Challenging the Increased Resistance of Regular Hash Functions Against Birthday Attacks

Nicky Mouha, Gautham Sekar and Bart Preneel

Abstract. At EUROCRYPT 2004, Bellare and Kohno presented the concept of a regular hash function. For a hash function to be regular, every hash value must have the same number of preimages in the domain. The findings of their paper remained unchallenged for over six years, and made their way into several research papers and textbooks. In their paper, Bellare and Kohno claim that regular hash functions are more resistant against the birthday attack than random hash functions. We counter their arguments, by showing that the success probability of the birthday attack against a regular hash function can be made arbitrarily close to that of a random hash function (for the same number of trials). Our analysis uses the fact that the choices of the attacker can be limited to any subset of the domain. Furthermore, we prove that it is not possible to construct a hash function that is regular for only a small fraction of subsets of the domain. In order to avoid these problems, we propose to model hash functions as random functions. Compared to regular functions, we argue that the statistics of random functions are more similar to hash functions used in practice, regardless of how the attacker chooses the domain points.

Keywords. Hash function, balance, regularity, birthday attack, (linear) subset regularity, random function.

2010 Mathematics Subject Classification. 94A60, 97K20.

1 Introduction

Let $h_{D,R} : \ D \rightarrow R$ be a function, where both the domain D and the range R are finite sets. Denote $|D|$ by d and $|R|$ by r. If $d > r$, $h_{D,R}$ is referred to as a hash function. Although strictly not required, D is finite for commonly used hash functions; e.g. for SHA-1, D consists of all strings of length at most $2^{64} - 1$

This work was supported in part by the Research Council K.U.Leuven: GOA TENSE, and by the IAP Program P6/26 BCRYPT of the Belgian State (Belgian Science Policy), and in part by the European Commission through the ICT program under contract ICT-2007-216676 ECRYPT II. Nicky Mouha is funded by a research grant of the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen). Part of this work was performed during the Gautham Sekar's PhD research at COSIC.

bits [8].

Any pair (x, y) where $x, y \in D$ for which $x \neq y$ and $h_{D,R}(x) = h_{D,R}(y)$, is denoted as a *collision* for hash function $h_{D,R}$. A *possibly trivial collision* is a pair (x, y) where $x, y \in D$ for which $h_{D,R}(x) = h_{D,R}(y)$. That is, unlike for a collision, it is allowed that $x = y$. Among other requirements [14], a cryptographic hash function should be collision resistant, i.e. it should be computationally infeasible to find a collision. In this paper, a "hash function" does not necessarily refer to a "cryptographic hash function." For any choice of h, a generic birthday attack can be used to find a collision.

In a birthday attack, points $x_1, \ldots, x_q$ are picked from D. For $i = 1, \ldots, q$, we compute $y_i = h_{D,R}(x_i)$. We say that the birthday attack is successful, if we find $h_{D,R}(x_i) = h_{D,R}(x_j)$, where $1 \leq i < j \leq q$. We refer to q as the number of trials of the birthday attack.

There are several variants of the birthday attack, that differ in the way that the points $x_1, \ldots, x_q$ are chosen. In their analysis [1, 2], Bellare and Kohno only consider the case where the domain points are chosen independently and uniformly at random from all m-bit strings (therefore $|D| = 2^m$). Yuval [26] instead suggests using q minor modifications of a message, in such a way that all messages are meaningful. Using distinguished points, Quisquater and Delescaille [18] showed that collisions for meaningful messages can also be found with negligible memory requirements, i.e. without storing all $(x_i, h_{D,R}(x_i))$ for $i = 1, \ldots, q$. An efficient parallel implementation of their algorithm was proposed by Van Oorschot and Wiener [22].

For most applications, only a small subset of all m-bit strings are meaningful. If, for example, the messages consist of only ASCII characters, a necessary (but not sufficient) requirement is that the most significant bit of every character is zero.

Let $C_{D,R}(q)$ be the probability that the birthday attack finds a possibly trivial collision for $h_{D,R}$ after q trials. If for every domain point, the corresponding range point of $h_{D,R}$ is chosen uniformly and independently at random from all r range points, we refer to $h_{D,R}$ as a random function. The success probability of the birthday attack for a random function is denoted as $C^{\$}_{D,R}(q)$.

Bellare and Kohno [2] point out that if $h_{D,R}$ is a random function, this does not necessarily mean that $h_{D,R}(x)$ is uniformly distributed in R. In order to have such a uniform distribution in R, every range point must have the same fraction of preimages under the hash function. We refer to such a hash function as a regular function, denoted by $h^{\text{reg}}_{D,R}$. This can be defined more formally as follows.

Definition 1.1 (Balance and Regularity). Let $h_{D,R} : D \to R$ be a hash function with domain D of size d and range $R = \{R_1, R_2, \ldots, R_r\}$ of size r. For $h_{D,R}$ to be a hash function, we must have $d > r$. For $1 \leq i \leq r$, $d_i = |{h_{D,R}}^{-1}(R_i)|$

denotes the size of the preimage of R_i under $h_{D,R}$. The *balance* of $h_{D,R}$ is then defined as

$$\mu(h_{D,R}) = \log_r \left(\frac{d^2}{d_1^2 + d_2^2 + \ldots + d_r^2} \right) \quad . \tag{1.1}$$

A hash function is regular iff $\mu(h_{D,R}) = 1$ (that is, iff $\forall\ 1 \leq i \leq r : d_i = d/r$) [2, §5].

If h is a regular function, the success probability of the birthday attack is denoted by $C^{\text{reg}}_{D,R}(q)$. Bellare and Kohno calculate[1] that $C^{\$}_{D,R}(q) > (8/5) \cdot C^{\text{reg}}_{D,R}(q)$, if $d = 2r \geq 10$. Therefore, they conclude that "regular functions fare better than random functions [against the birthday attack]."

We recall that their reasoning assumes that the attacker chooses the messages uniformly at random from D. In the following sections, we investigate the case where the attacker limits the choice of the domain points to subsets of D. We prove that it is not possible to construct a hash function that is regular with respect to only a small fraction of subsets of the domain. For this, we introduce the concepts of subset regularity and linear subset regularity.

