# A Mini-Course on the Theory of Cryptography

Charles Rackoff

Department of Computer Science, University of Toronto

Bonn, June 17-21, 2006

(Part 4)

## Review

We showed how two parties who share a random session key can use a pseudo-random function generator to talk securely over a totally insecure channel.

Next time we will discuss how to share a session key.

Today we will discuss some new primitives.

# One-Way Functions

A *one-way function* is a function that is easy to compute, but hard – on the average – to invert.

Let $f : \{0,1\}^* \to \{0,1\}^*$ be a polynomial-time computable function. We assume that the length of $f(x)$ is determined by the length of $x$; say $|f(x)| = \ell(|x|)$. (For convenience, assume $\ell$ is one-one.) We say $f$ is *one-way* if the following holds for every $I$:

Let $I$ be be a polynomially bounded adversary.
Define $q_I(n)$ to be the probability that if $x$ is randomly chosen from $\{0,1\}^n$ and $I$ is run on $f(x)$ and $I$ outputs $\beta$, then $f(\beta) = f(x)$.

**THEN** $q_I(n)$ is negligible.

**Remark:** It is not too hard to see that if a one-way function $f$ exists, we can convert it to a one-way function $f'$ such that $|f'(x)| = |x|$.

**Theorem:** ((Håstad,Impagliazzo, Levin, Luby))
There exist one-way functions $\Longleftrightarrow$
there exist pseudo-random generators.

**Proof of $\Longleftarrow$:** Let $G$ be a pseudo-random number generator with length function $\ell(n) = 2n$. It is easy to see that $G$ is one-way.

**Harder Fact:** *Every* pseudo-random number generator is a one-way function.

**Application**: A one-way function $f$ can be used to login to a computer: the computer need only store $f(\text{PASSWORD})$ instead of the PASSWORD.
Now assume DES is a pseudo-random function generator;
define $f : \{0,1\}^{56} \to \{0,1\}^{64}$ by $f(x) = \text{DES}_x(\bar{0})$.
Then the above fact shows that $f$ is one-way;
this $f$ was used by UNIX for logins.

**Proof of $\Longrightarrow$:**

We will consider a simpler case due to Yao, and Goldreich/Levin. Let $f$ be a one-way *permutation*, that is, for every $n$, $f$ maps $\{0,1\}^n$ one-one/onto $\{0,1\}^n$.

**Fact:** One can show that the following is hard for an adversary: random $r, x \in \{0,1\}^n$ are chosen, and the adversary is given $r$ and $f(x)$; he wants to find $(r \cdot x)$ (the inner product mod 2 of $r$ and $x$). Then his success probability is negligibly greater than 1/2.

We now see that the following number generator $G$ (that expands its input by one bit) is pseudo-random: $G(rx) = [r, f(x), (r \cdot x)]$.

Why? $r$ is random; because $x$ is random and $f$ is a permutation, $f(x)$ is random. So none of the first $2n$ bits are predictable. And the above fact shows that the last bit is also unpredictable.

## (Public Key) Signature Scheme (or primitive) consists of the following polynomial-time computable functions:

- $\textsc{KeyGen}(1^n, \textit{random bits}) = (pub, pri)$.
  (Assume size of keys determines $n$.)
- For every $m \in \{0,1\}^*$, $\textsc{Sign}(pri, m) \in \{0,1\}^n$.
- For every $m \in \{0,1\}^*$ and $\sigma \in \{0,1\}^n$, $\textsc{Ver}(pub, m, \sigma) \in \{0,1\}$.

We insist that if $\textsc{KeyGen}$ outputs $(pub, pri)$, then $\textsc{Ver}(pub, m, \textsc{Sign}(pri, m)) = 1$.

## Definition of Security for Signature Schemes

For every polynomially (in $n$) bounded adversary $A$:

$A$ sees $pub$, chooses $m_1$, queries $\textsc{Sign}_{pri}(m_1)$, chooses $m_2$, queries $\textsc{Sign}_{pri}(m_2)$, etc.

$A$ then chooses a message $m$ for which he has not queried $\textsc{Sign}_{pri}$ and produces an attempted signature $\sigma$.
(Note that because $A$ is polynomially bounded, the sizes of the $m_i$ and $m$ will be bounded by a polynomial in $n$.)

Let $q_A(n)$ be the probability that $\textsc{Ver}(pub, m, \sigma) = 1$.
Then $q_A(n)$ is negligible.

(Remark:) It does not help to allow $\textsc{Sign}$ to be probabilistic. If it were, we can make it deterministic by including in the private key the key $k$ for a pseudo-random function generator $F$; for each message $m$ to be signed, just use $F_k(m)$ instead of the random bits.

**Theorem:** (Goldwasser, Micali, Rivest)
There exist public key signature schemes $\Longleftrightarrow$
there exist one-way functions (or pseudo-random generators).

In practice, signature schemes proposed or in use are based on strong versions of assumptions from public-key encryption technology, rather than doing the construction of this theorem.

Ideas from this theorem are nonetheless useful.

