Cryptanalysis of the Yi-Lam hash

David Wagner
University of California, Berkeley
daw@cs.berkeley.edu

Abstract. This paper analyzes the security of a hash mode recently proposed by Yi and Lam. Given a block cipher with m-bit block size and $2m$-bit key, they build a hash function with $2m$-bit outputs that can hash messages as fast as the underlying block cipher can encrypt. This construction was conjectured to have ideal security, i.e., to resist all collision attacks faster than brute force. We disprove this conjecture by presenting a collision attack that is substantially faster than brute force and which could even be considered practical for typical security parameters.

1 Introduction

The public cryptographic community has over 20 years of experience in building secure block ciphers. In contrast, the design of cryptographic hash functions has received only about half as many years of research. Yet hash functions are still a very important primitive to practitioners. Therefore, there is much interest in the problem of building a secure, fast hash function out of a secure block cipher.

This research program has been troubled by two major challenges. First, most existing block ciphers have a 64 bit block size, but a hash function with a 64 bit output cannot possibly resist collision search. Therefore, one must somehow securely double the width of the internal state, and this appears to be a non-trivial endeavor. Second, it is hard to maintain efficiency without sacrificing security. The critical figure of merit is the *rate* of the hash function, which is defined as the number of m-bit message blocks hashed per encryption, where m is the block size of the underlying cipher. Many early proposals for building fast hash functions have been broken; in particular, Knudsen, Lai, and Preneel cryptanalyzed a large class of double-length hash functions of rate 1 [1–4, 8].

In *ACISP '97*, Yi and Lam proposed a new construction for building a hash function from a block cipher (e.g., IDEA) with m-bit block width and $2m$-bit key size [10]. Typically, we will have $m = 64$. The Yi-Lam scheme has rate 1 and yields $2m$-bit outputs. With such high performance, it is an attractive candidate for building a fast hash function.

One crucial feature of the Yi-Lam design is the inclusion of incompatible group operations (XOR and addition modulo 2^m) to combine internal state variables. Most previous work had used only XOR operations, so that the only nonlinear component was the block cipher; however, Knudsen, Lai, and Preneel's work broke all such hash functions of rate 1 using m-bit keys. The use of incompatible group operations in the Yi-Lam hash is apparently intended to help frustrate those types of attacks.

Nonetheless, in this paper we still manage to find fast collision attacks on Yi and Lam's proposal. Our attacks work by controlling the effect of the "carry bits." This shows that the incompatibility of XOR and addition does not add much strength to the Yi-Lam hash mode.

Our primary results are as follows. We describe how to find full collisions for the Yi-Lam hash with $2^{.71m}$ work. Also, we give a free-start collision attack (also known as a pseudo-collision attack [6, Section 9.7.2]) that requires only $2^{m/2}$ steps of computation, when the adversary can choose the initial starting state of the hash function. Both attacks are very practical to implement for $m = 64$. Note that Satoh, Haga, and Kurosawa have previously shown that there are second-preimage attacks against this hash with complexity about 2^m [9], in contrast to its conjectured 2^{2m} security level. Thus, we may conclude that the Yi-Lam hash is neither strongly collision-free nor strongly one-way.

Outline. This paper is organized as follows. Section 2 describes Yi and Lam's new scheme briefly, for reference, and Section 3 shows how to find collisions in their proposal. Finally, we conclude the paper in Section 4. Appendix A includes a proof of our main lemma on the correlation between XOR and addition.

2 Description of the Yi-Lam hash

A bit of notation is in order. We write $E(k, x)$ for the encryption of block x under key k. The block cipher is assumed to be free of any weaknesses. Also, $x||y$ stands for the concatenation of two blocks x and y. We let $x \oplus y$ stand for the XOR of x and y, and write $x + y$ for the addition modulo 2^m of x and y. Since Yi and Lam did not name their scheme, we simply call it the Yi-Lam hash in this paper.

