Administrivia - see TA's for "group issues"
- PS #1 due this week
- PS #2 out...

Outline
- Encryption
- One-time pad (OTP) & generating random bits
- RC4 (stream cipher)
- Block ciphers
- DES
- *AES*
- modes of operation
Encryption (classical) goal: confidentiality

\[ C = E_k(M) \]

- Bob must know something Eve doesn't
- Eve can't tell \( E_k(M_1) \) from \( E_k(M_2) \), even if she knows (or chooses) \( M_1 \) and \( M_2 \) (note: length!)

\[ |M_1| = |M_2| \]

- attacks:
  - known ciphertext
  - known PT/CT pairs (same key)
  - chosen PT

these assume key is re-used.

But let's do simpler situation first: OTP one-time pad
no key reuse (assumes synchronization...)
One-Time Pad (OTP)

- How it works

\[ \text{Message} = 101100 \ldots \]

\[ \text{Pad} = 011010 \ldots \quad (\text{mod-2 \ addition \ each \ column}) \]

\[ \text{Ciphertext} = 110110 \ldots \]

Pad is \underline{random \& secret} (known only to sender \& receiver). Note:
used by Russian spies (silk handkerchiefs, small pads... )
Joke: only changes 1/2 the bits - information leaks through!
why not change all the bits? (Desmedt rump session talk)

\underline{Proof of Security:} [Note: instantaneous - all c's equal likely, or
\[ P(M) = \text{probability of message } M \text{ (may be non-uniform)} \]
\[ = \text{adversary's initial information about } M \]
\[ P(\text{Pad}) = \text{probability at pad } = 2^{-n} \text{ for } n\text{-bit pad} \]
\[ P(C) = \text{probability of observing } c, \text{ given ciphertext} \]
\[ P(C|M) = \text{probability of observing } C, \text{ given } M \text{ = message} \]
\[ = \text{probability } \text{Pad} = C \oplus M \]
\[ = 2^{-n} \]
\[ P(C) = \sum P(C|M) \cdot P(M) \]
\[ = \sum 2^{-n} \cdot P(M) = 2^{-n} \cdot \sum P(M) = 2^{-n} \]

uniform
\[ P(M | C) = \frac{P(C | M) \cdot P(M)}{P(C)} \quad \text{(Bayes' Rule)} \]

\[ = 2^{-n} \cdot P(M) = P(M) \quad \text{No change!} \]

\[ \Downarrow \]

Adversary learns nothing!

\[ \Downarrow \]

Perfect security!

**Note:** Unconditionally secure \( \equiv \) no assumptions made or needed about adversary's computing power

\[ \text{(compare to computationally secure \( \equiv \) assume adversary has limits to his computing ability, } \]

\[ \text{e.g., both for an only computationally secure).} \]

Users need to:
- Generate large secrets
- Share them
- Keep them secret
- Avoid re-using them! (HW)

\[ \text{Note: OTP is malleable! (non-malleability not a requirement)} \]
How to generate pads?

- Coins, dice
- Radioactive sources (old memory chips?)
- Microphone
- Dish speed variations
- Intel chip set
- Back-biased diodes
- User mouse/typing
- Lavarand (lava lamp)
- Alpert & Schneider

A \rightarrow \text{eve} \rightarrow B

eve can't tell who sent message.
A & B randomly emit beeps
they get shared secret

Quantum Key Distribution

A \rightarrow \text{filter} \leftrightarrow \text{filter} \rightarrow + \text{filter}

A sends
B publishes around filter choice
this is known which bits they have should have in common.
- Satellite Based:

- Broadcast

- Idea: channels may be noisy, independent

- Eve may have limited memory

- Book-based pad:
  - take e.g. 4 or 5 texts (starting point)
  - XOR together

- "Pseudo-random" pad

  "Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin."

  (John von Neumann, 1951)
Pseudo-random pad (e.g., RC4)

pseudo-random generator:

\[
\begin{array}{c}
\text{seed} \rightarrow \text{pad} \\
\text{256 bits} \quad \text{arbitrarily long...}
\end{array}
\]

adversary can't tell PR pads from truly random pads of same length...

RC4: table \( S[0..255] \), permutation of \( 0..255 \) (init-from key)

\( i, j \in [0..255], 0 \leq i, j \leq 255 \) (points into table)

\[
\begin{align*}
i &= 0 \\
j &= 0 \\
\text{while true do} \\
&\quad \left[ \begin{array}{l}
\quad i = (i + 1) \mod 256 \\
\quad j = (j + S[i]) \mod 256 \\
\quad \text{swap} \left( S[i], S[j] \right) \\
\quad \text{output} \ S \left[ (S[i] + S[j]) \mod 256 \right]
\end{array} \right]
\end{align*}
\]

widely used

\[
\left[ \begin{array}{l}
\text{setup of } S \text{ from key is weak; good idea to} \\
\quad \text{discard first 1024 bytes of output...}
\end{array} \right]
\]

\[
\text{Note bug implementing } A \leftrightarrow B \text{ as } A = A \oplus B \\
\quad B = B \oplus A \\
\quad A = A \oplus B
\]

\[\text{doesn't work if } A, B \text{ the same variable - sets table } S \text{ to 0's!} \]
Block ciphers:

\[
\begin{align*}
\text{key } k & \rightarrow \text{P plaintext block} \\
& \downarrow \\
& \text{C ciphertext block}
\end{align*}
\]

Fixed length of P, C

- **DES:** |P| = |C| = 64 bits
- **AES:** |P| = |C| = 128 bits

Fixed length of K

- **DES:** |K| = 56 bits
- **AES:** |K| = 128, 192, or 256 bits

Then use "mode of operation" to handle variable-length input.
DES

standardized in 70's, now deprecated in favor of AES

Structure "Feistel design", roughly:

- One round
- X 16

- Note: invertible for any f or key schedule
- f uses 8 "S-boxes" mapping 6 bit \( \rightarrow \) 4 bit for nonlinearity
- key is too short! (breachable now in a few hours...)
- Subject to differential attacks
- Linear attacks

\[ M_3 \oplus M_5 \oplus C_2 \oplus k_{14} = 0 \]

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

need \( \frac{1}{\epsilon^2} \) samples to figure out \( k_{14} \)

- "Round keys" derived from key K (e.g. subset of key bits)
- "Key schedule"