1 Last lecture, and plan for today's lecture

Last lecture, we saw a construction of an encryption scheme satisfying No-Query-Semantic-Security, assuming the security of PRFs. Next, we wanted to show if a secure PRG implies a secure PRF. We saw a few attempts that did not work. Here is another attempt (proposed by one of the students after last class):

$$F(x,k) = \Big([G(x) \oplus G(k)]_{[1,n]}\Big) \oplus \Big([G(x) \oplus G(k)]_{[n+1,2n]}\Big)$$

That is, compute $G(x) \oplus G(k)$, this results in $2n$ bit output. Then compute XOR of the two halves.

Consider the following attack on this function F: query on 0^n, let the output be y_0. Next, query on 1^n, let the output be y_1. Compute $y_0 \oplus y_1$, and check if this is equal to the XOR of the two halves of $G(0^n) \oplus G(1^n)$. Note that G is a public function, and hence the adversary can compute the XOR of the two halves. If y_0 and y_1 are random, then we don't expect their XOR to be equal to the XOR of the two halves of $G(0^n) \oplus G(1^n)$.

2 PRG → PRF (not in course syllabus)

Let $G : \{0,1\}^n \to \{0,1\}^{2n}$ be a secure pseudorandom generator. Last class, we discussed how to extend the output space of a PRG via a tree-based approach. This approach, when extended to n levels, gives us a keyed function $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ as follows. Let $x = (x_1, x_2, \ldots, x_n) \in \{0,1\}^n$ be the input, and $k \in \{0,1\}^n$ the key. Let $x_{[1,t]}$ denote the first t bits of x, and let $G(k) = (y_0 \mid\mid y_1)$.

For $i = 2$ to n, do the following:

1. Given $y_{x_{[1,i]}}$, compute $G(y_{x_{[1,i]}}) = \Big(y_{(x_{[1,i]},0)} \mid\mid y_{(x_{[1,i]},1)}\Big)$. Note that either the blue part, or the purple part will be used for the next iteration.

Output y_x as the function evaluation on x. A pictorial representation of the construction is given in Figure 1.

Note that the evaluation does not require computing the entire tree. How to prove security? The stepping stone for this proof is Question 11.03 (from previous lecture).

This is one of the my favorite proofs in cryptography, and I strongly encourage you to attempt it (after attempting Question 11.03). The proof involves a clever choice of hybrid worlds. This construction/proof was given in the 80s by Oded Goldreich, Shafi Goldwasser and Silvio Micali (the construction is referred to as the GGM construction).

3 PRFs vs PRPs

3.1 PRPs from PRFs

Given a secure PRF $F : \{0,1\}^{\ell_{\text{in}}} \times \{0,1\}^n \to \{0,1\}^{\ell_{\text{out}}}$, can we build a secure PRP (with some domain, key space and range)?[1]

The initial attempts resulted in permutations that were either inefficient to implement, or the resulting construction was not a permutation. A simple way to verify if a function is a permutation is by demonstrating the inverse of the permutation.

1. **Attempt 1.** Consider the function $P_1 : \{0,1\}^{\ell_{\text{in}}+\ell_{\text{out}}} \times \{0,1\}^n \to \{0,1\}^{\ell_{\text{in}}+\ell_{\text{out}}}$ defined as follows.

$$P_1((x,y),\ k) = (x, y \oplus F(x,k)) \text{ where } x \in \{0,1\}^{\ell_{\text{in}}}, y \in \{0,1\}^{\ell_{\text{out}}}$$

[1]Note that for a PRF, the input length, key length and output length can all be different. In particular, if we need a secure PRF, only the key length needs to be as large as the security parameter; the input and output lengths can be small. Similarly, for a PRP, the input length and output length must be equal, but they need not be same as key length. Again, for a secure PRP, only the key size needs to be some polynomial in the security parameter, the input length can be small.

