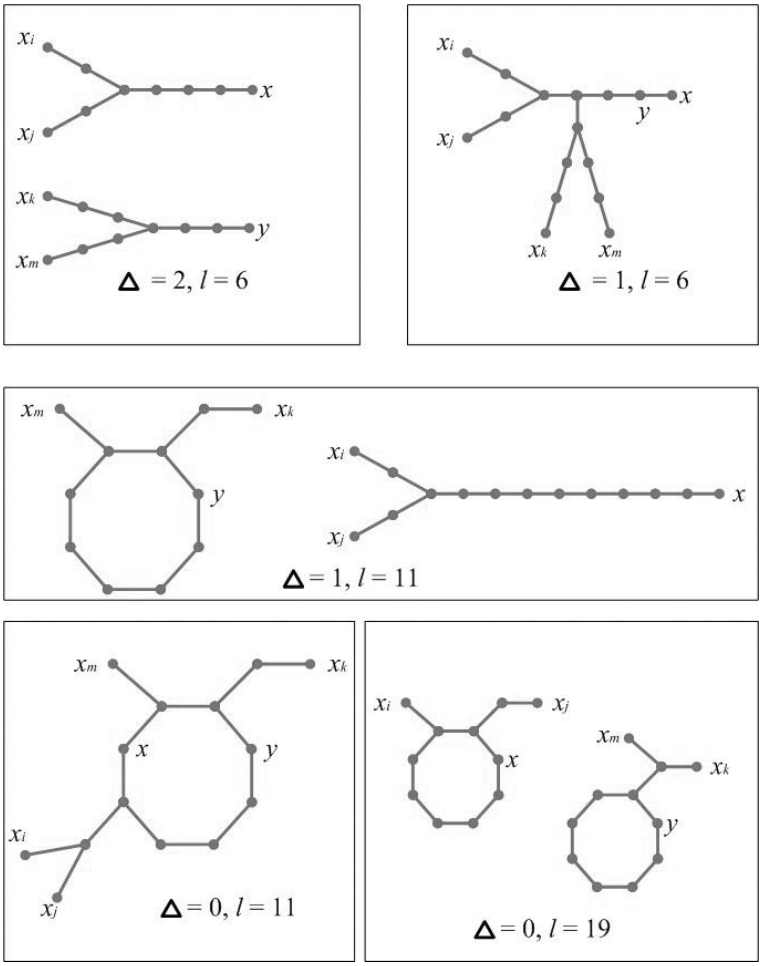

Figure 4.2: Possible function graphs for estimating $|\mathcal{F}_{i,j,k,m}|$

Chapter 5

An attack on PMAC

We provide an analysis of the collision probability for PMAC for suitably chosen two block messages, that is $\ell = 2$. We choose messages $(x_1, 0), \cdots, (x_q, 0) \in D^2$ and want to compute a lower bound for the collision probability for these messages. Note that, for $\ell = 2$, PMAC acts very similar to CBC-MAC with $\ell = 2$. We have the following main result for PMAC.

Theorem 5.1. *When $((x_1, 0), (x_2, 0), \cdots, (x_q, 0))$ are the queries (x_i's are distinct and not equal to zero), the advantage for distinguishing a PMAC oracle and a random oracle is at least $\Omega(\frac{q^2}{N}) - \frac{q}{N}$, when $\frac{q^2}{N} \leq c$ for some $d > 0$ such that $q \geq \frac{1}{d}, c \leq \frac{N}{3(1+d)}$, and $q \leq \frac{N}{4}$.*

Remark 5.2. We can think of a *random oracle* as a uniform random function $G : \{0,1\}^* \to \{0,1\}^n$. The PMAC oracle is a particular function $F : \{0,1\}^* \to \{0,1\}^n$. PMAC uses one random function $f : \{0,1\}^n \to \{0,1\}^n$.

We call a k-tuple $(x_1, \cdots, x_k)$ a *non-collision* k-tuple if all x_i's are distinct for $1 \leq i \leq k$. If a k-tuple is not a non-collision k-tuple then it is a *collision* k-tuple.

Collision probability for PMAC

Let $((x_1, 0), (x_2, 0), ..., (x_q, 0))$ be the q queries to the PMAC oracle (x_i's are distinct and not equal to zero). Let $y_i = x_i + cf(0), w_i = f(y_i), z_i = f(w_i)$, for $1 \leq i \leq q$. So the z_i's are the output of the PMAC. We want to find a lower bound on the probability that

$(z_1, \cdots, z_q)$ is a collision k-tuple. The underlying function f is called a *collision function* if the output tuple $(z_1, \cdots, z_q)$ is a collision k-tuple.

Now a collision can happen in one of the following two ways. (It can happen in other ways also, but we restrict ourselves to the following cases as we are interested in finding a lower bound.)

Case I:

$(0, y_1, \cdots, y_q)$ is a non collision $(q+1)$-tuple and $(w_1, w_2, \cdots, w_q)$ is a collision q-tuple, $w_i \neq 0$ for $1 \leq i \leq q$.

In this case we can choose the $(q+1)$-tuple $(f(0), w_1, w_2, \cdots, w_q)$ in

$$(N-q)((N^q - N(N-1)(N-2)\cdots(N-q))$$

ways. So f is fixed at $(q+1)$ points namely $0, y_1, \cdots, y_q$. The remainder of the $(N-q-1)$ points can be defined arbitrarily. Hence there are

$$(N-q)((N^q - (N-1)(N-2)\cdots(N-q))N^{N-q-1}$$

collision functions in case I.

