M. Tech. (Computer Science) Dissertation

Probabilistic Analysis of Cryptographic Hash Functions

A dissertation submitted in partial fulfillment
of the requirements for the award of
M.Tech.(Computer Science) degree

By

Somindu C R
Roll No: CS0711

under the supervision of

Professor Palash Sarkar
Applied Statistics Unit

I N D I A N S T A T I S T I C A L I N S T I T U T E
203, Barrackpore Trunk Road
Kolkata - 700 108

Acknowledgement

At the end of this course, it is my pleasure to thank everyone who has helped me along the way.

First of all, I want to express my sincere gratitude to my supervisor Prof. Palash Sarkar for introducing me to the world of hash functions and starting me on this interesting problem. I have learnt a lot from him. For his patience, for all his advice and encouragement and for the way he helped me think about problems with a broader perspective, I will always be grateful.

I would like to thank all the professors at ISI who have made my educational life exciting and helped me gain a better outlook on computer science. Special thanks to Prof. Bimal Roy who first inspired me to study Cryptography.

I would like to thank everybody at ISI for providing a wonderful atmosphere for pursuing my studies. I thank all my classmates who have made the academic and non-academic experience very delightful. Special thanks to my friends Sreevani, Krithika, Nargis, Debosmita, Richa, Sandeep, Sanjay and many others who made my campus life so enjoyable. It has been great having them around at all times, good or bad.

My most important acknowledgement goes to my family and friends who have filled my life with happiness. Most significantly to my parents who have always encouraged me to pursue my passions and instilled a love of knowledge in me; to my brothers Vishu and Vicky; to my sisters-in-law Veena and Vinutha and to my nephew and niece Harshi and Nikitha who have filled my heart with joy. I am indebted to my friends Vijay, Smitha and Prajna for their endless supply of encouragement, moral support and entertainment.

Abstract

A multicollision for a hash function is a set of two or more distinct domain points all mapping to the same range point. Multicollision freeness has been suggested as an important security property of hash functions. Joux has shown that multicollisions are not harder to find than ordinary collisions for hash functions based on an iterated construction. For general hash functions, the best known attack is the generic birthday attack. For truly random functions, the complexity of finding r-collisions is $\Theta(m^{(r-1)/r})$ where m is the size of the range of the hash function. But such functions are seldom encountered in practice.

For the case of $r = 2$, Bellare and Kohno analyze the success rate of the birthday attack on a specific hash function rather than analyzing one chosen at random. They define balance of a hash function h, denoted $\mu(h)$, which is a measure of the "amount of regularity" of h and study its impact on the birthday attack.

In this thesis we extend the notion of balance to that of r-balance. We then analyze the performance of the birthday attack via the r-balance. We derive bounds on the probability of finding r-collisions using the birthday attack for a given hash function h. Using these bounds we show that the complexity of finding r collisions is roughly $\Theta(m^{\left(\frac{r-1}{r}\right)\mu_r(h)})$ where $\mu_r(h)$ is the r-balance of h. Our results indicate that higher the r-balance, higher will be the complexity of finding r-collisions. For $r = 2$, our analysis provides slightly better bounds than the ones given by Bellare and Kohno.

Contents

List of Figures

Chapter 1

Introduction

Hash functions are of fundamental importance in cryptographic protocols. Informally, a hash function takes as input a bit string that is arbitrarily long and compresses it into a fixed length output. A hash function essentially produces a "digest" or a "fingerprint" of fixed length of a message of arbitrary length ([Sti02]). Most hash functions used in practice have an upper limit on the size of inputs, but this limit is so large that it would make sense to say that hash functions take arbitrarily sized inputs.

A cryptographic hash function is required to satisfy some security properties depending on the particular protocol in which it is being used. Some of the most important security properties are *collision resistance*, *preimage resistance* and *second preimage resistance*. One desirable property of a hash function is *"random behaviour"*. That is, given a hash value, it must be *hard to predict* another hash value. Some applications [RS96, GS94, BPVY00] rely on another property called *r-collision freeness*.

A collision is a pair of messages that hash to the same value. A hash function is said to be *collision resistant* if it is *"infeasible"* to construct a collision. Since the domain of a hash function is larger than its range, it follows from the pigeonhole principle that collisions will surely exist. A hash functions must be designed in such a way that finding these collisions is computationally difficult. Preimage resistance refers to the hardness of finding a preimage of a given hash value (note that there could be more than one preimage for a given hash value). Second preimage resistance requires that, given a hash value and one of its preimages, finding another message that maps to the same value is computationally hard. All hash functions can be attacked using the generic collision-finding attack called the *birthday attack*.

In cryptography, hash functions are most commonly encountered in digital signatures and data integrity. In case of digital signatures, a long message is hashed and then the hash value is signed. The sender sends the message and the signature. The receiver hashes the received message and verifies whether the received signature is correct for this hash value. This saves both time and space compared to signing

the entire message itself. The property of collision resistance comes into play here. Suppose one could construct two different messages that hash to the same value. Then the sender could send one of these messages and later claim to have signed and sent another. To prevent this from happening the hash function must be designed to be collision resistant.

Another application of hash function is in data integrity where it is used as follows. The hash value of the data is computed and stored. At any point of time if we want to verify the integrity of the data, we will compute the hash value and compare it to the stored value for equality. Note that we have assumed here that the integrity of the stored hash value is protected in some manner. Other applications of hash functions include identification schemes and micropayment schemes. A distinct class of hash functions, called *message authentication codes* (MACs), are used for message authentication ([MvOV97, Sti02]). A MAC differs from a hash function in that it takes as a second input a secret key with the security goal that it be infeasible to find a message that maps to the same output without the knowledge of the key.

The property of r-collision freeness has been suggested as a useful tool in building cryptographic protocols. It was used for the micropayment scheme Micromint of Rivest and Shamir [RS96], for identification schemes by Girault and Stern [GS94] and for signature schemes by Brickell *et. al.* [BPVY00]. An *r-way collision* (or *r-collision*) is a set of r messages that hash to the same value. This property requires that finding an r-collision should be computationally difficult. Consider, for example, the micropayment scheme of [RS96]. In this scheme, "coins" are produced by a broker who sells them to users. Users give these coins to vendors as payments. Vendors return coins to the broker in return for payment by other means. A coin is a bit-string whose validity can be easily checked by anyone, but which is hard to produce. Here an r-collision is used as a coin. Verification is simply done by checking whether the r messages are distinct and they all hash to the same value. The intuition behind using r-collisions is that forging a coin i.e., finding an r-collision is infeasible and much harder than finding ordinary 2-collisions.

It was shown by Joux [Jou04] that for iterated hash functions finding r-collisions is not much harder than finding 2-collisions. Following Joux's attack, Nandi and Stinson [NS07] studied multicollision attacks in a more general class of hash functions called *generalized sequential hash functions*. They showed the existence of multicollision attacks for this class of hash functions provided that every message block is used at most twice in the computation of the message digest. They rule out a large class of hash functions as candidates for multicollision secure hash functions. These attacks were extended and generalized by Hoch and Shamir [HS06] to tree-based hash functions that have a fixed *expansion rate*, which indicates the maximum number of times a message block is processed in the evaluation of a message digest. But when the hash function is truly random, it would be more difficult to find an r-collision than finding a 2-collision. A generic attack that finds r-collisions is the *generalized birthday attack*.

Bellare and Kohno [BK04] analyze the performance of the birthday attack on a hash function in terms of what they define as *balance* of the hash function. They show that the birthday attack fares well against functions with low balance as compared to highly balanced ones. Their results indicate that designing a hash function that has random behaviour and also high balance would ensure better security against birthday attacks.

The main question addressed here the following: What is the notion of balance of a hash function in the context of r-collisions? How will the balance affect the generalized birthday attack that finds r-collisions?

The rest of the thesis is organized as follows. In Chapter 2, we provide basic definitions concerning hash functions followed by a description of the birthday attack and its analysis in the random oracle model. In Chapter 3, we state the definition of balance and discuss its impact on the birthday attack. In Chapter 4, we will discuss multicollision attacks on iterated hash functions. Then we will review some statistics literature on the generalized birthday problem. In Chapter 5, we will define the notion of r-balance and analyze the performance of the generalized birthday attack via the r-balance.

Chapter 2

Hash Function Preliminaries

A function $h : X \to Y$ is called a *hash function* if the domain X and range Y are finite sets such that $|X| > |Y|$. If a hash function is secure then it should be infeasible for a bounded adversary to solve the following problems.

- **Preimage**
  Given $y \in Y$, find $x \in X$ such that $h(x) = y$
- **Second Preimage**
  Given $x \in X$, find $x' \in X$ such that $x \neq x'$ and $h(x') = h(x)$
- **Collision**
  Find $x, x' \in X$ such that $x \neq x'$ and $h(x) = h(x')$

We will now analyze the difficulty of solving the collision finding problem in a certain idealized model for a hash function called the *random oracle model*.

Random Oracle Model The random oracle model introduced by Bellare and Rogaway [BR93], provides a mathematical model for an "ideal" hash function. In this model a hash function h is chosen uniformly at random from the set $\mathcal{F}^{X,Y}$ of all functions from the set X to the set Y. There is no algorithm or formula that computes the values for h but we are provided only *oracle access* to the function h. Given $x \in X$, the only way to compute $h(x)$ is to query the oracle. As a consequence of these assumptions, the hash values for a function $h \in \mathcal{F}^{X,Y}$ are independently and uniformly distributed over Y. That is,

$$\Pr[h(x) = y] = \frac{1}{m}$$

for all $x \in X$ and $y \in Y$, where $m = |Y|$.

2.1 The Birthday Attack

As mentioned in Chapter 1, one of the most important security properties of a hash function is collision-resistance, which measures the ability of an adversary to

```
Choose X_0 ⊆ X such that |X_0| = q
For i = 1, ..., q do
        y_i ← h(x_i)
EndFor
If there is a pair (i, j) such that y_i = y_j
        then return x_i, x_j
EndIf
Return "failure"
```

Figure 2.1: Birthday attack on a hash function $h : X \to Y$ based on sampling without replacement.

