

Lecture Notes 7:**One-Way Functions****Recommended Reading.**

- Katz-Lindell 6.1, 8.3

1 Definition

We cannot demonstrate that an encryption scheme (or any other type of cryptosystem) is secure just by running it. This is why *proofs* of security are particularly important in cryptography. However, since the security of cryptosystem asserts that breaking the system requires lots of computation time and proving lower bounds on computation time is beyond current techniques in complexity theory, we cannot hope for absolute proofs of security. Instead, we will make (reasonable & precise) complexity-theoretic assumptions, and prove security based on these assumptions.

Ideally, we'd be able to prove a theorem of the form "If $\mathbf{P} \neq \mathbf{NP}$ then Encryption Scheme X is secure". Unfortunately, \mathbf{NP} -completeness seems to be insufficient for cryptography because it only refers to *worst-case* complexity. In cryptography, we need to be able to *efficiently generate* hard instances of a problem *together with a solution*: e.g. in encryption, the problem of recovering a message m from a ciphertext c should be hard and the pair (key,message) is "the solution." Of course, we've seen that secure encryption requires even more than infeasibility of recovering the plaintext but still this intuition points us to the kind of assumption we need for cryptography. This is captured by the notion of a one-way function — a function f which is easy to evaluate, but hard to invert. To generate a hard instance together with a solution, choose a random input x and evaluate $y = f(x)$. x is the "solution" to the hard problem of inverting f on y . The formal definition:

Definition 1 $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$ is a one-way function if:

1. f can be evaluated in polynomial time.
2. For every PPT A (poly-time in n), there is a negligible function ε such that

$$\Pr_{X \xleftarrow{R} \{0,1\}^n} [A(f(X), 1^n) \in f^{-1}(f(X))] < \varepsilon(n) \quad \forall n.$$

Concrete version: $f : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$ is (t, ε) -one-way if every A running in time $t \dots$ Time to evaluate f should be $\ll t$.

2 Some candidate one-way functions**2.1 Multiplication**

$f(x, y) = x \cdot y$, where $\|x\| = \|y\|$.

- How do we choose randomly two inputs? We break the random input of length $2n$ into two pieces of length n .
- Technicality: f is defined only for even-length inputs. To fix this, we can define the function $f(z) = z_1 \cdot z_2$ where $z = z_1 z_2$ ($\|z_1\| = \|z_2\| = \ell$) if $\|z\| = 2\ell$ and $z = z_1 z_2 b$ ($\|z_1\| = \|z_2\| = \ell$) if $\|z\| = 2\ell + 1$ – we drop the bit b . In general, we'll ignore this issue in the future, as it can usually be handled by similar "padding" tricks.
- Is it hard to factor a random number? If the random number is even or has small factors, it is easy to factor it. But still multiplication is a candidate one-way function because the adversary must find two *large* factors.

Definition 2 (Factoring Assumption) *For every PPT A , there is a negligible function ε such that:*

$$\Pr [A(N) \in \{P, Q\}] \leq \varepsilon(n)$$

where P and Q are random n -bit primes and $N = P \cdot Q$.

- RSA Challenges: the largest number factored so far is a 640-bit number. It took 3 months on 30 machines (30 2.2GHz CPU years).
- The best provable asymptotic algorithm known for factorization runs in time $\exp\left(O(n^{\frac{1}{2}}(\log n)^{\frac{1}{2}})\right)$
- The best heuristic asymptotic algorithm known for factorization runs in time $\exp\left(O(n^{\frac{1}{3}}(\log n)^{\frac{2}{3}})\right)$
- If we believe these are truly the best algorithms, then we can modify the factoring assumption to have a stronger assumption: we allow A to run in time say $\exp(n^{1/3})$ and the function $\varepsilon(n) = \exp(-n^{1/3})$

Fact 3 *If the Factoring Assumption holds, then multiplication is a one-way function.*

2.2 Subset sum

$f(x_1, \dots, x_n, S) = (x_1, \dots, x_n, \sum_{i \in S} x_i \bmod 2^n)$, where each x_i is a random n -bit number and S is a random subset of $\{1, \dots, n\}$.

The subset-sum problem is **NP**-complete but that doesn't imply that the function is one-way. **NP**-completeness refers to the worst case whereas one-way functions must be hard to invert for a randomly chosen input.

2.3 Block ciphers

Block ciphers are functions that are designed to be directly used for encryption. They are not based on any well-studied computational problem (such as factorization), but they are widely used in practice and extremely fast to compute. It is plausible that they already give secure encryption schemes (when used appropriately - we'll study this more later), but that is a very strong assumption. Modelling them simply as one-way functions is a much weaker assumption.

- DES (Data Encryption Standard) $DES_K(M) = C$ where K is a 56-bit key, M and C are 64-bit strings.
 $f(K) = DES_K(0^{64})$ for $|K| = 56$ is a candidate OWF. The best known attack of DES is an exhaustive key search. A specialized key-search machine that costs \$250K recovered a key in 56 hours.
- AES (Advanced Encryption Standard) The key length is variable ($|K| \in \{128, 192, 256\}$) and the message length is 128. $f(K) = AES_K(0^{128})$ for $|K| \in \{128, 192, 256\}$ is a candidate one-way function.
- For these block ciphers, exhaustive search is still essentially the best known attack in practice and takes time $\Theta(2^n)$ where n is the key length.

3 Limitations on “one-wayness”

- If $f : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$ can be computed in time t_0 , then it is not $(O(2^n t_0), 1)$ -one-way.
- $f : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$ cannot be $(O(n), \max\{1/2^n, 1/2^\ell\})$ -one-way.

These two ideas can be combined to yield a trade-off between time and success probability.

4 Additional Remarks

Evidence that one-way functions are a necessary assumption for cryptography comes from the following fact (which we will not prove).

Fact 4 *If secure encryption schemes exist, then one-way functions exist.*

It is an amazing theorem that the converse is also true — from any one-way function, one can build a secure encryption scheme. We will devote a number of upcoming lectures to proving a weaker form of this, constructing secure encryption from a more structured type of one-way function: f is a *one-way permutation* if f is a one-way function such that $|f(x)| = |x|$ and f is a permutation for each input length. In other words, if we let f_n denote the restriction of f to inputs of length n , then $f_n : \{0, 1\}^n \rightarrow \{0, 1\}^n$ is a permutation for each n .