Cryptography

Lecture 14

Announcements

- Midterm grades, solutions posted
  - Median was 68
  - Please read over solutions before re-grade requests
- Two extra credit opportunities
  - Current events (up to +5 added to midterm grade)
  - Scholarly paper (up to +10 added to midterm grade)

Agenda

- This time: New Unit
  - Practical constructions of Block Ciphers
    - SPN (K/L 6.2)
    - Feistel Networks (K/L 6.2)

Block Ciphers

(AES)

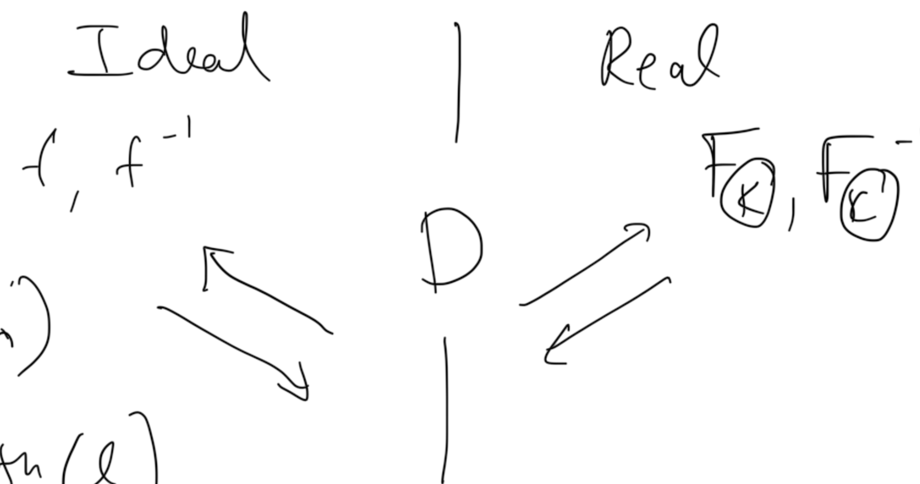
Pseudorandom Permutation (PRP)

Recall: A block cipher is an efficient, keyed permutation $F: \{0,1\}^n \to \{0,1\}^\ell$. This means the function $F_k(x) \coloneqq F(k, x)$ is a bijection, and moreover, F_k and its inverse F_k^{-1} are efficiently computable given k.

- n is the key length
- ℓ is the block length

AES

128	192	256	Key length (n)
128	128	128	block length (ℓ)



Block Cipher Security

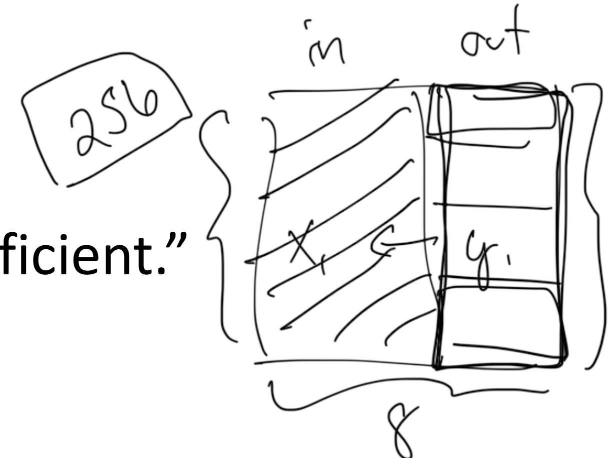
NIST

Call for proposals for the AES competition: 1997-2000

"The security provided by an algorithm is the most important factor… Algorithms will be judged on the following factors… The extent to which the algorithm output is indistinguishable from a random permutation…"

Note: It is assumed the adversary gets to query both F_k, F_k^{-1} or f, f^{-1}, which means we want a **strong** pseudorandom permutation.

First Idea



- Random permutations over small domains are “efficient.”
  - What does this mean?
- First attempt to define F_k:
  - The key k for F will specify 16 permutations $f_1, \dots, f_{16}$ that each have an 8-bit block length ($16 \cdot 8 = 128$ input length in total).
  - Given an input $x \in \{0,1\}^{128}$, parse it as 16 bytes $x_1, \dots, x_{16}$ and then set
$$F_k(x) = f_1(x_1) || \cdots || f_{16}(x_{16})$$
  - Is this a permutation? ✓ $y_1 || \cdots || y_{16}$ — is it (efficiently) invertible.
  - Is this indistinguishable from a random permutation?