The Yi-Lam hash operates as follows. We pad the message and divide the result into k blocks of size m, denoted by $M_0, \ldots, M_{k-1}$. The $2m$-bit internal state is named $G||H$ and is initialized to a fixed public value $G_0||H_0$. We denote the compression function by h, and define h by $h(G||H, M) = G'||H'$, where G', H' satisfy

$$t = E(H||M, G) \qquad G' = (t \oplus M) + H \bmod 2^m \qquad H' = t \oplus G.$$

The final hash digest is computed as $h(\ldots h(h(G_0||H_0, M_0), M_1) \ldots, M_{k-1})$. This construction is illustrated pictorially in Figure 1.

Caution. Beware: there are actually two hashing constructions which could be called "the Yi-Lam hash." One such scheme was proposed in *Electronics Letters* [11] (and subsequently cryptanalyzed [5]); another was published in *ACISP'97* [10]; and the two designs are quite different. In this paper, we focus on analysis of the *ACISP'97* proposal. To avoid confusion, phrases such as "the Yi-Lam hash" should be understood to refer to the *ACISP'97* proposal.

3 Collision resistance

In this section, we explore the collision-resistance of the Yi-Lam hash. We exhibit a free-start collision attack with complexity $2^{m/2}$, and then we extend this to a full

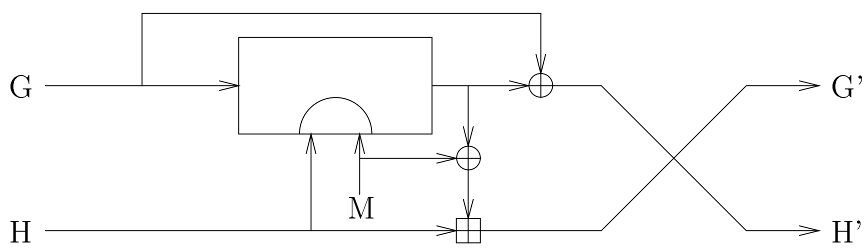

Fig. 1. The Yi-Lam hash.

collision attack with complexity $2^{.71m}$. This is substantially lower than the conjectured security factor of 2^m for security against collision attacks. For the suggested parameters (i.e., $m = 64$), even the full collision attack is well within reach of most adversaries in practice, since it requires only $2^{.71\times 64} < 2^{46}$ work.

The free-start attack. First, we describe a free-start collision attack on the Yi-Lam hash, as promised. Calculate

$$G'_i||H'_i = h(G_i||0, G_i) \qquad i = 1, \ldots, n,$$

for $n = 2^{m/2}$ values of G_i, observing that $G'_i = H'_i$ for all i. We search for a pair i, j with $H'_i = H'_j$. Then $G'_i = H'_i = H'_j = G'_j$, so this produces a pseudo-collision for the compression function.

The total computational complexity of this attack is $2^{m/2}$ encryptions. A naive implementation of this attack might require $2^{m/2}$ units of storage, but we note that using Floyd's cycle-finding algorithm or any of its improvements [7] reduces the storage complexity of the attack to a very small constant.

The full collision attack. In the remainder of this section, we will analyze the security of the Yi-Lam hash against full collision attacks. Let $G_i||H_i$ (for $i = 1, \ldots, n$) be any n values for the chaining variables. For instance, we could select $G_i||H_i = h(G_0||H_0, X_i)$ for n different message blocks X_i.

To aid the intuition, let's first consider a variant of the Yi-Lam hash where all additions are replaced by XORs. Imagine calculating the values

$$G'_i||H'_i = h(G_i||H_i, G_i \oplus H_i) \qquad i = 1, \ldots, n,$$

for $n = 2^{m/2}$ arbitrary values of $G_i||H_i$ (which could be obtained by hashing n different message prefixes). Then we would have the relation $G'_i = H'_i$ for all i, since

$$G'_i \oplus H'_i = t_i \oplus M_i \oplus H_i \oplus t_i \oplus G_i = 0. \tag{1}$$

After about $n = 2^{m/2}$ trial hashes, we would expect to find some $i \neq j$ such that $H'_i = H'_j$ by the birthday paradox. This would imply that $G'_i = H'_i = H'_j = G'_j$, so this algorithm would give a full collision in this Yi-Lam variant after $2^{m/2}$ trial hash computations.

