Reminders

HW3 Due Today

PR1 Due **March 12th (Thursday)**
• No late days
Previously on COS 433...
Confusion/Diffusion Paradigm
Confusion/Diffusion Paradigm

Third Attempt: Repeat multiple times!
Substitution Permutation Networks

Round

Round key

Potentially different

Final key mixing
AES

One fixed S-box, applied to each byte
• Step 1: multiplicative inverse over finite field \( \mathbb{F}_8 \)
• Step 2: fixed affine transformation
• Implemented as a simple lookup table
AES

Diffusion (not exactly a P-box):
• Step 1: shift rows
• Step 2: mix columns
AES

Shift Rows:
AES

Mix Columns

• Each byte interpreted as element of $\mathbb{F}_8$
• Each column is then a length-4 vector
• Apply fixed linear transformation to each column
Feistel Networks
Feistel Network

Convert functions into permutations

\[ F \oplus k_0, F \oplus k_1 \]

\( F \): round function
\( k_0, k_1 \): round keys
Luby-Rackoff

3- or 4-round Feistel where round function is a PRF

**Theorem:** If $F$ is a secure PRF, then 3 rounds of Feistel (with independent round keys) give secure PRP. 4 rounds give a strong PRP

- Proof non-trivial, won’t be covered in this class
Today

Constructing block ciphers part 2
Attacks on block ciphers
Constructing Round Functions

Ideally, ”random looking” functions

Similar ideas to constructing PRPs
• Confusion/diffusion
• SPNs, S-boxes, etc

Key advantage is that we no longer need the functions to be permutations
• S-boxes can be non-permutations
DES

Block size: 64 bits
Key size: 56 bits
Rounds: 16
DES

Key Schedule:
• Round keys are just 48-bit subsets of master key

Round function:
• Essentially an SPN network
DES S-Boxes

8 different S-boxes, each
• 6-bit input, 4-bit output
• Table lookup: 2 bits specify row, 4 specify column

• Each row contains every possible 4-bit output
• Changing one bit of input changes at least 2 bits of output
DES History

Designed in the 1970’s
- At IBM, with the help of the NSA
- At the time, many in academia were suspicious of NSA’s involvement
  - Mysterious S-boxes
  - Short key length
- Turns out, S-box probably designed well
  - Resistant to “differential cryptanalysis”
  - Known to IBM and NSA in 1970’s, but kept secret
- Essentially only weakness is the short key length
  - Maybe secure in the 1970’s, definitely not today
DES Security Today

Seems like a good cipher, except for its key length and block size

What’s wrong with a small block size?
• Remember for e.g. CTR mode, IV is one block
• If two identical IV’s seen, attack possible
• After seeing $q$ ciphertexts, probability of repeat IV is roughly $q^2/2^{\text{block length}}$
• Attack after seeing $\approx$ billion messages
3DES: Increasing Key Length

3DES key = Apply DES three times with different keys

Why three times?
• Later: “meet in the middle attack” renders 2DES no more secure than 3DES

Why inverted second permutation?
Attacks on block ciphers
Brute Force Attacks

Suppose attacker is given a few input/output pairs

Likely only one key could be consistent with this input/output

Brute force search: try every key in the key space, and check for consistency

Attack time: $2^{\text{key length}}$
Insecurity of 2DES

DES key length: 56 bits
2DES key length: 112 bits
Brute force attack running time: $2^{112}$
Meet In The Middle Attacks

For 2DES, can actually find key in $2^{56}$ time

• Also $\approx 2^{56}$ space
Meet In The Middle Attacks

\[ d = \text{DES}(k_0, m) \]

\[ d = \text{DES}^{-1}(k_1, m) \]

<table>
<thead>
<tr>
<th>( k_0 )</th>
<th>( d = \text{DES}(k_0, m) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>52</td>
</tr>
<tr>
<td>1</td>
<td>93</td>
</tr>
<tr>
<td>2</td>
<td>03</td>
</tr>
<tr>
<td>3</td>
<td>96</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>5</td>
<td>49</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( k_1 )</th>
<th>( d = \text{DES}^{-1}(k_1, m) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>69</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>86</td>
</tr>
<tr>
<td>3</td>
<td>49</td>
</tr>
<tr>
<td>4</td>
<td>99</td>
</tr>
<tr>
<td>5</td>
<td>08</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>
Meet In The Middle Attacks

Complexity of meet in the middle attack:
• Computing two tables: time, space $2 \times 2^{\text{key length}}$
• Slight optimization: don’t need to actually store second table

On 2DES, roughly same time complexity as brute force on DES
MITM Attacks on 3DES

MITM attacks also apply to 3DES...