Case II:

$(0, y_1, y_2, \cdots, y_q, w_1, w_2, \cdots, w_q)$ is a non collision $(2q+1)$-tuple and $(z_1, z_2, \cdots, z_q)$ is a collision q-tuple.

In this case we can choose the $(2q+1)$-tuple $(f(0), w_1, w_2, \cdots, w_q, z_1, z_2, \cdots, z_q)$ in

$$(N-q)(N-q-1)(N-q-2)\cdots(N-2q)(N^q - N(N-1)\cdots(N-q+1))$$

ways. So f is fixed at $(2q+1)$ points namely $0, y_1, \cdots, y_q, z_1, \cdots, z_q$. The remainder of the $(N-2q-1)$ points can be defined arbitrarily. Hence there are

$$(N-q)(N-q-1)(N-q-2)...(N-2q)(N^q - N(N-1)...(N-q+1))N^{N-2q-1}$$

collision functions in case II.

Clearly Case I and Case II are mutually exclusive. Hence we get the following result, which we state as Lemma 5.3.

Lemma 5.3. *There are at least* $(N-q)((N^q - N(N-1)(N-2)...(N-q))N^{N-q-1} + (N-q)(N-q-1)(N-q-2)...(N-2q)(N^q - N(N-1)...(N-q+1))N^{N-2q-1}$ *many collision functions.*

We are now ready to prove Theorem 5.1.

Theorem 5.1. *When* $((x_1,0),(x_2,0),\cdots,(x_q,0))$ *are the queries* (x_i*'s are distinct and not equal to zero), the advantage for distinguishing a PMAC oracle and a random oracle is at least* $\Omega(\frac{q^2}{N}) - \frac{q}{N}$*, when* $\frac{q^2}{N} \leq c$ *for some* $d > 0$ *such that* $q \geq \frac{1}{d}, c \leq \frac{N}{3(1+d)}$*, and* $q \leq \frac{N}{4}$*.*

Proof. From Lemma 5.3 we deduce that collision probability is at least

$$\Delta = (1-\frac{q}{N})\left(1-(1-\frac{1}{N})\dots(1-\frac{q-1}{N})\right)+(1-\frac{q}{N})\dots(1-\frac{2q}{N})\left(1-(1-\frac{1}{N})\dots(1-\frac{q-1}{N})\right).$$

Rearranging the above expression, we get

$$\Delta = \left(1-(1-\frac{1}{N})...(1-\frac{q-1}{N})\right) - \frac{q}{N}\left(1-(1-\frac{1}{N})...(1-\frac{q-1}{N})\right)$$
$$+(1-\frac{q}{N})...(1-\frac{2q}{N})\left(1-(1-\frac{1}{N})...(1-\frac{q-1}{N})\right).$$

We know that the collision probability in the case of a random oracle is $1-(1-\frac{1}{N})...(1-\frac{q-1}{N})$. Hence,

$$\mathbf{Adv} \geq (1-\frac{q}{N})...(1-\frac{2q}{N})\left(1-(1-\frac{1}{N})...(1-\frac{q-1}{N})\right) - \frac{q}{N}\left(1-(1-\frac{1}{N})...(1-\frac{q-1}{N})\right).$$

When $\frac{1}{d} \leq q \leq \frac{N}{4}$, we get

$$\mathbf{Adv} \geq \exp\left(\frac{-3(1+d)q^2}{N}\right)\left(1-\exp(\frac{-(1-d)q^2}{2N})\right) - \frac{q}{N}.$$

When $q^2 \leq \frac{N}{3(1+d)}$, the above expression is

$$\geq (\frac{1-d}{2})(\frac{q^2}{N})\left(1-3(1+d)(\frac{q^2}{N})\right)\left(1-(\frac{1-d}{4})(\frac{q^2}{N})\right) - \frac{q}{N}.$$

Now if there exists a constant c such that $\frac{q^2}{N} \leq c \leq \frac{1}{3(1+d)}$ then $\mathbf{Adv} \geq \Omega(\frac{q^2}{N}) - \frac{q}{N}$. □

Example 5.4. Distinguishing PMAC from a random oracle when $n = 128$.

Writing $\alpha = \frac{q^2}{N}$, we get $\mathbf{Adv} \approx \frac{\alpha}{2}(1 - \frac{3\alpha}{2})(1 - \frac{\alpha}{4})$. This expression attains a maximum value 0.083 at $\alpha = 0.3183$. Hence if we make $\sqrt{0.3183} \cdot 2^{64}$ queries we get at least 0.083 advantage. This means that our attack can distinguish between a PMAC oracle and a random oracle with advantage at least 0.083. □

Chapter 6

Improved security analysis of PMAC

6.1 Definition of PMAC

In this section we will describe PMAC. In the next section we will analyze its security. We would first like to make the following important comments to the reader. The definition of PMAC we provide is a slight modification of the original definition. In the original definition, the length of the message and the bitstring 10^s (for a suitably chosen s) are appended to the end of the message (this is called the *padding*). In this thesis, we consider a different (in fact, a simpler) padding which does not append the length of the message. All other padding rules and the definition of PMAC are exactly the same as the original one. There are some advantages in considering the modified definition.

1. First of all, it is more efficient as we may need one less invocation of the underlying pseudorandom function.

2. We do not have to store the length of the messages, resulting in a reduction of the internal memory requirement.

3. Finally (and most importantly), our modification allows messages of any length to be MACed. So, the message space for our version of PMAC is $\{0,1\}^*$. However for simplicity of our security analysis, we will take $\{0,1\}^{\leq L}$ as the memory space where L can be any large integer. Note that in the original definition L should be

less than $n2^n$. This choice of L is certainly large enough for all current applications. But it is always advantageous if we know that the same construction can be used for arbitrary-length messages.

