Printer - please drop in Elsevier (tree) logo

Computers & Security Vol. 17, No.2, pp. 171-176, 1998
© 1998 Elsevier Science Limited
All rights reserved. Printed in Great Britain
0167-4048/98 $19.00

On Probabilities of Hash Value Matches

Mohammad Peyravian[a], Allen Roginsky[a], Ajay Kshemkalyani[b]

[a]*IBM Corporation, Research Triangle Park, NC 27709, USA*
[b]*ECECS Department, University of Cincinnati, Cincinnati, OH 45221, USA*

Hash functions are used in authentication and cryptography, as well as for the efficient storage and retrieval of data using hashed keys. Hash functions are susceptible to undesirable collisions. To design or choose an appropriate hash function for an application, it is essential to estimate the probabilities with which these collisions can occur. In this paper we consider two problems: one of evaluating the probability of no collision at all and one of finding a bound for the probability of a collision with a particular hash value. The quality of these estimates under various values of the parameters is also discussed.

Keywords: hash functions, security, cryptography, indexing, databases

1. Introduction

A hash function takes a variable-length input string and maps it to a fixed-length (generally smaller) output string. Hash functions are extensively used for index management in database systems and file systems for efficient storage and retrieval of data based on hashed keys [1,9,15], digital signatures for authentication [6,12,16], and cryptography [4,5,11,12]. A good survey of classical hashing methods is given in [9].

A hash function is prone to *collisions* wherein two input strings map to the same output string. A good hash function minimizes the possibility of collisions; a hash function is said to be *collision resistant* if it is hard to find two input strings that map to the same hash value. The problem of constructing fast hash functions that also have low collision rates is studied in [5]. Key-to-address transformation techniques for file access and their performance have been studied in [8]. Given k, the number of hashed values that have been used in a hashing scheme in which input strings are mapped to random values, the expected number of locations that must be looked at until an empty hashed value is found is formulated in [10]. This result was improved for certain non-random hashing functions and certain values of k in [14]. The efficiency of multiple-key hashing in limiting the search of a given key value in a file and in minimizing the search in answering partial-match or multi-attribute queries is studied in [2]. The only known work that deals with the probability of collisions of hash functions is [3,13,16]. These papers dealt with the construction of universal hash functions which are classes of hash functions such that the functions in any class have the same bound on the probability of hash collisions.

For several applications such as cryptography, it is important to design or choose appropriate hash functions with an upper bound on the probability of collisions. To this end, our objective is to determine the probabilities of hash collisions for hash functions that are uniformly mapped into the codomain, i.e., each input string is equally likely to be mapped to any of the hash values of the codomain under different scenarios. Specifically, we address two problems: 1) determining the probability that there are no collisions at all, in Theorem 1, and 2) determining a bound on the probability of a collision with a particular hash value, in Theorem 2. For each of Theorems 1 and 2, we state some corollaries that give the minimum size of the hash function that is necessary to satisfy a desired upper bound on the probability of hash collisions.

2. Hash Functions

Definition 1: A hash function $H(M) = h$ maps an input bit string M from set A into a bit string of a fixed length in set B with the properties that

Every possible value of B is equally likely to be an image of an element in set A.
Given M, it is easy to compute h.

Hash functions considered in the literature, e.g. [3,5,8,10,13,14,16] satisfy the above definition.

We were prompted to address the problems on the probability of collisions while employing hash functions for cryptography and had to deal with potential collisions. In cryptography, the security of protocols that use hash functions would be undermined if the hash functions are not highly collision resistant. Additionally, it is should be extremely hard to reconstruct the input string from its hash value in a reasonable amount of time. Such hash functions are called *one-way hash functions*.

One-way hash functions are useful in cryptography because they enforce the property that the knowledge of a particular value of $H(M)$ does not help the attacker to guess the value M that was used and at the same time it provides a 'fingerprint' of M that is unique. There are many one-way hash functions such as SHA-1, MD2, MD4, MD5, and Snefru presently in use (see [12]).

Definition 2: Following [12], we define a one-way hash function $H(M) = h$ as a function that maps any input bit string M from set A into a bit string of a fixed length in set B with the properties that

- Every possible value of B is equally likely to be an image of an element in set A.
- Given M, it is easy to compute h.
- Given h, it is hard to compute M such that $H(M) = h$
- Given M, it is hard to find another message M' such that $H(M) = H(M')$.
- It is hard to find two random messages, M and M', such that $H(M) = H(M')$.

Although our results apply to all hash functions that satisfy Definition 1, we focus on the cryptography application (Definition 2) for the rest of this paper.