---

find a collision for the hash function. All hash functions suffer from the generic *birthday attack*. Figure 2.1 shows the classical birthday attack on a hash function h. The attack picks points $x_1, x_2, \cdots, x_q$ without replacement from the domain X and computes $y_i = h(x_i)$ for $i = 1, \cdots, q$. The attack is successful if there is a pair i, j such that x_i, x_j form a collision for h. The integer $q \geq 2$ is the number of trials. The attack returns a collision for h or returns "failure" if it fails to find one. The relation to birthdays arises from the question of how many people need be in a room before the probability of there being two people with the same birthday is close to one. Assuming birthdays are independently and uniformly distributed over the days of the year it turns out that when the number of people hits $\sqrt{365}$ the chance of a birthday collision is already quite high, around $1/2$.

We will now provide an analysis of the birthday attack of Figure 2.1 in the random oracle model.

Theorem 2.1.1. *Let $h : X \to Y$ be a hash function with $|X| = n$ and $|Y| = m$. Then for any integer $q \geq 2$,*

1. *The success probability of the classical birthday attack is given by*

$$p = 1 - \left(1 - \frac{1}{m}\right)\left(1 - \frac{2}{m}\right) \cdots \left(1 - \frac{q-1}{m}\right).$$

2. *The minimum number of trials required to find a collision with probability p is approximately determined by solving for q in*

$$q \approx \sqrt{2m \ln\left(\frac{1}{1-p}\right)}.$$

Proof. Let $X_0 = \{x_1, \cdots x_q\}$. For $i = 1, \cdots, q$, define E_i to be the event that $y_i \notin \{y_1, y_2, \cdots y_{i-1}\}$. According to the random oracle model, the y_i's are uniformly distributed in Y. If $y_1, \cdots, y_{i-1}$ are all distinct then $\Pr[E_i]$ is the probability that y_i takes one of the $m - i + 1$ remaining values from Y. Thus we have,

$$\Pr[E_i | E_1 \wedge E_2 \wedge \cdots \wedge E_{i-1}] = \frac{m - i + 1}{m}$$

for $2 \leq i \leq q$. By applying chain rule we obtain

$$\Pr[E_1 \wedge E_2 \wedge \cdots \wedge E_q] = \left(1 - \frac{1}{m}\right)\left(1 - \frac{2}{m}\right) \cdots \left(1 - \frac{q-1}{m}\right).$$

Now,

$$p = \Pr[\text{at least one collision}] = 1 - \Pr[\text{no collisions}] = 1 - \Pr[E_1 \wedge E_2 \wedge \cdots \wedge E_q]$$

and thus statement 1 follows. Now we proceed to the proof of statement 2. Using the fact that $1 - x \leq e^{-x}$ for $0 < x < 1$, we get

$$\begin{aligned} \prod_{i=1}^{q-1}\left(1 - \frac{i}{m}\right) &\leq \prod_{i=1}^{q-1} e^{-\frac{i}{m}} \\ &= e^{-\sum_{i=1}^{q-1}\frac{i}{m}} \\ &= e^{-\frac{q(q-1)}{2m}} \end{aligned}$$

Thus we have

$$\begin{aligned} p &\approx 1 - e^{-\frac{q(q-1)}{2m}} \\ e^{-\frac{q(q-1)}{2m}} &\approx 1 - p \\ -\frac{q(q-1)}{2m} &\approx \ln(1-p) \\ q^2 - q &\approx 2m \ln\left(\frac{1}{1-p}\right) \\ q &\approx \sqrt{2m \ln\left(\frac{1}{1-p}\right)} \end{aligned}$$

□

If we substitute $p = 0.5$ then we obtain

$$q \approx 1.17\sqrt{m}$$

This shows that around $\sqrt{m}$ trials are needed to find a collision for h with probability equal to 1/2.

2.2 Merkle-Damgård Construction

Hash functions map arbitrarily long strings to fixed length strings. In practice, building a cryptographic function with an input of variable size is not a simple task. Due to this reason, most hash functions are based on an *iterated construction* that makes use of a building block called the *compression function* whose inputs have fixed sizes ([Sti02]). The compression function f takes two inputs: a chaining variable and a message block; it outputs the next value of the chaining variable. The most popular generic construction is the Merkle-Damgård (MD) Construction ([Mer79, Dam89]). We will now give a brief description of the MD-construction. Throughout this section we will consider hash functions whose inputs and outputs are bit strings. Denote the length of a bit string x as $|x|$. Before processing, the message is first padded and split into elementary blocks.

Padding scheme Padding is done by appending a single '1' bit, followed by as many '0' bits as needed. To avoid some attacks, the binary encoding of the message length is also added to complete the padding. This is called Merkle-Damgård strengthening.

The iteration Once the padded message is split into ℓ blocks, $M_1, \cdots, M_\ell$, the chaining variable is set to some fixed initial value, IV, and the iteration is performed.

- Pad the original message and split it into ℓ blocks $M_1, \cdots, M_\ell$.
- Set H_0 to the initial value IV.
- For $i = 1, \cdots, \ell$, let $H_i = f(H_{i-1}, M_i)$.
- Output $h(M) = H_\ell$.

The following theorem states that if a collision can be found for h then a collision can be found for the compression function f. This means that it is enough to design a collision resistant compression function to ensure that the hash function is collision resistant.

Theorem 2.2.1 (Merkle-Damgård Theorem)**.** *If the compression function f is collision resistant then the hash function h is also collision resistant.*

Proof. Suppose that h is not collision resistant. This means that there are two messages M and M' such that $h(M) = h(M')$. Let $M = M_1 \| M_2 \| \cdots \| M_\ell$ and $M' = M'_1 \| M'_2 \| \cdots \| M'_k$ where M_i's and M'_j's are of the same length.

Case 1: The lengths of M and M' are different.
If the lengths of two messages are different then their last blocks must be different.

$$h(M) = h(M')$$

$$\implies f(H_{\ell-1}, M_\ell) = f(H'_{k-1}, M'_k)$$

which gives a collision for f.

Case 2: The lengths of M and M' are same.
Let the number of blocks be ℓ. If all the intermediate hash values are equal, i.e. $H_i = H'_i$ for all $i \leq l$ with some $M_j = M'_j$. In this case, a collision is obtained as follows.

$$H_j = f(H_{j-1}, M_j) = f(H'_{j-1}, M'_j) = H'_j$$

If some intermediate hash values are unequal, i.e., $H_i \neq H'_i$ for some $i \leq l$ then consider the maximum such i. A collision can be obtained as follows.

$$H_{i+1} = f(H_i, M_{i+1}) = f(H'_i, M'_{i+1}) = H'_{i+1}$$

□

Chapter 3

Balance and its impact on Birthday Attacks

In a birthday attack, we pick points $x_1, x_2, \cdots, x_q$ from X and compute $y_i = h(x_i)$ for $i = 1, \cdots, q$. The attack is successful if there is a pair i, j such that x_i, x_j form a collision for h. Here, q is called the number of *trials*. There are several variants of this attack which differ in the way the points $x_1, x_2, \cdots, x_q$ are chosen. The attack considered in [BK04] shown in Figure 3.1, considers points that are chosen independently and uniformly at random from X. This attack is as good as the classical birthday attack given in Figure 2.1 when the domain is substantially larger than the range (say, $n \geq 2m$). This is because the probability of two domain points being equal becomes negligible compared to probability of a collision when the domain is large enough. Let $C_h(q)$ be the probability that the birthday attack on hash function $h : X \to Y$ succeeds in finding a collision in q trials. For any real number c with $0 \leq c < 1$, let

$$Q_h(c) = \min\{q : C_h(q) \geq c\}. \tag{3.1}$$

The discussion in Section 2.1 suggests that the function $C_h(q)$ grows with q as follows.

$$C_h(q) \approx \binom{q}{2} \cdot \frac{1}{m}\ , \tag{3.2}$$

where $m = |Y|$ is the size of the range of h and $q \leq O(\sqrt{m})$. This implies that a collision is expected in about $m^{1/2}$ trials. This is obtained by viewing the range points $y_1, y_2, \cdots, y_q$ computed in the attack as being independently and uniformly distributed in Y.

Bellare and Kohno [BK04] explain why this argument is actually not correct. It is because the point $h(x)$, for x drawn at random from X, is not necessarily uniformly distributed in Y. Rather, the probability that $h(x)$ equals a particular range point y is $|h^{-1}(y)|/|X|$, where $h^{-1}(y)$ is the set of all preimages of y under h. So the range points computed in the attack are uniformly distributed over Y if and only if h is

```
For i = 1, ..., q do
        x_i ←$ X
        y_i ← h(x_i)
EndFor
If there is a pair (i, j) such that y_i = y_j and x_i ≠ x_j
        then return x_i, x_j
EndIf
Return "failure"
```

Figure 3.1: Birthday attack on a hash function $h : X \to Y$ based on sampling with replacement.

---

regular, meaning that every range point has the same number of preimages under h. Given a hash function h one cannot assume that h has "random behaviour" because the analysis of the birthday attack ignores the actual function entirely by looking at only random functions. One ends up not analyzing the given function h, but rather analyzing an abstract and ideal object which ultimately has no connection to h, regardless of the design principle underlying h. Hence in [BK04] the authors assess the success rate of birthday attack by looking at $C_h(q)$ for a specific h rather than one chosen at random. They define the *balance measure* for a hash function and then provide quantitative estimates of the success rate of the birthday attack as a function of the balance of the hash function being attacked.

3.1 Balance of a hash function

The balance of a hash function is a measure of the "amount of regularity" of the function. It is defined as follows.

Definition 3.1.1 (Balance [BK04])**.** Let $h : X \to Y$ be a hash function whose domain X and range $Y = \{y_1, y_2, \cdots, y_m\}$ have sizes $n, m \geq 2$, respectively. For $i = 1, 2, \cdots, m$, let $n_i = |h^{-1}(y_i)|$ denote the size of the preimage of y_i under h. The balance of h, denoted $\mu(h)$, is defined as

$$\mu(h) = \log_m \left(\frac{n^2}{n_1^2 + \cdots + n_m^2} \right), \tag{3.3}$$

where $\log_m(\cdot)$ denotes the logarithm in base m.

