Note:
- RSA conference, Discussion of dual-ec-drbg

Project Idea:
- "Security of routers"

Today: Block Ciphers
- DES (Data Encryption Standard)
- AES (Advanced Encryption Standard)
- Ideal Block Cipher
- "Modes of Operation," ECB, CTR, CBC, CFB
- IND-CCA security definition
- UFE mode
Block ciphers:

\[ \text{key } k \rightarrow \text{Enc} \rightarrow c \]

- Plaintext block
- Cipher text block

Fixed-length P, C, K:

- DES: \( |P| = |C| = 64 \text{ bits} \)  \( |K| = 56 \text{ bits} \)
- AES: \( |P| = |C| = 128 \text{ bits} \)  \( |K| = 128, 192, 256 \text{ bits} \)

Use a "mode of operation" to handle variable-length input.
DES

"Data Encryption Standard"
Standardized in 1976. Now deprecated in favor of AES.

"Feistel structure"

16 "rounds" total

one "round"

plaintext 64 bits

"round key"

all 16 round keys derived from 64-bit encryption key (only 56 bits are really used) via "key schedule"

ciphertext

Note: Invertible for any \( f \) and any key schedule.

\( f \) uses 8 "S-boxes" mapping 6-bits \( \rightarrow \) 4 bits non-linearly.

Key is too short! (Breakable now quite easily by brute-force.)

Subject to differential attacks:

\[
\begin{align*}
M & \rightarrow M \oplus A \\
L & \rightarrow DBS \\
K & \rightarrow DBS \\
C & \rightarrow C + 8
\end{align*}
\]

\( 2^{47} \) chosen pairs (Biham/Shamir)

Subject to linear attacks:

e.g., if \( M_3 \oplus M_5 \oplus C_9 \oplus K_{14} = 0 \) (e.g. on bits)

with prob \( p = \frac{1}{2} + \epsilon \)

then need \( \sqrt{\epsilon^2} \) samples to break (Matsui, \( 2^{43} \) PT/LT pairs)
AES

“Advanced Encryption Standard” (U.S. govt)

Replaces DES

AES "contest" 1997-1999:
15 algorithms submitted: RC6, Mars, Twofish, Rijndael, ...
Winner = Rijndael (by Joan Daemen & Vincent Rijmen, (Belgians))

Specs: 128-bit plain/text/cipher text blocks
128, 192, or 256-bit key
10, 12, or 14 rounds (dep. on key length)

Byte-oriented design (some math done in Galois field \( GF(2^8) \))

View input as 4x4 byte array:

\[
\begin{array}{cccc}
\text{4x4x8 = 128}
\end{array}
\]

For version with 128-bit keys, 10 rounds:

- Derive 11 "round keys", each 128 bits (4x4x byte)

- In each round:
  1. XOR round key
  2. Substitute bytes (lookup table)
  3. Rotate rows (by different amts)
  4. Mix each column (by linear opn)

- Output final state

See reading for details.
There are very fast implementations. Also Intel has put
supporting hardware into its CPUs.

Security: Good - perhaps # rounds should be a bit larger...
For practical purposes, can treat AES as ideal block cipher:

For each key, mapping $\text{Enc}(K, \cdot)$ is a random independent permutation of $\{0,1\}^b$ to itself.

Modes of Operation:

How to encrypt variable-length messages? (using AES)

- "ECB" = "Electronic code book"
- "CTR" = "Counter mode"
- "CBC" = "Cipher-block chaining" (CBC-MAC)
- "CFB" = "Cipher feedback"

ECB:

\[
\begin{align*}
\text{M}_1 &\rightarrow \text{E} & \text{C}_1 \\
\text{M}_2 &\rightarrow \text{E} & \text{C}_2 \\
&\vdots & \vdots \\
\text{M}_n &\rightarrow \text{E} & \text{C}_n
\end{align*}
\]

To handle data that is not a multiple of $b$ bits in length:

(1) Append a "1" bit (always)
(2) Append enough "0" bits to make length a multiple of $b$ bits.

This gives invertible $(a+1)$ "padding" operation. Pad before encryption; unpad after decryption.

ECB preserves many patterns: repeated message blocks $\Rightarrow$ repeated ciphertext blocks

ECB really only good for encrypting random data (e.g. keys)