Let $f : \{0,1\}^n \to \{0,1\}^n$ be a random function for some positive integer n. We write $N = 2^n$. Let $\mathcal{M} = \{0,1\}^{\leq L}$ for a sufficiently large integer L and $T = \{0,1\}^n$. Now we define a random function, known as the **PMAC** function, $\mathrm{P}_f : \mathcal{M} \to T$ based on f. We first define a *padding rule* which makes the message bitlength a multiple of n if it is not already so:

$$\mathbf{pad}(M) = \begin{cases} M \parallel 10^s, & \text{if } n \not| \, |M| \\ M, & \text{otherwise} \end{cases} \tag{6.1}$$

where $s = n\lceil(|M|+1)/n\rceil - |M| - 1$. If $n \not| \, |M|$ then $|\mathbf{pad}(M)| = |M| + s + 1 = n\lceil(|M|+1)/n\rceil$, which is the smallest multiple of n strictly bigger than $|M|$.

Note that if M_1, M_2 are two distinct messages such that $\mathbf{pad}(M_1) = \mathbf{pad}(M_2)$, then exactly one of M_1 and M_2 has size multiple of n (say $n \mid |M_2|$ and $n \not| \, |M_1|$) and $M_2 = \mathbf{pad}(M_1) = M_1 \parallel 10^s$.

Algorithm PMAC Input :M, Output :$Y = \mathrm{P}_f(M)$

1. Write $\mathbf{pad}(M) = x_1 \parallel \cdots \parallel x_\ell \parallel z$, where $\ell \geq 0$ and $|x_1| = \cdots = |x_\ell| = |z| = n$. [The x_i's and z are called *blocks*. If $\ell = 0$, then $\mathbf{pad}(M)$ is nothing but z. Thus, $\ell + 1$ is the total number of message blocks in $\mathbf{pad}(M)$.]

2. Compute $w = f(0)$. [Since f is a random function and kept secret, the value of $f(0)$ has some distribution and can be used as a part of the key of the algorithm.]

3. Compute $v_i = x_i + c_i \cdot w,\ 1 \leq i \leq \ell$. [The c_i's are some fixed distinct nonzero constants as given in [6]. For our security analysis, we only need that $c_i \neq 0$ and are distinct. $(\{0,1\}^n, +, \cdot)$ is any representation of the Galois field $GF(2^n)$. One can think of the addition operation $+$ as bitwise exclusive-or $\oplus$ as it is the simplest operation to implement in both hardware and software.]

4. Compute $w_i = f(v_i),\ 1 \leq i \leq \ell$.

5. Compute $v = z + \Delta + \sum_{1 \le i \le \ell} w_i$, where

$$\Delta = \begin{cases} c \cdot w, & \text{if } n \mid |M|, \\ 0, & \text{otherwise.} \end{cases} \tag{6.2}$$

   [Again, c is a nonzero fixed constant which is different from $c_1, c_2, \cdots, c_\ell$ and is specified in [6].]

6. Finally, $Y \stackrel{\Delta}{=} \mathrm{P}_f(M) = f(v)$.

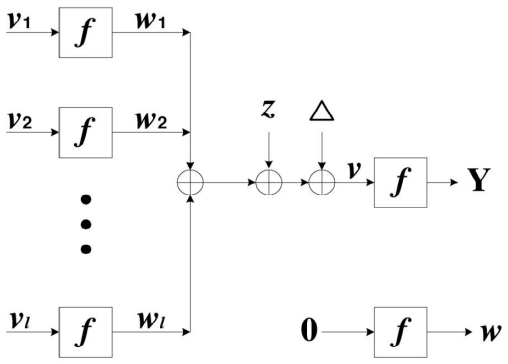

Figure 6.1: PMAC

6.2 Improved security analysis of PMAC

We are interested in computing the probability

$$\Pr[\mathrm{P}_f(M^1) = y^1, \cdots, \mathrm{P}_f(M^q) = y^q], \quad y^i \in \{0,1\}^n, \ M^i\text{'s are distinct.}$$

This probability is assessed under the probability distribution of f, a uniform random function, and it is known as the **interpolation probability**. Denote $\mathbf{M} = \{M^1, \cdots, M^q\}$ and $\ell_j = \parallel \mathbf{pad}(M^j) \parallel$ (the number of message blocks), $1 \le j \le q$. For each $1 \le j \le q$, we denote all variables in the computation of $\mathrm{P}_f(M^j)$ with a superscript j, that is,

$x_i^j, z^j, v_i^j, w_i^j, \Delta^j, v^j, Y^j$, $1 \leq i \leq \ell_j$. Among them, x_i^j and z^j (sometimes Δ^j when $|M^j|$ is not multiple of n) are not random variables, but are fixed. All other variables are random variables with a distribution induced from the distribution of the uniform random functions. Sometime we also write them as $w[f], v_i^j[f], w_i^j[f], v^j[f], \Delta^j[f], Y^j[f]$ to show the dependency of f. We call

- $0, v_i^j$ as *intermediate inputs* and v^j as a *final input*,
- w, w_i^j as *intermediate outputs* and Y^j as a *final output*.