The intuition behind the above definition is that

$$\frac{1}{m^{\mu(h)}} = \frac{n_1^2 + \cdots + n_m^2}{n^2}$$

is the probability that $h(a) = h(b)$ if a, b are drawn independently with replacement from the domain X. From the definition it follows that the balance is a real number between 0 and 1 i.e., $0 \leq \mu(h) \leq 1$. Balance 1 indicates that the hash function is regular and balance 0 indicates that it is a constant function.

3.2 Impact of balance on birthday attack

The following theorems summarize the results obtained in [BK04].

Theorem 3.2.1. *Let $h : X \to Y$ be a hash function. Let $n = |X|$ and $m = |Y|$ and assume $n > m \geq 2$. Let $\alpha \geq 0$ be any real number. Then for any integer $q \geq 2$*

$$(1 - \alpha^2/4 - \alpha) \cdot \binom{q}{2} \cdot \left(\frac{1}{m^{\mu(h)}} - \frac{1}{n}\right) \leq C_h(q) \leq \binom{q}{2} \cdot \left(\frac{1}{m^{\mu(h)}} - \frac{1}{n}\right), \tag{3.4}$$

the lower bound being true under the additional assumption that

$$q \leq \alpha \cdot \left(1 - \frac{m}{n}\right) \cdot m^{\mu(h)/2}. \tag{3.5}$$

It is important to have upper and lower bounds on $C_h(q)$ that are close to each other, because based on these estimates we are making very specific choices of hash function parameters, in particular, output lengths. Accordingly Theorem 3.2.1 strives for good bounds and achieves this since, as $\alpha \to 0$, the lower bound of Equation (3.4) approaches the upper bound. So the bounds can be made as close as we want. However, there is a tradeoff: as $\alpha \to 0$ the lower bound is valid across smaller and smaller ranges of q due to the restriction of Equation (3.5).

Suppose that $n \geq 2m \geq 4$. If we choose $\alpha = 2/5$, then for any integer $q \geq 2$, we have

$$0.28 \cdot \binom{q}{2} \cdot \frac{1}{m^{\mu(h)}} \leq C_h(q) \leq \binom{q}{2} \cdot \frac{1}{m^{\mu(h)}}. \tag{3.6}$$

These bounds show that

$$C_h(q) = \Theta(1) \cdot \frac{q^2}{m^{\mu(h)}},$$

as long as q is not too large.

Theorem 3.2.2. *Let $h : X \to Y$ be a hash function. Let $n = |X|$ and $m = |Y|$ and assume $n \geq 2m \geq 4$. Let $\alpha \geq 0$ be any real number such that $\beta = 1 - \alpha^2/4 - \alpha > 0$. Let c be a real number in the interval $0 \leq c < 1$. Then*

$$\sqrt{2c} \cdot m^{\mu(h)/2} \leq Q_h(c) \leq 1 + \sqrt{\frac{4c}{\beta}} \cdot m^{\mu(h)/2}\,, \tag{3.7}$$

the upper bound being true under the additional assumption that

$$c \leq (\alpha \cdot (1 - m/n) - m^{-\mu(h)/2})^2 \cdot \frac{\beta}{4}\,. \tag{3.8}$$

Substituting $\alpha = (\sqrt{17} - 3)/2$ will yield the following bounds.

$$\sqrt{2c} \cdot m^{\mu(h)/2} \leq Q_h(c) \leq 1 + 2.36 \cdot \sqrt{2c} \cdot m^{\mu(h)/2} \tag{3.9}$$

the upper bound being true under the additional assumption that $c \leq 0.0006$ and $m^{\mu(h)} \geq 2,200$. These results indicate that $Q_h(c) = \Theta(\sqrt{c}) \cdot m^{\mu(h)/2}$.

The above theorems show that a collision is expected in about $m^{\mu(h)/2}$ trials. Hence it is expected that the birthday attack fares better on hash functions with low balance as compared to highly balanced ones. Therefore designers should consider high balance as one of the design criteria for hash functions.

Regular vs. Random Functions A symmetry argument shows that if $h_1, h_2 : X \to Y$ are regular functions, then $C_{h_1}(q) = C_{h_2}(q)$. Denote this value by $C^{reg}_{X,Y}(q)$. Similarly $Q_{h_1}(c) = Q_{h_2}(c)$ and denote this value by $Q^{reg}_{X,Y}(c)$. It can be shown that

Proposition 3.2.3. *If $h : X \to Y$ is a hash functions then $C_h(q) > C^{reg}_{X,Y}(q)$ and $Q_h(c) < Q^{reg}_{X,Y}(c)$.*

In other words, regular functions are the best with regard to security against the birthday attack.

Designers of hash functions often have as target to make the hash function have "random behaviour". To assess how this impacts their security against the birthday attack consider the performance of the birthday attack when the function h is random. Let h be chosen randomly from $\mathcal{F}^{X,Y}$. Let $C^{\$}_{X,Y}(q)$ denote the probability that the attack succeeds in q trials. The probability is over the initial choice of h and the choice of $x_1, x_2, \cdots, x_q$ made in the attack. Let $Q^{\$}_{X,Y}(c)$ denote the smallest value of q for which $C^{\$}_{X,Y}(q) \geq c$. Then it can be shown that

Proposition 3.2.4. *$C^{\$}_{X,Y}(q) > C^{reg}_{X,Y}(q)$ and $Q^{\$}_{X,Y}(c) < Q^{reg}_{X,Y}(c)$*

This indicates that random functions offer less security than regular functions against the birthday attack. Hence designing hash functions that have random behaviour subject to being regular would improve security slightly.

Does MD transform preserve balance? Many popular hash functions are designed by applying the Merkle-Damgård (MD) transform to an underlying compression function. The results in [BK04] show that the MD transform does not preserve regularity or maintain balance. This indicates a weakness in the MD transform based design principle from the point of view of ensuring high balance. Also, it is shown that regularity not only of the compression function but also of certain associated functions does suffice to guarantee regularity of the hash function. The conclusion is that a general design principle, attempting to ensure high balance of a hash function by only establishing some properties of the compression function, cannot be recommended.

Chapter 4

Multicollisions

The notion of collision can be generalized to that of r-way collision (or r-collision). An r-collision is simply an r-tuple of distinct domain points $x_1, x_2, \cdots, x_r$ such that, $h(x_1) = h(x_2) = \cdots = h(x_r)$. If the hash values behave almost randomly, finding an r-collision could be done by hashing about $m^{(r-1)/r}$ points in the domain. This tends to m when r becomes large. Due to this fact relying on r-collision freeness in cryptographic constructions seems to be a good way to gain more security without increasing the size of the hash functions.

The intuition behind relying on r-collision freeness is that finding multicollisions is harder than finding collisions. This is true for a truly random function. But many of the hash functions used in practice are based on an iterated construction and Joux [Jou04] has demonstrated that r-collisions in iterated hash functions are not much harder to find than ordinary collisions, even for very large values of r.

4.1 Multicollision attacks on iterated hash functions

We now give a brief summary of Joux's attack. We will ignore the padding process in the MD-transform as long as we consider collisions between messages of same length since collisions without padding lead to collisions with padding on messages of same length. Using Joux's attack, one can find a 2^t-collision for an iterated hash function h that costs t times as much as building ordinary 2-collisions. Let f denote the compression function and let $H_0, H_1, \cdots$ denote the chaining values. Assume that we have access to a collision finding machine C, that given as input a chaining value H, outputs two different message blocks M and M' such that $f(H, M) = f(H, M')$. This collision finding machine may use the generic birthday attack or any specific attack based on a weakness of f. Using t-calls to C, we can build 2^t-collisions for h using the following method:

- Let H_0 be the initial value IV of h.
- For $i = 1, \cdots, t$, do

- Call C and find M_i and M_i' such that $f(H_{i-1}, M_i) = f(H_{i-1}, M_i')$
- Let $H_i = f(H_{i-1}, M_i)$.

- Output the 2^t messages of the form $(m_1, m_2, \cdots, m_t)$ where m_i is one of M_i and M_i'.

Following Joux's attack, Nandi and Stinson [NS07] studied multicollision attacks in a more general class of hash functions called *generalized sequential hash functions*. They showed the existence of multicollision attacks for this class of hash functions provided that every message block is used at most twice in the computation of the message digest. The expected complexity of their attack is $O\left(t^2 \ln t \left(\log_2 m + \ln \ln t\right)\sqrt{m}\right)$ to find 2^t-collisions. Thus they rule out a large class of hash functions as candidates for multicollision secure hash functions. These attacks were extended and generalized by Hoch and Shamir [HS06] to tree-based hash functions that have a fixed *expansion rate*, which indicates the maximum number of times a message block is processed in the evaluation of a message digest.

If a hash function is truly random then birthday attack is the best known attack for finding multicollisions. The problem of finding multicollisions is analogous to the *generalized birthday problem* which is well studied in both statistics and cryptography literature. The following subsections summarize different results concerning the probability of finding r-collisions when the hash function has random behaviour.

4.2 Generalized Birthday Problem

The problem of finding r-collisions is analogous to the *generalized birthday problem* described as follows:

> q people are selected at random. What is the probability that at least r of them will have the same birthday? What is the smallest value of q such that the probability is greater than or equal to $1/2$ that at least r people have the same birthday?

This problem is abstractly equivalent to the scheme of placing q balls in m cells. In the following section we will discuss McKinney's solution [McK66] to this problem.

4.2.1 Exact probability of finding multicollisions

The solution to the problem given by McKinney is as follows.

Let X_i $(i = 1, 2, \cdots, q)$ be independent, identically distributed random variables with uniform distribution over the set of cells. If $y_1, y_2, \cdots, y_m$ denote the cells, then

$$\Pr[X_i = y_j] = \frac{1}{m}.$$

The exact probability that r or more X_i's are equal is to be determined. Let the event E be defined as "no r of the random variables X_i's are equal" (or equivalently, at most $r-1$ of the X_i's are equal). Then

$$\Pr[r \text{ or more } X_i\text{'s are equal}] = 1 - \Pr[E]$$

$\Pr[E]$ is then computed by summing the probabilities of all ways in which q random variables can take on less than r equal values.