Bellare and Kohno pointed out in their analysis that there is only a small difference between regular and random functions in their resistance against the birthday attack. For random functions, the success probability of the birthday attack does not depend on how the attacker chooses the domain points.

NIST is currently holding a competition in search for a new hash function standard [15]. Our result may be relevant to the analysis of statistical properties of the hash functions in this competition.

Organization. This paper is organized as follows. In Section 2, we describe the birthday problem and its relation to the birthday attack. Section 3 provides a brief overview of some works that employ the notion of regularity. In Section 4, we compute the ratio of regular functions to all functions with the same domain and range. Our notions of subset regularity and linear subset regularity are introduced in Sections 5 and 6, respectively. The impact of our observations on the birthday attack is discussed in Section 7, where we show that the success probability of the birthday attack against a regular hash function can be made arbitrarily close to that of a random hash function (for the same number of trials). In Section 8, we describe the relation of regularity to the dictionary problem in computer science.

We propose in Section 9 that, to analyze the complexity of the birthday attack for commonly used hash functions, we model hash functions as random functions

[1] In this proof, the values from the domain are randomly chosen, but with replacement. The replacement can result in a possibly trivial collision with a collision in the domain. The authors show in [2, §7.2] that the higher success probability is not due to the possibility of such collisions.

instead of as regular functions. For random functions, we prove that the success probability of the birthday attack does not depend on how the attacker chooses the domain points. We conclude in Section 10. We show in Appendix A how the construction of a 3-to-1 bit linear subset regular hash function fails. In Appendix B, we calculate the inverse of some matrices that we use in Section 6.

2 The Birthday Problem

Assume that there are N people in one room. How large must N be, in order to have a probability of at least $1/2$ that two people share the same birthday? It is assumed that birthdays are independently and uniformly distributed over the 365 days of the year (leap years are ignored). This is the birthday problem (see Feller [9, §2.3]), which dates back to von Mises [23]. The answer to the problem is $N \geq 23$.

Bloom showed that the probability that two people share the same birthday, is the lowest when birthdays are uniformly distributed [3]. Nunnikhoven [16] analyzed the birthday problem for nonuniform birth frequencies.

Based on the mathematics of the birthday problem, Yuval proposed the birthday attack for hash functions [26]. In the attack, a large number of messages are generated, until two messages are found that result in the same hash value. The attack complexity depends on the distribution of the hash values. If the hash values are uniformly distributed, the analysis of the original birthday problem applies. In case of a nonuniform distribution, collision probabilities were calculated by Cachin [5, §3.2.5], as well as Bellare and Kohno [2].

In this paper, we point out that the distribution of the hash values not only depends on the hash function, but also on how the attacker chooses the input messages. This is different from the birthday problem, where the probability distribution of the birthdays is fixed in advance (to have a uniform distribution). In the following sections, we investigate the impact of the attacker's choice of the messages.

3 Balance and Regularity in Existing Literature

The results of [2] not only remained unchallenged for over six years, but were also often cited in papers on cryptographic theory, in cryptanalysis papers and in textbooks. In this section, we give a brief overview of some of the most notable results.

Since Bellare and Kohno introduced their balance measure $\mu(h_{D,R})$ in [1, 2] (defined in (1.1)), this measure has been applied to several hash functions. Already in their original paper, the balance measures of truncated variants of SHA-1 were analyzed. Later, Yoshida *et al.* calculated the balance of a reduced version of MAME [25]. Ødegård and Gligoroski recently computed the balance measures of reduced versions of EDON-$\mathcal{R}$ [17].

In each of these papers, hash function balances are calculated. However, the results show that not a single one of the hash functions variants under consideration is regular, and the balance measure $\mu(h_{D,R})$ seems to decrease if the number of output bits of $h_{D,R}$ is increased. The balance of the actual (untruncated) hash functions is never calculated, because this would be computationally infeasible. Because of this difficulty, we question the applicability of the balance measure to analyze practical hash functions.

The notions of balance and regularity also appear in several textbooks. In [11], Goldwasser and Bellare state that "If $h_{D,R}$ is not regular, it turns out the [birthday] attack succeeds even faster, telling us that we ought to design hash functions to be as "close" to regular as possible." In this paper, we explain why we counter this design criterion.

Buchmann's book [4] states: "We assume that strings from [the domain] can be chosen such that the distribution on the corresponding hash values is the uniform distribution." However, it is the attacker who can freely determine how strings are chosen from the domain. In this paper, we show that there always exists a way for the attacker to restrict the domain so that the resulting function is not regular.

In the first edition of his book [19], Stinson describes the birthday attack under the assumption that the hash function is regular. This assumption is dropped in the second edition [20], in favor of random oracles [10].

In [13], Joux refers to [1] for a more precise analysis of collisions in hash functions for the unbalanced case. Bellare and Kohno provide bounds for this unbalanced case [2], which they refer to as "the generalized birthday problem". The reader should not confuse this with the generalized birthday problem that Wagner studied earlier [24].

4 Fraction of Regular Functions

Recall that the total number of hash functions $|h_{D,R}|$ is given by r^d. We now introduce the following lemma.

Lemma 4.1. *The total number of functions $|{h_{D,R}}^{\mathrm{reg}}|$ that are regular is given by*

$$|{h_{D,R}}^{\mathrm{reg}}| = \begin{cases} {}^{d!}/_{((d/r)!)^r} & \text{if } r \mid d \ , \\ 0 & \text{if } r \nmid d \ . \end{cases} \tag{4.1}$$

Proof. For a function to be regular, each range point must have the same number of preimages under the function. This is achieved if and only if $r \mid d$. Given that the function is regular, the first range point that we consider has one of $\binom{d}{d/r}$ possible sets of d/r preimages mapping to it. Here, $\binom{u}{v}$ denotes the quantity $u!/(v! \cdot (u - v)!)$. Any domain point in the set that maps to this range point cannot map to any other range point; otherwise the mappings do not constitute a function. Therefore, the second range point that we consider will have one of only $\binom{d-d/r}{d/r}$ possible sets of d/r preimages mapping to it. Similarly, the i-th range point will have one of $\binom{d-(i-1)\cdot d/r}{d/r}$ sets of domain points mapping to it. In total, therefore, we have

$$\prod_{i=1}^{r} \binom{d-(i-1)\cdot d/r}{d/r} = \frac{d!}{((d/r)!)^r} \tag{4.2}$$

functions that are regular. Figure 1 illustrates the above arguments with an example. □