Note that the intermediate and final inputs are really inputs of f while computing $\mathrm{P}_f(M^j)$, and intermediate and final outputs are outputs of f. We will show that for some small ϵ, the interpolation probability $\Pr[\mathrm{P}_f(M^1) = y^1, \cdots, \mathrm{P}_f(M^q) = y^q] \geq (1-\epsilon)/N^q$.

Definition 6.1. *An m-tuple $(a_1, a_2, \ldots, a_m)$ is* new *in an r-tuple $(b_1, b_2, \ldots, b_r)$ if for all $1 \leq i \leq m$ and $1 \leq j \leq r$ we have $a_i \neq b_j$ and the a_i's are pairwise distinct. Note that $m = 1$ is allowed in which case, we say that a_1 is new in $(b_1, b_2, \ldots, b_r)$.*

Let us denote by D the event that all final inputs are distinct and different from all other intermediate inputs. More precisely, $(v^1, \cdots, v^q)$ is new in $(0, v_1^1, \cdots, v_{\ell_1}^1, v_1^2, \cdots, v_{\ell_q}^q)$. Now we prove that the interpolation probability conditioned on D is $1/N^q$. Intuitively, it is clear that the value of $(f(v^1), \cdots, f(v^q))$ follows a uniform distribution conditioned on the assumption that the v^j's do not occur as intermediate inputs (which is guaranteed by the event D). We, next provide a more precise proof of this statement.

Lemma 6.2. $\Pr[\mathrm{P}_f(M^1) = y^1, \cdots, \mathrm{P}_f(M^q) = y^q \mid D] = \frac{1}{N^q}$.

Proof. Let $\mathcal{F}_D$ denote the set of all functions from $\mathcal{F}$ which satisfies the event D.

$$\mathcal{F}_D = \{f_0 \in \mathcal{F} : (v^1[f_0], \cdots, v^q[f_0]) \text{ is new in } (0, v_1^1[f_0], \cdots, v_{\ell_q}^q[f_0])\}.$$

Let $\mathcal{F}_{D_1} = \{f_0 \in \mathcal{F} : (v^1[f_0], \cdots, v^q[f_0])$ is new in $(0, v_1^1[f_0], \cdots, v_{\ell_q}^q[f_0])$ and $Y^j[f_0] = y^j, 1 \leq j \leq q\}$. Thus, $\Pr[\mathrm{P}_f(M^1) = y^1, \cdots, \mathrm{P}_f(M^q) = y^q \mid D] = |\mathcal{F}_{D_1}|/|\mathcal{F}_D|$. Now consider the mapping α from $\mathcal{F}_D$ to $\mathcal{F}_{D_1}$ defined as follows:

$$\alpha(f_0)(x) = \begin{cases} f_0(x), & \text{if } x \neq v^j[f_0] \text{ for all } j, \\ y^j, & \text{if } x = v^j[f_0] \text{ for some } j. \end{cases} \tag{6.3}$$

Note that α is an N^q-to-1 mapping. That is, for every $f_1 \in \mathcal{F}_{D_1}$, there exists exactly N^q many f_0's such that $\alpha(f_0) = f_1$. Given f_1, the f_0's are exactly the same as f_1 except that it can take any of the N^q possible values on $v^j[f_1]$'s. This is well defined since the values of $f_1(v^j[f_1])$'s do not have any effect on the whole computations of $\mathrm{P}_{f_1}(M^j)$'s except the final output. Thus, $|\mathcal{F}_D| = N^q|\mathcal{F}_{D_1}|$ and hence, $\Pr[\mathrm{P}_f(M^1) = y^1, \cdots, \mathrm{P}_f(M^q) = y^q \mid D] = \frac{1}{N^q}$. □

Now we give a lower bound on $\Pr[D]$, or equivalently, an upper bound on $\Pr[\overline{D}]$. Let D^{j_1,j_2} be the event that (v^{j_1}, v^{j_2}) is new in $(0, v_1^{j_1}, \cdots, v_{\ell_{j_1}}^{j_1}, v_1^{j_2}, \cdots, v_{\ell_{j_2}}^{j_2})$, $j_1 \neq j_2$. It is easy to check that $\overline{D} = \cup_{1 \leq j_1 < j_2 \leq q} \overline{D^{j_1,j_2}}$. Thus if $\Pr[\overline{D^{j_1,j_2}}] \leq \delta$ for some δ and all choices of $j_1 < j_2$, then $\Pr[D] \geq (1 - \binom{q}{2}\delta)$. Without loss of generality, we compute $\Pr[D^{1,2}]$ for the messages M^1 and M^2. We have several cases depending on the messages M^1 and M^2.

Lower bound on $\Pr[D^{1,2}]$

Case 1 : $\ell_1 = \ell_2 = \ell$ (say) and $x_1^1 = x_1^2, \cdots, x_\ell^1 = x_\ell^2, z^1 \neq z^2$.

Let us denote $v_1 = v_1^1 = v_1^2, \cdots, v_\ell = v_\ell^1 = v_\ell^2$ and $w_1 = w_1^1 = w_1^2, \cdots, w_\ell = w_\ell^1 = w_\ell^2$. Recall that $w_i = f(v_i), v_i = x_i + c_i w, v^j = \Sigma_{i=1}^{\ell} w_i + z^j + \Delta^j$, for $1 \leq i \leq \ell, j = 1, 2$.

