CSE 599b: Cryptography **(Winter 2006)**

Lecture 14: Cryptographic Hash Functions

17 February 2006

Lecturer: Paul Beame *Scribe: Paul Beame*

1 Hash Function Properties

A *hash function family* $H = \{H_K\}_{K \in \mathcal{K}}$ is a function $H : \mathcal{K} \times D \to R$ where $|D| < |R|$. We generally view $\mathcal{K}$ as a probability distribution on the set of possible keys but here we also use $\mathcal{K}$ to denote the set of possible keys.

Universal Hash Function Families

Definition 1.1 (Carter-Wegman)**.** *$H : \mathcal{K} \times D \to R$ is a* universal hash function family *if and only if for all $x_1 \neq x_2 \in D$,*

$$\Pr[H_K(x_1) = H_K(x_2) \mid K \leftarrow \mathcal{K}] = 1/|R|,$$

where $\mathcal{K}$ is a probability distribution.

This is equivalent to saying that, for all $a_1, a_2 \in R$ and all $x_1 \neq x_2 \in D$,

$$\Pr_{K \leftarrow \mathcal{K}}[H_K(x_2) = a_2 \mid \ H_K(x_2) = a_2] = 1/|R|,$$

and thus being a universal hash function family is equivalent to having a probability distribution on functions from D to R that maps elements of D in a uniform pairwise independent fashion.

Typically we will consider $D = \{0,1\}^n$ and $R = \{0,1\}^m$ for $m < n$. The following construction due to Dietzfelbinger is particularly convenient: The space of keys is all strings $K = (a, b)$ where $a, b \in \{0,1\}^{m+n}$ and $H_K(x)$ consists of the middle m bits of $ax + b$. ($ax + b$ will naturally have $2n + m$ bits.)

In keeping with our choice of considering PPT adversaries for our formal definitions we will use infinite hash function families and allow probability slop that is negligible. We will also want our hash function families to be efficiently computable. Before we consider the cryptographic versions we state a relaxation of the universal hash function family definition.

Definition 1.2. *An* almost universal hash function family $\{H^k : \{0,1\}^k \times D_k \to R_k\}_{k \geq 1}$ *for $|R_k| < |D_k|$ is a collection of functions satisfying*

- *there is a PPT algorithm $\mathcal{K}$ that on input 1^k produces a random key K from $\{0,1\}^k$.*
- *there is a deterministic polynomial time algorithm H that for any k on input a key $K \in \{0,1\}^k$ and $x \in D_k$ outputs $H_K(x) = H^k_K(x)$.*

- *The function*
$$\epsilon(k) = \max_{x_1 \neq x_2 \in D_k} \Pr[H_K(x_1) = H_K(x_2) \mid K \leftarrow \mathcal{K}(1^k)]$$
is negligible.

Universal One-Way Hash Function Families (UOWHFFs) We now consider cryptographic requirements. The first of these to be defined historically was collision resistance whose properties were formalized by Damgard. Collision resistance (which we define later) requires that given the key K it should be hard to find a pair of distinct inputs that map to the same place. Subsequently Naor and Yung realized that a simpler property suffices in many applications. This property which they called universal one-way hash function families is also know as target collision resistance. It says that given the key K and a challenge point x_1, it is hard to find a second point x_2 that maps to the same place as x_1.

Definition 1.3. *A* universal one-way hash function family (UOWHFF) $\{H^k : \{0,1\}^k \times D_k \rightarrow R_k\}_{k \geq 1}$ *for* $|R_k| < |D_k|$ *is a collection of functions satisfying*

- *there is a PPT algorithm* $\mathcal{K}$ *that on input* 1^k *produces a random key* K *from* $\{0,1\}^k$.
- *there is a deterministic polynomial time algorithm* H *that for any* k *on input a key* $K \in \{0,1\}^k$ *and* $x \in D_k$ *outputs* $H_K(x) = H_K^k(x)$.
- *For any PPT algorithm* A*, the function*
$$\epsilon(k) = \Pr[H_K(x_1) = H_K(x_2) \text{ and } x_1 \neq x_2 \mid K \leftarrow \mathcal{K}(1^k);\ x_1 \leftarrow \mathcal{D}_k;\ x_2 \leftarrow A(K, x_1)]$$
is negligible where $\mathcal{D}_k$ *is the uniform distribution over* D_k

In this definition we think of $H_K(x_1)$ as being the target.