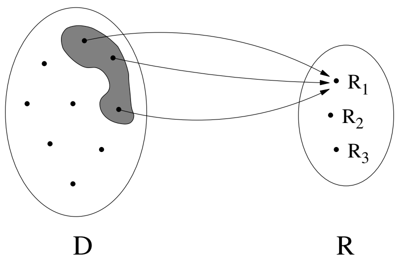

Figure 1. In this example, $d = 9$ and $r = 3$; the shaded area represents one of the $\binom{9}{3}$ possible sets of 3 domain points that can map to the range point R_1 given that the function is regular; for R_2 there are only $\binom{6}{3}$ sets

Theorem 4.2. *Assume $r \mid d$. The probability that a random function is also a regular function, is given by*

$$\frac{|h_{D,R}{}^{\mathrm{reg}}|}{|h_{D,R}|} \approx 2^{-(r/2)\cdot\log_2(2\pi d/r)} \quad . \tag{4.3}$$

Proof. Recall Stirling's approximation:

$$\log_2(z!) \approx \frac{1}{2}\log_2(2\pi z) + z\log_2\left(\frac{z}{e}\right) \quad . \tag{4.4}$$

Using Lemma 4.1, we obtain:

$$\begin{aligned}
\log_2\left(\frac{|h_{D,R}{}^{\text{reg}}|}{|h_{D,R}|}\right) &= \log_2\left(|h_{D,R}{}^{\text{reg}}|\right) - \log_2\left(|h|\right) \\
&= \log_2\left(d!/(\tfrac{d}{r}!)^r\right) - \log_2\left(r^d\right) \\
&= \log_2\left(d!\right) - r\log_2\left(\frac{d}{r}!\right) - d\log_2(r) \\
&\approx \frac{1}{2}\log_2\left(2\pi d\right) + d\log_2\left(\frac{d}{e}\right) \\
&\quad - \frac{r}{2}\log_2\left(2\pi d/r\right) - d\log_2\left(\frac{d}{re}\right) \\
&\quad - d\log_2(r) \\
&= \frac{1}{2}\log_2\left(2\pi d\right) - \frac{r}{2}\log_2\left(2\pi d/r\right) \\
&\approx -\frac{r}{2}\log_2\left(\frac{2\pi d}{r}\right) \quad . \qquad (4.5)
\end{aligned}$$

□

Let us consider a random hash function with $d = 2^{161}$ and $r = 2^{160}$. According to Theorem 4.2, the probability[2] that this function is a regular function, is $2^{-2^{160.9}}$. We note that it is therefore extremely unlikely that a hash function chosen uniformly at random from the set of r^d hash functions is regular. This relates to the observations made in the literature study of Section 3, where we discuss papers that analyze the balance of several hash function variants.

5 Subset Regularity

First, we recall a rather obvious point from [2]. Assume that for an n-bit hash function h, we restrict the input of $h_{D,R}$ to messages of at most m bits. Let $g_{D'',R}$ be a hash function, such that the domain is restricted to at most m'' bits, where $m'' \geq m$. Suppose $g_{D'',R}(x) = h_{D,R}(x),\ \forall\ x : |x| \leq m$. Then, a collision for $h_{D,R}$ will also be a collision for $g_{D'',R}$. If $g_{D'',R}$ is SHA-1, then m'' can be at most $2^{64} - 1$. A collision for, say, $h_{D,R} : \{0,1\}^{161} \rightarrow \{0,1\}^{160}$ is a collision for SHA-1 for any $m'' > 161$. In other words, as separately stated by Bellare and Kohno [2, §7.2],

[2] Although Stirling's approximation (equation (4.4)) is used for a small value of z, namely $d/r = 2$, all digits of the calculated probability using the approximation are correct.

"[A]n adversary attacking a hash function with a very large domain D might restrict its choices of domain elements to some smaller subset of D."

One possibility is to restrict the domain elements to sets of size 2^a, where $a \in \mathbb{N}$ and $2^a > r$. In this paper, we assume that the attacker chooses to make such a restriction. We also assume that $|D|$ is even, and that the size of the restricted domain is always half the size of D.

For certain applications, the domain D must be restricted to a smaller subset. For example, if a message consists of ASCII characters, the most significant bit of every character must be zero.

Definition 5.1 (Subset regularity). Let $h_{D,R} : D \to R$ be a hash function with domain D and range $R = \{R_1, R_2, \ldots, R_r\}$ of size d and r respectively. Assuming $|D|$ is even, the attacker can restrict the elements of D to a subset S such that $|S| = |D|/2$. For $1 \le i \le r$, $s_i = |{h_{D,R}}^{-1}(R_i) \in S|$ denotes the size of the preimage of R_i under $h_{D,R}$, when the domain is restricted to S. We say that a hash function is *subset regular* with respect to S, if it is not only regular, but also $\forall\ 1 \le i \le r : s_i = d/(2r)$. That is, it must also be regular when the domain is restricted to subset S. We impose the condition $d > 2r$, to ensure that $|S| > |R|$.

We now introduce the following lemma.

Lemma 5.2. *The total number of hash functions $|{h_{D,R}}^{\text{sreg}}|$ that are subset regular with respect to S, is given by*

$$|{h_{D,R}}^{\text{sreg}}| = \begin{cases} \left({(d/2)!}/{((d/2r)!)^r}\right)^2 & \text{if } 2r \mid d\ , \\ 0 & \text{if } 2r \nmid d\ . \end{cases} \tag{5.1}$$

Proof. Suppose that $|D|$ is even. Let the domain D be partitioned into two equally-sized sets D_1 and D_2, and consider only domain elements in one of these sets (D_1 or D_2). Then every range point can have the same number of preimages, if and only if $2r \mid d$. This also implies $r \mid d$, which is required for the regularity criterion on the entire domain. The reasoning now is exactly the same as for Lemma 4.1, but with d replaced by $d/2$ as the regularity criterion holds on the smaller domain as well. Because the subset regularity criterion has to hold on the other subset of the domain, we square the entire expression. If $|D|$ is not even, it is not possible that $h_{D,R}$ is subset regular with respect to S. □