To derive a lower bound on the probability of the event $D^{1,2}$, i.e. (v^1, v^2) is *new* in $(0, v_1, \cdots, v_\ell)$, we define another event A as follows. Let A be the event that v_1 is new in $(0, v_2, \cdots, v_\ell)$ and $\Delta^1 + z^1 \neq \Delta^2 + z^2$. This implies that whenever event A is true, the random variable $w_1 = f(v_1)$ follows the uniform distribution. Using this fact we can easily calculate a lower bound for $\mathbf{Pr}[D^{1,2} \mid A]$. Then we can get lower bound

$$\mathbf{Pr}[D^{1,2}] \geq \mathbf{Pr}[D^{1,2} \mid A]\mathbf{Pr}[A]$$

because $A \cap D^{1,2} \subseteq D^{1,2}$.

- A is the event that v_1 is new in $(0, v_2, \cdots, v_\ell)$ and $\Delta^1 + z^1 \neq \Delta^2 + z^2$. Hence, for $2 \leq i \leq \ell$, $w \neq -\frac{x_1^1}{c_1}, -\frac{x_1^1 - x_i^1}{c_1 - c_i}, \frac{z^2 - z^1}{c}$ (assume that $|M^1|$ is a multiple of n and $|M^2|$ is not; if both are or are not multiples of n then we necessarily have $\Delta^1 + z^1 \neq \Delta^2 + z^2$). So $\Pr[A] \geq \frac{N-\ell-1}{N} = 1 - \frac{\ell+1}{N}$. This follows because A holds whenever $w = f(0)$ takes any value other than those $(\ell+1)$ values listed above, as there are $(\ell+1)$ restrictions on the values of w.

- $D^{1,2}$ is the event that (v^1, v^2) is new in $(0, v_1, \cdots, v_\ell)$. If event A holds, then for each $1 \leq i \leq \ell$ we have,
  - $w_1 + z^1 + (w_2 + \cdots + w_\ell) + \Delta^1 \neq v_i, 0$,
  - $w_1 + z^2 + (w_2 + \cdots + w_\ell^2) + \Delta^2 \neq v_i, 0$ and
  - $w_1 + z^1 + (w_2 + \cdots + w_\ell) + \Delta^1 \neq w_1 + z^2 + (w_2 + \cdots + w_\ell^2) + \Delta^2$.

  Thus we get $\mathbf{Pr}[D^{1,2} \mid A] \geq \frac{N-2\ell-2}{N} = (1 - \frac{2\ell+2}{N})$. Note that w_1 is the output of v_1 which is new in $(0, v_2, \cdots, v_\ell)$, and there are at most $2(\ell+1)$ restrictions on the values of w_1.
- Hence $\Pr[D^{1,2}] \geq (1 - \frac{\ell+1}{N})(1 - \frac{2\ell+2}{N}) \geq 1 - \frac{3\ell+3}{N}$.

Case 2 : $\ell_1 = \ell_2 = \ell$ (say) and $x_1^1 = x_1^2, \cdots, x_\ell^1 = x_\ell^2, z^1 = z^2$.

This case can happen only if $\mathbf{pad}(M^1) = M^1 = M^2 \parallel 10^s = \mathbf{pad}(M^2)$ (there is one more similar case where $|M^2|$ is a multiple of n and $|M^1|$ is not). We denote $v_1 = v_1^1 = v_1^2, \cdots, v_\ell = v_\ell^1 = v_\ell^2$ and $w_1 = w_1^1 = w_1^2, \cdots, w_\ell = w_\ell^1 = w_\ell^2$.

- Let A be the event such v_1 is new in $(0, v_2, \cdots, v_\ell)$ and $\Delta^1 + z^1 \neq \Delta^2 + z^2$. Hence, for $2 \leq i \leq \ell$, $w \neq -\frac{x_1^1}{c_1}, -\frac{x_1^1 - x_i^1}{c_1 - c_i}, \frac{z^2 - z^1}{c}$. So $\Pr[A] = \frac{N-\ell-1}{N} = 1 - \frac{\ell+1}{N}$.
- Let $D^{1,2}$ be the event that (v^1, v^2) is new in $(0, v_1, \cdots, v_\ell)$. If event A holds, then for each $1 \leq i \leq \ell$ we have,
  - $w_1 + z^1 + (w_2 + \cdots + w_\ell) + \Delta^1 \neq v_i, 0$,
  - $w_1 + z^2 + (w_2 + \cdots + w_\ell^2) + \Delta^2 \neq v_i, 0$ and
  - $w_1 + z^1 + (w_2 + \cdots + w_\ell) + \Delta^1 \neq w_1 + z^2 + (w_2 + \cdots + w_\ell^2) + \Delta^2$.

  Thus, we get $\Pr[D^{1,2} \mid A] \geq \frac{N-2\ell-2}{N} = (1 - \frac{2\ell+2}{N})$. Note that w_1 is the output of v_1 which is new in $(0, v_2, \cdots, v_\ell)$.
- Now, $A \cap D^{1,2} \subseteq D^{1,2}$ and hence $\Pr[D^{1,2}] \geq (1 - \frac{\ell+1}{N})(1 - \frac{2\ell+2}{N}) \geq 1 - \frac{3\ell+3}{N}$.

Case 3 : $\ell_1 = \ell_2 = \ell$ (say) and $x_1^1 x_2^1 \dots x_\ell^1 \neq x_1^2 x_2^2 \dots x_\ell^2$.

Without loss of generality we can assume $x_1^1 \neq x_1^2$.

