8 Detecting collision attacks

Contents

8.1 Extra line of defense

It has been known since 2004 and 2005 that MD5 and SHA-1 have weaknesses. In Chapters 6 and 7 we have improved attacks against MD5 and SHA-1 and the future will tell how far these attacks can be further improved. So it is clear that MD5 and SHA-1 should be replaced by more secure hash functions such as SHA-2 or the upcoming SHA-3 standard for all security purposes that depend on the collision resistance of MD5 and SHA-1. Completely banishing all such occurrences of MD5 and SHA-1 in software and hardware might be desirable in light of this, but might very well be impossible due to the high costs and effort required. Even replacing MD5 and SHA-1 in the security-wise most important software and hardware may be difficult due to for instance compatibility issues. Therefore, the most likely scenario is the one that we currently see with respect to MD5, namely that more secure hash functions such as SHA-2 are used where possible however that MD5 and SHA-1 remain supported.

Except when MD5 and SHA-1 have been explicitly disallowed, this scenario still leaves users vulnerable to collision attacks. The technique presented in this section allows to add an extra line of defense in security applications by detecting collision attacks. Obviously when given two messages that result in the same hash value, it is almost certain that a collision attack is involved. The first MD5 collision attack [WY05] could easily be detected by applying its specific message differences to a message and checking whether a collision has occurred. Since other message differences that lead to fast near-collision attacks were found and our differential path construction algorithm was published, there is an infinite number of ways to construct a collision attack and thereby making it impossible to check for all possible collision attacks in this manner. Moreover, our chosen-prefix collision attack is completely undetectable in this manner since the message differences that alter the first chosen-prefix into the second chosen-prefix cannot be guessed in general.

Our technique allows to efficiently detect collisions attacks for MD5 and SHA-1, both identical-prefix collision attacks and chosen-prefix collision attacks, given only one of the two colliding messages. It has been tested successfully in a simple proof of concept plug-in for Microsoft's Internet Explorer. For this test this plug-in detects only chosen-prefix collisions in the certificate-chain of secure websites when connecting to a secure web server before any sensitive information has been sent. It has been

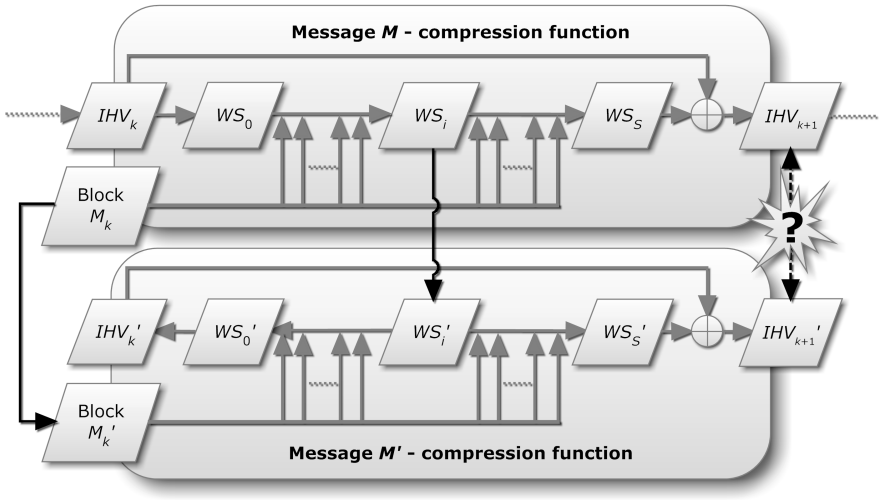

Figure 9: *Principle of detecting near-collisions*

successfully used to detect man-in-the-middle attacks using our rogue Certification Authority described in Section 4.2.

8.2 Core principle

The core principle of our technique of detecting collision attacks is to detect the last near-collision block of a collision attack and uses two key observations:

- There are only a small number of possible message block differences that may lead to feasible near-collision attacks.
- All published MD5 and SHA-1 collision attacks use a differential path that at some step has no differences at all in the working state, or in the case of MD5 it also can use differences $(2^{31}, 2^{31}, 2^{31}, 2^{31})$ (see [dBB93]).[43]

Due to these observations it is possible to check for collision attacks given only one message of a colliding pair of messages.

