University of Maryland
CMSC456 — Introduction to Cryptography
Professor Jonathan Katz

Lecture 5

1 More on One-Way Functions

At the end of last lecture, we showed that multiplication is *not* a one-way function, since factoring numbers is "easy" when there is a 3/4 probability that the number is even! Of course, it is reasonable to assume that multiplication is a *weak* one-way function, and then there is a theorem stating that this implies the existence of some (strong) one-way function. But can we do better and give a more natural construction?

In fact, we can. We do not need to restrict ourselves to functions whose domain is $\{0,1\}^*$; instead, we can consider functions over arbitrary domains $D \subset \{0,1\}^*$ *as long as D is efficiently sampleable*. What we mean by this is the following: let $D_k = D \cap \{0,1\}^k$ (i.e., D_k represents strings in D of length k). Then we require that it be possible to uniformly sample an element of D_k in time polynomial in k. More formally, there exists a PPT algorithm $\mathcal{S}$ such that $S(1^k)$ returns a uniformly distributed element in D_k.

With this in mind, we can now define a one-way function as an efficiently computable function which is hard to invert in the following sense (you should check that this is equivalent to the previous definition when $D = \{0,1\}^*$): for all PPT algorithms A there is a negligible function $\epsilon(\cdot)$ such that:

$$\Pr[x \leftarrow D_k; y = f(x); x' = A(y) : f(x') = y] \leq \epsilon(k).$$

With these concepts in mind, let's see how we might turn multiplication into a one-way function. Instead of allowing the domain of f to be $\mathbb{Z} \times \mathbb{Z}$ (or, equivalenty, $\{0,1\}^*$ if we parse things appropriately), let's set the domain D of f to be $D \stackrel{\text{def}}{=} \mathcal{P} \times \mathcal{P}$, where we let $\mathcal{P} \subset \mathbb{Z}$ be the set of prime numbers. Now we can view D_k as a pair (x, y) of primes *each of length $k/2$*. And in fact it is a very reasonable conjecture that factoring integers N, where N is a product of two equal-length primes, is "hard" for any polynomial-time algorithm; more formally (defining $\mathcal{P}_k$ as the set of primes of length k): for all PPT algorithms A there is a negligible function $\epsilon(\cdot)$ such that:

$$\Pr[x, y \leftarrow \mathcal{P}_k; z = x \cdot y : A(z) = (x, y)] \leq \epsilon(k).$$

(Note that because x and y are primes, multiplication is now one-to-one so we do not need to consider the case when A outputs $(x', y') \neq (x, y)$.)

The careful reader will note that we omitted one important consideration: can we in fact sample from $\mathcal{P}$ efficiently? (Equivalently, can we efficiently sample random k-bit primes?) The answer is yes; we will not give details now, but this will be discussed in the guest lecture on Monday.

2 Number Theory

We gave a brief review of modular arithmetic, and what it means to compute "modulo n". We also introduced the notation $\mathbb{Z}_n \stackrel{\text{def}}{=} \{0, 1, \ldots, n-1\}$. We then defined the notion of a group (see algebra handout), and defined the set:

$$\mathbb{Z}_N^* \stackrel{\text{def}}{=} \{x : 1 \leq x \leq N \text{ and } \gcd(x, N) = 1\}.$$

We defined $\varphi(n) \stackrel{\text{def}}{=} |\mathbb{Z}_N^*|$. Note that $\mathbb{Z}_p^* = \{1, \ldots, p-1\}$ when p is prime and hence $\varphi(p) = p - 1$. We also showed that if $N = pq$ is the product of two distinct primes p, q then $\varphi(N) = |\mathbb{Z}_N^*| = (p-1)(q-1)$.

We stated the important fact that, for any N, the set $\mathbb{Z}_N^*$ can be viewed as a multiplicative group.

We also stated the following important lemma:

Lemma 1 *Let m be the order of (finite) group G. Then $g^m = 1$ for any nonzero $g \in G$.*

There is a nice (simple) proof for this; see [Ch] for details. This simple lemma can be used to demonstrate a very useful fact which we state in its own lemma because it is so important:

Lemma 2 *Let G be a finite group of order m. Let $g \in G$ and x be an integer. Then:*

$$g^x = g^{x \bmod m}.$$

Proof Let $x = x' \bmod m$. Then we can write $x = km + x'$. But now we have $g^x = g^{km+x'} = g^{km} g^{x'} = (g^m)^k g^{x'} = (1)^k g^{x'} = g^{x'}$. ∎