- Let A denote the event that (v_1^1, v_1^2) is new in $(0, v_2^1, v_2^2, \dots, v_\ell^1, v_\ell^2)$.
  Hence $w \neq \frac{x_1^1 - x_i^1}{c_i - c_1}, \frac{x_1^2 - x_i^2}{c_i - c_1}, \frac{x_1^1 - x_j^2}{c_j - c_1}, \frac{x_1^2 - x_j^1}{c_j - c_1}, -\frac{x_1^1}{c_1}, -\frac{x_1^2}{c_1}$ for $2 \leq i, j \leq \ell$. So $\Pr[A] \geq (1 - \frac{4\ell - 2}{N})$
- Let B_1 denote the event that v^1 is new in $(0, v_1^1, v_1^2, \dots, v_\ell^1, v_\ell^2)$. Hence $w_1^1 \neq -(z^1 + w_2^1 + \cdots w_\ell^1), -(z^1 + w_2^1 + \cdots w_\ell^1) + v_i^1, -(z^1 + w_2^1 + \cdots w_\ell^1) + v_i^2$ for $1 \leq i \leq \ell$. So $\Pr[B_1 \mid A] \geq (1 - \frac{2\ell + 1}{N})$
- Let B_2 denote the event that v^2 is new in $(0, v_1^1, v_1^2, \dots, v_\ell^1, v_\ell^2, v^1)$. Hence $w_1^2 \neq -(z^2 + w_2^2 + \cdots w_\ell^2), -(z^2 + w_2^2 + \cdots w_\ell^2) + v_i^1, -(z^2 + w_2^2 + \cdots w_\ell^2) + v_i^2, -(z^2 + w_2^2 + \cdots w_\ell^2) + w_1^1 + (z^1 + w_2^1 + \cdots + w_\ell^1)$ for $1 \leq i \leq \ell$. So $\Pr[B_2 \mid B_1 \cap A] \geq (1 - \frac{2\ell + 2}{N})$.
- Now, $A \cap B_1 \cap B_2 \subseteq D^{1,2}$ and hence $\Pr[D^{1,2}] \geq (1 - \frac{4\ell - 2}{N})(1 - \frac{2\ell + 1}{N})(1 - \frac{2\ell + 2}{N}) \geq 1 - \frac{8\ell + 1}{N}$.

Case 4 : $\ell_1 \neq \ell_2$ and $x_1^1 \neq x_1^2$.

Assume $\ell_2 > \ell_1$.

- Let A denote the event that $(v_1^1, v_{\ell_2}^2)$ is new in $(0, v_2^1, \cdots, v_{\ell_1}^1, v_1^2, v_2^2, \cdots, v_{\ell_2 - 1}^2)$. Hence $w \neq \frac{x_1^1 - x_i^1}{c_i - c_1}, \frac{x_1^1 - x_j^2}{c_j - c_1}, -\frac{x_1^1}{c_1}$ for $2 \leq i \leq \ell_1, 1 \leq j \leq \ell_2$ and $w \neq \frac{x_{\ell_2}^2 - x_i^1}{c_i - c_{\ell_2}}, \frac{x_{\ell_2}^2 - x_j^2}{c_j - c_{\ell_2}}, -\frac{x_{\ell_2}^2}{c_{\ell_2}}$ for $1 \leq i \leq \ell_1, 1 \leq j \leq \ell_2 - 1$. So $\Pr[A] \geq (1 - \frac{2(\ell_1 + \ell_2)}{N})$.
- Let B_1 denote the event that $(v_1^1, v_{\ell_2}^2, v^1)$ is new in $(0, v_2^1, \cdots, v_{\ell_1}^1, v_1^2, v_2^2, \cdots, v_{\ell_2 - 1}^2)$. Hence we have $w_1^1 \neq -(z^1 + w_2^1 + \cdots + w_{\ell_1}^1), -(z^1 + w_2^1 + \cdots + w_{\ell_1}^1) + v_i^1, -(z^1 + w_2^1 + \cdots + w_{\ell_1}^1) + v_j^2$ for $1 \leq i \leq \ell_1, 1 \leq j \leq \ell_2$. So $\Pr[B_1 | A] \geq (1 - \frac{\ell_1 + \ell_2 + 1}{N})$.
- Let B_2 denote the event that $(v_1^1, v_{\ell_2}^2, v^1, v^2)$ is new in $(0, v_2^1, \cdots, v_{\ell_1}^1, v_1^2, v_2^2, \cdots, v_{\ell_2 - 1}^2)$. Hence we have $w_{\ell_2}^2 \neq -(z^2 + w_1^2 + \cdots + w_{\ell_2 - 1}^2), -(z^2 + w_1^2 + \cdots + w_{\ell_2 - 1}^2) + v_i^1, -(z^2 + w_1^2 + \cdots + w_{\ell_2 - 1}^2) + v_j^2, -(z^2 + w_1^2 + \cdots + w_{\ell_2 - 1}^1) + w_1^1 + (z^1 + w_2^1 + \cdots + w_\ell^1)$ for $1 \leq i \leq \ell_1, 1 \leq j \leq \ell_2$. So $\Pr[B_2 | A \cap B_1] \geq (1 - \frac{\ell_1 + \ell_2 + 2}{N})$.
- Now, $A \cap B_1 \cap B_2 \subseteq D^{1,2}$ and hence $\Pr[D^{1,2}] \geq (1 - \frac{2(\ell_1 + \ell_2)}{N})(1 - \frac{\ell_1 + \ell_2 + 1}{N})(1 - \frac{\ell_1 + \ell_2 + 2}{N}) \geq 1 - \frac{4(\ell_1 + \ell_2) + 3}{N}$.