Theorem 5.3. *If $2r \mid d$, the probability that a regular function chosen uniformly at random is also subset regular with respect to S, is given by*

$$\frac{|{h_{D,R}}^{\mathrm{sreg}}|}{|{h_{D,R}}^{\mathrm{reg}}|} \approx 2^{(-r/2)\cdot\log_2(\pi d/2r)} \quad . \tag{5.2}$$

Proof. Using Lemma 4.1 and Lemma 5.2, we obtain:

$$\begin{aligned}
\log_2\left(\frac{|{h_{D,R}}^{\mathrm{sreg}}|}{|{h_{D,R}}^{\mathrm{reg}}|}\right) &= \log_2\left(|{h_{D,R}}^{\mathrm{sreg}}|\right) - \log_2\left(|{h_{D,R}}^{\mathrm{reg}}|\right) \\
&= \log_2\left(\left(\tfrac{d}{2}!/(\tfrac{d}{2r}!)^r\right)^2\right) - \log_2\left(d!/(\tfrac{d}{r}!)^r\right) \\
&= 2\log_2\left(\frac{d}{2}!\right) - 2r\log_2\left(\frac{d}{2r}!\right) - \log_2(d!) + r\log_2\left(\frac{d}{r}!\right) \\
&\approx \log_2(\pi d) + d\log_2\left(\frac{d}{2e}\right) - r\log_2(\pi d/r) - d\log_2\left(\frac{d}{2re}\right) \\
&\quad - \frac{1}{2}\log_2(2\pi d) - d\log_2\left(\frac{d}{e}\right) + \frac{r}{2}\log_2(2\pi d/r) \\
&\quad + d\log_2\left(\frac{d}{re}\right) \\
&= \frac{1}{2}\log_2\left(\frac{\pi d}{2}\right) + \frac{r}{2}\log_2\left(\frac{2r}{\pi d}\right) \approx -\frac{r}{2}\log_2\left(\frac{\pi d}{2r}\right) \quad .
\end{aligned} \tag{5.3}$$

□

Assume that for a regular hash function, $d = 2^{162}$ and $r = 2^{160}$. The attacker decides to restrict the choice of the domain points to a smaller subset, consisting of 2^{161} elements. According to Theorem 5.3, the probability[3] that a randomly chosen regular function is also subset regular with respect to S, is $2^{-2^{160.5}}$.

This leads us to conclude that if $h_{D,R}$ is a regular function chosen uniformly at random (from all regular functions with the same domain and range), the probability that $h_{D,R}$ is also a regular function for a particular subset is negligible.

6 Linear Subset Regularity

In Section 5, we showed that a randomly chosen regular hash function is also subset regular with respect to S with a probability of almost zero. Our calculations assumed that r was at least reasonably large, otherwise finding collisions using the birthday attack becomes feasible in practice.

[3] The approximation given by equation (5.2) results in $2^{-2^{160.4}}$, but we have calculated this value more accurately by including additional terms in Stirling's approximation (equation (4.4)).

One might therefore propose the design of a hash function h that is not only regular, but also subset regular with respect to arbitrary subsets. We now prove that no such h exists, by showing that a hash function can be subset regular with respect to only a negligible fraction of all $\binom{d}{d/2}$ possible subsets. In order to do this, we first introduce the definition of linear subset regularity.

Definition 6.1 (Linear subset regularity). Let $h_{D,R} : D \to R$ be a hash function with $d = |D| = 2^m$ and $r = |R|$. Every element of D consists of m bits, which we label from x_0 to x_{m-1}, where x_0 represents the least significant bit. The attacker can restrict the elements of D to a smaller subset, including only domain points that satisfy $a_{m-1}x_{m-1} \oplus a_{m-2}x_{m-2} \oplus \ldots \oplus a_0 x_0 = 0$, where $a_i \in \{0, 1\}$. We can therefore construct $2^m - 1$ subsets of D, for all choices of a_i, $0 \le i < m$, except the all-zero case. We impose $d > 2r$, to ensure that each of these subdomains is larger than the range R. We say that a hash function is *linear subset regular*, if it is not only regular for the domain D, but also for each of the $2^m - 1$ subsets of the domain that we defined.

We first prove that there are no m-to-1 bit hash functions that are linear subset regular. Using this, we prove that there are also no m-to-n bit hash functions that are linear subset regular.

Theorem 6.2. *There does not exist an m-to-1 bit hash function that is linear subset regular.*

Proof. A necessary condition for a 3-to-1 bit hash function to be linear subset regular, is that exactly four hash values are 0, and that for every linear subset exactly two hash values are 0. This condition can be described by the following system of linear equations:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} h(000) \\ h(001) \\ h(010) \\ h(011) \\ h(100) \\ h(101) \\ h(110) \\ h(111) \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \end{bmatrix} . \tag{6.1}$$

We find that there is only one solution, namely $h_{D,R}(000) = h_{D,R}(001) = \ldots = h_{D,R}(111) = 1/2$. As none of these range points are in the set $\{0, 1\}$, we conclude that there does not exist a 3-to-1 bit hash function that is linear subset regular. In Appendix A, we show how the explicit construction of a 3-to-1 bit linear subset regular hash function fails.

Let the 8×8 matrix in equation (6.1) be denoted as A_8. By $\bar{A}_d$ we denote the matrix that results when the logical negation operator is applied to every element of A_d. Matrices A_d can then be constructed as follows:

$$A_1 = [1] \, , \tag{6.2}$$

$$A_d = \begin{bmatrix} A_{d/2} & A_{d/2} \\ A_{d/2} & \bar{A}_{d/2} \end{bmatrix}, \quad \text{for } 1 \leq \log_2(d) \in \mathbb{N} \; . \tag{6.3}$$

Every row of A_d corresponds to a subset of the domain defined by the linear expression $a_{m-1}x_{m-1} \oplus a_{m-2}x_{m-2} \oplus \ldots \oplus a_0x_0$, where $a_i \in \{0, 1\}$, $0 \leq i < m$ indicates if a linear term is included or not, and x_0 refers to the least significant bit. By definition, we assume that a linear expression containing zero terms corresponds to the regularity condition. The values of a_i are different for every row, and A_d is constructed such that the top $d/2$ rows have $a_{m-1} = 0$ and the bottom $d/2$ rows have $a_{m-1} = 1$.

