Substitution-Permutation Networks (SPNs)
Recall.

- Want keyed permutation

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

\( n = \text{key length}, \ \ell = \text{block length} \)

- Want \( F_k \) (for uniform, unknown key \( k \)) to be indistinguishable from a uniform permutation over \( \{0, 1\}^\ell \)
Designing block ciphers

- If $x$ and $x'$ differ in one bit, what should the relation between $F_k(x)$ and $F_k(x')$ be?
  - How many bits should change (on average)?
Designing block ciphers

- If $x$ and $x'$ differ in one bit, what should the relation between $F_k(x)$ and $F_k(x')$ be?
  - How many bits should change (on average)? $\frac{n}{2}$
  - Which bits should change?
Designing block ciphers

- If $x$ and $x'$ differ in one bit, what should the relation between $F_k(x)$ and $F_k(x')$ be?
  - How many bits should change (on average)? $n/2$
  - Which bits should change? unpredictable

- How to achieve this?
Confusion/Diffusion

- Confusion
  - Small change in input should result in local, “random” change in output

- Diffusion
  - Local change in output should be propagated to entire output
Substitution-Permutation Networks (SPNs)

- Build random-looking perm on large input from rand perms on small inputs
- E.g. assume 8-byte block length
- 
  \[ F_k(x) = f_{k1}(x_1)f_{k2}(x_2) \ldots f_{k8}(x_8) \]
  
  where each \( f_{ki} \) is a random permutation of \( n/8 \) numbers.
- Need \( k \) to code 8 perms of \( n/8 \) numbers. Clunky.
  Need the perms to be fast AND random-looking. Hard!
  **Punchline:** Won’t be using this but pretend for now to see what we aspire to.
Substitution-Permutation Networks (SPNs)

Is this a pseudorandom function? Vote
Substitution-Permutation Networks (SPNs)

Is this a pseudorandom function? Vote \textcolor{red}{No} Too Local
This has confusion but no diffusion. Random-looking locally but not globally.

- Add a *mixing permutation*...
SPN
Invertibility

- Note that the structure is invertible (given the key) since the $f$s are permutations
Mixing permutation is public
  - Chosen to ensure good diffusion

Does this give a pseudorandom function?

What if we repeat for another round (with independent, random functions)?
  - What is the minimal \# of rounds we need?
  - **Avalanche effect**: Small change in input leads to global change.
1. From key $k$ get 8 random perms on $n/8$ bit
2. $F_k(x) = f_{k1}(x_1) \cdots f_{k8}(x_8)$ where $x = x_1 \cdots x_8$.
3. Permute the blocks.
4. Lather, Rinse, Repeat many times.

**PRO:** Provably gives pseudorandom perm

**CON:** Hard to generate fast random perms.
Key will not code perms. Key will be $k = k_1 \cdots k_{n/8}$ and $k_i$’s will be used along with public S-box to create perms.

- $f_{k_i}(x) = S_i(k_i \oplus x)$, where $S_i$ is a public permutation
- $S_i$ are called “S-boxes” (substitution boxes)
- XORing the key is called “key mixing”
- Note that this is still invertible (given the key)
Avalanche effect

- Design S-boxes and mixing permutation to ensure avalanche effect
  - Small differences should eventually propagate to entire output

- S-boxes: 1-bit input change $\implies \geq 2$-bit output change

- Mixing permutation
  - Each bit output from a given S-box should feed into a different S-box in the next round
S-Boxes are HARD to Create

Building them is a major challenge.

Titles of Papers that tried:

*The Design of S-Boxes by Simulated Annealing*

*A New Chaotic Substitution Box Design for Block ciphers*

*Perfect Nonlinear S-Boxes*
S-Boxes are HARD to Create

Building them is a major challenge.

Titles of Papers that tried:

*The Design of S-Boxes by Simulated Annealing*

*A New Chaotic Substitution Box Design for Block ciphers*

*Perfect Nonlinear S-Boxes*

If you type in S-Boxes into Google Scholar how many papers to you find?
S-Boxes are HARD to Create

Building them is a major challenge.

Titles of Papers that tried:

*The Design of S-Boxes by Simulated Annealing*

*A New Chaotic Substitution Box Design for Block ciphers*

*Perfect Nonlinear S-Boxes*

If you type in **S-Boxes** into Google Scholar how many papers to you find?

