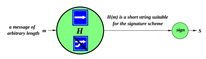
New Techniques for Cryptanalysis of Cryptographic Hash Functions

RAFI CHEN ADVISOR: PROF. ELI BIHAM

Cryptographic Hash Functions

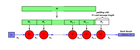
A cryptographic hash function takes a message of arbitrary length and computes a short fingerprint of the message.
A cryptographic hash function should be a **one–way** as well as **Collision Resistant**, i.e., it is *computationally* infeasible to find a message with a given hash value, and it is *computationally* infeasible to find two messages that have the same hash value.
Hash functions are used for digital signatures, information authentication, pseudo random generators and many other applications.



Introduction

In most of the attacks on hash functions the attacker tries to find two colliding messages by using differential cryptanalysis techniques. Up to 2004 all of the known attacks concentrated on collisions of single–block messages.
In our research we improved the attacks two–fold, both on the search level and on the meta–structure. The former led to the **neutral bits technique** that substantially reduces the attack complexity, while the latter led to the **multi–block technique** which shows that finding a multi–block collision may be easier than finding a collision of a single block. These two techniques led to the first known collision of SHA-0: a four-block collision by Joux. We also constructed the first attacks on reduced SHA-1.

Description of SHA-0 and SHA-1



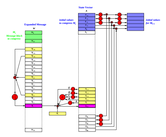
1. A message M is appended with '1', followed by a variable number of '0's, followed by a 64-bit representation of the message length. The total length of the padded message is a multiple of 512 bits.
2. The padded message is divided into blocks of 512 bits: $M_1, \ldots, M_n$, and a 5-word chaining variable h_0 is initialized.
3. A compression function processes each message block with the last chaining variable, and generates a new chaining variable
$$h_j = C(M_j, h_{j-1}).$$
4. h_n is the output of the hash function.

The Compression Function

1. **Divide** the message block M_j into 16 words $W_0, \ldots, W_{15}$, and expand the 16 words to 80 words by:
$$W_i = (W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16}) \lll s, \qquad i = 16, \ \ldots, \ 79,$$
where s is 0 for SHA-0 and 1 for SHA-1.
2. **Divide** h_{j-1} to the five first entries of a state vector A by:
$$(A_0, A_{-1}, A_{-2}, A_{-3}, A_{-4}) = (h^0_{j-1}, h^1_{j-1}, h^2_{j-1} \ggg 30, h^3_{j-1} \ggg 30, h^4_{j-1} \ggg 30).$$
3. **Compute** the entries $1, \ldots, 80$ of the state vector by:
$$A_i = A_{i-1} \lll 5 + f_i(A_{i-2}, A_{i-3} \lll 30, A_{i-4} \lll 30) + A_{i-5} \lll 30 + W_{i-1} + K_i.$$
Where the functions and constants used in each round are:

Round	Function	$f_i(X,Y,Z)$	K_i
$0 \le t \le 19$	IF	$XY \vee XZ$	5A827999
$20 \le t \le 39$	XOR	$X \oplus Y \oplus Z$	6ED9EBA1
$40 \le t \le 59$	MAJ	$XY \vee XZ \vee YZ$	8F1BBCDC
$60 \le t \le 79$	XOR	$X \oplus Y \oplus Z$	CA62C1D6

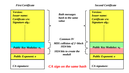
4. Denote $g_j = (A_{80}, A_{79}, A_{78} \lll 30, A_{77} \lll 30, A_{76} \lll 30)$.
5. **The output** of the compression function is:
$h_j = h_{j-1} + g_j$ (wordwise).



The Importance of Collision Resistance

We demonstrate the importance of the collision resistant property by three examples: The first uses X.509 certificates and the second uses postscript files. These two examples utilize pseudo-random collisions. The third example is a direct collision with partial meaningful English words.

Two MD5 Certificates with the Same Signature



The first parts of the two certificates are identical and consist of the issuer and subject details. The public moduli is stored in the next 2048 bits. The first 1024 bits of the moduli are created by generating a two-block collision. The remaining 1024 bits are same in both, and are computed such that the factors of both full 2048–bit moduli are known. Then, the same public exponent is appended.
The hash values of the two certificates is equal. Once the CA signs one of the certificates, the attacker has the signature of the other as well.