\[
m \xrightarrow{k_0} \text{DES} \xrightarrow{k_1} \text{DES} \xrightarrow{k_2} c
\]
MITM for 3DES

\[ m \xrightarrow{k_0} \text{DES} \xrightarrow{k_1} \text{DES} \xrightarrow{m,c} \xrightarrow{k_2} \text{DES}^{-1} \xrightarrow{c} \]

<table>
<thead>
<tr>
<th>(k_0)</th>
<th>(k_1)</th>
<th>(d = \text{DES}(k_0,m))</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>52</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>93</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>03</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>96</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>20</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>49</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(k_2)</th>
<th>(d = \text{DES}^{-1}(k_2,m))</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>69</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>86</td>
</tr>
<tr>
<td>3</td>
<td>49</td>
</tr>
<tr>
<td>4</td>
<td>99</td>
</tr>
<tr>
<td>5</td>
<td>08</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
MITM for 3DES

No matter where “middle” is, need to have two keys on one side
• Must go over $2^{112}$ different keys

Space?

While 3DES has 168 bit keys, effective security is 112 bits
Generalizing MITM

In general, given $r$ rounds of a block cipher with $t$-bit keys,

• Attack time: $2^{t\lceil r/2 \rceil}$

• Attack space: $2^{t\lfloor r/2 \rfloor}$
Brute Force vs. Generic Attacks

MITM attacks on iterated block ciphers are generic
  • Attack exists independent of implementation details of block cipher

However, still beats a brute force
  • Doesn’t simply try every key
MITM Attacks

MITM attacks can also be applied to plain single block ciphers

Can yield reasonable attacks in some regimes
Time-Space Tradeoffs

MITM attack requires significant space

In contrast, brute force requires essentially no space, but runs slower

Known as a time-space tradeoff
Another Time-Space Trade-off Example

Given $y = F(k,x)$, find $x$

- Allowed many queries to $F(k,x)$ oracle
  (That is, standard block cipher oracle)
- Assume $|k| >> |x|$

Option 1:
- Brute force search over entire domain looking for $x$
- Time: $2^n$ ($n=|x|$)
- Space: 1
Another Time-Space Trade-off Example

Given $y = F(k, x)$, find $x$

- Allowed many queries to $F(k, x)$ oracle
  (That is, standard block cipher oracle)
- Assume $|k| >> |x|$

Option 2: Preprocessing

- Before seeing $y$, compute giant table of $(x, F(k, x))$ pairs, sorted by $F(k, x)$
- Preprocessing Time: $2^n$
- Space: $2^n$
- Online time: ?
Option 3: Hellman’s Attack

For simplicity, assume $F(k,\bullet)$ forms a cycle covering entire domain
- $\{0, F(k,0), F(k, F(k,0)), F(k, F(k, F(k,0))),\ldots\} = X$

In preprocessing stage:
- Attacker iterates over entire cycle, saving every $t^{th}$ term in a table $(x_1,\ldots,x_{N/t})$ where $N=2^n$
Option 3: Hellman’s Attack
Option 3: Hellman’s Attack

\[ y = F(k, x) \]
Option 3: Hellman’s Attack

\[ y = F(k, x) \]
Option 3: Hellman’s Attack

\[ y = F(k, x) \]
Option 3: Hellman’s Attack

\[ y = F(k, x) \]
Option 3: Hellman’s Attack

\[ y = F(k, x) \]
Option 3: Hellman’s Attack

Preprocessing Time: \( N = 2^n \)
Space: \( N/t \)
Online Time: \( t \)

Time-space tradeoff: \( space \times online \ time \approx N \)

For non-cycles, attack is a bit harder, but nonetheless possible
Differential Cryptanalysis

Suppose there were $\Delta x, \Delta z$ such that, for random key $k$ and random $x_1, x_2$ such that $x_1 \oplus x_2 = \Delta x$,

$$F(k, x_1) \oplus F(k, x_2) = \Delta z$$

with probability $p \gg 2^{-n}$

- Call $(\Delta x, \Delta z)$ a differential
- $p$ is probability of differential
- $\approx 2^{-n}$ is probability of differential for random permutation

Yields distinguishing attack. With some effort, can also recover secret key
Differential Cryptanalysis

\[ F'(k,x) \]

\[ F(k,x) \]
Differential Cryptanalysis

Attack:
• Suppose we have differential $(\Delta_x, \Delta_y)$ for $F'$
• Choose many random pairs $(x_1, x_2)$ s.t. $x_1 \oplus x_2 = \Delta_x$
• Make queries on each $x_1, x_2$, obtaining $y_1, y_2$
• Guess final round key $k_r'$,
  • Use differentials to determine if guess was correct
Differential Cryptanalysis

\[
\begin{align*}
(y_0^1, y_1^1), & \quad (y_0^2, y_1^2), \\
(y_0^3, y_1^3), & \quad (y_0^4, y_1^4), \\
\ldots
\end{align*}
\]

\[
\begin{align*}
(x_0^1, x_1^1), & \quad (x_0^2, x_1^2), \\
(x_0^3, x_1^3), & \quad (x_0^4, x_1^4), \\
\ldots
\end{align*}
\]