Attack:

$x_1 = 0 \cdots 0 \; 1 \cdots 1$ (8 | 128 − 8) → $y_1 \, y_2 \cdots y_{16}$

$x_2 = 0 \cdots 0 \; 0 \cdots 0$ → $y_1 \, y_2' \cdots y_{16}'$

full key recovery?

Shannon's Confusion-Diffusion Paradigm

Above step is called the "confusion" step. It is combined with a "diffusion" step: The bits of the output are permuted or "mixed," using a mixing permutation.

- Confusion/Diffusion steps taken together are called a round
- Multiple rounds required for a secure block cipher

Example: First compute intermediate value $y = f_1(x_1)||\cdots||f_{16}(x_{16})$. Then permute the bits of y.

$S_1(x_1)||\cdots||S_{16}(x_{16})$

Substitution-Permutation Network (SPN)

In practice, round-functions are not random permutations, since it would be difficult to implement this in practice.

- Why?

- Instead, round functions have a specific form:
- Rather than have a portion of the key k specify an arbitrary permutation f, we instead fix a public "substitution function" (i.e. permutation) S, called an S-box.
- Let k define the function f given by $f(x) = S(k \oplus x)$.

Informal Description of SPN

1. Key mixing: Set $x := x \oplus k$, where k is the current round sub-key.
2. Substitution: Set $x := S_1(x_1)|| \cdots ||S_8(x_8)$, where x_i is the i-th byte of x.
3. Permutation: Permute the bits of x to obtain the output of the round.
4. Final mixing step: After the last round there is a final key-mixing step. The result is the output of the cipher.
   - Why is this needed?

- Different sub-keys (round keys) are used in each round.
  - Master key is used to derive round sub-keys according to a key schedule.