3. Problem Statement

Let us consider the following scenario. Let A. be a set consisting of m different messages $M_1, M_2, \ldots, M_m$. Let M denote a particular message in A with a corresponding hash element $H(M)$ in set B. Two problems then arise.

Problem 1: Given sets A and B, determine the probability that no two elements in A mesh into the same element of B, that is, determine the probability that there are no collisions.

Problem 2: Given our particular message M, determine the probability that no other element in A meshes into the value, $H(M)$, of the hash function at M.

Answers to Problems 1 and 2 give two measures of goodness of a hash function. An application can choose or design a hash function such that the goodness of the hash function satisfies the application's requirements. In cryptography, Problem 1 is also important since a high probability of collisions between any messages opens the door to the so-called 'birthday attacks' (see [12].) Problem 2 is also relevant because a user is concerned with only his or her message M having a unique image in set B.

In practical applications, Problems 1 and 2 can be stated as follows: given the size m of set A and the maximum permissible value Q_{max} of the collision probability, what should be the minimum length, k, in bits of the values of the hash function? We answer these questions associated with Problems 1 and 2 by deriving corollaries to two theorems that answer Problem 1 and Problem 2, respectively.

4. Estimates of Collision Probabilities

We will prove two theorems (Theorems 1 and 2) associated with Problems 1 and 2 stated above and also consider some special cases.

Theorem 1: For $k \geq 3$ and $m < 2^{k-1}$, the probability Q that there exists collisions between the hash values satisfies the following inequality:

$$Q < 1 - \exp\left(-\frac{(m+1)^2}{2(2^k+1-m)}\right) \qquad (1)$$

Proof:
Let P = 1-Q be the probability that no collisions occur. There will be no collisions when the second message is hashed into a value in set B different from that of the first message, the third message is hashed into yet another value, etc. The probability of this is equal to

$$P = \frac{b-1}{b} \times \frac{b-2}{b} \times \cdots \times \frac{b-m+1}{b} = \frac{b!}{b^m (b-m)!} \qquad (2)$$

where $b=2k$ is the number of possible values of the hash function. In terms of the Gamma function $\Gamma(x)$, (2) can be rewritten

$$P = \frac{\Gamma(c)}{b^m \Gamma(c-m)} \qquad (3)$$

where $c = b+1=2^k+1$. Hence

$$\ln(P) = \ln(G(c)) - m\ \ln(b) - \ln(\Gamma(c-m)). \qquad (4)$$

A logarithm of a Gamma function for the large values of the argument can be very well estimated by using formula 8.344 from [7]. It states that for any $j \geq 1$,

$$\ln(\Gamma(z)) = z\ \ln(z) - z - 0.5\ \ln(z) + \ln\left(\sqrt{(2\pi)}\right) + \sum_{i=1}^{j-1} \frac{B_{2i}}{2i(2i-1)z^{2i-1}} + R_j(z) \qquad (5)$$

where

$$|R_j(z)| < \frac{|B_{2j}|}{2j(2j-1)|z|^{2j-1} \cos^{2j-1}(0.5 \cdot \mathrm{Im}(z))}$$

and B_{2i} is the corresponding Bernoulli number. The reason for the cosine term is that the formula is also applicable to the non-real values of z. We will use this formula only with $j=1$, so that the summation term will not be present, the only applicable Bernoulli number is $B_2 = 1/6$, and with z equal first to c and then to c-m. Therefore, z is real and positive, the cosine function of one half of its imaginary part is equal to 1 and thus $|R_1(z)| < 1/12z$. In the following calculations we will omit a subscript for $R_1(z)$ and simply call it $R(z)$ Applying (5) to formula (4), we get

$$\begin{aligned}\ln(P) &= c\ \ln(c) - c - 0.5\ \ln(c) + \ln\left(\sqrt{2\pi}\right) + R(c) - m\ \ln(b) - (c-m)\ \ln(c-m) + (c-m) \\ &\quad + 0.5\ \ln(c-m) - \ln\left(\sqrt{2\pi}\right) - R(c-m) \\ &= c\ \ln(c) - c - 0.5 \cdot \ln(c) - m\ \ln(b) - c\ \ln(c-m) + m\ \ln(c-m) + c - m + 0.5\ \ln(c-m) \\ &\quad + (R(c) - R(c-m)) \\ &= c\ \ln(c/(c-m)) - 0.5\ \ln(c/(c-m)) - m\ \ln((c-1)/(c-m)) - m + (R(c) - R(c-m)) \\ &> c\ \ln(c/(c-m)) - 0.5\ \ln(c/(c-m)) - m\ \ln(c/(c-m)) - m + (R(c) - R(c-m)) \\ &= (c - m - 0.5)\ \ln(1 + m/(c-m)) - m + (R(c) - R(c-m)).\end{aligned} \qquad (6)$$

Let us denote $n = c$ - m. From the assumptions of the theorem it follows immediately that $m < n$.