Collision Resistant Hash Function Families

Definition 1.4. *A* collision-resistant hash function family (CRHFF) $\{H^k : \{0,1\}^k \times D_k \rightarrow R_k\}_{k \geq 1}$ *for* $|R_k| < |D_k|$ *is a collection of functions satisfying*

- *there is a PPT algorithm* $\mathcal{K}$ *that on input* 1^k *produces a random key* K *from* $\{0,1\}^k$.
- *there is a deterministic polynomial time algorithm* H *that for any* k *on input a key* $K \in \{0,1\}^k$ *and* $x \in D_k$ *outputs* $H_K(x) = H_K^k(x)$.
- *For any PPT algorithm* A*, the function*
$$\epsilon(k) = \Pr[H_K(x_1) = H_K(x_2) \text{ and } x_1 \neq x_2 \mid K \leftarrow \mathcal{K}(1^k);\ (x_1, x_2) \leftarrow A(K)]$$
is negligible.

Collision-resistant hash function families are sometimes referred to as *collision-intractable* or even *collision-free* hash function families.

To understand the relationship between the definitions observe that the difference between the cryptographic notions and the non-cryptographic ones is that the non-cryptographic definitions require that the points x_1 and x_2 are fixed before the key is chosen whereas the cryptographic notions allow these points to be chosen by an adversary depending upon the key.

Birthday Attack To see the difference between the CRHFF and UOWHFF definitions consider the so-called birthday attack discussed earlier in which the adversary A simply applies H_K randomly to q random elements of D_k. Suppose that $|D_k| >> |R_k|$ which is the typical case.

For any two elements of D_k, the probability that H_K maps them to the same element of R_k is at least $1/|R_k|$ (it is exactly $1/|R_k|$ if each point in R_k has an equal number of pre-images under H_K). There are $\binom{q}{2}$ pairs of elements so the probability that at least one pair collides in roughly at least $q^2/|R_k|$. In particular, this means that after $q = \sqrt{|R_K|}$ random queries there is a constant probability of finding a collision. (Since $|R_k| << |D_k|$ the queries are much less likely to be the same than the places where they are mapped.)

On the other hand, for a uniformly distributed H_k, the chance that one of q queries will map to a target $H_K(x_1)$ is at most $q/|R_k|$.

Therefore, if a hash function family maps inputs to a space $R_k = \{0,1\}^m$ then it is insecure under obvious attacks for collision-resistance that run in time $2^{m/2}$ and for target collision-resistance that run in time 2^{m-1}.

2 The MD paradigm, MD4, MD5, SHA-1, SHA256, etc.

For practical examples we deal with non-asymptotic versions of the definitions of hash function families (using only one value of k). Moreover, the functions will work with only one fixed value of the key K.

In 1988 Merkle and Damgard devised a method for building collision resistant hash functions that work for (essentially) arbitrarily long strings from collision resistant hash functions that operate on fixed length strings.

Based on the Merkle and Damgard paradigm, Rivest in 1990 defined a candidate hash function family MD4 and a related larger scheme MD5 in 1992. In 1995 insecurities were found the NSA modified MD5 to add an error-correcting code and produced the SHA1, the "secure hash algorithm". Subsequently, related definitions have been developed using similar ideas that have longer key size.

$$
\begin{aligned}
SHA1 &: \{0,1\}^{<2^{64}} \to \{0,1\}^{160} \\
SHA256 &: \{0,1\}^{<2^{64}} \to \{0,1\}^{256} \\
SHA384 &: \{0,1\}^{<2^{64}} \to \{0,1\}^{384} \\
SHA512 &: \{0,1\}^{<2^{64}} \to \{0,1\}^{512}
\end{aligned}
$$

The hope for each of these functions is collision-resistance and therefore security roughly 2^{80} for SHA1, 2^{128} for SHA256, etc.

We describe the SHA1 scheme in some detail to give an idea about the MD construction and the ideas involved in these schemes. The basic structure of SHA1 involves breaking the message being hashed up into blocks of 512 bits and iterating a fixed-length function $shf1$ on these bits.

$\texttt{SHA1}_K(M)$:
  $y \leftarrow pad(M) = M10^d\ell$ where ℓ is the 64-bit representation of $|M|$
    and d is chosen so that $|y|$ is a multiple of 512.
  Write $y = M_1M_2 \ldots M_n$ where $|M_i| = 512$.
  $V_0 \leftarrow IV \in \{0,1\}^{160}$
  For $i = 1$ to n
    $V_i \leftarrow shf1_K(M_i, V_{i-1})$
  EndFor
  Output V_n

In SHA1 the key $K = K_1K_2K_3K_4$ is a fixed 128 bits consisting of 4 sub-keys of 32 bits each. K_1 is the binary approximation of $\sqrt{2}$; K_2 is the binary approximation of $\sqrt{3}$; K_3 is the binary approximation of $\sqrt{5}$ and K_4 is the binary approximation of $\sqrt{10}$. Also, the initialization vector IV is an ad-hoc choice of string. (We'll see that its exact value is not particularly important.)