Guess \( k_r' \)
Differential Cryptanalysis

Attack:
• Choose many random pairs \((x_1, x_2)\) s.t. \(x_1 \oplus x_2 = \Delta_x\)
• Make queries on each \(x_1, x_2\), obtaining \(y_1, y_2\)
• For each possible final round key guess \(k_r'\),
  • Undo last round assuming \(k_r'\), obtaining \((z_1', z_2')\)
  • Look for \(z_1' \oplus z_2' = \Delta_z\)
• If right guess, expect \(\approx p\) fraction
• If wrong guess, expect \(\approx 2^{-n}\) fraction
Differential Cryptanalysis

\[ (x_0^1, x_1^1), (x_0^2, x_1^2), (x_0^3, x_1^3), (x_0^4, x_1^4), \ldots \]

\[ (y_0^1, y_1^1), (y_0^2, y_1^2), (y_0^3, y_1^3), (y_0^4, y_1^4), \ldots \]

Guess \( k_r' \)
Differential Cryptanalysis

So far, inefficient since we have to iterate over all $2^n$ possible round keys.

Instead, we can learn $k_r$ 8 bits at a time:

- Guess 8 bits of $k_r$ at a time
- Iterate through all $2^8$ possible values for those 8 bits
  - Compute 8 bits of $z_1', z_2'$, look for (portion of) differential
- Which bits to choose?
Differential Cryptanalysis

\[(x_0^1, x_1^1), (x_0^2, x_1^2), (x_0^3, x_1^3), (x_0^4, x_1^4), \ldots\]

Guess \(k_r'\)

PRP Oracle

\[(y_0^1, y_1^1), (y_0^2, y_1^2), (y_0^3, y_1^3), (y_0^4, y_1^4), \ldots\]
Differential Cryptanalysis

Extending to further levels:

- One $k_r$ is known, can un-compute last layer
- Now perform same attack on round-reduced cipher
- Repeat until all round keys have been found
Finding Differentials

So far, assumed differential given

How do we find it?
• Can’t simply brute force all possible differentials
Finding Differentials

Solution: look for differentials in S-boxes

- Only $2^8$ possible differences, so we can actually look for all possible differentials

- Then trace differentials through the evaluation
  - Key mixing does not affect differentials
  - Diffusion steps just shuffle differential bits
Differential Cryptanalysis
Differential Cryptanalysis in Practice

Used to attack real ciphers
• FEAL-8, proposed as alternative to DES in 1987
  • requires just 1000 chosen input/output pairs, 2 minutes computation time in 1990’s

• Also theoretical attacks on DES
  • Requires $2^{47}$ chosen input/output pairs
  • Very difficult to obtain in real world applications
  • Therefore, DES is still considered relatively secure
  • Small changes to S-boxes in DES lead to much better differential attacks
Linear Cryptanalysis

High level idea: look for linear relationships that hold with too-high a probability

- E.g. $x_1 \oplus x_5 \oplus x_{17} \oplus y_3 \oplus y_6 \oplus y_{12} \oplus y_{21} = 0$

Can show that if happen with too-high probability, can completely recover key

Important feature: only requires known plaintext as opposed to chosen plaintext
- Much easier to carry out in practice
- Ex: DES can be broken with $2^{43}$ input/output pairs
Block Cipher Design

S-boxes are designed to minimize differential and linear cryptanalysis
- Cannot completely remove differentials/linear relations, but can minimize their probability

Increasing number of rounds helps
- Likelihood of differential decreases each round
Related Key Attacks

Properly designed crypto will always use random, independent keys for every application.

However, sometimes people don’t follow the rules.

Related key attack: have messages encrypted under similar keys.

(Recall RC4 used for encryption, $\text{RC4(IV,k)}$)

For AES 256, can attack in $2^{110}$ space/time.
Limitations of Feistel Networks

Turns out Feistel requires block size to be large
• If number of queries $\sim 2^{\text{block size}/2}$, can attack

Format preserving encryption:
• Encrypted data has same form as original
• E.g. encrypted SSN is an SSN
• Useful for encrypting legacy databases

Sometimes, want a very small block size
Holiwudd Criptoe!

Device is top of the line. AES cipher locks, brute force decryption is the only way…. It's effective, but slow. Very slow.
Reminders

HW3 Due Today

PR1 Due March 12th (Thursday)
• No late days