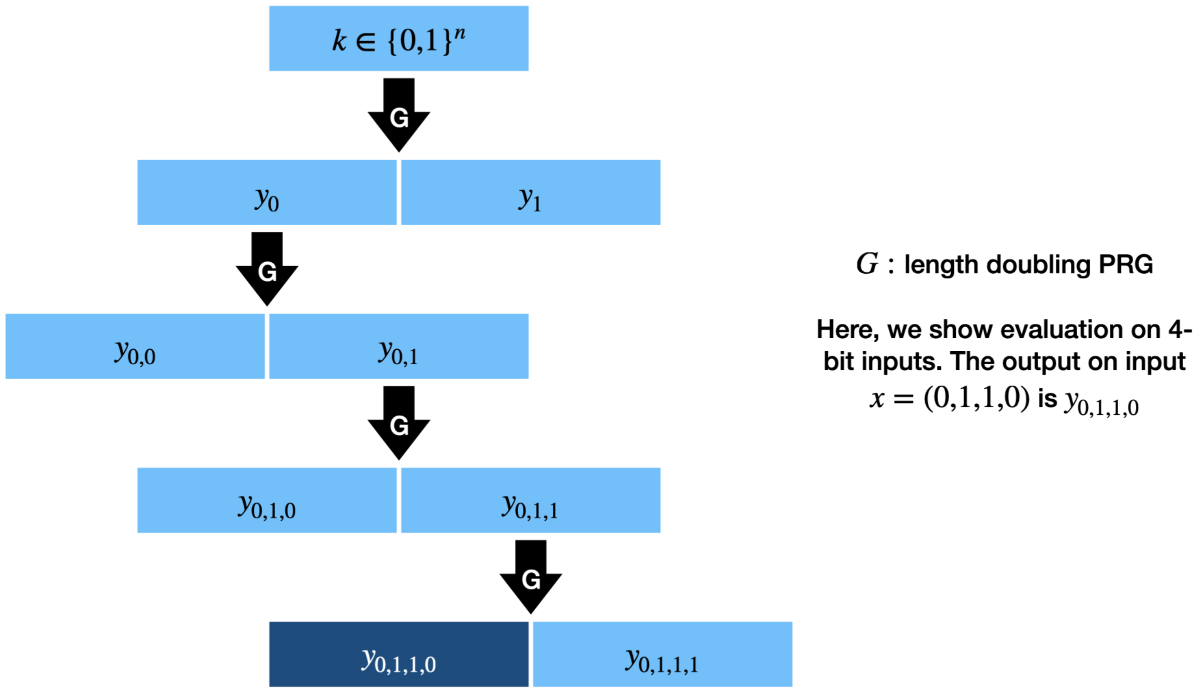
$k \in \{0,1\}^n$
G
y_0
y_1
G
$y_{0,0}$
$y_{0,1}$
G
$y_{0,1,0}$
$y_{0,1,1}$
G
$y_{0,1,1,0}$
$y_{0,1,1,1}$
G : length doubling PRG
Here, we show evaluation on 4-bit inputs. The output on input $x = (0,1,1,0)$ is $y_{0,1,1,0}$

Figure 1: PRF evaluation using length-doubling PRG

(☕) **Challenge Question:** Assuming G is a secure PRG, show that the keyed function F is a secure PRF. As a first step, try to prove security against adversaries that make at most 2 queries. Extend this idea to handle unbounded number of adversarial queries.

Check that $P_1(\cdot, k) : \{0,1\}^{\ell_{\text{in}}+\ell_{\text{out}}} \to \{0,1\}^{\ell_{\text{in}}+\ell_{\text{out}}}$ is indeed a permutation. This is because $P_1(\cdot, k)$ is the inverse of $P_1(\cdot, k)$.

But this is not a secure permutation, since the output contains part of the input, and just one query is enough to distinguish the keyed permutation from a uniformly random permutation.

2. **Attempt 2.** If we apply P_1 twice with the same key, then we get back the original input (that is, $P_1(\cdot, k) \circ P_1(\cdot, k)$ is the identity function, where $\circ$ denotes composition of functions). What if we add a swap operation between the two applications of P_1, and use same key for the two applications of P_1? More formally, let $\mathsf{swap}(a, b) = (b, a)$ for all a, b. Consider the function $P_2(\cdot, k) \equiv P_1(\cdot, k) \circ \mathsf{swap} \circ P_1(\cdot, k)$. A pictorial representation of P_2 is given in Figure 2 below.

   Check that P_2 is indeed a permutation. Is this a secure PRP, assuming F is a secure PRF? For any input (x, y), the output is $\Big(y \oplus F(x, k), x \oplus F\Big(y \oplus F(x, k), k\Big)\Big)$. Unfortunately, not, and this can be broken with two queries! If the function $P_2(\cdot, k)$ is queried on (x, y) and $(x, y \oplus 1^{\ell_{\text{out}}})$, then you can find some correlation in the output, something that should not happen we want P_2 to be indistinguishable from a random permutation.