For a given q, let

q_1 = number of non-repeated X_i's
q_2 = number of pairs of equal X_i's
$\cdots$
q_{r-1} = number of $(r-1)$-tuples of equal X_i's, where

$$q = \sum_{i=1}^{r-1} i q_i. \tag{4.1}$$

Then $\Pr[E]$ is given by $\sum \Pr[q;\ q_1, q_2, \cdots, q_{r-1}]$ where the summation extends over all q_i $(i = 1, 2, \cdots, q_{r-1})$ which satisfy Equation (4.1). The general term of the summation is the probability that there are exactly q_1 non-repeated items, q_2 pairs, $\cdots$, q_{r-1} $(r-1)$-tuples of equal X_i's. This takes the form

$$\Pr[q;\ q_1, q_2, \cdots, q_{r-1}] = \frac{q!}{\prod_{j=1}^{r-1}(q_j!)(j!)^{q_j}} \cdot \frac{\mathrm{P}\left(m, \sum_{i=1}^{r-1} q_i\right)}{m^q}, \tag{4.2}$$

where $\mathrm{P}(a, b)$ denotes the number of permutations of a things taken b at a time.

4.2.2 A representation using multinomial c.d.f.'s

The multinomial distribution is described in [Fel08] as follows.

Consider a succession of N independent trials where each trial can have one of several outcomes. Denote the possible outcomes of each trial by $E_1, E_2, \cdots, E_t$ and suppose that the probability of the realization of E_i in each trial is p_i $(i = 1, \cdots, t)$. For $t = 2$, we have Bernoulli trials. In general, the numbers p_i are subject only to the condition

$$p_1 + \cdots + p_t = 1$$

Let $N_1, N_2, \cdots, N_t$ denote the number of occurrences of the events $E_1, E_2, \cdots, E_t$ respectively. $N_1, N_2, \cdots, N_t$ are said to follow a t-category multinomial distribution with *sample size* N and *parameters* $p_1, p_2, \cdots, p_t$. The p.m.f. is given by the probability that in N trials, E_1 occurs k_1 times, E_2 occurs k_2 times, etc., which takes the form

$$\Pr[N_1 = k_1, N_2 = k_2, \cdots, N_t = k_t] = \frac{N!}{k_1! k_2! \cdots k_t!} p_1^{k_1} p_2^{k_2} \cdots p_t^{k_t},$$

where the k_i's are arbitrary non-negative integers subject to the obvious condition

$$k_1 + k_2 + \cdots + k_t = N.$$

The multinomial cumulative distribution function is defined as follows.

$$\Pr[N_1 \leq a_1, \cdots, N_t \leq a_t] = \sum_{\substack{k_1 \leq a_1 \\ \cdots \\ k_t \leq a_t}} \Pr[N_1 = k_1, N_2 = k_2, \cdots, N_t = k_t]$$

One can express $\Pr[E]$ (defined in Section 4.2.1) using a multinomial cumulative distribution function. Let $N_1, N_2, \cdots, N_m$ denote the number of balls in cells $y_1, y_2, \cdots, y_m$ respectively. Then $(N_1, N_2, \cdots, N_m)$ will have an m-category multinomial distribution with sample size q and parameters $p_i = \frac{1}{m}$ $(i = 1, \cdots, m)$. Then $\Pr[E]$ can be expressed as follows.

$$\Pr[E] = \Pr[N_1 \leq r-1, N_2 \leq r-1, \cdots, N_m \leq r-1]$$

Levin [Lev81] provides an efficient way to compute a multinomial c.d.f. by expressing it as the conditional distribution of independent Poisson random variables given fixed sum. The previous methods could be applied only in the equiprobable case whereas Levin's approximation works even when probabilities are different.

Theorem 4.2.1. *Let $(N_1, N_2, \cdots, N_t)$ have a t-category multinomial distribution with sample size N and parameters $(p_1, p_2, \cdots, p_t)$. Let $(a_1, a_2, \cdots, a_t)$ be non-negative integers, and define*

$$p_N = \Pr[N_1 \leq a_1, \cdots, N_t \leq a_t].$$

Then for any real number $s > 0$,

$$p_N = \frac{N!}{s^N e^{-s}} \left(\prod_{i=1}^{t} \Pr[X_i \leq a_i] \right) \Pr[W = N], \tag{4.3}$$

where $X_i \sim$ indep $\mathcal{P}(sp_i)$ = independent Poisson r.v.'s with mean sp_i and W is a sum of independent truncated Poisson r.v.'s, namely $W = \sum_{i=1}^{t} Y_t$ where $Y_i \sim \mathcal{TP}_{a_i}(sp_i)$ = truncated Poisson(sp_i) with range $0, 1, \cdots, a_i$.

Proof. The theorem may be proved by applying Bayes' Theorem to the usual representation of the multinomial frequencies conditional on their sum being fixed. Let A_i denote the event $X_i \sim$ indep $\mathcal{P}(sp_i)$. Then the multinomial c.d.f. is

$$\Pr[A_1 \cdots A_t | \Sigma_1^t X_i = N] = \frac{\Pr[A_1 \cdots A_t]}{\Pr[\Sigma_1^t X_i = N]} \Pr[\Sigma_1^t X_i = N | A_1 \cdots A_t].$$

The result follows by noting that $\sum X_i \sim$ indep $\mathcal{P}(s)$ and that the conditional distribution of X_i given A_i is $\mathcal{TP}_{a_i}(sp_i)$. □

For large t the Central Limit Theorem offers an approximation to the last term i.e., $\Pr[W = N]$. Levin suggests an Edgeworth expansion which provides better accuracy than just a first order normal approximation.

Diaconis and Mosteller [DM89] suggest an approximation that is valid for fixed r and large m. The number of balls required to have probability p of r or more balls in the same cell is approximately given by solving for q in

$$qe^{-q/(rm)}\left(1-\frac{q}{(r+1)m}\right)^{-1/r} \approx \left(m^{(r-1)}r!\ln\left(\frac{1}{1-p}\right)\right)^{1/r} \tag{4.4}$$

It follows from the above expression that for fixed p, the complexity of finding an r-collision is $\Theta(rm^{(r-1)/r})$ using the birthday attack. For fixed p and r, the complexity is $\Theta(m^{(r-1)/r})$.

4.3 Complexity of finding r-collisions

As mentioned in the Section 4.2.2, the approximate complexity of finding an r-collision using the birthday attack is $\Theta(rm^{(r-1)/r})$. We will now give a proof of this result which appears in [Pre93]. The following notation will be used in the analysis: for any positive integers d and r such that $d \geq r \geq 2$, $(d)_r = d(d-1)\cdots(d-r+1)$.

Theorem 4.3.1. *Let q balls be distributed in m cells independently and uniformly at random. Then the number t of cells containing exactly r balls is given by*

$$\binom{m}{t}\frac{(q)_{tr}}{(r!)^t}\frac{\left(1-\frac{t}{m}\right)^{q-tr}}{m^{tr}}\sum_{v=0}^{m-t}(-1)^v\binom{m-t}{v}\frac{(q-tr)_{vr}}{(r!)^v m^{vr}}\frac{\left(1-\frac{v}{m-t}\right)^{q-(v+t)r}}{\left(1-\frac{t}{m}\right)^{vr}} \tag{4.5}$$

Proof. We start by calculating the probability that one of the cells contains exactly r balls.

$$p_1 = \binom{q}{r}\frac{(m-1)^{q-r}}{m^q}.$$

Here $\binom{q}{r}$ is the number of selections of r balls and $(m-1)^{q-r}$ is the number of ways in which the remaining $q-r$ balls can be distributed over the remaining $m-1$ cells. The total number of distributions is given by m^q. Similarly, for two cells this probability is equal to

$$p_2 = \frac{\binom{q}{r}\binom{q-r}{r}(m-2)^{q-2r}}{m^q}.$$

The product of the two binomial coefficients can be simplified to

$$\frac{q!}{(r!)^2(q-2r)!} = \frac{(q)_{2r}}{(r!)^2}$$

The general expression for $v \leq m$ is then

$$p_v = \frac{(q)_{vr}}{(r!)^v}\frac{\left(1-\frac{v}{m}\right)^{q-vr}}{m^{vr}} \tag{4.6}$$

The probability that v cells contain r balls is given by

$$S_v = \binom{m}{v} p_v, \tag{4.7}$$

as there are $\binom{m}{v}$ ways to select v cells out of m. The next step is to calculate the probability that no cell contains exactly r balls. This can be done using the inclusion-exclusion principle:

$$P(m,q,r,0) = \sum_{v=0}^{m-1} (-1)^v S_v. \tag{4.8}$$

Consider now a distribution where t cells contain exactly r balls. These t cells can be chosen in $\binom{m}{t}$ ways and the balls in these cells can be chosen in $(q)_{tr}/(r!)^t$ ways. The remaining $q - tr$ balls are distributed over the remaining cells so that none of these cells contains r balls; the number of such distributions is $(m-t)^{q-tr} P(m - t, q - tr, r, 0)$. Dividing by m^q one obtains for the probability that exactly t cells contain r balls as

$$\begin{aligned} & P(m,q,r,t) \\ &= \frac{1}{m^q}\binom{m}{t}\frac{(q)_{tr}}{(r!)^t}(m-t)^{q-tr}P(m-t,q-tr,r,0) \\ &= \binom{m}{t}\frac{(q)_{tr}}{(r!)^t}\frac{\left(1-\frac{t}{m}\right)^{q-tr}}{m^{tr}}\sum_{v=0}^{m-t}(-1)^v\binom{m-t}{v}\frac{(q-tr)_{vr}}{(r!)^v m^{vr}}\frac{\left(1-\frac{v}{m-t}\right)^{q-(v+t)r}}{\left(1-\frac{t}{m}\right)^{vr}} \end{aligned}$$

This completes the proof. □

It is not feasible to evaluate (4.5) for large values of m and q. We will hence study the asymptotic behaviour of this function. Let $r > 1$. If q/m is too small, then we can expect no cells containing r balls; in this case $P(m,q,r,0)$ is near unity and all $P(m,q,r,t)$ with $t \geq 1$ are very small. If q/m is very large, then most cells will contain about $r = q/m$ balls. We will discuss only the intermediate case.