Using the bound for $R(z)$ and the fact that $n < c$, we have $|R(c) - R(n)| < 1/6n$ so (6) can be written as

$$\begin{aligned}\ln(P) &> (n-0.5)\ \ln(1+m/n) - m - 1/6n \\ &> (n - 0.5)(m/n - m^2/2n^2) - m - 1/6n \\ &= -\frac{m}{2n} - \frac{m^2}{2n} + \frac{m^2}{4n^2} - 1/6n \\ &= -\frac{(m+1)^2}{2n} + \left(\frac{m}{2n} + \frac{m^2}{4n^2} + 1/3n\right) \\ &> -(m+1)^2/2n.\end{aligned}$$

Here we used the well-known and easily-verified fact that $\ln(1+x)>x-x^2/2$ for $x>0$. Hence $P>e^{-(m+1)2/2n}$ and the statement of Theorem 1 follows from here. This completes the proof of Theorem 1.

QED

Theorem 1a: Under the assumptions of Theorem 1, the probability Q that there exist collisions between the hash values satisfies that following inequality:

$$Q<\frac{(m+1)^2}{2(2^k+1-m)} \qquad (7)$$

Proof:
It follows immediately from Theorem 1 and from the fact that $e^x \geq 1+x$, for all x.

QED

Theorem 1a offers a very close approximation to the true probability of having collisions if m is significantly smaller than n.

Corollary 1: Given $Q_{\max}$ and m, it is sufficient to have

$$k \geq \log_2\left(\frac{(m+1)^2}{2Q_{\max}}+m-1\right) \qquad (8)$$

Proof:
If k satisfies (8) then

$$2^k \geq \frac{(m+1)^2}{2Q_{\max}}+m-1,$$

so

$$Q_{\max} \geq \frac{(m+1)^2}{2(2^k+1-m)} \qquad (9)$$

According to Theorem 1a, the collision probability Q satisfies

$$Q<\frac{(m+1)^2}{2(2^k+1-m)} \leq Q_{\max}$$

QED

The following two corollaries may prove to be very useful in all practical applications of the above results. The upper bound for the probability in Theorem 1 can be very closely approximated by $m2/2^{-(k+1)}$. While there is no absolute guarantee that the latter expression provides an upper bound for Q, its closeness to the estimate in (1) makes it fully acceptable for all reasonable values of r, k and $Q_{\max}$. With this understanding we can obtain the following results.

Corollary 1a: To achieve a target probability of $Q_{\max}$ or less for the probability of having hash collisions, it is sufficient to have a hash function with the value of k that satisfies

$$k \geq 2\cdot\log_2 m-\log_2 Q_{\max}-1$$

QED

Corollary 1b: Let r be the length of a message in the A set and suppose that the A set may include all of the 2^r different messages of length r bits. Suppose also that $Q_{\max}$ is expressed as 2^{-t} for some t. Then it is sufficient to have k that satisfies

$$k \geq 2r+t-1$$

Now we turn to Problem 2 and prove the following result.

Theorem 2: The probability Q that there exist collisions with a given hash value satisfies the following inequality:

$$Q<1-\exp\left(-\frac{m}{2^k}\right)$$

Proof:
We can assume that $m \leq 2^k$; otherwise, there is nothing to prove.

As in the proof of Theorem 1, let us denote P as the complementary probability. Thus $P = 1-Q$ is the probability that no collisions with a given value $H(M)$ take place. The probability that a message $M_i \neq M$ hashes into a value different from $H(M)$ is $(b-1)/b$ where, as before, $b=2^k$.

The properties of our hash function allow us to assume the independence of these collisions, so $P = (1-1/b)^{m-1}$. We have

$$\begin{aligned}\ln(P) &= (m-1)\ln(1-1/b)\\ &= (m-1)(-1/b-1/2b^2-1/3b^3-1/4b^4-\cdots)\\ &> (m-1)(-1/b-1/2b^2-1/2b^3-1/2b^4-\cdots)\\ &= (m-1)\left(-1/b-\frac{1}{b^2}\frac{b}{2(b-1)}\right)\\ &> (m-1)(-1/b-1/b^2)\\ &= -m/b+(1/b+1/b^2-m/b^2) > -m/b, \qquad \text{when } m \le b.\end{aligned}$$

Therefore $P>e^{-m/b}$, so $Q<1-e^{-m/b}$. Theorem 2 is proved.