20,000. Given repeats and conference-Journal repeats, there are approx 10,000 papers on S-boxes.
PLAINTEXT

$S_1 \quad S_2 \quad S_3 \quad S_4$

$P$

KEY

$K_0$

$K_1$

$K_2$

$K_3$

CIPHERTEXT
One round of an SPN involves

- Key mixing
  - Ideally, round keys are independent
  - In practice, derived from a master key via *key schedule*
- Substitution (S-boxes)
- Permutation (mixing permutation)

*r*-round SPN has *r* rounds as above, plus a final key-mixing step

- Why?

Since S-boxes and Perms are invertible and public, if there was no final key-mixing stage then the last stage would be pointless.
Key-Recovery Attacks

- Key-recovery attacks are even more damaging than distinguishing attacks
  - As before, a cipher is secure only if the best key-recovery attack takes time $\approx 2^n$
  - A fast key-recovery attack represents a complete break of the cipher
Key-recovery attack, 1-round SPN

Consider case where there is no final key-mixing step.

1. Public input $x_1$
2. Then get $x_2 = k \oplus x_1$ where $k$ is private
3. Then get $x_2$ and do $S$-box stuff to it, and Perm to it, to get $x_3$

If see all of this then Eve knows $x_1, x_3$. Can she find $k$? **Discuss**
Consider case where there is no final key-mixing step.

1. Public input \( x_1 \)
2. Then get \( x_2 = k \oplus x_1 \) where \( k \) is private
3. Then get \( x_2 \) and do S-box stuff to it, and Perm to it, to get \( x_3 \)
4. Output \( x_3 \). Public.

If see all of this then Eve knows \( x_1, x_3 \). Can she find \( k \)? Discuss

Yes
1) From \( x_3 \) can find \( x_2 \) since S-box stuff and Perm are all invertible.
2) Compute \( x_1 \oplus x_2 = x_1 \oplus x_1 \oplus k = k \)
Key-recovery attack, 1-round SPN

There is a final key-mixing step. Key $k_1, k_2 \in \{0, 1\}^{n/2}$.

1. Public input $x_1$
2. $x_2 = k_1 \oplus x_1$ where $k_1$ is private
3. $x_2$ and do S-box stuff to it, and Perm to it, to get $x_3$
4. Output $x_4 = x_3 \oplus k_2$ where $k_2$ is private.

Eve sees $x_1, x_4$.
For each $(k_1, k_2)$ see if $x_1, x_4$ is consistent with it. There may be many candidates. As Eve sees more input-output pairs she can zero in on the right candidate with roughly $2^n$ input-output pairs. Can Eve do better?
Discuss
Key-recovery attack, 1-round SPN

There is a final key-mixing step. Key \( k_1, k_2 \in \{0, 1\}^{n/2} \).

1. Public input \( x_1 \)
2. \( x_2 = k_1 \oplus x_1 \) where \( k_1 \) is private
3. \( x_2 \) and do S-box stuff to it, and Perm to it, to get \( x_3 \)
4. Output \( x_4 = x_3 \oplus k_2 \) where \( k_2 \) is private.

Eve sees \( x_1, x_4 \).
For each \( (k_1, k_2) \) see if \( x_1, x_4 \) is consistent with it. There may be many candidates. As Eve sees more input-output pairs she can zero in on the right candidate with roughly \( 2^n \) input-output pairs. Can Eve do better?

Discuss

For each \( k_2 \in \{0, 1\}^{n/2} \) view the SPN as in prior slide- no last key-mixing stage. Hence can derive \( k_1 \). Have only \( 2^{n/2} \) candidates. Eve needs only \( 2^{n/2} \) input-output pairs.
Key-recovery attack, 1-round SPN, Better Attack

Work S-box-by-S-box. Assume $\frac{n}{a}$ a-to-a S-boxes.
Each guess of the first $a$ bits of $k_2$ determines some $a$ bits of $k_1$.
So have $2^a$ possibilities for $2a$-bits
Do this for first, second, ..., $\frac{n}{a}$ part of $k_2$
This took time
$$2^a + 2^a + \cdots + 2^a = \frac{n2^a}{a}$$
$\frac{n}{a}$ times
steps. Still have $2^{n/2}$ possibilities for the key but took less time to find them.

Given an input-output pair it will likely eliminate many of the
1) Can extend to $r$ rounds but time complexity goes up.
2) Better than naive but still too slow.
3) Considered secure if $r$ is large enough.
4) AES uses 8-bit S-boxes and at least 9 rounds (and other things) and is thought to be secure.
1) Can extend to $r$ rounds but time complexity goes up.
2) Better than naive but still too slow.
3) Considered secure if $r$ is large enough.
4) AES uses 8-bit S-boxes and at least 9 rounds (and other things) and is thought to be secure. For now.