In order to extend our result from 3-to-1 bit hash functions to m-to-1 bit hash functions, we must prove that the following system of equations has no solutions that consist of only elements in $\{0, 1\}$:

$$A_d X = \frac{d}{4} \begin{bmatrix} 2 & 1 & 1 & \ldots & 1 \end{bmatrix}^T \; . \tag{6.4}$$

We find that $X = \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^T / 2$ is always a valid solution, by counting the number of ones in every row of A_d. This is the only solution if A_d is invertible. In Appendix B, we prove that matrices A_d are invertible, by showing their relation to Hadamard matrices. As none of the elements of X are in the set $\{0, 1\}$, there are no m-to-1 bit hash functions that are linear subset regular. □

We now show that if no m-to-1 bit linear subset regular hash functions exist, there exist no m-to-n bit linear subset regular hash functions.

Theorem 6.3. *There exists no m-to-n bit hash function that is linear subset regular.*

Proof. We show by induction on n. Let $P(n)$ denote the proposition:

There exists no m-to-n bit hash function that is linear subset regular.

The base case $P(1)$ is true by Theorem 6.2. Let $P(i)$ be true for some $i < n$. Then, we derive the truth table of Table 1.

Table 1. Truth table for an m-to-i bit hash function $h_{D,R}$; $\alpha_{j,\ell} \in \{0,1\}\ \forall\ j \in \{0,\ldots,2^m-1\}$ and $\ell \in \{0,\ldots,i-1\}$

x				$h(x)$			
x_{m-1}	x_{m-2}	$\cdots$	x_0	α_{i-1}	α_{i-2}	$\cdots$	α_0
0	0	$\cdots$	0	$\alpha_{0,i-1}$	$\alpha_{0,i-2}$	$\cdots$	$\alpha_{0,0}$
0	0	$\cdots$	1	$\alpha_{1,i-1}$	$\alpha_{1,i-2}$	$\cdots$	$\alpha_{1,0}$
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
0	1	$\cdots$	1	$\alpha_{2^{m-1}-1,i-1}$	$\alpha_{2^{m-1}-1,i-2}$	$\cdots$	$\alpha_{2^{m-1}-1,0}$
1	0	$\cdots$	0	$\alpha_{2^{m-1},i-1}$	$\alpha_{2^{m-1},i-2}$	$\cdots$	$\alpha_{2^{m-1},0}$
1	0	$\cdots$	1	$\alpha_{2^{m-1}+1,i-1}$	$\alpha_{2^{m-1}+1,i-2}$	$\cdots$	$\alpha_{2^{m-1}+1,0}$
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
1	1	$\cdots$	1	$\alpha_{2^{m-1},i-1}$	$\alpha_{2^{m-1},i-2}$	$\cdots$	$\alpha_{2^{m-1},0}$

Now, if a hash function is linear subset regular then it is

- regular, and
- subset regular under all linear conditions, each of which partitions the domain into two equally-sized sets.

Therefore, if a hash function is not linear subset regular then it is either (*i*) not regular or (*ii*) not subset regular with respect to at least one linear condition. Given case (*ii*), without loss of generality, let us assume that subset regularity does not hold for $x_{m-1} = 0$. Then, in the truth table in Table 1, at least one of the 2^{m-1} i-bit outputs corresponding to $x_{m-1} = 0$ appears more than the expected $2^{m-1}/2^i = 2^{m-i-1}$ times. Again, without loss of generality, let the output $\{0\}^i$ appear $t > 2^{m-i-1}$ times (i.e., say, $\alpha_{j,\ell} = 0\ \forall\ j \in \{0,\ldots,t-1\}$ and $\ell \in \{0,\ldots,i-1\}$ in Table 1). Then, if we append one bit to each of the t output strings $\{0\}^i$, one of the strings $\{0\}^i||0$ and $\{0\}^i||1$ appears *strictly* more than $2^{m-1}/2^{i+1} = 2^{m-i-2}$ times. Since $\{0\}^i||0$ and $\{0\}^i||1$ should each appear exactly 2^{m-i-2} times when the m-to-$(i+1)$ bit hash function is subset regular under the linear condition $x_{m-1} = 0$, $P(i+1)$ is true when $P(i)$ is true. Given case (*i*), following a similar line of reasoning, replacing 2^{m-1} with 2^m and recalculating the formulas, we obtain that

$P(i) \Rightarrow P(i+1)$. Therefore, by the principle of mathematical induction, $P(n)$ is true. □

Let us again consider a regular hash function with $d = 2^{162}$ and $r = 2^{160}$. If we require this function to be linear subset regular as well, the function must be subset regular for a fraction of $d-1$ out of all $\binom{d}{d/2}$ possible subsets consisting of half of the domain elements. For $d = 2^{162}$, this fraction evaluates to about $2^{-2^{162}}$. Therefore, by imposing subset regularity for only an extremely small fraction of the possible subsets that we consider, we prove that no linear subset regular functions exist.

In the previous section, we showed that the fraction of subset regular hash functions was negligible. In this section, we obtained an even stronger result: there does not exist a hash function that is regular for more than a negligibly small fraction of subsets of the domain.

Therefore, in the birthday attack, the attacker can always restrict the domain in such a way that the resulting hash function is not regular. This counters Bellare and Kohno's interpretation of why regular functions fare better than random functions against the birthday attack. However, we do not dispute the mathematics of their analysis.

7 Impact on the Birthday Attack

In the previous sections, we showed how unlikely it is that a hash function is regular, if the attacker restricts the inputs to a particular subset. We now use this observation to increase the success probability of the birthday attack against a regular hash function (for the same number of trials), compared to Bellare and Kohno's analysis.

Bellare and Kohno [2, §7.2] see a possibility for the attacker to restrict the domain to a smaller subset of $d = 2r > 10$ elements, and calculate that in this case, $C^{\$}_{D,R}(q) > (8/5) \cdot C^{\mathrm{reg}}_{D,R}(q)$. From this, they conclude that regular hash functions fare better than random hash functions against the birthday attack. However, Bellare and Kohno's analysis assumes that the attacker restricts the domain in such a way, that D consists of all strings of length $\log_2(r) + 1$ bits.