3. **Attempt 3:** Suppose we repeat $P_1(\cdot, k)$ thrice, and interleaved with swap operations. Would that be a secure PRP? More formally, consider the permutation $P_3(\cdot, k) = P_1(\cdot, k) \circ \mathsf{swap} \circ P_1(\cdot, k) \circ \mathsf{swap} \circ P_1(\cdot, k)$. This is shown pictorially in Figure 3.

(☕) **Challenge Question:** The GGM construction is inherently sequential (requires n sequential applications of the PRG G). Suppose we want to reduce the number of sequential applications of G. Is it possible to modify the construction to ensure fewer sequential operations of G?

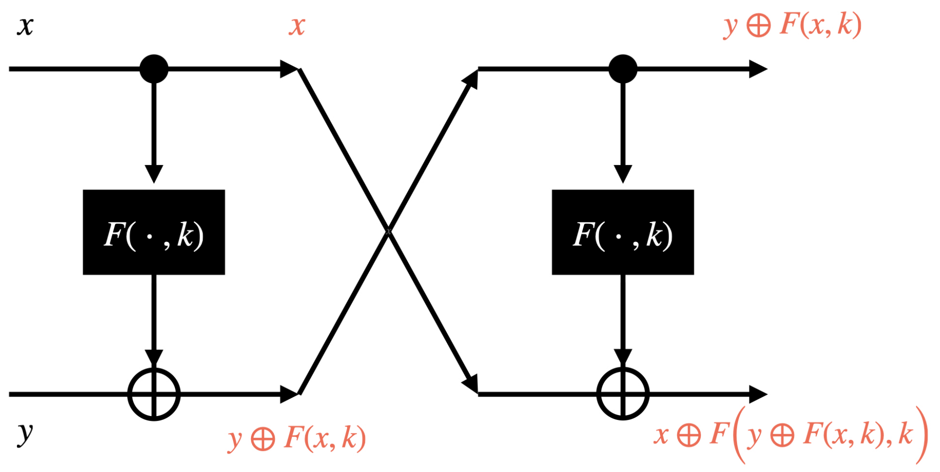

Figure 2: Implementation of keyed permutation P_2

This scheme is also not secure. In general, this strategy is bound to fail, since the inverse of $P_3(\cdot,k)$ is $P_3(\cdot,k)$. As a result, an adversary can first query the PRP challenger on (x_1,x_2), receive a response (y_1,y_2), and then query the PRP challenger on (y_1,y_2). If the PRP challenger chose $P_3(\cdot,k)$, then the adversary receives (x_1,x_2). On the other hand, if the challenger chose a uniformly random permutation, then the adversary will not receive (x_1,x_2) as response to the second query.

Can we make a slight modification to Attempt 3 so that in the resulting scheme, applying the same permutation twice does not result in identity? We will discuss this in the next lecture. You all are very close to a provably secure construction, so I encourage you to think about this!

3.2 PRFs from PRPs

Last lecture, we briefly discussed that any PRP is already a PRF. This proof used the idea that a uniformly random permutation is indistinguishable from a uniformly random function *if only polynomially many queries are made* by the adversary. Let us spend some time understanding this proof, since this idea will show up again, later in the course.

The core idea is a popular probability question/paradox called the 'birthday paradox'. Suppose there are k students in a class, what is the probability that at least two of them share their birthday (assuming birthdays are distributed independently, and uniformly throughout the year)? Intuitively, it feels that this probability should be about $k/365$. However, our intuition is misleading in this case, and this probability is about $k^2/365$. [2]

Let us look at this problem more abstractly: we are sampling t numbers out of $[N] = \{1,2,\ldots,N\}$, uniformly at random and independently. What is the probability that at least two of the sampled numbers are equal? This probability is equal to

$$1 - \frac{N \cdot (N-1) \cdot \ldots \cdot (N-t+1)}{N^t}.$$

This expression is somewhat complicated, but there is an easy upper bound:

[2]This was 'experimentally verified' in today's class. Thirty one students entered their birthdates, and there were two collisions!