Theorem 4.3.2. *Let q balls be distributed in m cells independently and uniformly at random. Then the number t of cells containing exactly r balls follows asymptotically a Poisson distribution with*

$$P(m,q,r,t) = e^{-\lambda_r}\frac{{\lambda_r}^t}{t!} \text{ and } \lambda_r = \frac{me^{-\frac{q}{m}}}{r!}\left(\frac{q}{m}\right)^r \tag{4.9}$$

This holds when q and m tend to infinity such that λ_r remains bounded.

Proof. First we will estimate the quantity S_v of (4.7). Based on the inequality $(x)_k \leq x^k$ for $s \geq 1$ one obtains

$$v! S_v \leq \frac{m^v}{m^{vr}}\frac{q^{vr}}{(r!)^v}\left(1-\frac{v}{m}\right)^{q-vr}$$

From Taylor's expansion, for $0 < x < 1$, we have

$$\frac{-x}{1-x} < \ln(1-x) < -x. \tag{4.10}$$

Therefore

$$v!S_v < \frac{m^v}{(r!)^v}\left(\frac{q}{m}\right)^{vr} e^{-\left(\frac{q-vr}{m}\right)v}.$$

Using the inequality $(x)_k \geq (x-k)^k$ for $s \geq 1$, we get the lower bound as follows.

$$v!S_v \geq \frac{(m-v)^v}{m^{vr}}\frac{(q-vr)^{vr}}{(r!)^v}\left(1-\frac{v}{m}\right)^{q-vr}$$

Using (4.10), we get

$$v!S_v > \frac{m^v}{(r!)^v}\left(\frac{q}{m}\right)^{vr}\left(1-\frac{vr}{q}\right)^{vr} e^{-\left(\frac{q-v(r-1)}{m-v}\right)v}.$$

Now define

$$\lambda_r = \frac{me^{-\frac{q}{m}}}{r!}\left(\frac{q}{m}\right)^r$$

and suppose that q and m increase in such a way that λ_r remains constrained to a finite interval $0 < a < \lambda_r < b$. For each fixed v the ratio of the upper and lower bounds tends to unity, under the condition that $vr \ll q$. Hence

$$0 \leq \frac{{\lambda_r}^v}{v!} - S_v \to 0. \tag{4.11}$$

This relation holds trivially when $\lambda_r \to 0$ Hence (4.11) holds whenever q and m increase in such a way that λ_r remains bounded. Now

$$e^{-\lambda_r} - P(m,q,r,0) = \sum_{v=0}^{\infty}(-1)^v\left(\frac{{\lambda_r}^v}{v!} - S_v\right) \tag{4.12}$$

and (4.11) implies that the right side tends to zero. The observation that (4.5) can be rewritten as $S_t P(m,q,r,0)$ shows that for each fixed t

$$P(m,q,r,t) - e^{-\lambda_r}\frac{{\lambda_r}^t}{t!} \to 0.$$

This completes the proof. □

The probability that exactly one r-collision occurs is $\lambda_r e^{-\lambda_r}$ and the probability that at least one r-collision occurs is $1 - e^{\lambda_r}$. If one wants to calculate the number of trials in order to have at least one r-collision with probability $1 - e^{-1} \approx 0.63$, one has to solve numerically the equation $\lambda_r = 1$ or

$$qe^{-\frac{q}{mr}} = m^{\frac{r-1}{r}}(r!)^{\frac{1}{r}}.$$

If $q \leq m$ then the following is a good approximation.

$$q \approx m^{\frac{r-1}{r}}(r!)^{\frac{1}{r}}.$$

This shows that the number of trials is $\Theta(rm^{(r-1)/r})$.

Chapter 5

Balance-based Analysis of Generalized Birthday Attack

As mentioned in Chapters 1 and 4, r-collision freeness is a desirable security property of a hash function. One natural question that arises from the discussions in the previous sections is the following.

> Can we extend balance-based analysis to the generalized birthday attack? What would be the equivalent notion of balance in the context of r-collisions?

The generalized birthday attack, that we consider, for finding r-collisions ($r \geq 2$) for a given hash function $h : X \rightarrow Y$ is shown in Figure 5.1. It picks points $x_1, x_2, \cdots, x_q$ from the domain X independently and uniformly at random. If any r of these points map to the same range point then it returns them, and if no r-collision is found it returns "failure". The integer $q \geq r$ is the number of trials. Note that for $r = 2$, the algorithms given in Figures 5.1 and 3.1 are the same.

Our goal here is to analyze the performance of the generalized birthday attack on a hash function $h : X \rightarrow Y$ in terms of what we call r-balance of h. Equivalently, we want to analyze how the following metrics vary with r-balance.

- $C_h^{(r)}(q)$: probability that the birthday attack of Figure 5.1 successfully finds an r-collision for h in q trials ($q \geq r$).
- $Q_h^{(r)}(c)$: the minimum number of trials required to obtain an r-collision with probability greater than or equal to c. That is,

$$Q_h^{(r)}(c) = \min\{q : C_h^{(r)}(q) \geq c\}. \tag{5.1}$$

Note that, for a balance-based analysis of the generalized birthday attack, the definition of balance given in [BK04] will not suffice. We need an equivalent formulation in the context of r-collisions. In the following section we will provide the definition of r-balance and discuss its properties.

For $i = 1, ..., q$ do

$x_i \xleftarrow{\$} X$

$y_i \leftarrow h(x_i)$

EndFor

If there are indices $i_1, \cdots, i_r$ such that $h(x_{i_1}) = \cdots = h(x_{i_r})$ and $x_{i_1}, \cdots, x_{i_r}$ are distinct

then return $x_{i_1}, \cdots, x_{i_r}$

EndIf

Return "failure"

Figure 5.1: Generalized birthday attack on a hash function $h : X \to Y$ for finding r-collisions.

5.1 Notation

We will use the following notation for our analysis.

- If d is a non-negative integer, then $[d] = \{1, 2, \cdots, d\}$.
- For an integer $r \geq 2$, $[d]_r$ denotes the set of all r-element subsets of $[d]$.
- $[d]_{r,2}$ denotes the set of all 2-element subsets of $[d]_r$.
- For any $y \in Y$, $h^{-1}(y) = \{x \in X : h(x) = y\}$.
- Let $r \geq 2$ and $d \geq 0$ be integers. Then $(d)_r$ is defined as follows.

$$(d)_r = \begin{cases} d(d-1)\cdots(d-r+1) & \text{if } d \geq r \\ 0 & \text{otherwise} \end{cases}$$

- $\mathrm{P}(a, b)$ denotes the number of permutations of a things taken b at a time. Clearly $\mathrm{P}(a, b) = (a)_b$.

5.2 r-balance and its properties

Recall that in [BK04], balance was defined as a function of the probability that two uniformly chosen (with replacement) domain points map to the same range point. It is defined this way because in case of 2-collisions the probability that the domain points coincide can be ignored compared to the probability of r-collisions when the size of the domain is at least twice the size of the range. But in case of r-collisions, where r is large, we cannot ignore the probability of domain points coinciding. So if we are looking for r-collisions, a natural way to define the r-balance of h would be in terms of the probability of finding r-collisions for h.

Definition 5.2.1. Let $h : X \to Y$ be a hash function whose domain X and range $Y = \{y_1, y_2, \cdots, y_m\}$ have sizes $n, m \geq r$, respectively. For $i \in [m]$, let $n_i = |h^{-1}(y_i)|$ denote the size of the preimage of y_i under h. The *r-balance of h*, denoted $\mu_r(h)$, is defined as

$$\mu_r(h) = \frac{1}{r-1} \cdot \log_m \left(\frac{1}{p_r} \right), \tag{5.2}$$

where p_r denotes the probability of an r-collision when r elements are chosen independently and uniformly at random from the domain X.

For $r = 2$, we have

$$\begin{aligned} m^{-\mu_2(h)} &= \frac{\sum_{i=1}^{m} n_i(n_i - 1)}{n^2} \\ &= \frac{\sum_{i=1}^{m} {n_i}^2}{n^2} - \frac{\sum_{i=1}^{m} n_i}{n} \\ &= m^{-\mu(h)} - \frac{1}{n} \end{aligned}$$

This shows that $\mu_2(h)$ is always greater than $\mu(h)$. The difference gets smaller as n grows larger.

Proposition 5.2.1. *Let r elements be chosen independently and uniformly at random from the domain X. The probability that they form an r-collision is determined by*

$$p_r = \frac{\sum_{i=1}^{m} (n_i)_r}{n^r}.$$

Proof. Let r elements $w_1, w_2, \cdots, w_r$ be picked independently and uniformly at random from the domain X. Let E be the event that these elements form an r-collision. Let A denote the event that these are distinct and for $1 \leq i \leq m$, let B_i be the event that $h(w_1) = \cdots = h(w_r) = y_i$. Then

$$E = AB_1 \cup AB_2 \cup \cdots \cup AB_m.$$

Since B_i's are mutually exclusive events, we have

$$\begin{aligned} \Pr[E] &= \sum_{i=1}^{m} \Pr[AB_i] \\ &= \sum_{i=1}^{m} \Pr[A|B_i] \cdot \Pr[B_i] \\ &= \sum_{i=1}^{m} \frac{n_i(n_i - 1) \cdots (n_i - r + 1)}{n_i^r} \cdot \frac{n_i^r}{n^r} \\ &= \sum_{i=1}^{m} \frac{n_i(n_i - 1) \cdots (n_i - r + 1)}{n^r} \end{aligned}$$

Since $p_r = \Pr[E]$, the proposition follows. □

The following lemma will be used in obtaining bounds on the r-balance of a hash function.