For two colliding messages M and M', let (M_k, M'_k) be the last near-collision block pair of a collision attack. Let IHV_{k+1} and IHV'_{k+1} be the intermediate hash values just after applying the compression function to M_k and M'_k in the hash value computation of M and M', respectively. It follows that

$$IHV_{k+1} = IHV'_{k+1}.$$

43. The reason for this is simple: these working state differences can be maintained at every step of the 64 steps of MD5Compress with probability at least 1/2 if not 1.

Suppose we are only given the message M, the message block differences δM_k and a step i after which the working state difference in the near-collision attack should be either zero or, in the case of MD5, $(2^{31}, 2^{31}, 2^{31}, 2^{31})$. Then as illustrated in Figure 9, we can easily compute IHV_{k+1} and IHV'_{k+1} and test for the telltale condition $IHV_{k+1} = IHV'_{k+1}$, even though both the second message M' and the message block M'_k were not given.

The hash value computation of M gives us values for IHV_k, IHV_{k+1} and WS_i which is the working state before step i of the compression function applied to IHV_k and M_k. Since we know the message block differences and the working state differences, we can determine the message block M'_k and the working state WS'_i associated with the message M' that collides with M. Computing steps $i+1, \ldots, S$ of the compression function using M'_k and WS'_i, we obtain working states $WS'_{i+1}, \ldots, WS'_S$. As the step function of MD5 and SHA-1 is reversible (see Section 5.4.1) we can also compute working states $WS'_{i-1}, \ldots, WS'_0$. The value of IHV'_k can be derived from WS'_0 and the value of IHV'_{k+1} can be computed from IHV'_k and WS'_S. Finally, if $IHV'_{k+1} = IHV_{k+1}$ then we have confirmed that (M_k, M'_k) is a near-collision block pair.

This principle leads to our collision detection algorithm presented in Algorithm 8-1 that works for any Merkle-Damgård hash function that uses a compression function in $\mathcal{F}_{\text{md4cf}}$ (see Section 5.3, p. 65). Let C be the number of possible combinations of values for message block differences δB, step i and working state differences δWS_i belonging to a feasible near-collision attack. Then the runtime-complexity of Algorithm 8-1 for a message M is approximately $C+1$ times the runtime-complexity of computing the hash value of M.

The probability of a false positive is at least $C \cdot 2^{-L}$ where L is the bit length of the hash value, since $IHV'_{k+1} = IHV_{k+1}$ holds with probability 2^{-L} for randomly selected IHV'_{k+1} and IHV_{k+1}. The false positive probability may be higher when there exists a differential path compatible with one of the combinations of δB, i and δWS_i that holds with probability less than 2^{-L}.

8.3 Application to MD5

Algorithm 8-1 can be directly applied to MD5. What remains is to determine possible combinations of values for message block differences δB, step i and working state differences δWS_i that belong to a feasible near-collision attack. The message block differences are additive in $\mathbb{Z}_{2^{32}}$ and for each message block M_k the message block M'_k can be either $M_k + \delta B$ or $M_k - \delta B$. There are two different working state differences δWS_i that can be used for MD5, namely $(0,0,0,0)$ and $(2^{31}, 2^{31}, 2^{31}, 2^{31})$, written more compactly as $\underline{0}$ and $\underline{2^{31}}$.

Used non-zero message block differences in published near-collision attacks are (together with selected values for i and δWS_i):

- $\delta B = \pm(\delta m_{11} = 2^{15}, \delta m_4 = \delta m_{14} = 2^{31})$ [WY05]: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_2 = 2^8, \delta m_{11} = 2^{15}, \delta m_4 = \delta m_{14} = 2^{31})$ [SSA+09b]: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;

Algorithm 8-1 Last near-collision block detection

This algorithm returns `True` if a near-collision attack is detected and `False` otherwise. For a given message M, let $M_0, \ldots, M_{N-1}$ be the N message blocks that result from the padding and the splitting of M by the hash function. For $k \in \{0, \ldots, N-1\}$ do the following:

