Pseudorandom functions. Instead of exhaustively listing the properties we want from a “good” blockcipher, we define a “master property” that implies everything we could want in principle. This master property is known as being a “pseudorandom function.” (Here “pseudorandom” means that not actually random, but behaving like random, and “function” is a bit misnomer as we will actually be talking of function families defined next.) We work in a more general setting than blockciphers and define the notion of a function family. A function family is a map $F: \text{Keys}(F) \times \text{Dom}(F) \to \text{Rng}(F)$. For any key $K \in \text{Keys}(F)$ define $F_K(x) = F(K, x)$ for all $x \in \text{Dom}(F)$. Note that a blockcipher is a particular case that $\text{Dom}(F) = \text{Rng}(F)$ and the inverse map is efficiently computable. For a function family $F: \text{Keys}(F) \times \text{Dom}(F) \to \text{Rng}(F)$ define the games below:

<table>
<thead>
<tr>
<th>Game</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRF-REAL$_F$</td>
<td>$	ext{proc}$ \text{INITIALIZE} \hspace{1cm} \text{proc}$ \text{FINALIZE}(b)$</td>
</tr>
<tr>
<td>PRF-RAND$_F$</td>
<td>$	ext{proc}$ \text{INITIALIZE} \hspace{1cm} \text{proc}$ \text{FINALIZE}(b)$</td>
</tr>
</tbody>
</table>

Then the PRF-advantage of an adversary $A$ is

$\text{Adv}^{\text{prf}}_F(A) = \Pr[\text{PRF-REAL}_F(A) \text{ outputs 1}] - \Pr[\text{PRF-RAND}_F(A) \text{ outputs 1}].$

So the adversary is trying to guess which game it is in. A high advantage means $A$ is doing well and is not PRF-secure. A low advantage means $A$ is doing poorly and $F$ resists this specific attack. An adversary’s advantage depends not only on its strategy but its resources: its running time $t$ and number of queries $q$. $F$ is a PRF (or PRF-secure) if $\text{Adv}^{\text{prf}}_F(A)$ is “small” for all “practical” $A$, meaning its running-time and number of queries are not too large. One talks about “bits of security;” 80-bit security could mean $\text{Adv}^{\text{prf}}_F(A) \leq 2^{-n}$ for all adversaries $A$ with running-time and queries at most $2^{80-n}$. Question: Is the one-time pad a secure PRF?

PRF Security implies KR Security. We have said that PRF security is a “master property” that implies everything we could want from a blockcipher. Let’s illustrate this for the particular case of security against key recovery. We prove the following theorem:

Theorem 0.1 Let $E: \{0, 1\}^k \times \{0, 1\}^\ell \to \{0, 1\}^\ell$ be a blockcipher. Let $B$ be a $q$-query KR-adversary. Then there exists a $q$-query PRF-adversary $A$ such that

$\text{Adv}^{\text{kr}}_E(B) \leq \text{Adv}^{\text{prf}}_E(A) + 2^{k-q\ell}.$
The running-time of \( A \) is the time to run \( B \) plus \( O(q\ell) \).

**Proof:** To be filled in...

**Birthday attack.** There is a generic “birthday attack” on any blockcipher under the PRF security notion. Namely, consider the adversary which chooses \( q \) distinct points, queries its oracle on these points, and returns 1 if there is no collision. What is the PRF-advantage of this adversary (as a function of \( q \))? This is the famous birthday problem (or paradox); its advantage is roughly \( q^2/2^\ell \), or concretely at least \( 0.3 \cdot q^2/2^\ell \). Thus after about \( 2^{\ell/2} \) queries this advantage is 1. This means that any blockcipher has roughly at most \( \ell/2 \)-bits of security under the PRF notion.

**Pseudorandom permutations.** Is the birthday attack the “best” generic attack against a blockcipher under the PRF notion? Intuitively, it seems that the answer is “yes” but how do we prove this? To prove this we will first formalize the notion of pseudorandom permutations, which are in some sense a better model for capturing what blockciphers actually are. For a function family \( F: \text{Keys}(F) \times \text{Dom}(F) \rightarrow \text{Rng}(F) \) define the games below:

<table>
<thead>
<tr>
<th>Game PRP-REAL(_F)</th>
<th>Game PRP-RAND(_F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>proc INITIALIZE</td>
<td>proc INITIALIZE</td>
</tr>
<tr>
<td>( K \leftarrow K )</td>
<td>( R \leftarrow \emptyset )</td>
</tr>
<tr>
<td>proc FN(x)</td>
<td>proc FN(x)</td>
</tr>
<tr>
<td>Return ( F_K(x) )</td>
<td>If ( T[x] = \perp ) then ( T[x] \leftarrow \text{Rng}(F) \setminus R )</td>
</tr>
<tr>
<td>proc FINALIZE(b)</td>
<td>Return ( T[x] )</td>
</tr>
<tr>
<td>Return ( b )</td>
<td>proc FINALIZE(b)</td>
</tr>
</tbody>
</table>

Then the PRP-advantage of an adversary \( A \) is

\[
\text{Adv}^\text{prp}_F(A) = \Pr[\text{PRP-REAL}_F(A) \text{ outputs } 1] - \Pr[\text{PRP-RAND}_F(A) \text{ outputs } 1].
\]

The Fundamental Lemma of Game-Playing. We will now state a fundamental lemma we will use. Let \( G_0, G_1 \) be games (in the usual framework we have defined) which set a flag \( \text{bad} \). Call \( G_0, G_1 \) identical-until-bad if their code differs only after this flag is set.

**Lemma 0.2** Let \( G_0, G_1 \) be identical-until-bad. Then for any output \( y \)

\[
\Pr[G_1 \text{ outputs } y] - \Pr[G_0 \text{ outputs } y] \leq \Pr[G_0 \text{ sets } \text{bad}].
\]

Another variant of the lemma can also be useful:

**Lemma 0.3** Let \( G_0, G_1 \) be identical-until-bad. Then for any output \( y \)

\[
\Pr[G_1 \text{ outputs } y \land G_1 \text{ doesn’t set } \text{bad}] = \Pr[G_0 \text{ outputs } y \land G_0 \text{ doesn’t set } \text{bad}].
\]
The PRP/PRF Switching Lemma. We can now prove that there is no better attack on a blockcipher under the PRF notion than the birthday attack, via the following lemma.

**Lemma 0.4** Let $F : \{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^n$ be a function family. Then for any $q$-query adversary $A$

$$\text{Adv}^\text{prf}_F(A) \leq \text{Adv}^\text{prp}_F(A) + \frac{q(q - 1)}{2^{n+1}}.$$ 

In other words an adversary’s advantage can be larger under the PRF notion than the PRP notion, but only by $\frac{q(q - 1)}{2^{n+1}}$. If there was a significantly better attack on a blockcipher under the PRF notion than the birthday attack then this difference would be larger.

**Proof:** To be filled in...