Same Signature on Two Different Documents

Daum and Lucks showed how two postscript files with the same signature prints two totally different documents. They use pseudo–random collision of MD5.

May,22,2005
Order:
Alice Falbalais given a full access to all confidential and secret information about GAUL.
Sincerely,
Julius Caesar

May,22,2005
To Whom it May Concern:
Alice Falbalais fulfilled all the requirements of the Roman Empire intern position. She was excellent at translating Roman into her gaul native language, learned very rapidly, and worked with considerable independence and confidence.
Her basic work habits such as punctuality, interpersonal deportment, communication skills, and completing assigned and self-determined goals were all excellent.
I recommend Alice for challenging positions in which creativity, reliability, and languages skills are required.
I highly recommend hiring her. If you would like to discuss her attributes in more detail, please don't hesitate to contact me.
Sincerely,
Julius Caesar

The two postscript files contain a collision M and M^* respectively, concatenated to the two letters, and a conditional statement that displays the first or second letter in accordance with the value of M or M^*. Both have the same signature, and thus a signature on one leaks a signature on the other.
Another approach was also described for MS-WORD files.

Colliding Messages with Meaningful Words

A 34–round reduced version of SHA-1 has the following collisions with partial English text:

Message 1:
`I Am OilMANgujnPay916472136314$USAkNOWwTkjepMFXGlmfHNGcpodElGfvL`
Message 2:
`I am KilMANgunfPay11607213.312$USASNOWSTknipMFtGlmnHNGkpodmlGbvL`
Message 1:
`OhG, not this mess,age notThat onenot U, oh noHRtBMTkKllLlIluvpB`
Message 2:
`Ohg, jot this$eess$aga notLhar oneVot q, kd nodRtBETkKdlLlalurpB`

Differential Cryptanalysis of SHA-0 and SHA-1

Local Collisions with Differential Attack

The attacks on SHA-0 and SHA-1 compression functions are based on the ability to create local collisions by applying patterns of differences.
A pattern that creates a local collision starts with a difference of one bit in $\mathbf{W}^j_i$ that we call **disturbance**. The disturbance creates a **difference** in $\mathbf{A}^j_{i+1}$. The pattern continues with five additional differences that we call **corrections**. These corrections avoid the propagation of the difference from A^j_{i+1} to other bits. The computation of the rounds with these differences are as follows:

$$\begin{aligned}
\mathbf{A}^j_{i+1} &= \mathbf{W}^j_i + A^{j-5}_i + f^j_{i+1}(A^j_{i-1}, A^{j-30}_{i-2}, A^{j-30}_{i-3}) + A^{j-30}_{i-4} \\
A^{j+5}_{i+2} &= \mathbf{W}^{j+5}_{i+1} + \mathbf{A}^j_{i+1} + f^{j+5}_{i+2}(A^{j+5}_{i-2}, A^{j-25}_{i-3}, A^{j-25}_{i-4}) + A^{j-25}_{i-5} \\
A^j_{i+3} &= \mathbf{W}^j_{i+2} + A^{j-5}_{i+2} + \mathbf{f}^j_{i+3}(\mathbf{A}^j_{i+1}, A^{j-30}_i, A^{j-30}_{i-1}) + A^{j-30}_{i-2} \\
A^{j+30}_{i+4} &= \mathbf{W}^{j+30}_{i+3} + A^{j+25}_{i+3} + \mathbf{f}^{j+30}_{i+4}(A^{j+30}_{i+2}, \mathbf{A}^j_{i+1}, A^j_i) + A^j_{i-1} \\
A^{j+30}_{i+5} &= \mathbf{W}^{j+30}_{i+4} + A^{j+25}_{i+4} + \mathbf{f}^{j+30}_{i+5}(A^{j+30}_{i+3}, A^j_{i+2}, \mathbf{A}^j_{i+1}) + A^j_i \\
A^{j+30}_{i+6} &= \mathbf{W}^{j+30}_{i+5} + A^{j+25}_{i+5} + f^{j+30}_{i+6}(A^{j+30}_{i+4}, A^j_{i+3}, A^j_{i+2}) + \mathbf{A}^j_{i+1}
\end{aligned}$$

At the end of the pattern there is no difference in the state vector bits.
The expansion process of SHA preserves the structure of these patterns, thus if all the patterns of the expanded message succeed on creating local collisions, then a collision of the compression function is found.