Case 5 : $\ell_1 < \ell_2$ **and** $x_1^1 = x_1^2, \cdots, x_{\ell_1}^1 = x_{\ell_1}^2$**.**

- Let A denote the event that $(v_1^2, v_{\ell_2}^2)$ is new in $(0, v_2^2, \cdots, v_{\ell_2-1}^2)$. Hence we have $w \neq -\frac{x_1^2}{c_1}, -\frac{x_{\ell_2}^2}{c_{\ell_2}}, -\frac{x_{\ell_2}^2 - x_1^2}{c_{\ell_2} - c_1}, -\frac{x_i^2 - x_1^2}{c_i - c_1}, -\frac{x_{\ell_2}^2 - x_j^2}{c_{\ell_2} - c_j}$ for all $2 \leq i \leq \ell_2 - 1,\ 2 \leq j \leq \ell_2 - 1$. So $\Pr[A] \geq (1 - \frac{2\ell_2 - 1}{N})$.

- Let B denote the event that v^1 is new in $(0, v_1^2, \cdots, v_{\ell_2}^2)$. Hence we have $w_1^2 \neq -(z^1 + \Delta^1) - (w_2^2 + \cdots + w_{\ell_1}^2), -(z^1 + \Delta^1) - (w_2^2 + \cdots + w_{\ell_1}^2) + v_i^2$ for $1 \leq i \leq \ell_2$. So $\Pr[B|A] \geq (1 - \frac{\ell_2 + 1}{N})$.

- As defined before $D^{1,2}$ is the event that (v^1, v^2) is new in $(0, v_1^2, \cdots, v_{\ell_2}^2)$. Hence we have $w_{\ell_2}^2 \neq -(z^2 + \Delta^2) - (w_1^2 + \cdots + w_{\ell_2-1}^2), -(z^2 + \Delta^2) - (w_1^2 + \cdots + w_{\ell_2-1}^2) + v_i^2, -(z^2 + \Delta^2) - (w_1^2 + \cdots + w_{\ell_2-1}^2) + v^1$ for all $1 \leq i \leq \ell_2$. So $\Pr[D^{1,2}|A \cap B] \geq (1 - \frac{\ell_2 + 2}{N})$.

Hence $\Pr[D^{1,2}] \geq (1 - \frac{2\ell_2 - 1}{N})(1 - \frac{\ell_2 + 1}{N})(1 - \frac{\ell_2 + 2}{N}) \geq 1 - \frac{4\ell_2 + 2}{N}$.
Now we are in a position to give a lower bound for the interpolation probability.

Lemma 6.3. *Let $M^1, \cdots, M^q$ be distinct messages from $\mathcal{M}$, and let $y^1, \cdots, y^q \in T$ (not necessarily distinct). Then*

$$\Pr[\mathrm{P}_f(M^1) = y^1, \cdots, \mathrm{P}_f(M^q) = y^q] \geq \frac{1 - \epsilon}{N^q} = (1 - \epsilon) \times \Pr[F(M^1) = y^1, \cdots, F(M^q) = y^q],$$

where $\epsilon = \frac{4(q-1)\sigma}{N}$ and F is a uniform random function on $\mathbf{Func}(\{0,1\}^{\leq L}, \{0,1\}^n)$.

Proof. From the above five cases we can say that for any two messages M^{j_1} and M^{j_2},

$$\Pr[\overline{D^{j_1, j_2}}] \leq \frac{4((\ell_{j_1} - 1) + (\ell_{j_2} - 1)) + 3}{N}.$$

(Here ℓ_i denotes total number of blocks including z.) Thus,

$$\Pr[\overline{D}] \leq \sum_{1 \leq j_1 < j_2 \leq q} \frac{4(\ell_{j_1} + \ell_{j_2}) - 5}{N} = \frac{4(q-1)\sum_j \ell_j}{N} - \frac{3q(q-1)}{2N} \leq \frac{4(q-1)\sigma}{N}.$$

Hence by Lemma 6.2 we get

$$\Pr[\mathrm{P}_f(M^1) = y^1, \cdots, \mathrm{P}_f(M^q) = y^q] \geq \frac{1 - \epsilon}{N^q}, \text{ where } \epsilon = \frac{4(q-1)\sigma}{N}.$$

The rest of the claim in the statement of the Lemma follows trivially. □

Lemma 6.3 precisely gives the precondition for applying Theorem 2.9. With the help of Remark 2.10 we get the final result of this section which we mention as Theorem 6.4.

Theorem 6.4. $\mathbf{Adv}_{\mathbf{PMAC}}(q, \sigma, t) \leq \frac{4(q-1)\sigma}{N}$.

To conclude the chapter, here we have shown that for any attacker (making q queries with σ blocks in total) trying to distinguish a PMAC oracle and a random oracle the advantage is bounded by $O(\frac{q\sigma}{N})$.

Chapter 7

Conclusion

We have seen how counting arguments and a combinatorial approach give rise to improved security bounds for PMAC and distinguishing attacks in the case of CBC-MAC and PMAC. In all the cases we have assumed that the underlying compression functions in the MAC algorithms are random functions. However in practical implementations generally random permutations are preferred instead of random functions, mainly due to simplicity of implementations of pseudorandom permutations. Our work in the case of CBC-MAC indicates that not only are random permutations simpler to implement, but CBC-MAC based on random permutations tends to be more secure than than CBC-MAC based on random functions. This is because we have shown one attack with advantage $\Omega(\frac{\ell q^2}{N})$ for CBC-MAC based on random functions, whereas no similar attack is known for CBC-MAC based on random permutations. And it is not trivial to extend our attack idea to the case of random permutations. Bellare, Pietrzak and Rogaway [2] have shown a security bound of $O(\frac{\ell q^2}{N})$ in the case of CBC-MAC based on random permutations, whereas Bernstein [4] has shown a security bound of $O(\frac{\ell^2 q^2}{N})$ in the case of CBC-MAC based on random functions. And in the case of CBC-MAC based on random permutations the best attack known is based on the birthday attack with advantage $\Omega(\frac{q^2}{N})$. So we see that in both the cases there is a gap between the security bound and the best attack known. It would be a nice result if that gap can be removed.

