Modern stream ciphers (eSTREAM project: 2004-2008)

- Core design maps 256-bit key, 64-bit nonce, 64-bit counter onto a 512-bit output
  - Design is more complex: relies on a sequence of rounds
  - Each round consists of 32-bit additions, xor's, and bit-shifts

- Very fast even in software (4-14 CPU cycles/output byte) — used to encrypt TLS traffic between Android and Google services

Recall: the one-time pad is not reusable (i.e., the two-time pad is totally broken)

NEVER REUSE THE KEY TO A STREAM CIPHER!

But wait... we "proved" that a stream cipher was secure, and yet, there is an attack?

Recall security game:

\[
\begin{align*}
\text{adversary} & : m_0, m_1 \quad \rightarrow \quad k \in \mathbb{K} \quad \rightarrow \quad C_b \leftarrow \text{Encrypt}(k, m_b) \\
\text{challenger} & : k \
\end{align*}
\]

Observe: adversary only sees one ciphertext key is only used once

\[b \in \{0, 1\}\]

⇒ Security in this model says nothing about multiple messages/ciphertexts

Problem: If we want security with multiple ciphertexts, we need a different or stronger definition (CPA security)

Reusable security: security against chosen-plaintext attacks (CPA-security)

⇒ Semantic security should hold even if adversary sees multiple encrypted messages of its choosing, encrypted!

⇒ Captures many settings where adversary might know the message that is encrypted (e.g., predictable headers or site content in web traffic) or be able to influence it (e.g., client replies to an email sent by adversary)

⇒ Goal is to capture as broad of a range of attacks as possible
Definition: An encryption scheme $TSE = (\text{Encrypt}, \text{Decrypt})$ is secure against chosen-plaintext attacks (CPA-secure) if for all efficient adversaries $A$:
\[
\text{CPAAD}_{\text{UA}}[A, TSE] = |\Pr[b = 1] - \Pr[b = 1]| \leq \text{negl},
\]
where $W_b$ is the output of the following experiment:

\[
\begin{aligned}
\text{adversary} & \quad \rightarrow \quad \text{challenger} \\
\{m, m' \in \mathcal{M} \} & \quad \text{Choose} \quad \{m, m' \in \mathcal{M} \} \\
\text{Encrypt}(k, m) & \quad \text{where} \quad m \neq m' \\
\downarrow & \quad k \leftarrow \mathcal{K} \\
W_b & \quad \text{Output of experiment } W_b
\end{aligned}
\]

$A$'s goal is to guess which of $m$ or $m'$ was encrypted, given access to an encryption oracle (i.e., $A$ gets to see encryptions of messages of its choice).

Claim. A stream cipher is not CPA-secure.

Proof. Consider the following adversary:

\[
\begin{aligned}
\text{adversary} & \quad \rightarrow \quad \text{challenger} \\
\{m, m' \in \mathcal{M} \} & \quad \text{Choose} \quad \{m, m' \in \mathcal{M} \} \\
\text{Encrypt}(k, m) & \quad \text{where} \quad m \neq m' \\
\downarrow & \quad k \leftarrow \mathcal{K} \\
W_b & \quad \text{Output of experiment } W_b
\end{aligned}
\]

\[
\begin{aligned}
\text{Output} 0 & \quad \text{if} \quad c = c' \\
\text{Output} 1 & \quad \text{if} \quad c 
eq c'
\end{aligned}
\]

Observe: Above attack works for any deterministic encryption scheme.

\[
\Rightarrow \text{CPA-secure encryption must be randomized!}
\]

\[
\Rightarrow \text{To be reusable, cannot be deterministic. Encrypting the same message twice should not reveal that identical messages were encrypted.}
\]

To build a CPA-secure encryption scheme, we will use a "block cipher":

- Block cipher is an invertible keyed function that takes a block of $n$ input bits and produces a block of $n$ output bits.
- Examples include 3DES (key size 168 bits, block size 64 bits), AES (key size 128 bits, block size 128 bits).

Will define block ciphers abstractly first: pseudorandom functions (PRFs) and pseudorandom permutations (PRPs).

General idea: PRFs behave like random functions, PRPs behave like random permutations.
Definition. A function $F : \mathbb{K} \times X \rightarrow Y$ with key space $\mathbb{K}$, domain $X$, and range $Y$ is a pseudorandom function (PRF) if for all efficient adversaries $A$, $\left| W_0 - W_1 \right| = \text{negl}$, where $\text{Pr}[A \text{ outputs } 1 | b = 0] - \text{Pr}[A \text{ outputs } 1 | b = 1] = \text{negl}$.

Intuitively, input-output behavior of a PRF is indistinguishable from that of a random function (to any computationally bounded adversary).

| Block Cipher | Block Size | Key Size | Functions $|\mathbb{F}^{|X|}|$ | $2^{n(n-1)/2}$ Functions $|\mathbb{P}(\mathbb{F})|$ |
|-------------|------------|----------|-----------------|-------------------------------|
| 3DES        | 64 bits    | 168 bits | $2^{168}$       | $2^{64(2^{64})}$ |
| AES         | 128 bits   | 128 bits | $2^{128}$       | $2^{128(2^{128})}$ |

No_to: a block cipher is another term for PRP (just like stream ciphers are PRGs).
Observe that a block cipher can be used to construct a PRG:

\[ F: \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^n \]

be a block cipher

Define \( G: \{0,1\}^n \rightarrow \{0,1\}^n \) as

\[ G(k) = F(k, 1) || F(k, 2) || \ldots || F(k, n) \]

we said PRP above (will revisit this)

Theorem. If \( F \) is a secure PRF, then \( G \) is a secure PRG.

Proof. As usual, we show the contrapositive: if \( G \) is not a secure PRG, then \( F \) is not a secure PRF.

Suppose we have efficient adversary \( A \) for \( G \). We use \( A \) to build adversary for \( F \):

Not quite... for a random function, \( f(x) = f(x) \) with probability \( \frac{1}{2^n} \) but \( 2^n \) might be very very small...

for a random permutation, \( f(x) \neq f(x) \) with probability 0 adversary won't notice unless it sees a "collision" [i.e., two values \( x, y \) where \( f(x) = f(y) \)]

PRF Switching Lemma. Let \( F: K \times X \rightarrow X \) be a secure PRF. Then, for any \( Q \)-query adversary \( A \):

\[ | \text{PRFAdv}[A,F] - \text{PRFAdv}[A,F] | \leq \frac{Q^2}{2^n} \]

Proof Idea. Adversary essentially cannot tell the difference unless it sees a collision. If there is no collision, then it is just seeing random values. How many queries before there is a collision? Birthday paradox: \( Q \sim \sqrt{Q^n} \)

Take-away: If \( Q^n \) is large (e.g., exponential), then we can use a PRP as a PRF.

- 3DES: \( n = 64 \) so \( Q^n = 2^{64} \) [it adversary makes \( \ll 2^{32} \) queries, then can use it as a PRF]
- AES: \( n = 128 \) so \( Q^n = 2^{128} \) [if adversary makes \( \ll 2^{64} \) queries, then can use it as a PRF]