As Bellare and Kohno already pointed out, $C^{\mathrm{reg}}_{D,R}(q)$ approaches $C^{\$}_{D,R}(q)$ if the length of the input strings increases. Therefore, to increase the success probability of the birthday attack against a regular hash function with $d = 2r$ (for the same number of trials q), the attacker can consider long input messages. The attacker will then restrict these long input messages to a set of $d = 2r$ elements, and perform the birthday attack. Therefore, by increasing the length of the input messages (but still restricting the domain points in the birthday attack to $d = 2r$ elements), the success probability of the birthday attack against a regular hash function can be made arbitrarily close to that of a random hash function, for the same number of trials q. This contradicts Proposition 7.4 of [2], which states that if $|D| = 2|R| \geq 10$, then $C^{\$}_{D,R}(q) > (8/5) \cdot C^{\mathrm{reg}}_{D,R}(q)$ for all q satisfying $2 \leq q \leq 0.1 \cdot r^{1/2}$.

8 Related Work

In 1956, Dumey introduced the concept of (non-cryptographic) hashing [7]. It was proposed as a solution to the dictionary problem. In the dictionary problem, a sequence of operations $\textsc{Insert}(k,x)$, $\textsc{Delete}(k)$ and $\textsc{Lookup}(k)$ are given. They are used to respectively insert, delete and look up key-value pairs (k,x), and are performed on an initially empty table of key-value pairs. The goal is to minimize the time and memory used by these operations.

Let $h'_{D',R'} : D' \rightarrow R'$ be a hash function, where both the domain $|D'| = d'$ and the range $|R'| = r'$ are finite, and $d' > r'$. The construction known as *chained hashing* is then described as follows. We initialize an array $A[1 \ldots r']$, and let $A[i]$ contain a linked list of all key-value pairs (k,x) for which $h'(k) = i$.

Assume that $r' \mid d'$. For chained hashing, $h'_{D',R'}$ is ideally chosen such that every $A[i]$ contains the same number of key-value pairs. This is related to the notion of a regular hash function by Bellare and Kohno, where every hash value has the same number of preimages in the domain D'. If D' is the set of all keys that are added to the table, then the number of key-value pairs that have to be read when either of the three operations are performed, is at most d'/r'. If there exists an $A[i]$ with fewer than d'/r' elements, then there also exists an $A[j]$ where $i \neq j$ with more than d'/r' elements. Therefore, regular hash functions obtain the best performance in the worst-case scenario.

Doing a rigorous analysis of chained hashing is difficult, because the calculations strongly depend on the sequence of keys k. For example, by the pigeonhole principle there always exists a sequence of keys k that all map to the same hash value $h'_{D',R'}(k)$. Sometimes assumptions are placed on the sequence of keys k, but these may be very difficult (or even impossible) to guarantee in practice. This is also evident from the analysis in our paper.

As a novel solution to the dictionary problem, we mention the universal classes of hash functions proposed by Carter and Wegman [6]. In their paper, it is proposed that $h'_{D',R'}$ is chosen uniformly at random, but not from the set of all possible functions. The class of hash functions $H'_{D',R'}$ is chosen in such a way, that the average performance (for all $h' \in H'$) for the worst case input is bounded.

More formally, let $h'_{D',R'} : D' \rightarrow R'$ be a hash function, where both the domain $|D'| = d'$ and the range $|R'| = r'$ are finite, and $d' \geq r'$. A universal class of hash functions is then defined such that for a randomly chosen $h'_{D',R'} \in H'$, the probability that any $h'_{D',R'}(x) = h'_{D',R'}(y)$ is at most $1/r'$ for any two distinct x and y.

However, not all protocols allow a hash function to be selected uniformly at random from a class of hash functions. In that case, the notion of universal classes of hash functions is not meaningful.

9 Random Functions

Bellare and Kohno showed [2] that several reduced versions of SHA-1 do not behave as regular functions. This indicates that regular functions may not be a suitable theoretical model to analyze the collision resistance of commonly used hash functions. In previous sections, we also made an observation on Bellare and Kohno's claim that regular hash functions fare better than random hash functions against the birthday attack. Based on this, we suggest not to model hash functions as regular functions.

Instead, we propose to model hash functions as random functions when analyzing the complexity of the birthday attack. We agree with Bellare and Kohno that "the design principle of attempting to make hash functions have random behavior [...] is sound and central to security" [2]. We now explain why random functions do not suffer from any of the problems described in this paper.

A random function can be defined as follows:

Definition 9.1 (Random Function). Let $F : \{0,1\}^* \rightarrow \{0,1\}^n$ be a random function. If $x_i \in \{0,1\}^n$ has not been queried before, the random function chooses y_i uniformly at random from all 2^n range points, and outputs $y_i = F(x_i)$. Otherwise, if $x_i = x_j$ where $j < i$, the random function outputs $y_j = F(x_j) = F(x_i)$.

Unlike for a regular hash function, it is not necessary for a random function to require that the domain consists of a finite number of elements. Also, it is clear from the random function definition, that for any subset of the domain, the range points y_i are chosen randomly and independently from a uniform distribution as well. The statistics of a random function are the same, no matter how the domain points are chosen. Therefore, for a random function, the success probability of the birthday attack does not depend on how the domain points are chosen.

10 Conclusions

The notion of a regular hash function was introduced by Bellare and Kohno at EUROCRYPT 2004, and has subsequently appeared in several research papers. It is defined as a hash function that has the same number of preimages in the domain for every hash value. In their original paper, Bellare and Kohno state that "regular functions fare better than random functions [against the birthday attack]".

We explain that this statement, which until now remained unchallenged, is based on the assumption that the attacker chooses the domain points uniformly at random. However, Bellare and Kohno note that "there are several variants of [the birthday attack] which differ in the way the [domain] points $x_1, \ldots, x_q$ are chosen." One possible restriction is that domain points correspond to meaningful messages. For example, if messages consist of only ASCII characters, the most significant bit of every character must be zero.

For simplicity, we assumed that the choices of the attacker are restricted to half of the domain points. In that case, we calculate that the probability that the resulting function is still regular under this restriction is very close to zero.

We then attempt to extend the concept of regularity, and require that a hash function is also regular under subsets of the domain. We prove that no such hash function exists, even if we consider only a very small fraction of all possible ways to divide the domain into subsets.