Lemma 5.2.2. *Let $r \geq 2$ be an integer. Let $n_1, n_2, \cdots, n_m$ be non-negative integers such that $\sum_{i=1}^{m} n_i = n$. Then*

$$m \cdot \left(\frac{n}{m}\right)_r \leq \sum_{i=1}^{m} (n_i)_r \leq (n)_r$$

Proof. We will prove the bounds using a counting argument. Let $S(n_i)$ denote the set of all distinct arrangements of n_i things taken r at a time. Then $|S(n_i)| = \mathrm{P}(n_i, r)$ for $i = 1, \cdots m$. If $n_j \leq r-1$ for some j then $S(n_j) = \emptyset$. Assume, without loss of generality, that the first k of the n_i's are greater than $r-1$. By definition $n = \sum_{i=1}^{m} n_i$. Let S denote the set of all distinct arrangements of n things taken r at a time. Each arrangement in $S(n_i)$ is also present in S. This show that $S(n_1) \cup S(n_2) \cup \cdots \cup S(n_k) \subseteq S$. Also since the $S(n_i)$'s are disjoint, we have

$$\mathrm{P}(n_1, r) + \mathrm{P}(n_2, r) + \cdots + \mathrm{P}(n_k, r) \leq \mathrm{P}(n_1 + n_2 + \cdots + n_k, r) = \mathrm{P}(n, r)$$

Equality occurs when $k = 0$ i.e., one of the n_i's is equal to n and the rest are zero. This gives an upper bound on $\sum_{i=1}^{m} (n_i)_r$.

$$\sum_{i=1}^{m} (n_i)_r \leq (n)_r \tag{5.3}$$

Now we claim that $\sum_{i=1}^{m} (n_i)_r$ attains its minimum when all n_i's are equal i.e., $n_1 = n_2 = \cdots = n_m = \frac{n}{m}$. Suppose there exist n_i and n_j such that $n_i > \frac{n}{m}$ and $n_j < \frac{n}{m}$. Assume, without loss of generality, that $i = 1$ and $j = 2$. To prove the claim, we need but show that

$$\mathrm{P}(n_1 - 1, r) + \mathrm{P}(n_2 + 1, r) + \cdots + \mathrm{P}(n_k, r) < \mathrm{P}(n_1, r) + \mathrm{P}(n_2, r) + \cdots + \mathrm{P}(n_k, r).$$

Let T_i denote the set containing n_i items. Clearly, $T_1 \cup T_2 \cup \cdots \cup T_m = X$. Let $x \in T_1$. The number of arrangements of items in T_1 taken r at a time that contain x is equal to $r\mathrm{P}(n_1 - 1, r - 1)$. Suppose we remove x from T_1 and put it in T_2. Then the number of arrangements of items in T_2 taken r at a time that contain x is equal to $r\mathrm{P}(n_2, r-1)$. Thus we have

$$\begin{aligned}
&(\mathrm{P}(n_1, r) + \mathrm{P}(n_2, r) + \cdots + \mathrm{P}(n_k, r)) - (\mathrm{P}(n_1 - 1, r) + \mathrm{P}(n_2 + 1, r) + \cdots + \mathrm{P}(n_k, r)) \\
&= |S(n_1) \cup S(n_2) \cup \cdots \cup S(n_m)| - |S(n_1 - 1) \cup S(n_2 + 1) \cup \cdots \cup S(n_m)| \\
&= |S(n_1) \cup S(n_2)| - |S(n_1 - 1) \cup S(n_2 + 1)| \\
&= |S(n_1 - 1)| + r\mathrm{P}(n_1 - 1, r - 1) + |S(n_2)| - |S(n_1 - 1)| - |S(n_2)| - r\mathrm{P}(n_2, r - 1) \\
&= r\mathrm{P}(n_1 - 1, r - 1) - r\mathrm{P}(n_2, r - 1) \\
&> 0
\end{aligned}$$

since $n_1 - 1 > n_2$. Thus we have a lower bound on $\sum_{i=1}^{m}(n_i)_r$.

$$\sum_{i=1}^{m}(n_i)_r \geq m \cdot \left(\frac{n}{m}\right)_r \tag{5.4}$$

From (5.3) and (5.4), we have

$$m \cdot \left(\frac{n}{m}\right)_r \leq \sum_{i=1}^{m}(n_i)_r \leq (n)_r$$

□

Now we present upper and lower bounds on the r-balance of a hash function and see when they are attained.

Proposition 5.2.3. *Let h be a hash function. Then*

$$\frac{1}{r-1}\log_m \frac{n^r}{(n)_r} \leq \mu_r(h) \leq \frac{1}{r-1}\log_m \frac{n^r}{m \cdot \left(\frac{n}{m}\right)_r} \tag{5.5}$$

The lower bound is attained when h is a constant function and the upper bound is attained when h is a regular function.

Proof. From Lemma 5.2.2, we have

$$m \cdot \left(\frac{n}{m}\right)_r \leq \sum_{i=1}^{m}(n_i)_r \leq (n)_r$$

Dividing throughout by n^r we get,

$$\frac{m \cdot \left(\frac{n}{m}\right)_r}{n^r} \leq \frac{\sum_{i=1}^{m}(n_i)_r}{n^r} \leq \frac{(n)_r}{n^r}$$

$$\frac{n^r}{(n)_r} \leq \frac{n^r}{\sum_{i=1}^{m}(n_i)_r} \leq \frac{n^r}{m \cdot \left(\frac{n}{m}\right)_r}$$

Taking $\log_m$ and dividing by $r-1$ we obtain the required bounds on r-balance. From Lemma 5.2.2, we know that the lower bound of $\sum_{i=1}^{m}(n_i)_r$ is attained when h is a regular function and the upper bound is attained when h is a constant function. From this the proposition follows. □

Note that the lower bound in Equation (5.5) is close to zero and the upper bound is close to 1 when the size of the domain n is substantially large.

5.3 Bounds on $C_h^{(r)}(q)$

In this section we provide upper and lower bounds on $C_h^{(r)}(q)$ as functions of the r-balance. Consider the generalized birthday attack as shown in Figure 5.1. For every $I \in [q]_r$, $I = \{i_1, i_2, \cdots, i_r\}$, define a random variable Z_I as follows.

$$Z_I = \begin{cases} 1 & \text{if } x_{i_1}, x_{i_2}, \cdots, x_{i_r} \text{ form an } r\text{-collision} \\ 0 & \text{otherwise} \end{cases}$$

From Proposition 5.2.1 and the definition of r-balance we have

$$\mathbf{E}[Z_I] = \Pr[Z_I = 1] = \frac{\sum_{i=1}^{m}(n_i)_r}{n^r} = m^{-(r-1)\mu_r(h)} = p_r \tag{5.6}$$

Let $Z = \sum_{I \in [q]_r} Z_I$. Z denotes the number of r-collisions.

Theorem 5.3.1 (Upper Bound on $C_h^{(r)}(q)$)**.** *Let $h : X \to Y$ be a hash function with $|X| = n$ and $|Y| = m$. Assume $m \geq r \geq 2$. Then for any integer $q \geq r$*

$$C_h^{(r)}(q) \leq \binom{q}{r} m^{-(r-1)\mu_r(h)}. \tag{5.7}$$

Proof. By linearity of expectation we have

$$\mathbf{E}[Z] = \sum_{I \in [q]_r} \mathbf{E}[Z_I] = \binom{q}{r}\mathbf{E}[Z_I] = \binom{q}{r} \cdot p_r \tag{5.8}$$

The upper bound is obtained by a direct application of Markov's inequality.

$$C_h^{(r)}(q) = \Pr[Z \geq 1] \leq \frac{\mathbf{E}[Z]}{1} = \binom{q}{r} \cdot p_r \tag{5.9}$$

□

To obtain a lower bound on $C_h^{(r)}(q)$, we will need the following lemma.

Lemma 5.3.2. *Let ℓ be an integer such that $\ell > r$. Then*

$$p_\ell \leq m^{-\ell(\frac{r-1}{r})\mu_r(h)}$$

Proof. By definition,

$$p_\ell = \frac{\sum_{i=1}^{m}(n_i)_\ell}{n^\ell}.$$

We need an upper bound on $\sum_{i=1}^{m}(n_i)_\ell$ in terms of $\mu_r(h)$, m and r. From Lemma 5.2.2 we know that $\sum_{i=1}^{m}(n_i)_\ell$ attains a maximum when $n_1 = n$ and all the remaining n_i's

are zero. When this happens, p_r will be determined by $p_r = \frac{n(n-1)\cdots(n-r+1)}{n^r}$ and $(n)_r = n^r \cdot p_r$.

$$p_r = \left(1 - \frac{1}{n}\right)\left(1 - \frac{2}{n}\right)\cdots\left(1 - \frac{r-1}{n}\right) \leq \left(1 - \frac{r-1}{n}\right)^r$$

$$p_r{}^{1/r} \leq \left(1 - \frac{r-1}{n}\right)$$

For all $j > r-1$,

$$\left(1 - \frac{j}{n}\right) > \left(1 - \frac{r-1}{n}\right) \tag{5.10}$$

Using Equation (5.10), we have

$$\begin{aligned}
(n)_\ell &= (n)_r \cdot (n-r)(n-r-1)\cdots(n-\ell+1) \\
&= n^r \cdot p_r \cdot n^{\ell-r}\left(1 - \frac{r}{n}\right)\left(1 - \frac{r+1}{n}\right)\cdots\left(1 - \frac{\ell-1}{n}\right) \\
&\leq n^\ell \cdot p_r \cdot \left(1 - \frac{r-1}{n}\right)^{\ell-r} \\
&\leq n^\ell \cdot p_r \cdot p_r{}^{\frac{\ell-r}{r}} \\
&\leq n^\ell \cdot p_r{}^{\ell/r}
\end{aligned}$$

The bound can now be obtained as follows.

$$\begin{aligned}
p_\ell &= \frac{\sum_{i=1}^m (n_i)_\ell}{n^\ell} \\
&\leq \frac{n^\ell \cdot p_r{}^{\ell/r}}{n^\ell} \\
&\leq p_r{}^{\ell/r} \\
&= \left(m^{-(r-1)\mu_r(h)}\right)^{\ell/r} \\
&= m^{-\ell(\frac{r-1}{r})\mu_r(h)}
\end{aligned}$$

This completes the proof. □

Theorem 5.3.3 (Lower Bound on $C_h^{(r)}(q)$)**.** *Let $h : X \to Y$ be a hash function with $|X| = n$ and $|Y| = m$. Assume $m \geq r \geq 2$. Let $\alpha \geq 0$ be a real number such that*

$$r \leq q \leq \alpha \cdot m^{\left(\frac{r-1}{r}\right)\mu_r(h)}. \tag{5.11}$$

Then

$$C_h^{(r)}(q) \geq \left(\frac{3}{2} - \frac{1}{2}(\alpha+1)^r\right) \cdot \binom{q}{r} \cdot m^{-(r-1)\mu_r(h)}. \tag{5.12}$$

The reason for making the assumption (5.11) is as follows. In Sections 4.2 and 4.3, we stated that it would approximately take $\Theta(m^{(r-1)/r})$ trials to find a collision for a fixed probability. Intuitively one would expect that the number of trials needed when the balance is also considered would be roughly $\Theta(m^{\left(\frac{r-1}{r}\right)\mu_r(h)})$.