Selecting a Disturbance Vector

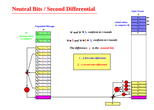
The **disturbance vector**, i.e., the pattern of disturbances and corrections, determines the attack complexity. Typically, the disturbance vector with the least Hamming weight has the highest probability.
In SHA-0 we select the disturbances in $j = 1$ to eliminate the probabilistic behavior of the carry in the corrections $W^{31}_{i+3}, W^{31}_{i+4}, W^{31}_{i+5}$. With this selection we try the 2^{16} possible selections of disturbances and select the **disturbance vector** with the least Hamming weight.
In SHA-1 the selection is more complicated due to the additional rotation. We observed that by selecting bits in adjacent locations, e.g., bits 0 and 1, some disturbances cancel each other and a low Hamming weight is achieved.
In addition, a single block collision requires no disturbance in the last five rounds since five corrections follow the last disturbance. We observed that there are disturbance vectors for more than 80 rounds with least Hamming weight that satisfy this condition, e.g., a disturbance vector for 82 rounds has a complexity of 2^{44} compared with 2^{57} for 80 rounds. We concluded that the strength of SHA is not monotonous with the number of rounds and that near-collisions are easier to find than collisions.

Complexity vs. Number of Rounds

# Rounds	Comp.	# Rounds	Comp.	# Rounds	Comp.
64	2^{29}	77	2^{66}	83	2^{65}
65	2^{29}	78	2^{56}	84	2^{64}
68	2^{43}	79	2^{56}	85	2^{71}
74	2^{50}	80	2^{56}	86	2^{95}
75	2^{52}	81	2^{43}	87	–
76	–	82	2^{43}	92	2^{74}

Neutral Bits Technique

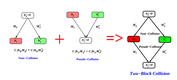
Generating local collisions by using the above patterns is a probabilistic process. In order to eliminate the probabilistic behavior of the first rounds, an attacker chooses and fixes some message bits such that the local collisions of the first rounds occur with probability 1.
We observed that given a pair of messages with these patterns of differences, there are many bits whose values do not affect the local collisions many rounds afterwards. Similarly, there are pairs, triplets, quadruplets and quintuplets of bits whose simultaneous complementation does not affect the local collision. We call these bits **Neutral bits**.
Among the neutral bits there are large sets of bits whose neutrality is mutually independent. Such sets of n mutually independent neutral bits are used to construct 2^n pairs of messages from any given pair by complementing all possible subsets. In a large fraction of the 2^n pairs the local collisions are not affected. This technique improves the complexity of a collision search by a factor of 2^6 and more. It also shows that the avalanche effect of SHA-0 and SHA-1 is poor.



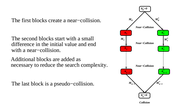
Multi-Block Technique

In our work on SHA-0 and SHA-1 we show that finding near-collisions and pseudo-collisions may be much easier than finding a collision. This observation led us to the development of the **Multi-Block** technique.
A **Near-Collision** is a result of a compression function applied to two different messages with the same IV's, where the results differ by a small number of bits.
Pseudo-Collision is a collision of the compression function applied to different messages with different IV's.
Near-Pseudo collision is the results of a compression function applied to two different messages, with IV's that differ by a small number of bits, and where the results also differ by a small number of bits.

Near-Collision, Pseudo-Collision, Two-Block Collision



Multi-Block Collision



Follow-Ups to Our Research

The first follow-ups to our research was the four-block collision of SHA-0 by Joux with a complexity of 2^{51}. His work is based on the basic differential technique he published in 1998, improved with our neutral bits and multi-block techniques.
Wang's breaking of SHA-1 with a complexity of 2^{63} uses some of our techniques.
Other researches used our techniques on other hash functions as well.

Further Research

- Develop new techniques to generalize the neutral bit and message modification techniques, to attack hash function more efficiently.
- Implement the new technique on SHA-1 in order to find a collision, and to better understand the weaknesses of the SHA and MD family of hash functions.
- Extend the cryptanalytic techniques developed for hash functions to block ciphers.