Thus, the attacker can restrict the domain points in the birthday attack in such a way that the resulting hash function is not regular. This is our point of disagreement with Bellare and Kohno's analysis of why regular functions perform better than random functions against the birthday attack.

We show how the success probability of the birthday attack against a regular hash function can be increased (for the same number of trials), compared to Bellare and Kohno's analysis. Our results hold even for a highly restricted domain.

If hash functions are modeled as random functions, the choice of the domain points does not change the success probability of the birthday attack.

A Linear Subset Regularity for 3-to-1 Bit Hash Functions

Here, we will attempt to construct a 3-to-1 bit linear subset regular hash function $h_{D,R}(x)$. Let the input x be a binary string, resulting from the concatenation of the three input bits, such that $x \leftarrow x_2 \parallel x_1 \parallel x_0$. We set $h_{D,R}(000) = A$, where A can be either 0 or 1. The other output symbol will then be denoted by B. We now consider three cases, as shown in Table 2.

- **Case 1:** Assume $h_{D,R}(001) = A$. Subset regularity with respect to x_2 then leads to $h_{D,R}(010) = h_{D,R}(011) = B$. Furthermore, subset regularity with respect to x_1 results in $h_{D,R}(100) = h_{D,R}(101) = B$. For $h_{D,R}$ to be regular, we must have $h_{D,R}(110) = h_{D,R}(111) = A$. However, we now find that restricting the inputs to $x_2 \oplus x_1 = 0$ results in a constant function.

Table 2. Constructing a 3-to-1 bit linear subset regular hash function $h_{D,R}(x)$, where $x \leftarrow x_2 \parallel x_1 \parallel x_0$; the values in bold were set initially, the others are derived from the linear subset regular conditions

Case 1				Case 2				Case 3			
x_2	x_1	x_0	$h_{D,R}(x)$	x_2	x_1	x_0	$h_{D,R}(x)$	x_2	x_1	x_0	$h_{D,R}(x)$
0	0	0	***A***	0	0	0	***A***	0	0	0	***A***
0	0	1	***A***	0	0	1	***B***	0	0	1	***B***
0	1	0	B	0	1	0	***A***	0	1	0	***B***
0	1	1	B	0	1	1	B	0	1	1	A
1	0	0	B	1	0	0	B	1	0	0	B
1	0	1	B	1	0	1	A	1	0	1	A
1	1	0	A	1	1	0	B	1	1	0	A
1	1	1	A	1	1	1	A	1	1	1	B

- **Case 2:** Assume $h_{D,R}(001) = B$ and $h_{D,R}(010) = A$. Subset regularity with respect to x_2 then leads to $h_{D,R}(011) = B$. Furthermore, subset regularity with respect to x_0 results in $h_{D,R}(100) = h_{D,R}(110) = B$. For $h_{D,R}$ to be regular, we must have $h_{D,R}(101) = h_{D,R}(111) = A$. However, we now find that restricting the inputs to $x_2 \oplus x_0 = 0$ results in a constant function.
- **Case 3:** Assume $h_{D,R}(001) = B$ and $h_{D,R}(010) = B$. Subset regularity with respect to x_2 then leads to $h_{D,R}(011) = A$. Furthermore, subset regularity with respect to $x_1 \oplus x_0$ results in $h_{D,R}(100) = h_{D,R}(111) = B$. For $h_{D,R}$ to be regular, we must have $h_{D,R}(101) = h_{D,R}(110) = A$. However, we now find that restricting the inputs to $x_2 \oplus x_1 \oplus x_0 = 0$ results in a constant function.

Consequently, there are no 3-to-1 bit hash functions that are linear subset regular. Also note that imposing all but one linear subset regular condition in Table 2 leads to an affine hash function. We found by exhaustive search that all 3-to-1 bit hash functions where all but one linear subset regular conditions are imposed, result in affine hash functions.

B Calculating the Inverses of Matrices A_d

In this section, we prove that the matrices A_d of Theorem 6.2 are invertible, by showing their relation to Hadamard matrices. We give an explicit formula for their inverses.

Hadamard matrices are square matrices of which all elements are either 1 or -1. They were initially proposed by Sylvester [21]. Hadamard [12] later showed that these matrices are the solution to his maximum determinant problem. A $d \times d$ Hadamard matrix H_d can be defined a matrix satisfying

$$H_d H_d^T = dI_d \ , \tag{B.1}$$

where I_d denotes the $d \times d$ identity matrix.
If d is a power of two, Sylvester [21] proposed the following construction for H_d:

$$H_1 = [1] \,, \tag{B.2}$$

$$H_d = \begin{bmatrix} H_{d/2} & H_{d/2} \\ H_{d/2} & -H_{d/2} \end{bmatrix}, \quad \text{for } 1 \le \log_2(d) \in \mathbb{N} \ . \tag{B.3}$$

Let J_d be the $d \times d$ matrix where every element is equal to one. Matrix K_d is the $d \times d$ matrix where every element of the first column is 1, and all other elements are zero. Note that $K_d K_d^T = J_d$. Matrices A_d of Theorem 6.2 satisfy the equation $H_d = 2A_d - J_d$. We now show that matrices A_d are invertible, and calculate their inverse. Using equation (B.1), we obtain

$$\begin{aligned}
(2A_d - J_d)(2A_d - J_d)^T &= dI_d && \text{(B.4)}\\
\Leftrightarrow (2A_d - J_d)(2A_d^T - J_d^T) &= dI_d \\
\Leftrightarrow 4A_dA_d^T - 2A_dJ_d^T - 2J_dA_d^T + J_dJ_d^T &= dI_d \\
\Leftrightarrow 4A_dA_d^T - d(K_d^T + J_d) - d(K_d + J_d^T) + dJ_d &= dI_d \\
\Leftrightarrow 4A_dA_d^T - dK_d^T - dK_d - dJ_d^T &= dI_d \\
\Leftrightarrow 4A_dA_d^T - dK_d^T - dK_d - dK_dK_d^T &= dI_d \\
\Leftrightarrow 4A_dA_d^T = d(K_d + I_d)(K_d + I_d)^T & && \text{(B.5)}
\end{aligned}$$

As $K_dK_d = K_d$, we have $(K_d + I_d)(I_d - K_d/2) = K_d - K_dK_d/2 + I_d - K_d/2 = I_d$. Therefore, $(K_d + I_d)^{-1} = I_d - K_d/2$. From equation (B.5), we then obtain

$$A_dA_d^T(2I_d - K_d)^T(2I_d - K_d) = dI_d \ . \tag{B.6}$$

This equation shows that A_d is invertible, and that its inverse is given by

$$A_d^{-1} = \frac{1}{d}A_d^T(2I_d - K_d)^T(2I_d - K_d) \ . \tag{B.7}$$

Bibliography

[1] Mihir Bellare and Tadayoshi Kohno, Hash Function Balance and Its Impact on Birthday Attacks, in: *EUROCRYPT* (Christian Cachin and Jan Camenisch, eds.), LNCS 3027, pp. 401–418, Springer, 2004.