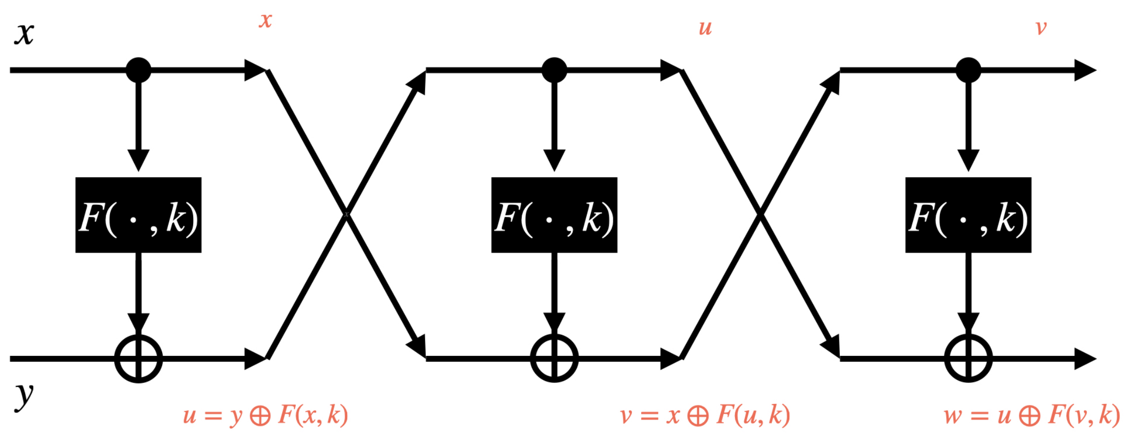

Figure 3: Implementation of keyed permutation P_3. The keyspace is $\{0,1\}^n$ here, input/output space is $\{0,1\}^{\ell_{\text{in}}+\ell_{\text{out}}}$ and it maps (x, y) to (v, w).

Claim 12.01 (Birthday Bound).

$$\Pr\left[\text{There exist repetitions when sampling } t \text{ numbers, u.a.r. with replacement}\right] = \Theta\left(\frac{t^2}{N}\right)$$

Proof. **Upper Bound:**

$$\begin{aligned}&\Pr\left[\text{There exist repetitions when sampling } t \text{ numbers, u.a.r. with replacement}\right]\\ &\leq \sum_{i \neq j} \Pr\left[i^{\text{th}} \text{ sampled number } = j^{\text{th}} \text{ sampled number }\right]\\ &= \frac{t(t-1)}{2} \cdot \frac{1}{N} < \frac{t^2}{N}\end{aligned}$$

Here, the first step follows from union bound (if there exists repetition, then there exists a pair of indices i and j such that the i^{th} sampled number is equal to the j^{th} sampled one). Once we fix our attention to the i^{th} and j^{th} numbers, the probability that they are equal is $1/N$, and there are $t(t-1)/2$ such pairs of indices. This concludes our proof of the upper bound.

Lower Bound: You can assume the lower bound as a fact. For completeness, I am including a sketch of the proof here.

Note that $N \cdot (N-1) \cdot \ldots \cdot (N-t+1)/N^t$ can be expressed as $(1-1/N) \cdot \ldots \cdot (1-(t-1)/N)$. Hence, the probability of at least two samples being equal is

$$1 - \prod_{i=1}^{t-1} \left(1 - \frac{i}{N}\right).$$

Next, we use the following inequalities: $(1-1/e) \cdot x \leq 1 - e^{-x} \leq x$ for all $x \in [0,1]$. Using this inequality, we can upper-bound $1 - i/N$ with $e^{-i/N}$, and therefore our probability of interest is at least

$$1 - \prod_i e^{-i/N} = 1 - e^{-t^2/(cN)}$$

for some constant c. Finally, we can lower-bound $1 - e^{-t^2/cN}$ with $(1 - 1/e) \cdot (t^2/cN)$. $\square$

How do we use the birthday bound to show that a uniformly random permutation is indistinguishable from a uniformly random function, if only t queries are allowed?

4 Lecture summary, plan for next lecture, additional resources

Summary: Here are the main take-aways from this lecture:

- PRFs can be constructed from secure PRGs (you are encouraged to think about the security proof, but it is not part of the course syllabus).
- We saw a few attempts at constructing PRPs from PRFs, and you should understand why Attempt 1, Attempt 2 and Attempt 3 (that is, the keyed permutations P_1, P_2 and P_3) are not secure PRPs.
- Any PRP is also a PRF. That is, a random permutation 'looks like' a random function if the domain is much larger than the number of queries to the permutation. The main take-away for this part is the birthday bound/paradox.

Next Lecture: We will first complete our discussion for the relations between PRPs and PRFs. Next, we will define semantic security for encryption schemes against general 'read-only' adversaries, and we will see that our current constructions can be tweaked slightly to achieve security against such adversaries.

Relevant sections from textbook [Boneh-Shoup]: Section 4.4.3 discusses why a PRP is also a secure PRF. Our proof (which we will complete next lecture) will be slightly different. Section 4.5 contains the PRF $\rightarrow$ PRP transformation.