Daily Notes:

Project!
Post #2 out

Today:

- AES
- Ideal cipher
- Modes of operation (ECB, CTR, CBC)
- IND-CCA security
- MACs
- Combining Enc & MACs - authenticated encryption modes

Readings:

- Katz - Chapter 6 & 4
- Wikipedia - AES
AES

“Advanced Encryption Standard” (U.S. gov't)

Replaces DES

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

Specs:
- 128-bit plaintext/ciphertext 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 $4 \times 4$ byte array:

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$4 \times 4 \times 8 = 128$

For version with 128-bit keys, 10 rounds:

- Derive 11 “round keys”, each 128 bits ($4 \times 4 \times 8$ 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 CPU's.

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

- For each key, mapping $\text{Enc}(K, \cdot)$ is a random independent permutation of $\{0, 1\}^{128}$ to itself.

**Modes of Operation:**

- How to encrypt variable-length messages? (using AES)
  - "ECB" = "Electronic code book"
  - "CTR" = "Counter mode"
  - "CBC" = "Cipher-block chaining" (CMC/CBC-MAC)
  - "CFB" = "Cipher feedback" (others...)

**ECB:**

- Divide data into $b$-bit blocks, where $b = \text{input block size}$
- To handle data that is not a multiple of $b$ bits in length:
  - Append a "1" bit (always)
  - Append enough "0" bits to make length a multiple of $b$ bits.
- This gives invertible (1-1) "padding" operation.
- Pad before encryption; unpadd after decryption, always!

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

**ECB really only good for encrypting random data**
- (e.g. keys)
CTR (Counter mode):

Generate a PR (pseudo random) sequence by encrypting $i, (i+1), \ldots$

XOR with message to obtain ciphertext.

\[
\begin{align*}
K &\rightarrow E & K &\rightarrow E & K &\rightarrow E \\
E &\downarrow & E &\downarrow & E &\downarrow \\
X_i &\rightarrow & X_{i+1} &\rightarrow & X_{i+2} & \rightarrow \\
\downarrow & & \downarrow & & \downarrow & \leftarrow \text{"padded" like OTP} \\
M_1 &\rightarrow \oplus & M_2 &\rightarrow \oplus & M_3 &\rightarrow \oplus \\
\downarrow & & \downarrow & & \downarrow & \\
C_1 & & C_2 & & C_3 & \\
\end{align*}
\]

Initial counter value can be transmitted first:

$i, C_1, C_2, \ldots$

Of course, no counter value should be re-used!

Message does not need to be padded to be a multiple of block length

\[|C| = |M| \quad \text{(not counting initial counter value)}\]