[2] Mihir Bellare and Tadayoshi Kohno, *Hash Function Balance and Its Impact on Birthday Attacks*, Full version available online, 2004, `http://cseweb.ucsd.edu/~mihir/papers/balance.html`.

[3] D. M. Bloom, A birthday problem, *American Mathematical Monthly* **80** (1973), 1141–1142.

[4] Johannes A. Buchmann, *Introduction to Cryptography, Second Edition*, Springer, 2004.

[5] Christian Cachin, *Entropy Measures and Unconditional Security in Cryptography*, Ph.D. thesis, ETH Zurich, 1997, Reprint as vol. 1 of *ETH Series in Information*

Security and Cryptography, ISBN 3-89649-185-7, Hartung-Gorre Verlag, Konstanz, 1997.

[6] J. Lawrence Carter and Mark N. Wegman, Universal Classes of Hash Functions, *Journal of Computer and System Sciences* **18** (1979), 143–154.

[7] Arnold I. Dumey, Indexing for Rapid Random Access Memory Systems, *Computers and Automation* **5** (1956), 6–9.

[8] Federal Information Processing Standards, *FIPS 180-1: Secure Hash Standard*, `http://www.itl.nist.gov/fipspubs/fip180-1.htm`, 1995.

[9] W. Feller, *An Introduction to Probability Theory and its Applications*, John Wiley, New York, NY, USA, 1950.

[10] Amos Fiat and Adi Shamir, How to Prove Yourself: Practical Solutions to Identification and Signature Problems, in: *CRYPTO* (Andrew M. Odlyzko, ed.), LNCS 263, pp. 186–194, Springer, 1986.

[11] Shafi Goldwasser and Mihir Bellare, *Lecture Notes on Cryptography*, Available at: `http://cseweb.ucsd.edu/~mihir/papers/gb.pdf`, 2008.

[12] Jacques Hadamard, Résolution d'une question relative aux déterminants, *Bulletin des Sciences Mathématiques* **17** (1893), 240–246.

[13] Antoine Joux, *Algorithmic Cryptanalysis*, Chapman & Hall / CRC, 2009.

[14] Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996.

[15] National Institute of Standards and Technology, Announcing Request for Candidate Algorithm Nominations for a New Cryptographic Hash Algorithm (SHA-3) Family, *Federal Register* **27** (2007), 62212–62220, Available: `http://csrc.nist.gov/groups/ST/hash/documents/FR_Notice_Nov07.pdf` (2008/10/17).

[16] Thomas S. Nunnikhoven, A Birthday Problem Solution for Nonuniform Birth Frequencies, *The American Statistician* **46** (1992), 270–274.

[17] Rune Steinsmo Ødegård and Danilo Gligoroski, *On the Randomness and Regularity of Reduced EDON-$\mathcal{R}$ Compression Function*, Cryptology ePrint Archive, Report 2009/234, 2009, `http://eprint.iacr.org/`.

[18] Jean-Jacques Quisquater and Jean-Paul Delescaille, How Easy is Collision Search? Application to DES (Extended Summary), in: *EUROCRYPT*, pp. 429–434, 1989.

[19] Douglas Robert Stinson, *Introduction to Cryptography, First Edition*, CRC-Press, 1995.

[20] Douglas Robert Stinson, *Introduction to Cryptography, Second Edition*, Chapman & Hall / CRC, 2002.

[21] James Joseph Sylvester, Thoughts on inverse orthogonal matrices, simultaneous signsuccessions, and tessellated pavements in two or more colours, with applications to Newton's rule, ornamental tile-work, and the theory of numbers, *Philosophical Magazine* **34** (1867), 461–475.

[22] Paul C. van Oorschot and Michael J. Wiener, Parallel Collision Search with Cryptanalytic Applications, *J. Cryptology* **12** (1999), 1–28.

[23] R. von Mises, Über Aufteilungs- und Besetzungswahrscheinlichkeiten. (On Partitioning and Occupation Probabilities), *İstanbul Üniversitesi Fen Fakültesi Mecmuasi* **4** (1939), 145–163.

[24] David Wagner, A Generalized Birthday Problem, in: *CRYPTO* (Moti Yung, ed.), LNCS 2442, pp. 288–303, Springer, 2002.

[25] Hirotaka Yoshida, Dai Watanabe, Katsuyuki Okeya, Jun Kitahara, Hongjun Wu, Özgül Küçük and Bart Preneel, MAME: A Compression Function with Reduced Hardware Requirements, in: *CHES* (Pascal Paillier and Ingrid Verbauwhede, eds.), LNCS 4727, pp. 148–165, Springer, 2007.

[26] Gideon Yuval, How to Swindle Rabin, *Cryptologia* **3** (1979), 187–189.

Author information

Nicky Mouha, Katholieke Universiteit Leuven, Dept. ESAT/SCD-COSIC and IBBT,
Kasteelpark Arenberg 10, 3001 Heverlee, Belgium.
E-mail: `Nicky.Mouha@esat.kuleuven.be`

Gautham Sekar, Indian Statistical Institute, Chennai Centre, SETS Campus,
MGR Knowledge City, CIT Campus, Taramani, Chennai 600113, India.
E-mail: `sgautham@isichennai.res.in`

Bart Preneel, Katholieke Universiteit Leuven, Dept. ESAT/SCD-COSIC and IBBT,
Kasteelpark Arenberg 10, 3001 Heverlee, Belgium.
E-mail: `Bart.Preneel@esat.kuleuven.be`