Of course, the real Yi-Lam hash mixes addition with XOR to prevent these sorts of attacks. So we need to extend our analysis to handle the carry bits. We begin by establishing a simple lemma on the incompatibility of XOR and addition.

Lemma 1. *If a, b, c are independently and uniformly distributed over all m-bit values, then*

$$\Pr[(a \oplus c) - (b \oplus c) \equiv a - b \bmod 2^m] = (3/4)^{m-1} > 2^{-.42m}.$$

Proof. See Appendix A.

For the collision attack on the full Yi-Lam hash, we suggest calculating the values

$$G'_i||H'_i = h(G_i||H_i, G_i \oplus (-H_i)) \qquad i = 1, \ldots, n,$$

for n values of $G_i||H_i$. where $-H_i$ denotes the additive inverse of H_i modulo 2^m. The analysis goes as follows. Suppose that, for some pair $i \neq j$, we have $H'_i = H'_j$. Then $G'_i = (H'_i \oplus (-H_i)) + H_i$, and similarly $G'_j = (H'_j \oplus (-H_j)) + H_j = (H'_i \oplus (-H_j)) + H_j$. Now the lemma ensures that

$$(H'_i \oplus (-H_i)) - (H'_i \oplus (-H_j)) \equiv (-H_i) - (-H_j) \equiv H_j - H_i \bmod 2^m$$

holds with probability $> 2^{-.42m}$, so $G'_i = G'_j$ with the same probability. With $n = 2^{.71m}$ trials, we expect to see $2^{.42m-1}$ pairs i, j such that $H'_i = H'_j$, which is enough that with non-negligible probability we expect to see one of them where $G'_i = G'_j$ also holds.

To summarize, this shows how to find a collision in the Yi-Lam hash with about $2^{.71m}$ offline hash computations. We expect that the storage complexity of the collision-finding attack will be negligible, if parallel collision search techniques are used to implement the attack [7].

4 Conclusions

We have shown that the Yi-Lam hash has serious flaws, and it is clear that the construction offers only minor benefits over traditional single-length hash functions.

The fundamental problem is that the Yi-Lam construction relied on the non-linearity of the carry bits found in modular addition. However, addition possesses only mild nonlinearity properties, and we have seen that in this case they were not enough to resist attack.

We considered a Yi-Lam variant where the addition operation is replaced by an XOR operation, and found a relation (see Equation 1) between the inputs and outputs of the compression function that is linear over XOR. This linear relation allows one to break the variant with a trivial attack. With the addition operation in place, the corresponding relation is no longer XOR-linear, but it is nearly so, and this permits a simple extension to the previous attack to handle the slight nonlinearity.

Therefore, our results provide some evidence to suggest that merely including an incompatible group operation in the hash design may not be sufficient to assure security. This does not rule out the possibility that mixing incompatible operations might improve the security of some cipher-based hash functions—at least in the case of the Yi-Lam hash, the inclusion of the addition operation did appear to frustrate certain trivial attacks—but we advise caution. We propose that in the future it would be wise for designers to avoid relying on addition modulo 2^m (instead of XOR) too much.

5 Acknowledgements

The author wishes to gratefully acknowledge a number of helpful comments from Lars Knudsen and from the anonymous reviewers of FSE'99 and ASIACRYPT 2000.

References

1. L. Knudsen and X. Lai, "New attacks on all Double Block Length Hash Functions of Hash Rate 1, including the Parallel DM," *EUROCRYPT'94*, Springer Verlag, LNCS 950.
2. L.R. Knudsen, X. Lai, and B. Preneel, "Attacks on fast double block length hash functions," *Journal of Cryptology*, Winter 1998, vol.11, (no.1):59–72.
3. L. Knudsen and B. Preneel, "Hash Functions Based on Block Ciphers and Quaternary Codes," *ASIACRYPT'96*, Springer Verlag, LNCS 1163.
4. X. Lai and L. Knudsen, "Attacks on Double Block Length Hash Functions," *Fast Software Encryption '93*, Springer Verlag, LNCS 809.
5. K.M. Martin and C.J. Mitchell, "Analysis of hash function of Yi and Lam," *Electronics Letters*, vol.34, no. 24, 1998, pp.2327–2328.
6. A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, Boca Raton, 1997.
7. P.C. van Oorschot and M.J. Wiener, "Parallel Collision Search with Cryptanalytic Applications," *Journal of Cryptology*, vol.12, no. 1, 1999, pp.1–28.