1. Let IHV_k be the intermediate hash value before the message block M_k is processed.
2. Initialize the working state WS_0 with IHV_k, compute all S working states $WS_1, \ldots, WS_S$ and determine IHV_{k+1} using IHV_k and WS_S.
3. For each possible combination of values for message block differences δB, step i and working state differences δWS_i belonging to a feasible near-collision attack do the following:
   (a) Apply the message block differences δB to M_k to obtain M'_k.
   (b) Apply the working state differences δWS_i to WS_i to obtain WS'_i.
   (c) Compute the working states $WS'_{i-1}, \ldots, WS'_0$ backwards.
   (d) Compute the working states $WS'_{i+1}, \ldots, WS'_S$ forward.
   (e) Determine IHV'_k from WS'_0 and IHV'_{k+1} from IHV'_k and WS'_S.
   (f) If $IHV'_{k+1} = IHV_{k+1}$ then (M_k, M'_k) is a near-collision block pair: return `True`
4. Return `False`

- $\delta B = \pm(\delta m_{11} = 2^b)$ for $b \in \{0, \ldots, 30\}$ [SLdW07c]: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = (\delta m_{11} = 2^{31})$ [SLdW07c]: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_5 = 2^{10}, \delta m_{10} = 2^{31})$ [XF10]: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = (\delta m_8 = 2^{31})$ [XLF08]: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_6 = 2^8, \delta m_9 = \delta m_{15} = 2^{31})$ [XFL08]: $i = 37$, $\delta WS_{37} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_9 = 2^{27}, \delta m_2 = \delta m_{12} = 2^{31})$ [VJBT08]: $i = 37$, $\delta WS_{37} \in \{\underline{0}, \underline{2^{31}}\}$.

Other possible non-zero message block differences taken from [XF09] and [XLF08] are:

- $\delta B = \pm(\delta m_4 = 2^{20}, \delta m_7 = \delta m_{13} = 2^{31})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_2 = 2^8)$: $i = 37$, $\delta WS_{37} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_5 = 2^{10}, \delta m_{11} = 2^{21})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_5 = 2^{10}, \delta m_{11} = 2^{31})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = (\delta m_5 = 2^{31}, \delta m_8 = 2^{31})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;

- $\delta B = \pm(\delta m_2 = 2^8, \delta m_{14} = 2^{31})$: $i = 37$, $\delta WS_{37} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = (\delta m_4 = 2^{31})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = (\delta m_5 = 2^{31})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = (\delta m_{14} = 2^{31})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_4 = 2^{25})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_5 = 2^{10})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_8 = 2^{25})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_{11} = 2^{21})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_{14} = 2^{16})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_4 = 2^{20})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_6 = 2^8)$: $i = 50$, $\delta WS_{50} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_9 = 2^{27})$: $i = 50$, $\delta WS_{50} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_5 = 2^{10}, \delta m_9 = 2^{27})$: $i = 37$, $\delta WS_{37} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = (\delta m_5 = 2^{31}, \delta m_{11} = 2^{31})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_8 = 2^{31}, \delta m_{11} = 2^{21})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$;
- $\delta B = \pm(\delta m_8 = 2^{25}, \delta m_{13} = 2^{31})$: $i = 44$, $\delta WS_{44} \in \{\underline{0}, \underline{2^{31}}\}$.

The number of all combinations given in the two lists above is $(36 \cdot 4 + 2 \cdot 2) + (16 \cdot 4 + 5 \cdot 2) = 222$. We do not guarantee that the lists of combinations given above form the exhaustive list of all combinations that lead to feasible near-collision attacks. Nevertheless, other combinations arising from future collision attacks can easily be added to the above lists.

All published near-collision attacks require complex differential steps in the first round, thereby requiring a high number of bitconditions, say at least 200. E.g., the differential paths by Wang et al. require roughly 300 bitconditions (see Table 2-4 and Table 2-6 on pages 28 and 30). This implies that the probability of a false positive is dominated by the general $C \cdot 2^{-L}$ term explained earlier. Hence, the probability of a false positive is estimated as $222 \cdot 2^{-128}$ and thus negligible.