In the function $shf1$, the 512-bit message M_i is first encoded using a simple binary code that expands it to $2560 = 80x32$ bits. That is, M_i is broken into 16 blocks of 32 bits, $W_0, \ldots, W_{15}$, and then $W_{16}, \ldots, W_{79}$ are defined where W_j is the $\oplus$ of 4 previous blocks.

Now the vector V_{i-1} of 160 bits is broken into 5 blocks of 32 bits each, A, B, C, D, E. This operates in a slightly Feistel-style fashion for 80 rounds. Each iteration sends $E' \leftarrow D$, $D' \leftarrow C$, $C' \leftarrow B$, $B' \leftarrow Rot(A)$ were $Rot(A)$ is some rotation of the bits of A. The main action in the j-th iteration is in creating A' which is defined by

$$A' = W_j + K_{\lceil j/20 \rceil} + f_{\lceil j/20 \rceil}(B, C, D) + Rot'(A) + E$$

where addition is modulo 2^{32}, Rot' is a different rotation, and f_1, f_2, f_3, f_4 are each very simple functions of B, C, and D, one is simply their $\oplus$, others involve simple bit-wise $\wedge$s and $\vee$s.

Finally, after the 80-th version of A, B, C, D, E have been defined, each of them is added back into its corresponding block of V_{i-1} (again taken modulo 2^{32} in each block) and that value is returned.

2.1 Attacks on SHA1

As mentioned above the birthday attack on SHA1 should be expected to succeed after 2^{80} queries. In early 2005, Wang, Yin, and Yu used differential cryptanalysis to derive a collision finding algorithm that succeeds in only 2^{69} queries. (Xing Wang had previously shown that not only was MD5 not collision-resistant, it was not even target collision-resistant.) In August 2005, follow-on work by Wang with Frances Yao and Andy Yao reduced the number of queries to 2^{63}. Since, a couple of years ago, there was a successful distributed attack on a different primitive that involved

2^{64} queries, this clearly should not be considered secure against collision-finding attacks from a practical point of view. (Target collision-resistance is still open but the success in breaking MD5 suggests that this may also be poor.)

SHA1 was scheduled to be replaced as a standard in 2009 but since SHA256, SHA384, and SHA512 use similar design principles it is unclear how much one should rely on any of them for collision-resistance. We shall see, however, the insecurity of SHA1 is a property of $shf1$ rather than the Merkle Damgard paradigm.

2.2 The MD Paradigm

Given a hash function family $h_{KK\in\{0,1\}^k}$ that each maps $(b+m)$-bit strings to m-bit strings the MD method shows how to build a hash function family $H_{KK\in\{0,1\}^k}$ that maps strings of up to $2^{m/2}$ bits to m-bit strings as follows:

$H_K(M)$
  $y \leftarrow pad(M)$ where $pad(M)$ is uniquely decodable as M, is a multiple of b-bits long and contains $|M|$ in the last block.
  Write $y = M_1 M_2 \ldots M_n$ where $|M_i| = b$.
  $V_0 \leftarrow IV \in \{0,1\}^m$
  For $i = 1$ to n
    $V_i \leftarrow h_K(M_i, V_{i-1})$
  EndFor
  Output V_n

Theorem 2.1 (Merkle, Damgard)**.** *The collision resistance of* $\{H_K\}_{K\in\{0,1\}^k}$ *is the same as that of* $\{h_K\}_{K\in\{0,1\}^k}$.

Proof. Let A_H be an adversary and let

$$\epsilon = \Pr[H_K(x_1) = H_K(x_2) \text{ and } x_1 \neq x_2 \mid K \leftarrow \mathcal{U}_k;\ (x_1, x_2) \leftarrow A_H(K)].$$

We now define an adversary A_h that achieves the same probability for h:
On input K,

1. Run A to produce a pair (x_1, x_2).
2. Run the MD code using h_K to produce $H_K(x_1)$ and $H_K(x_2)$ and keep track of the blocks M_i^1 and M_i^2 for x_1 and x_2 respectively as well as the corresponding V_i^1 and V_i^2 during the computation and let n_1 and n_2 be the number of blocks in their encoding.
3. If $H_K(x_1) \neq H_K(x_2)$ or if $x_1 = x_2$ then FAIL and HALT.
4. If $|x_1| \neq |x_2|$ then return the pair (x_1', x_2') where $x_1' = (M_{n_1}^1, V_{n_1-1}^1)$ and $x_2' = (M_{n_2}^2, V_{n_2-1}^2))$.
5. If $|x_1| = |x_2|$ then let $i \leftarrow n_1$.