Say that we have a secure signature scheme $\mathcal{S}$, but we can only sign messages of length $n$ (for example) with it.

Say that we have a *hash* family $h$: For every $k \in \{0,1\}^n$ and every $m \in \{0,1\}^*$, $h_k(m) \in \{0,1\}^n$ and $h_k(m)$ can be computed in time polynomial in $n + |m|$.

Define the (full) signature scheme $\mathcal{S}'$:
Add to the keys of $\mathcal{S}$ a key $k$ for a hash family. Define
$\textsc{Sign}'_{pri,k}(m) = \mathrm{Sign}_{pri}(h_k(m))$, and
$\textsc{Ver}'_{pub,k}(m, \sigma) = \textsc{Ver}_{pub}(h_k(m), \sigma)$.

For this to be secure, we need the hash family $h$ to be *strongly collision resistant.*

# Definitions of collision resistance for hash family $h$ against a polynomially (in $n$) bounded adversary $A$

**private collision resistance:** If $A$ sees $1^n$ but isn't given random key $k$, then the probability is negligible that $A$ finds $m_1 \neq m_2$ such that $h_k(m_1) = h_k(m_2)$.
These hash families provably exist.

**weak (public) collision resistance:** If $A$ sees $1^n$, chooses $m_1$, is then given random key $k$, then the probability is negligible that $A$ finds $m_2 \neq m_1$ such that $h_k(m_1) = h_k(m_2)$.
These hash families exist $\Longleftrightarrow$ one-way functions exist.
(Naor/Yung, Rompel)

**strong (public) collision resistance:** If $A$ sees $k$, then the probability is negligible that $A$ finds $m_1 \neq m_2$ such that $h_k(m_1) = h_k(m_2)$. To prove the existence of these hash families, it appears that we need *more* than one-way functions.

Say that we have a secure signature scheme $\mathcal{S}$, but we can only sign messages of length $2n$ with it, and say that we have a *hash* family $h$ that is only guaranteed to satisfy *weak* collision resistance?

Can we get a secure (full) signature scheme from it?

Define the (full) *probabilistic* signature scheme $\mathcal{S}'$:
To sign $m$ using *pri*, randomly choose a key $k$ for the hash family, $|k| = n$, and let $\text{SIGN}'_{pri}(m) = [k, \text{Sign}_{pri}[k, h_k(m)]]$, and $\text{VER}'_{pub,k}(m, [k, \sigma]) = \text{VER}_{pub}([k, h_k(m)], \sigma)$.

Can you prove that this is secure?

# Distribution Smoothing Hash Family

Let $h$ be a hash family, $h_k : \{0,1\}^m \rightarrow \{0,1\}^n$, $m > n$. For a set $S \subseteq \{0,1\}^m$ of size much bigger than $2^n$, we wish that for most $k$, the distribution $h_k(S)$ on $\{0,1\}^n$ is very close to uniform.

If $D$ is a distribution on $\{0,1\}^n$ and $U$ is the uniform distribution, we define
$\text{DIST}(D,U) = \frac{1}{2}\sum_{x\in\{0,1\}^n} |D(x) - U(x)|$ = the maximum, over every (not necessarily polynomial-time) algorithm, of the probability of accepting a random member of $D$ minus the probability of accepting a random member of $U$.

It turns out a sufficient property for $h$ is *2-way independence*: For every two distinct $x, y \in \{0,1\}^m$, $\text{prob}_k(h_k(x) = h_k(y)) = 1/2^n$.

**Theorem:** Let $S \subseteq \{0,1\}^m$, $|S| \geq 2^t$.
Let h be a 2-way independent hash family of functions mapping $\{0,1\}^m$ to $\{0,1\}^n$.
Then for all but a negligible fraction of the keys $k$,
$\mathrm{DIST}(h_k(S), U) \leq \frac{1}{2^{(t-n)/3}}$.

**Example of a 2-way independent $h$:** View $k$ as a pair $[a, b]$ of elements from the field $\mathrm{GF}(2^m)$ and view $x \in \{0,1\}^m$ as members of $\mathrm{GF}(2^m)$.

Then define
$h_{[a,b]}(x) =$ the rightmost $n$ bits of $ax + b$ (with arithmetic in $\mathrm{GF}(2^m)$).

## Application

Let $p$ be an $n$-bit prime, and say $p - 1 = 2q$ where $q$ is prime.

Let $\mathbb{Z}_p^*$ be the group $\{1, 2, \ldots, p-1\}$ with multiplication mod $p$.

Let $g$ be a generator of a subgroup $G$ of $\mathbb{Z}_p^*$ of size $q$.
So $G$ has at least $2^{n-2}$ elements, each one an $n$-bit string.

We will see later that there is a session-key exchange protocol in which the two parties share a random member $\alpha$ of $G$.
But they *really* want to share a random *string*.

So they use a 2-way independent hash family of functions mapping $\{0,1\}^n$ to (say) $\{0,1\}^{n/2}$. They chose a random (public) $k$, and use as the session key $h_k(\alpha)$.