Due to the *pseudo-collision* attack against MD5's compression function by den Boer and Bosselaers [dBB93], there is also a special near-collision attack with zero message block differences and with $\delta WS_i = \underline{2^{31}}$ for all $i \in \{0, \ldots, 64\}$. One can test for this pseudo-collision attack using $\delta B = 0$, $i = 32$ and $\delta WS_{32} = \underline{2^{31}}$. The probability of a false positive is 2^{-48} which is not negligible. However, since it requires $\delta WS_0 = \underline{2^{31}}$ and thus $IHV_{\text{in}} = \underline{2^{31}}$, this pseudo-collision attack requires at least one other preceding near-collision block to form a collision attack against MD5.

This observation calls for the following modification of Algorithm 8-1 for MD5 to reduce the chance of a false positive to $222 \cdot 2^{-128} \cdot 2^{-48}$ for $\delta B = 0$. Whenever a near-collision block is detected in step 3.(f) for the combination $\delta B = 0$, $i = 32$ and $\delta WS_{32} = \underline{2^{31}}$ and before returning `True`, perform steps 1–4 of Algorithm 8-1 on the previous message block M_{k-1} using all combinations that have $\delta B \neq 0$ and using

the condition $IHV'_k = IHV_k + \underline{2^{31}}$ instead of the condition $IHV'_k = IHV_k$. If this sub-instance of Algorithm 8-1 returns `False` then the main instance continues with the next combination of δB, i and δWS_i. Otherwise, the main instance returns `True`.

Given a message M, the average complexity to detect whether M is constructed by a collision attack against MD5 using one of the given message differences is about $222 + 1 + 1 = 224$ times the complexity of computing the MD5 hash of M.

This technique has been tested successfully in a simple proof of concept plug-in for Microsoft's Internet Explorer. For this test this plug-in detects only chosen-prefix collisions in the certificate-chain of secure websites when connecting to a secure web server before any sensitive information has been sent. It has been successfully used to detect man-in-the-middle attacks using our rogue Certification Authority described in Section 4.2.

8.4 Application to SHA-1

Algorithm 8-1 can be directly applied to SHA-1. Note that this is possible even though no practical collision attacks against SHA-1 or actual colliding messages are known yet. What remains is to determine possible combinations of values for message block differences δB, step i and working state differences δWS_i that belong to a feasible near-collision attack. For SHA-1 the message block differences δB are given as $M_k \oplus M'_k$ which is the bitwise XOR of the message block pair (M_k, M'_k) and can be derived from a disturbance vector as $(DW_t)_{t=0}^{15}$ (see Section 7.3.2). For disturbance vector $\mathrm{I}(j, b)$ or $\mathrm{II}(j, b)$ there are no differences at step $j + 8$, hence to test for near-collision block pair using this disturbance vector we use Algorithm 8-1 with the combination $(DW_t)_{t=0}^{15}$, $i = j + 8$ and $\delta WS_i = (0, 0, 0, 0, 0)$.

We arbitrarily choose all disturbance vectors $(DV_t)_{t=0}^{79}$ in Appendix F where for some $u \in \{0, \ldots, 7\}$ and $\epsilon \leq 0.5$ it holds that $\mathrm{FDC}_{u,20,\epsilon}((DV_t)_{t=0}^{79}) \geq 2^{-74}$:

I(46,0),	I(48,0),	I(49,0),	I(50,0),	I(51,0),	I(48,2),	I(49,2),
II(46,0),	II(50,0),	II(51,0),	II(52,0),	II(53,0),	II(54,0),	II(56,0).

Similar to the case of MD5, it is always possible to add extra disturbance vectors to the above list in the future whenever it is believed it can lead to a feasible collision attack. Ignoring the first round, each disturbance vector has a probability in the order of 2^{-70} that a false positive occurs. Taking into account the complex differential steps necessary in the first round, we can safely assume that the probability of a false positive is negligible.

Given a message M, the average complexity to detect whether M is constructed by one of the above possibly feasible collision attacks against SHA-1 is about 15 times the complexity of computing the SHA-1 hash of M.