Now we will give a proof of Theorem 5.3.3.

Proof. Let $[q]_{r,2}$ denote the set of all 2-element subsets of $[q]_r$. By the principle of inclusion and exclusion, we have

$$\begin{aligned} C_h^{(r)}(q) &= \Pr\left[\bigvee_{I\in[q]_r} Z_I = 1\right] && (5.13)\\ &= \sum_{I\in[q]_r} \Pr[Z_I = 1] - \sum_{\substack{I,J\in[q]_r\\ I\neq J}} \Pr[Z_I = 1 \wedge Z_J = 1] \\ &\quad + \cdots + (-1)^{\binom{q}{r}-1} \Pr\left[\bigwedge_{I\in[q]_r} Z_I = 1\right] && (5.14)\end{aligned}$$

The first two terms in the above equation gives us a lower bound on $C_h^{(r)}(q)$.

$$C_h^{(r)}(q) \geq \sum_{I\in[q]_r} \Pr[Z_I = 1] - \sum_{\{I,J\}\in[q]_{r,2}} \Pr[Z_I = 1 \wedge Z_J = 1] \qquad (5.15)$$

Equation (5.8) tells us that the first term in the above equation is

$$\sum_{I\in[q]_r} \Pr[Z_I = 1] = \binom{q}{r} \Pr[Z_I = 1] = \binom{q}{r} \cdot p_r \qquad (5.16)$$

In order to obtain a lower bound we need an upper bound on the second term of Equation (5.15). We now claim that

$$\sum_{\{I,J\}\in[q]_{r,2}} \Pr[Z_I = 1 \wedge Z_J = 1] \leq \frac{1}{2}\binom{q}{r} \cdot p_r \cdot ((\alpha+1)^r - 1) \qquad (5.17)$$

Combining Equations (5.15), (5.16) and (5.17), we obtain the lower bound stated in Equation (5.12) as follows.

$$\begin{aligned} C_h^{(r)}(q) &\geq \binom{q}{r} \cdot p_r - \sum_{\{I,J\}\in[q]_{r,2}} \Pr[Z_I = 1 \wedge Z_J = 1] \\ &\geq \binom{q}{r} \cdot p_r - \frac{1}{2}\binom{q}{r} \cdot p_r \cdot ((\alpha+1)^r - 1) \\ &= \left(\frac{3}{2} - \frac{1}{2}(\alpha+1)^r\right) \cdot \binom{q}{r} \cdot p_r \end{aligned}$$

It remains to prove the claim stated in (5.17).

For $k = 0, 1, \cdots, r-1$, let N_k be the number of sets $\{I, J\} \in [q]_{r,2}$ such that $|I \cap J| = k$. The k common elements can be chosen in $\binom{q}{k}$ ways. The remaining $r-k$ elements in I can be chosen in $\binom{q-k}{r-k}$ ways and for each such I, we can chose the remaining $r-k$ elements in J in $\binom{q-r}{r-k}$ ways. But this way we are counting every unordered pair twice (i.e., $\{I, J\}$ and $\{J, I\}$ are indistinguishable but counted twice). Therefore, we have

$$N_k = \frac{1}{2}\binom{q}{k}\binom{q-k}{r-k}\binom{q-r}{r-k} = \frac{1}{2}\binom{q}{r}\binom{r}{k}\binom{q-r}{r-k} \tag{5.18}$$

We can now break up the second term in Equation (5.15) as follows:

$$\sum_{\{I,J\}\in[q]_{r,2}} \Pr[Z_I = 1 \wedge Z_J = 1] = \sum_{k=0}^{r-1} N_k \cdot \Pr[Z_I = 1 \wedge Z_J = 1 : |I \cap J| = k] \tag{5.19}$$

When $k = 0$, the events $Z_I = 1$ and $Z_J = 1$ are independent and hence for $k = 0$,

$$\Pr[Z_I = 1 \wedge Z_J = 1] = \Pr[Z_I = 1] \cdot \Pr[Z_J = 1] = {p_r}^2 \tag{5.20}$$

When $k \geq 1$, the events $Z_I = 1$ and $Z_J = 1$ indicate that the elements in I map to a common point and so do the elements in J. Since $I \cap J \neq \emptyset$, the common image of the elements of both I and J must be the same. Hence $\Pr[Z_I = 1 \wedge Z_J = 1]$ is the probability that the $2r-k$ distinct elements in $I \cup J$ form a $2r-k$-collision. That is,

$$\Pr[Z_I = 1 \wedge Z_J = 1] = p_{2r-k} \tag{5.21}$$

Combining Equations (5.18), (5.19), (5.20) and (5.21), we obtain the following:

$$\Pr[Z_I = 1 \wedge Z_J = 1] = N_0 \cdot {p_r}^2 + \sum_{k=0}^{r-1} N_k \cdot p_{2r-k} \tag{5.22}$$

To obtain an upper bound on the above expression, we need an upper bound on p_{2r-k}. From Lemma 5.3.2, we have

$$p_{2r-k} \leq m^{-(2r-k)(\frac{r-1}{r})\mu_r(h)} \tag{5.23}$$

Let

$$a = \sum_{\{I,J\}\in[q]_{r,2}} \Pr[Z_I = 1 \wedge Z_J = 1].$$

Combining Equations (5.11), (5.18), (5.22) and (5.23), we obtain

$$\begin{aligned}
a &= \frac{1}{2}\binom{q}{r}\binom{q-r}{r}{p_r}^2 + \frac{1}{2}\binom{q}{r}\sum_{k=1}^{r-1}\binom{r}{k}\binom{q-r}{r-k}p_{2r-k} \\
&\leq \frac{1}{2}\binom{q}{r}\left((q^r \cdot p_r)\cdot p_r + \sum_{k=1}^{r-1}\binom{r}{k}q^{r-k}\cdot m^{-(2r-k)(\frac{r-1}{r})\mu_r(h)}\right) \\
&= \frac{1}{2}\binom{q}{r}\left((q^r m^{-(r-1)\mu_r(h)})\cdot p_r + m^{-(r-1)\mu_r(h)}\sum_{k=1}^{r-1}\binom{r}{k}q^{r-k}\cdot m^{-(r-k)(\frac{r-1}{r})\mu_r(h)}\right) \\
&= \frac{1}{2}\binom{q}{r}\left(\left(qm^{-(\frac{r-1}{r})\mu_r(h)}\right)^r\cdot p_r + p_r\sum_{k=1}^{r-1}\binom{r}{k}\left(q\cdot m^{-(\frac{r-1}{r})\mu_r(h)}\right)^{r-k}\right)
\end{aligned}$$

Using the assumption that $q \leq \alpha \cdot m^{(\frac{r-1}{r})\mu_r(h)}$, we obtain

$$\begin{aligned}
a &\leq \frac{1}{2}\binom{q}{r}p_r\left(\alpha^r + \sum_{k=1}^{r-1}\binom{r}{k}\alpha^{r-k}\right) \\
&= \frac{1}{2}\binom{q}{r}p_r\left((\alpha+1)^r - 1\right)
\end{aligned}$$

which proves the claim in (5.17). □

Comparison with $C_h(q)$

Substituting $r = 2$ in (5.7) and (5.12), we get the following bounds on $C_h^{(2)}(q)$.

$$\left(1 - \frac{\alpha^2}{2} - \alpha\right)\cdot\binom{q}{2}\cdot m^{-\mu_2(h)} \leq C_h^{(2)}(q) \leq \binom{q}{2}\cdot m^{-\mu_2(h)}$$

the lower bound being true provided $q \leq \alpha \cdot m^{\mu_2(h)/2}$. Replacing α with 0.4 gives us

$$0.52\cdot\binom{q}{2}\cdot m^{-\mu_2(h)} \leq C_h^{(2)}(q) \leq \binom{q}{2}\cdot m^{-\mu_2(h)}$$

which are slightly better than the bounds on $C_h(q)$ stated in (3.6). The difference is due to the way we have defined $\mu_2(h)$. Note that we do not need any additional restrictions on the sizes of the range and domain for our bounds to hold.

5.4 Bounds on $Q_h^{(r)}(c)$

Now we will provide upper and lower bounds on $Q_h^{(r)}(c)$.

Theorem 5.4.1. *Let $h : X \to Y$ be a hash function with $|X| = n$ and $|Y| = m$. Assume $m \geq r \geq 2$. Let $\alpha \geq 0$ be a real number such that $\beta = \left(\frac{3}{2} - \frac{1}{2}(\alpha+1)^r\right) > 0$. Let c be a real number such that $0 \leq c < 1$. Then*

$$c^{1/r}\left(\frac{r}{e}\right)m^{\left(\frac{r-1}{r}\right)\mu_r(h)} \leq Q_h^{(r)}(c) \leq \left(\frac{c}{\beta}\right)^{1/r} rm^{\left(\frac{r-1}{r}\right)\mu_r(h)} \tag{5.24}$$

the upper bound being true under the additional assumption that

$$c \leq \left(\frac{\alpha}{r}\right)^r \cdot \beta \tag{5.25}$$

Proof. From Theorem 5.3.1 we have

$$C_h^{(r)}(q) \leq \underbrace{\binom{q}{r}m^{-(r-1)\mu_r(h)}}_{U(q)}$$

To get the lower bound of Equation (5.24) we need to solve for q in the equation $U(q) = c$.

$$\begin{aligned} c &= \binom{q}{r}m^{-(r-1)\mu_r(h)} \\ &\leq \left(\frac{qe}{r}\right)^r \frac{1}{m^{(r-1)\mu_r(h)}} \end{aligned}$$

$$q \geq c^{1/r}\left(\frac{r}{e}\right)m^{\left(\frac{r-1}{r}\right)\mu_r(h)}$$

This proves the lower bound of Equation (5.24). We now obtain the upper bound. From Theorem 5.3.3 we have

$$C_h^{(r)}(q) \geq \underbrace{\beta\binom{q}{r}m^{-(r-1)\mu_r(h)}}_{L(q)}$$

Solving for q in the equation $L(q) = c$ we get

$$\begin{aligned} c &= \beta\binom{q}{r}m^{-(r-1)\mu_r(h)} \\ &\geq \beta\left(\frac{q}{r}\right)^r \frac{1}{m^{(r-1)\mu_r(h)}} \end{aligned}$$

$$q \leq \underbrace{\left(\frac{c}{\beta}\right)^{1/r} \cdot r \cdot m^{\left(\frac{r-1}{r}\right)\mu_r(h)}}_{q_u}$$

r	$cmax_r$
2	2.34375×10^{-2}
3	4.32376×10^{-4}
4	4.28984×10^{-6}
5	2.66499×10^{-8}
6	1.13553×10^{-10}
7	3.52733×10^{-13}
8	8.34584×10^{-16}

Table 5.1: Table of values of $cmax_r$ for r ranging from 1 to 8.