QED

Theorem 2a: The probability Q that there exist collisions with a given hash value satisfies the inequality

$$Q<\frac{m}{2^k}$$

Proof:
Theorem 2a follows immediately from Theorem 2 and the fact that $e^{-x} \ge 1-x$ for all x.

QED

Corollary 2: Given $Q_{\max}$ and m, it is sufficient to have

$$k \ge \log_2 m - \log_2 Q_{\max}$$

Proof:
The proof logic follows that of the proof of Corollary 1.

QED

Corollary 2a: Under the assumptions made in Corollary 1b, it is sufficient for k to satisfy $k \ge r+t$.

QED

Example: If one considers the hashes of all messages 100 bits long and wants the probability of collisions with a particular hash value not to exceed 2^{-60}, then it is safe to use the hash function SHA-1 (k = 160) for this purpose.

5. Conclusion

Analyzing the hash collision probability is important for choosing or designing appropriate hash functions for critical applications such as cryptography. A large number of hash functions studied in the literature satisfy the property that each hash value is equally likely to be the image of any given input. This paper analyzed the hash collision probability by answering the following two questions for all hash functions that satisfy the above property.

What is the probability that there are no collisions at all?

The answer to this question is particularly important in knowing the susceptibility of the hash function-based encryption to 'birthday attacks'.

Given any particular input string, what is the probability that no other input string in the domain gets hashed to the same value that the given input string hashed into?

The answer to this question gives the user of the crypto-system based on the hash function the probability that his particular input string will not be involved in a collision with any other input.

The theorems that answer the above questions are used to derive corollaries that answer the following questions for the two types of probabilities addressed:

Given the size of input strings and the maximum permissible value $Q_{\max}$ of the collision probability, what should be the minimum length k in bits of the value of the hash function?

An application would choose or design a hash function of a length indicated by answering this question in order to guarantee an upper bound on the probability of a hash collision.

References

[1] A. Aho, J. Hopcroft, J. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, Mass., 1974.

[2] A. Bolour, "Optimality Properties of Multiple Key Hash Functions," *Journal of the ACM*, 26(2), 196-210, 1979.

[3] J. L. Carter, M. N. Wegman, "Universal Classes of Hash Functions," *Journal of Computer and System Sciences*, 18, 143-154, 1979.

[4] I. B. Damgard, "Collision Free Hash Functions and Public Key Signature Schemes," Advances in Cryptography, Eurocrypt '87, LNCS 304, 203-216, 1987, Springer-Verlag.

[5] I. B. Damgard, "A Design Principle for Hash Functions," Advances in Cryptology, Crypto '89, LNCS 435, 416-427, 1989, Springer-Verlag.

[6] S. Goldwasser, S. Micali, R. Rivest, "A Paradoxical Solution to the Signature Problem," Proc. 25th IEEE Conf. on Foundations of Computer Science, 441-448, 1984.

[7] I. S. Gradshtein, I. M. Ryzhik, *Tables of Integrals, Series, and Products*, Academic Press, 1980.

[8] V. Lum, "General Performance Analysis of Key-to-Address Transformation Methods Using an Abstract File Concept," *Communications of the ACM*, 16(10), 603-612, October 1973.

[9] W. D. Maurer, T. G. Lewis, "Hash Table Methods," ACM Computing Surveys, 7(1), 5-20, 1975.

[10] R. Morris, "Scatter Storage Techniques," *Communications of the ACM*, 11(1), 38-44, Jan. 1968.

[11] B. Preneel, "Cryptographic Hash Functions," *European Trans. Telecommunications*, 5, 431-448, 1994.

[12] B. Schneier, *Applied Cryptography*, 2nd edition, John Wiley and Sons, Inc, 1996.

[13] D. Stinson, "Combinatorial Techniques for Universal Hashing" *Journal of Computer and System Sciences*, 48, 337-346, 1994.

[14] J. D. Ullman, "A Note on the Efficiency of Hashing Functions", *Journal of the ACM*, 19(3), 569-575, July 1972.

[15] J. D. Ullman, *Principles of Database Systems*, Computer Science Press, 1980.

[16] M. N. Wegman, J. L. Carter, "New Hash Functions and Their Use in Authentication and Set quality," *Journal of Computer and System Sciences*, 22, 265-279, 1981.