2.1 Chinese Remaindering

Let $N = pq$ be a product of two distinct primes. Chinese remaindering is an equivalent way of viewing $\mathbb{Z}_N^*$ as $\mathbb{Z}_p^* \times \mathbb{Z}_q^*$. The way this works is as follows: for any element $x \in \mathbb{Z}_N^*$, we can view x as $(x_p, x_q) \in \mathbb{Z}_p^* \times \mathbb{Z}_q^*$ where $x_p = x \bmod p$ and $x_q = x \bmod q$. Let's look at a particular example for $N = 15 = 3 \cdot 5$. Element $7 \in \mathbb{Z}_{15}^*$ can be written as $(1, 2)$ since $7 = 1 \bmod 3$ and $7 = 2 \bmod 5$. Doing this for all elements of the group gives the following table:

$$\begin{array}{rcl}
1 & \leftrightarrow & (1,1) \\
2 & \leftrightarrow & (2,2) \\
4 & \leftrightarrow & (1,4) \\
7 & \leftrightarrow & (1,2) \\
8 & \leftrightarrow & (2,3) \\
11 & \leftrightarrow & (2,1) \\
13 & \leftrightarrow & (1,3) \\
14 & \leftrightarrow & (2,4)
\end{array}$$

(Note as a sanity check that the number of elements in $\mathbb{Z}_{15}^*$ is indeed given by $2 \cdot 4 = 8$.) Note that each element (pair) of $\mathbb{Z}_3^* \times \mathbb{Z}_5^*$ appears once and only once on the right hand side above. This suggests that there is a *bijection* between $\mathbb{Z}_{15}^*$ and $\mathbb{Z}_3^* \times \mathbb{Z}_5^*$. The Chinese remainder theorem (which we do not prove here) can be viewed as stating that this is indeed a bijection for any N which is a product of two primes (actually, the Chinese remainder theorem is more general and can be extended for N of various other forms). Note also that the Chinese remainder theorem gives an alternate proof of the value of $\varphi(N)$:

$$\varphi(N) = |\mathbb{Z}_N^*| = |\mathbb{Z}_p^* \times \mathbb{Z}_q^*| = |\mathbb{Z}_p^*| \cdot |\mathbb{Z}_q^*| = (p-1) \cdot (q-1).$$

Now one important fact about this alternate representation of $\mathbb{Z}_N^*$ is the following: if $x, y \in \mathbb{Z}_N^*$ with $x \leftrightarrow (x_p, x_q)$ and $y \leftrightarrow (y_p, y_q)$, then $x \cdot y \bmod N \leftrightarrow (x_p \cdot y_p \bmod p, x_q \cdot y_q \bmod q)$. This is a very useful fact when doing computation modulo large numbers N: instead of computing $x \cdot y$ and then reducing modulo N, we can convert x and y to their alternate representations (x_p, x_q) and (y_p, y_q), do our multiplication modulo p and q, and then convert the answer back to an element in $\mathbb{Z}_N^*$. So if p, q are k-bit primes, then instead of doing one multiplication modulo a $2k$-bit number N, we instead do two multiplications modulo k-bit numbers. This extends for the case of exponentiation, as we will see next time.

As a final remark, note that is is "easy" to convert $x \in \mathbb{Z}_N^*$ to $(x_p, x_q) \in \mathbb{Z}_p^* \times \mathbb{Z}_q^*$ (simply compute the necessary remainders). How can we go in the opposite direction? Well, say we have found (in advance) values $X, Y \in \mathbb{Z}_N$ such that $X \leftrightarrow (1,0)$ and $Y \leftrightarrow (0,1)$. (We are abusing notation here, since in fact $(1,0) \notin \mathbb{Z}_p^* \times \mathbb{Z}_q^*$ since $0 \notin \mathbb{Z}_q^*$. This is why we say $X, Y \in \mathbb{Z}_N$ and not $X, Y \in \mathbb{Z}_N^*$.) Then, given (x_p, x_q) we have:

$$(x_p, x_q) = x_p \cdot (1,0) + x_q \cdot (0,1) = x_p \cdot X + x_q \cdot Y,$$

and this final computation can be done modulo N. As an example in $\mathbb{Z}_{15}^*$, we have $(1,0) \leftrightarrow 10$ (since $10 = 1 \bmod 3$ and $10 = 0 \bmod 5$) and $(0,1) \leftrightarrow 6$. So we can convert $(1,3)$ by doing:

$$(1,3) = 1 \cdot (1,0) + 3 \cdot (0,1) = 1 \cdot 10 + 3 \cdot 6 \bmod 15 = 13,$$

which is the correct answer.