This proves the upper bound of Equation (5.24) provided that $q = q_u$ meets the restriction stated in Equation (5.11).

$$\begin{aligned} q &\leq \left(\frac{c}{\beta}\right)^{1/r} r m^{\left(\frac{r-1}{r}\right)\mu_r(h)} \\ &\leq \left(\left(\frac{\alpha}{r}\right)^r \beta\right)^{1/r} \left(\frac{1}{\beta}\right)^{1/r} r m^{\left(\frac{r-1}{r}\right)\mu_r(h)} \\ &\leq \alpha m^{\left(\frac{r-1}{r}\right)\mu_r(h)}. \end{aligned}$$

Thus Equation (5.25) is true. □

Comparison with $Q_h(c)$

For $r = 2$, (5.24) gives the following bounds on $Q_h^{(2)}(c)$.

$$\sqrt{c} \cdot \frac{2}{e} \cdot m^{\mu_2(h)} \leq Q_h^{(2)}(c) \leq \sqrt{\frac{4c}{\beta}} \cdot m^{\mu_2(h)}$$

provided $c \leq \alpha^2\beta/4$. Substituting $\alpha = 0.5$ which maximizes the expression $\alpha^2\beta/4$ we get

$$0.7357\sqrt{c} \cdot m^{\mu_2(h)} \leq Q_h^{(2)}(c) \leq 3.26599\sqrt{c} \cdot m^{\mu_2(h)}$$

the upper bound being true under the restriction $c \leq 0.0234375$. Our bounds are valid over a wider range of c compared to the bounds stated in (3.9).

How good is the upper bound?

We will now analyze the upper bound on $Q_h^{(r)}(c)$ given in Theorem 5.4.1. To get the range of c for which the upper bound is valid for different values of r we need to look at the following expression (obtained by substituting for β in (5.25)).

$$\left(\frac{\alpha}{r}\right)^r \left(\frac{3}{2} - \frac{1}{2}(\alpha+1)^r\right) \tag{5.26}$$

The constraint that β must be positive we can say that $\alpha < (3)^{1/r} - 1$. It is now clear that $0 \leq \alpha < 1$. The maximum range of c can be obtained by finding the maximum value that (5.26) attains. If we differentiate this expression with respect to α, we will get a polynomial $s(\alpha)$ of degree $2r-1$ which has exactly one term with a negative sign. Hence $s(\alpha)$ will have only one real root. At this value of α, (5.26) will attain a maximum.

For any $r \geq 2$, let $cmax_r$ denote the maximum value of the expression (5.26). Table 5.1 shows values of $cmax_r$ for r ranging from 2 to 8.

One can observe that the value of $cmax_r$ is decreasing rapidly with increasing values of r which means that as r grows larger the upper bound of Theorem 5.4.1 is valid across smaller ranges of c.

Chapter 6

Conclusion

The results stated in sections 5.3 and 5.4 suggest that an r-collision can be found in about $m^{\left(\frac{r-1}{r}\right)\mu_r(h)}$ trials. This indicates that functions with high r-balance fare better against birthday attacks than the ones with low r-balance. Hence ensuring high r-balance provides good security against birthday attacks for finding multicollisions.

One could look at several other problems related to birthday attacks and balance-based analysis of this attack. Here are some interesting problems.

1. Bellare and Kohno [BK04] show that the MD-transform does not preserve balance by considering degenerate compression functions. They also give some experimental results that suggest that SHA-1 actually preserves balance. If the compression function is balanced and non-degenerate then what can be said about the balance of the MD iterates?

2. There are several space efficient algorithms that find cycles in random graphs. These methods can be used to find collisions in a hash function. Is there a space efficient algorithm to find multicollisions?

Cycle detection algorithms

Cycle detection is the algorithmic problem of finding a cycle of the following type: For any function f that maps a finite set S to itself, and any initial value x_0 in S, the sequence of iterated function values $x_0,\ x_1 = f(x_0),\ x_2 = f(x_1),\ \cdots,\ x_i = f(x_{i-1}),\ \cdots$ must eventually use the same value twice: there must be some $i \neq j$ such that $x_i = x_j$. Once this happens, the sequence must continue by repeating the cycle of values from x_i to x_{j-1}.

Floyd's cycle-finding algorithm ([Flo67, Knu97]) is a pointer algorithm that uses only two pointers, which move through the sequence at different speeds. Brent's algorithm [Bre80] is also a pointer algorithm using only two pointers but its underlying principle is different. It has been shown that, on an average, Brent's algorithm is faster than Floyd's. There is a discussion of these algorithms in Exercise 3.1-6 of

[Knu97]. Nivasch [Niv04] describes an algorithm that does not use a fixed amount of memory, but for which the expected amount of memory used is logarithmic in the sequence length. This algorithm uses a stack to store the sequence values. The same algorithm can be run with multiple stacks allowing a time-space tradeoff similar to the previous algorithms.

A note on Problem 2

Problem 2 has been addressed recently by Joux and Lucks in [JL09]. They give an algorithm to find 3-collisions that roughly uses m^{δ} storage and whose running time is $m^{1-\delta}$ for $\delta \leq 3$. This shows that finding 3-collisions in time $m^{2/3}$ would require $m^{1/3}$ units of storage.

The basic idea is as follows: First an array is initialized with N^{δ} collisions for the hash function h. Then simply create N^{γ} images of random points until we hit one of the known collisions.

To make this algorithm space efficient each of the N^{δ} collisions can be generated using one of the cycle finding algorithms mentioned earlier. It can be shown that the time complexity of this algorithm is roughly $O(m^{1-\delta})$ and storage required is $O(m^{\delta})$

Bibliography

[BK04] M. Bellare and T. Kohno. Hash function balance and its impact on birthday attacks. In C.Cachin and J.Camanisch, editors, *Advances in Cryptology - EUROCRYPT '04*, volume 3027 of *Lecture Notes in Computer Science*. Springer-Verlag, 2004.

[BPVY00] E. Brickell, D. Pointcheval, S. Vaudenay, and M. Yung. Design validation for discrete logarithm based signature schemes. In *PKC'2000*, volume 1751 of *Lecture Notes in Computer Science*, pages 276–292. Springer-Verlag, 2000.

[BR93] M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *Proceedings of the First Annual Conference on Computer and Communications Security*, pages 62–73. ACM Press, 1993.

[Bre80] R. P. Brent. An improved monte carlo factorization algorithm. *BIT*, 20:176–184, 1980.

[Dam89] I. Damgård. A design principle for hash functions. In G. Brassard, editor, *Advances in Cryptology - CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 416–427. Springer-Verlag, 1989.

[DM89] P. Diaconis and F. Mosteller. Methods for studying coincidences. *Journal of the American Statistical Association*, 84:853–861, 1989.

[Fel08] W. Feller. *An introduction to probability theory and its applications*, volume I. Wiley India, 3 edition, 2008.

[Flo67] R. W. Floyd. Non-deterministic algorithms. *Journal of the ACM*, 14(4):636–644, 1967.

[GS94] M. Girault and J. Stern. On the length of cryptographic hash-values used in identification schemes. In *Advances in Cryptology - CRYPTO 1994*, volume 839 of *Lecture Notes in Computer Science*, pages 202–215. Springer-Verlag, 1994.

[HS06] J. J. Hoch and A. Shamir. Breaking the ice - finding multicollisions in iterated concatenated and expanded (ice) hash functions. In *Fast*

Software Encryption 2006, volume 4047 of *Lecture Notes in Computer Science*, pages 179–194, Berlin, Germany, 2006. Springer-Verlag.

[JL09] A. Joux and S. Lucks. Improved generic algorithms for 3-collisions. Cryptology ePrint Archive, Report 2009/305, 2009. `http://eprint.iacr.org/`.

[Jou04] A. Joux. Multicollisions in iterated hash functions. application to cascaded constructions. In *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 474–490, Berlin, Germany, 2004. Springer-Verlag.

[Knu97] D. E. Knuth. *The Art of Computer Programming, vol. II: Seminumerical Algorithms*. Reading. Addison-Wesley, MA, 3 edition, 1997.

[Lev81] B. Levin. A representation for multinomial cumulative distribution functions. *The Annals of Statistics*, 9(5):1123–1126, September 1981.

[McK66] E. H. McKinney. Generalized birthday problem. *The American Mathematical Monthly*, 73(4):385–387, April 1966.

[Mer79] R. C. Merkle. *Secrecy, authentication and public key systems*. PhD thesis, Stanford University, 1979.

[MvOV97] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1997. Available at `http://www.cacr.math.waterloo.ca/hac/`.

[Niv04] G. Nivasch. Cycle detection using a stack. *Information Processing Letters*, 90/3:135–140, 2004.

[NS07] M. Nandi and D. R. Stinson. Multicollision attacks on some generalized sequential hash functions. *IEEE transactions on Information Theory*, 53(2):759–767, February 2007.

[Pre93] B. Preneel. *Analysis and Design of Cryptograhic Hash Functions*. PhD thesis, Katholieke Universiteit Leuven, Leuven, Belgium, 1993.

[RS96] R. Rivest and A. Shamir. Payword and micromint - two simple micropayment schemes. *CryptoBytes*, 2(1):7–11, Spring 1996.

[Sti02] D. R. Stinson. *Cryptography theory and practice*. Chapman and Hall/CRC, 2 edition, 2002.