CSE 484: Computer Security and Privacy

Cryptography
[Symmetric Encryption]

Winter 2021

David Kohlbrenner
dkohlbre@cs.washington.edu

Thanks to Franzi Roesner, Dan Boneh, Dieter Gollmann, Dan Halperin, Yoshi Kohno, Ada Lerner, John Manferdelli, John Mitchell, Vitaly Shmatikov, Bennet Yee, and many others for sample slides and materials...
Admin

• Lab 1 ongoing

• Homework 2 (crypto) will be out soon
  • Due on 2/10 (designed to give you hands-on experience with crypto concepts, not be tricky – not intended to take you a full 2 weeks)

• Watching the recording? Please fill out: https://forms.gle/5qoJSfxap3mmBsJk6
Reducing Key Size

• What to do when it is infeasible to pre-share huge random keys?
  • When one-time pad is unrealistic...

• Use special cryptographic primitives: block ciphers, stream ciphers
  • Single key can be re-used (with some restrictions)
  • Not as theoretically secure as one-time pad

56 / 128/256
6.4
Block Ciphers

- Operates on a single chunk ("block") of plaintext
  - For example, 64 bits for DES, 128 bits for AES
  - Each key defines a different permutation
  - Same key is reused for each block (can use short keys)
Keyed Permutation

<table>
<thead>
<tr>
<th>input</th>
<th>possible output</th>
<th>possible output</th>
<th>etc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>010</td>
<td>111</td>
<td>...</td>
</tr>
<tr>
<td>001</td>
<td>111</td>
<td>110</td>
<td>...</td>
</tr>
<tr>
<td>010</td>
<td>101</td>
<td>000</td>
<td>...</td>
</tr>
<tr>
<td>011</td>
<td>110</td>
<td>101</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>111</td>
<td>000</td>
<td>110</td>
<td>...</td>
</tr>
</tbody>
</table>

For N-bit input, $2^N!$ possible permutations
For K-bit key, $2^K$ possible keys

Key = 00
Key = 01

For N-bit input, 2^N! possible permutations
For K-bit key, 2^K possible keys
Keyed Permutation

- Not just shuffling of input bits!
  - Suppose plaintext = “111”.
  - Then “111” is not the only possible ciphertext!
- Instead:
  - **Permutation of possible outputs**
  - Use secret key to pick a permutation
Block Cipher Security

• Result should look like a random permutation on the inputs
  • Recall: not just shuffling bits. N-bit block cipher permutes over $2^N$ inputs.

• Only computational guarantee of secrecy
  • Not impossible to break, just very expensive
    • If there is no efficient algorithm (unproven assumption!), then can only break by brute-force, try-every-possible-key search
  • Time and cost of breaking the cipher exceed the value and/or useful lifetime of protected information
Block Cipher Operation (Simplified)

1. Block of plaintext
2. Add some secret key bits to provide confusion
   - Each S-box transforms its input bits in a “random-looking” way to provide diffusion (spread plaintext bits throughout ciphertext)
3. Repeat for several rounds
4. Block of ciphertext
5. Procedure must be reversible (for decryption)
Standard Block Ciphers

• **DES: Data Encryption Standard**
  • **Feistel structure**: builds invertible function using non-invertible ones
  • Invented by IBM, issued as federal standard in 1977
  • 64-bit blocks, 56-bit key + 8 bits for parity
DES and 56 bit keys

- 56 bit keys are quite short

<table>
<thead>
<tr>
<th>Key Size (bits)</th>
<th>Number of Alternative Keys</th>
<th>Time required at 1 encryption/μs</th>
<th>Time required at $10^6$ encryptions/μs</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>$2^{32} = 4.3 \times 10^9$</td>
<td>$2^{31} \mu s = 35.8$ minutes</td>
<td>2.15 milliseconds</td>
</tr>
<tr>
<td>56</td>
<td>$2^{56} = 7.2 \times 10^{16}$</td>
<td>$2^{55} \mu s = 1142$ years</td>
<td>10.01 hours</td>
</tr>
<tr>
<td>128</td>
<td>$2^{128} = 3.4 \times 10^{38}$</td>
<td>$2^{127} \mu s = 5.4 \times 10^{24}$ years</td>
<td>$5.4 \times 10^{18}$ years</td>
</tr>
<tr>
<td>168</td>
<td>$2^{168} = 3.7 \times 10^{50}$</td>
<td>$2^{167} \mu s = 5.9 \times 10^{36}$ years</td>
<td>$5.9 \times 10^{30}$ years</td>
</tr>
<tr>
<td>26 characters (permutation)</td>
<td>$26! = 4 \times 10^{26}$</td>
<td>$2 \times 10^{26} \mu s = 6.4 \times 10^{12}$ years</td>
<td>$6.4 \times 10^6$ years</td>
</tr>
</tbody>
</table>