(a) While $(M_i^1, V_{i-1}^1) = (M_i^2, V_{i-1}^2)$ Do $i \leftarrow i - 1$;

(b) Return (x'_1, x'_2) where $x'_1 = (M_i^1, V_{i-1}^1)$ and $x'_2 = (M_i^2, V_{i-1}^2)$.

Note that by construction of the padding, if $|x_1| \neq |x_2|$ then the last block of the padded version of x_1 will be different from that of x_2. Thus in this case x'_1 will be different from x'_2, and, moreover, $h_K(x'_1) = h_K(M_{n_1}^1, V_{n_1-1}^1) = H_K(x_1)$ $h_K(x'_2) = h_K(M_{n_2}^2, V_{n_2-1}^2) = H_K(x_2)$ so $h_K(x'_1) = h_K(x'_2)$.

On the other hand if $|x_1| = |x_2$ then the algorithm follows the two (M, V) pairs back from the end of the padded string until the first time that $(M_i^1, V_{i-1}^1) \neq (M_i^2, V_{i-1}^2)$. However, in this case $h_K(M_i^1, V_{i-1}^1) = V_i^1 = V_i^2 = h_K(M_i^2, V_{i-1}^2)$ and thus the returned values of x'_1 and x'_2 form a collision for h_K.

It follows that in either case A_h succeeds whenever A_H succeeds. □

It remains to derive fixed-length collision-resistant hash functions. There is a general (slow) construction of such schemes based on collections of pairs of claw-free functions (or directly using specific number-theoretic assumptions) but we won't have time to discuss them. Instead we will focus more on UOWHFFs which are easier to construct.

3 Universal One-Way Hash Function Families (UOWHFFs)

Why might it be difficult to find a second pre-image for a target that is a hash function applied to a randomly chosen x_1? There are two natural reasons:

- The hash function is 1-1 on the point x_1.
- The hash function is hard to invert on $H_K(x_1)$; i.e. H_K is one-way.

Let $p_1 = \Pr[|H_K^{-1}(H_K(x))| = 1 \mid K \leftarrow \mathcal{U}_k; x \leftarrow \mathcal{D}_k]$. Then since each element of $y \in R_k$ is associated with most one $x \in D_k$ such that $|H_K^{-1}(H_K(x))| = 1$, namely $H_K{-}1(y) = \{x\}$, it follows that $p_1 \leq |R_k|/|D_k|$. For most hashes we will consider $|R_k|/|D_k|$ is negligible in k so p_1 is negligible in k.

Let A be a PPT and consider the probability that the one-wayness of H_K fails,

$$\epsilon(k) = \Pr[A(y) \in H_K^{-1}(y) \mid K \leftarrow \mathcal{U}_k;\ x \leftarrow \mathcal{D}_k;\ y = H_K(x)].$$

If $A(y) \in H_K^{-1}(y)$ then since $\mathcal{D}_k$ is uniform over D_k, the chance that $A(y)$ will return the element x used to construct y is precisely $1/|H_K^{-1}(y)|$, which is at most $1/2$ for $|H_K^{-1}(y)| > 1$. Therefore if $A'(K, x_1)$ runs A on K and $y = h_K(x_1)$ then A' will produce a second pre-image with probability at least $(\epsilon(k) - p_1)/2$ which is non-negligible if and only if $\epsilon(k)$ is non-negligible and the domain/range ratio is large. Thus UOWHFFs must be collections of one-way functions.

The existence of UOWHFFs is actually equivalent to the existence of one-way functions. We give the original construction due to Naor and Yung [1990] based on one-way permutations. To do this we will also need universal hash function families from $\{0,1\}^n \rightarrow \{0,1\}^{n-1}$ for each n that have one additional property.