8. B. Preneel, "Analysis and design of cryptographic hash functions," Doctoral dissertation, Katholieke Universiteit Leuven, 1993.
9. T. Satoh, M. Haga, and K. Kurosawa, "Towards Secure and Fast Hash Functions," *IEICE Trans. Fundamentals*, vol. E82-A, no. 1, Jan. 1999.
10. X. Yi and K.-Y. Lam, "A new hash function based on block cipher," *ACISP'97*, Second Australasian Conference on Information Security and Privacy, Springer, LNCS 1270.
11. X. Yi and K.-Y. Lam, "Hash function based on block cipher," *Electronics Letters*, 6 Nov 1997, vol.33, (no.23):1938–1940, IEEE.

A Proof of Lemma 1

Lemma 1. *If a, b, c are independently and uniformly distributed over all m-bit values, then*

$$\Pr[(a \oplus c) - (b \oplus c) \equiv a - b \bmod 2^m] = (3/4)^{m-1} > 2^{-.42m}.$$

Proof. We rearrange the equation to become

$$(a \oplus c) + b \equiv a + (b \oplus c) \bmod 2^m.$$

Let d stand for the left-hand carry bits, and e the right-hand carry bits. Then $(a \oplus c) + b \equiv (a \oplus c) \oplus b \oplus d \bmod 2^m$, and $a + (b \oplus c) \equiv a \oplus (b \oplus c) \oplus e \bmod 2^m$, so our required equation becomes

$$(a \oplus c) \oplus b \oplus d \equiv a \oplus (b \oplus c) \oplus e \bmod 2^m.$$

In short, we need only calculate the probability that $d = e$. Let d_j be the j-th bit position in d, for $j = 0, \ldots, m-1$, where d_0 is the least significant bit of d. Define e_j and c_j similarly. We have $d_0 = e_0 = 0$ trivially. Now we proceed inductively. Suppose that $d_i = e_i$ for $i = 0, \ldots, j$ where $0 \le j < 31$. We know that d_{j+1} is computed as the majority function of $a_j \oplus c_j$, b_j, and d_j; also, e_{j+1} is calculated as the majority of a_j, $b_j \oplus c_j$, and $e_j = d_j$. If $c_j = 0$, then clearly $d_{j+1} = e_{j+1}$. On the other hand, if $c_j = 1$, then $d_{j+1} = e_{j+1}$ holds with probability exactly $1/2$ (i.e., in the case that $a_j \neq b_j$). Moreover, the probabilities for each bit position are independent, so we can multiply them. Therefore, for any fixed value of c the probability is $2^{-W(\overline{c})}$, where $W(\overline{c})$ denotes the Hamming weight of $\overline{c}$, and $\overline{c}$ denotes the $m-1$ least significant bits of c.

Summing over c and applying the binomial theorem, we get that the desired probability p satisfies

$$\begin{aligned}
p &= 2^{-m} \sum_{c} 2^{-W(\overline{c})} \\
&= 2^{-m+1} \sum_{c' < 2^{m-1}} 2^{-W(c')} \\
&= 2^{-m+1} \sum_{w=0}^{m-1} \binom{m-1}{w} 2^{-w} \\
&= 2^{-m+1} \cdot (1 + 1/2)^{m-1} \\
&= 2^{-m+1} \cdot (3/2)^{m-1} \\
&= (3/4)^{m-1} \\
&\approx 2^{-.415(m-1)} \\
&> 2^{-.42m}.
\end{aligned}$$

This completes the proof. ∎