Formal Description of SPN

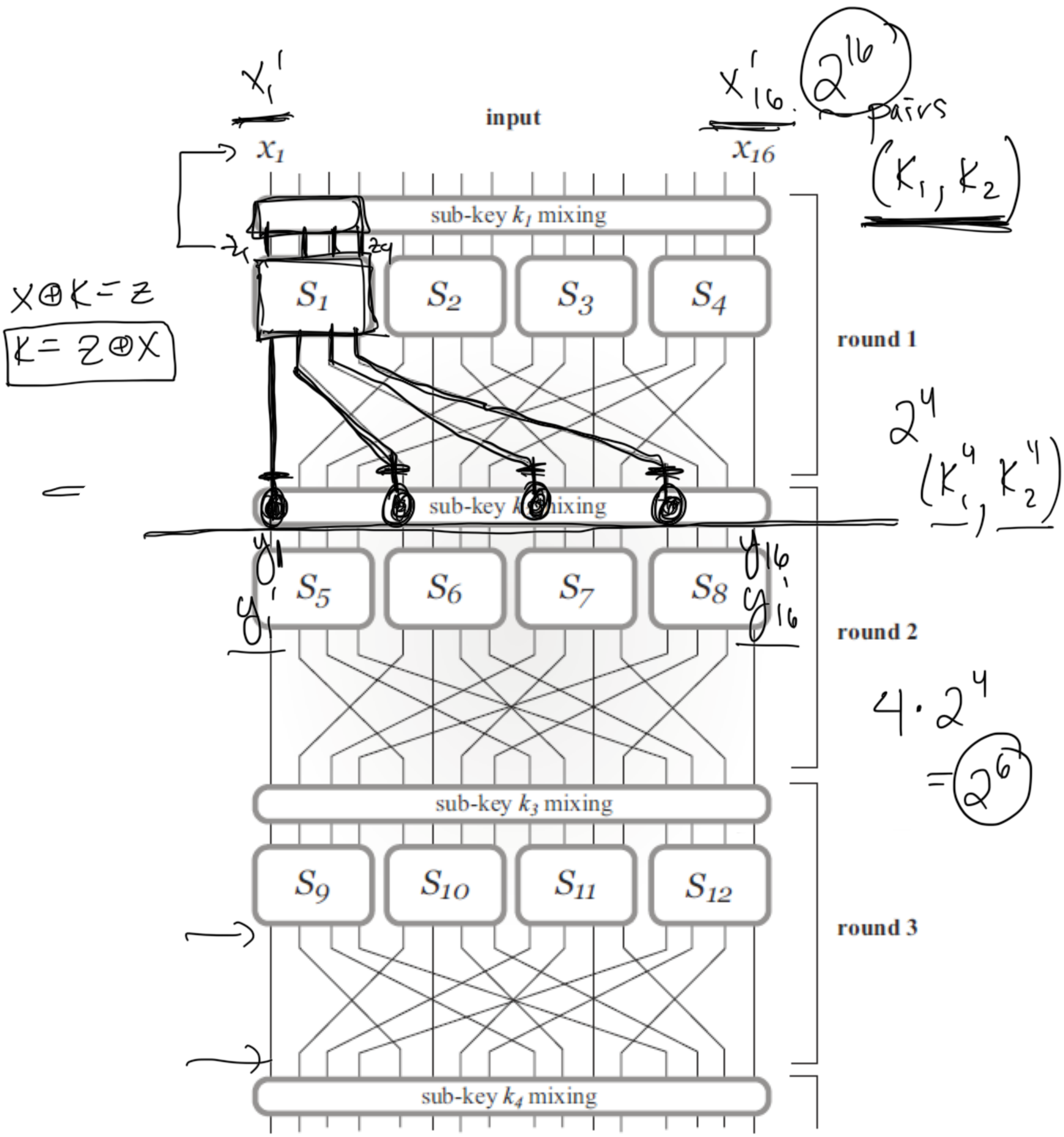


FIGURE 6.2: A substitution-permutation network.

SPN is a permutation

Proposition: Let F be a keyed function defined by an SPN in which the S-boxes are all permutations. Then regardless of the key schedule and the number of rounds, F_k is a permutation for any k.

How many rounds needed for security?

The avalanche effect.

Random permutation: When a single input bit is changed to go from x to x', each bit of $f(x)$ should be flipped with probability ½.

- S-boxes are designed so that changing a single bit of the input to an S-box changes at least two bits in the output of the S-box.
- The mixing permutations are designed so that the output bits of any given S-box are used as input to multiple S-boxes in the next round.

The Avalanche Effect

$f(x)$ vs. $f(x')$ where x, x' differ in one bit:

1. After the first round the intermediate values differ in exactly two bit-positions. Why?
2. The mixing permutation spreads these two bit positions into two different S-boxes in the second round.
   - At the end of the second round, intermediate values differ in 4 bits.
3. Continuing the same argument, we expect 8 bits of the intermediate value to be affected after the 3rd round, 16 after the 4th round, and all 128 bits of the output to be affected at the end of the 7th round.

Practical SPN

- Usually use more than 7 rounds
- S-boxes are NOT random permutations.

changing 1 bit of input → changing 2 bits of output

Attacking Reduced-Round SPN

Trivial case: Attacking one round SPN with no final key-mixing step.

Attacking Reduced-Round SPN

One-round SPN: 16-bit block length. S -boxes with 4-bit input. Independent, 16-bit subkeys.

First attempt at attack:

- Give an input/output pair (x, y)
- Enumerate over all possible values for the second-round subkey k_2.
- For each such value, invert the final key-mixing step to get a candidate output y'.
- Given (x, y'), the first round subkey k_1 is determined.
- Use additional input-output pairs to determine the correct $(k_1||k_2)$ pair.

How long does this attack take?

Attacking Reduced-Round SPN

One-round SPN: 16-bit block length. S -boxes with 4-bit input. Independent, 16-bit

subkeys. Improved attack—work byte-by-byte:

- Given an input/output pair (x, y)
- Enumerate over all possible values for the 4 bit positions corresponding to the output of the first S -box for the second-round subkey k_2.
- For each such value, invert the final key-mixing step to get a candidate 8-bt output y'.
- Given (x, y') the first 4-bits of the first-round subkey k_1 are determined.
- Construct a table of 2^4 possible key values for each block of 4-bits of k_1, k_2.
- Use additional input-output pairs to determine the correct 4-bits of k_1 and first 4 bits of k_2.