We also have shown an improved security bound for PMAC based on random functions, following the ideas of Bernstein and Nandi [4, 13]. The underlying idea is based on a

counting principle and can easily be adopted to prove security for various other MAC constructions. Also unlike the previous case, our analysis can be extended to PMAC based on random permutations. In fact, recently Minematsu and Matsushima [16] (to be published in FSE-2007 proceedings) have obtained the same bound as us for PMAC based on random permutations. But in some sense our bound is a little more general than theirs. Their security bound is $O(\frac{\ell_{max}q^2}{N})$, while ours is $O(\frac{q\sigma}{N})$. The security analysis is made on a slight modification of PMAC (without length padding), but the analysis also holds for the original PMAC definition. Hence one can use PMAC for arbitrary length messages. As a future research work, we hope that our security analysis can be extended to obtain improved bounds on a general class called *PRF domain extension using directed acyclic graph* given in [10, 13].

Bibliography

[1] M. Bellare, A. Boldyreva, L. Knudsen and C. Namprempre. On-Line Ciphers and the Hash-CBC Constructions. Advances in Cryptology – CRYPTO 2001. Lecture Notes in Computer Science, Volume **2139**, pp 292-309, Springer-Verlag 2001.

[2] M. Bellare, K. Pietrzak and P. Rogaway. Improved Security Analysis for CBC MACs. Advances in Cryptology – CRYPTO 2005. Lecture Notes in Computer Science, Volume **3621**, pp 527-545, Springer-Verlag 2005.

[3] M. Bellare, J. Killan and P. Rogaway. The Security of the Cipher Block Chaining Message Authentication Code. Advances in Cryptology – CRYPTO 1994. Lecture Notes in Computer Science, Volume **839**, pp 341-358, Springer-Verlag 1994.

[4] Daniel J. Bernstein. A Short Proof of the Unpredictability of Cipher Block Chaining (2005). URL: http://cr.yp.to/papers.html#easycbc.

[5] J. Black and P. Rogaway. CBC MACs for Arbitrary Length Messages. Advances in Cryptology – CRYPTO 2000. Lecture Notes in Computer Science, Volume **1880**, pp 197-215, Springer-Verlag 2000.

[6] J. Black and P. Rogaway. A Block-Cipher Mode of Operations for Parallelizable Message Authentication. Advances in Cryptology – Eurocrypt 2002. Lecture Notes in Computer Science, Volume **2332**, pp 384-397, Springer-Verlag 2002.

[7] J. Daemen and V. Rijmen. Resistance Against Implementation Attacks. A Comparative Study of the AES Proposals. In Proceedings of the Second

AES Candidate Conference (AES2), Rome, Italy, March 1999. Available at http://csrc.nist.gov/encryption/aes/aes_ home.htm.

[8] H. Krawczyk. LFSR-based Hashing and Authenticating. Advances in Cryptology – CRYPTO 1994, Lecture Notes in Computer Science, Volume **839**, pp 129-139, Springer-Verlag 1994.

[9] T. Iwata and K. Kurosawa. OMAC : One-Key CBC MAC. Fast Software Encryption, 10th International Workshop, FSE 2003. Lecture Notes in Computer Science, Volume **2887**, pp 129-153, Springer-Verlag 2003.

[10] C. S. Jutla. PRF Domain Extension Using DAG. Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006. Lecture Notes in Computer Science, Volume **3876**, pp 561-580, Springer-Verlag 2006.

[11] K. Kurosawa and T. Iwata. TMAC : Two-Key CBC MAC. Topics in Cryptology - CT-RSA 2003: The Cryptographers' Track at the RSA Conference 2003. Lecture Notes in Computer Science, Volume **2612**, pp 33-49, Springer-Verlag 2003.

[12] M. Luby and C. Rackoff. How to Construct Pseudo-random Permutations from Pseudo-random Functions. Advances in Cryptology, CRYPTO' 85, Lecture Notes in Computer Science, Volume **218**, pp 447, Springer-Verlag 1985.

[13] M. Nandi. A Simple and Unified Method of Proving Indistinguishability. Indocrypt 2006, Lecture Notes in Computer Science, Volume **4329**, pp 317-334, Springer-Verlag 2007.

[14] P. Rogaway. Bucket Hashing and Its Application to Fast Message Authentication. Advances in Cryptology, CRYPTO 1995, Lecture Notes in Computer Science, Volume **963**, pp 29-42, Springer-Verlag 1995.

[15] D. R. Stinson. On the Connections between Universal Hashing, Combinatorial designs and Error-correcting codes. Congressus Numerantium **114**, 1996, pp 7-27.

[16] K. Minematsu and T. Matsushima. Improved Security Bounds for PMAC, TMAC, and XCBC. Proceedings of FSE 2007, to appear.

[17] W. Diffie and M. Hellman. New Directions in Cryptography. IEEE transactions on Information Theory, Volume **22**, pp 644-654, 1976.