Definition 3.1. *An infinite family of hash functions $\{G_K\}_{K\in\{0,1\}^k}$ for $k \geq 1$ is* collision accessible *if and only if there is a PPT algorithm $\mathcal{C}$ such that for each k and $x_1, x_2 \in D_k$, $\mathcal{C}(x_1, x_2)$ produces a uniformly chosen $K \in \{0,1\}^k$ conditioned on $G_K(x_1) = G_K(x_2)$.*

Most of the constructions of universal hash function families are easily seen to be collision accessible. The Naor-Yung construction requires the existence of one-way permutations and a collision accessible universal family of hash functions that compress their input by precisely one 1 bit.

Suppose that $\{G_n\}_{n\geq 1}$ is a collision accessible universal hash function family where $G_n : \{0,1\}^{k_n} \times \{0,1\}^n \to \{0,1\}^{n-1}$. Let f be a one-way permutation. Define a hash function family $\{H_n\}_{n\geq 1}$ by $H_n(K, x) = G_n(K, f(x))$. That is for $x \in \{0,1\}^n$, $H_K(x) = G_K(f(x))$.

Theorem 3.2. *If f is a one-way permutation and $\{G_n\}_{n\geq 1}$ is a family of collision-accessible universal hash functions that compress their inputs by 1 bit then $\{H_n\}$ as defined above is a UOWHFF.*

Proof. Suppose that A is an algorithm that breaks the UOWHFF property with probability $\epsilon(n)$; i.e.

$$\epsilon(n) = \Pr[H_K(x_1) = H_K(x_2) \text{ and } x_1 \neq x_2 \mid K \leftarrow \mathcal{K}(1^{k_n});\ x_1 \leftarrow \mathcal{U}_n;\ x_2 \leftarrow A(K, x_1)].$$

We use algorithm A to create an algorithm A' that inverts f as follows:
On input $y \in \{0,1\}^n$,

1. Choose x_1 from $\mathcal{U}_n$
2. Evaluate f on x_1; if $f(x_1) = y$ then return x_1.
3. Apply the collision accessibility algorithm $\mathcal{C}$ for G to produce a random $K \in \{0,1\}^{k_n}$ such that $G_K(y) = G_K(f(x_1))$.
4. Run A on input K and x_1 to produce x_2.
5. If $H_K(x_2) \neq H_K(x_1)$ or $x_2 = x_1$ then FAIL
6. Otherwise return x_2.

Observe that if A succeeds then $G_K(y) = G_K(f(x_1)) = H_K(x_1) = H_K(x_2) = G_K(f(x_2))$. The key idea for the correctness of the construction is that since each G is a universal hash function family that compresses by precisely one bit, each G_K is precisely a 2-1 map. Furthermore, since f is a permutation on $\{0,1\}^n$ if $x_1 \neq x_2$ then $f(x_1) \neq f(x_2)$ are the only two pre-images of $G_K(y)$ so one of $f(x_1)$ and $f(x_2)$ must be y. Thus if the call to A succeeds the algorithm will succeed in inverting f on y.

However, we have to argue that this use of A leads to the same probability of success for breaking f as in the definition of $\epsilon(n)$. For this we observe that since x_1 is chosen from $\mathcal{U}_n$, and f is permutation then $f(x_1)$ is a random element of $\{0,1\}^n$. In the definition of the one-way property for f, the element y is the image of a random element of $\{0,1\}^n$ and is therefore also random. Thus the call to $\mathcal{C}$ is for two random elements of $\{0,1\}^n$. By the definition of universal hash function

families, in the experiment in which we choose a random pair of inputs y, y' and then choose a random K such that $G_K(y) = G_K(y')$, each K will be equally likely to be chosen. Thus the success probability of A' is precisely that of A. □

This construction produced an infinite family $\{H_n\}_{n\geq 1}$ that is a UOWHFF but compresses only by 1 bit at a time. It is easy to observe that if we want more compression then we can obtain this by concatenating keys and composing the functions. That is, we can define a family of UOWHFFs mapping $\{0,1\}^n \to \{0,1\}^{m(n)}$ for any $m(n) < n$, by defining

$$H_{K_{m(n)+1}\cdots K_{n-1}K_n} = H_{K_{m(n)+1}} \circ \cdots \circ H_{K_{n-1}} \circ H_{K_n},$$

where each $K_j \in \{0,1\}^{k_j}$. It is easy to check that the definition of UOWHFF is preserved.

Note that this construction requires one one-way permutation evaluation for every bit of compression. Kim, Simon, and Tetali have shown that any black-box construction needs to have a number of evaluations that is polynomial in the amount of compression (roughly at least the square root).

Furthermore, in a natural oracle model, Simon has shown that the existence of collision-resistant hash functions does not follow from that of UOWHFFs by given an oracle relative to which the later exist but the former do not.