How long does this attack take? $4 \cdot 2^4 = 2^6$.

Can be improved: Use additional input/output pairs. Incorrect pair $(k_1 || k_2)$ will work on two pairs with probability 2^{-4}. Can use small number of input/output pairs to narrow down all tables to a single value each at which point the entire master key is known. In expectation, a single additional pair will reduce each table to a single consistent key value.

Lessons Learned

It should not be possible to work independently on different parts of the key.

More diffusion is required. More rounds are necessary to achieve this.

Sample Quiz Questions:

Given a block cipher with key length 256 and block length 128, how many pairs of input/outputs are expected to be needed to eliminate all keys but 1 in a brute force search?

$$2^{256} \cdot \boxed{\frac{1}{2^{128}}} \cdot \boxed{\frac{1}{2^{128}}} = 1$$

Given the avalanche effect, what is the minimum number of rounds needed for an SPN with block length 64?

$$\log_2 64 = 6 \text{ rounds.}$$

$X_1 \; X_2 \ldots$

$\bar{X}_1 \; X_2 \ldots$

Feistel Networks
An alternative approach to Block Cipher Design

Feistel Networks

- The underlying round functions do not need to be invertible.
- Feistel network allows us to construct an invertible function from non-invertible components.
- With enough rounds, can construct a PRP from a PRF.

(Balanced) Feistel Network

- The ith round function $\hat{f}_i$ takes as input a sub-key k_i and an $\ell/2$-bit string and outputs an $\ell/2$-bit string.
- Master key k is used to derive sub-keys for each round.
- Note that the round functions $\hat{f}_i$ are fixed and publicly known, but the $f_i(R) \coloneqq \hat{f}_i(k_i, R)$ depend on the master key and are not known to the attacker.

i-th Feistel Round

- If the block length of the cipher is ℓ bits, then L_{i-1} and R_{i-1} each has length $\ell/2$.
- The output (L_i, R_i) of the round is:

$L_i \coloneqq R_{i-1}$ and $R_i \coloneqq L_{i-1} \oplus f_i(R_{i-1})$

A three-round Feistel Network

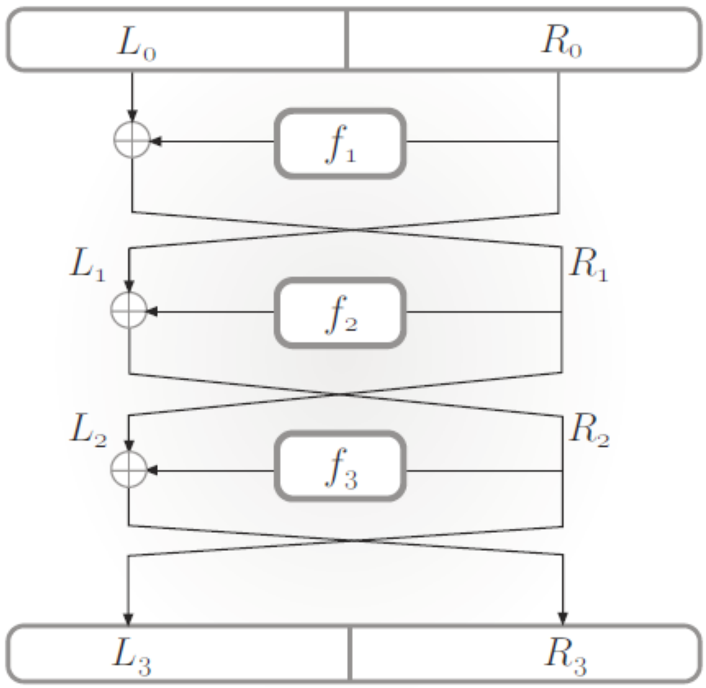


FIGURE 6.4: A 3-round Feistel network.

Feistel Networks are invertible

Proposition: Let F be a keyed function defined by a Feistel network. Then regardless of the round functions $\{\hat{f}_i\}$ and the number of rounds, F_k is an efficiently invertible permutation for all k.