- 1999: EFF DES Crack + distributed machines
  - < 24 hours to find DES key
- DES --> 3DES
  - 3DES: DES + inverse DES + DES (with 2 or 3 diff keys)
But wait... what about 2DES?

- Meet-in-the-middle attack

\[ \text{adv: P/C pair} \]

\[ \text{encrypt} \]

\[ \begin{array}{c|cc}
0 & y_0 \\
1 & y_1 \\
2 & y_2 \\
3 & y_3 \\
\end{array} \]

\[ \text{decrypt} \]

\[ \begin{array}{c|cc}
0 & z_0 \\
1 & z_1 \\
2 & z_2 \\
3 & z_3 \\
\end{array} \]

\[ P \rightarrow \text{DES}_{K_1} \rightarrow \text{DES}_{K_2} \rightarrow C \]

\[ y_1 = z_m \]

\[ K_1 = K_n \quad K_2 = K_m \]
Defining the strength of a scheme

- **Effective Key Strength**
  - Amount of ‘work’ the adversary needs to do
- **DES**: 56-bits
  - 2^56 encryptions to try ‘all keys’
- **2DES**: 57-bits
  - 2*(2^56) encryptions = 2^57
- **3DES**: 112-bits (or sometimes 80-bits)
  - Meet-in-the-middle + more work = 2^112 (for 3 keys, e.g. K1, K2, K3)
  - Various attacks = 2^80 (for 2 keys, e.g. K1, K2, K1)
Standard Block Ciphers

• **DES: Data Encryption Standard**
  - Feistel structure: builds invertible function using non-invertible ones
  - Invented by IBM, issued as federal standard in 1977
  - 64-bit blocks, 56-bit key + 8 bits for parity

• **AES: Advanced Encryption Standard**
  - New federal standard as of 2001
    - NIST: National Institute of Standards & Technology
  - Based on the Rijndael algorithm
    - Selected via an open process
  - 128-bit blocks, keys can be 128, 192 or 256 bits
Encrypting a Large Message

• So, we’ve got a good block cipher, but our plaintext is larger than 128-bit block size

  
  128-bit plaintext
  (arranged as 4x4 array of 8-bit bytes)

  
  128-bit ciphertext

• What should we do?
Electronic Code Book (ECB) Mode

plaintext

<table>
<thead>
<tr>
<th>key</th>
<th>plaintext</th>
<th>key</th>
<th>ciphertext</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>block cipher</td>
<td>key</td>
<td>block cipher</td>
</tr>
<tr>
<td>key</td>
<td>block cipher</td>
<td>key</td>
<td>block cipher</td>
</tr>
<tr>
<td>key</td>
<td>block cipher</td>
<td>key</td>
<td>block cipher</td>
</tr>
</tbody>
</table>

Confidentiality, Integrity, Availability, Authenticity

128 bits
Electronic Code Book (ECB) Mode

- Identical blocks of plaintext produce identical blocks of ciphertext
- No integrity checks: can mix and match blocks
Information Leakage in ECB Mode

Encrypt in ECB mode

[Wikipedia]
Oops

Move Fast and Roll Your Own Crypto
A Quick Look at the Confidentiality of Zoom Meetings

By Bill Marczak and John Scott-Railton       April 3, 2020

- Zoom documentation claims that the app uses “AES-256” encryption for meetings where possible. However, we find that in each Zoom meeting, a single AES-128 key is used in ECB mode by all participants to encrypt and decrypt audio and video. The use of ECB mode is not recommended because patterns present in the plaintext are preserved during encryption.

Cipher Block Chaining (CBC) Mode: Encryption

- Identical blocks of plaintext encrypted differently
- Last cipherblock depends on entire plaintext
- Still does not guarantee integrity
CBC Mode: Decryption

Initialization vector

plaintext

ciphertext

decrypt

decrypt

decrypt

decrypt

key

key

key

key
ECB vs. CBC

AES in ECB mode

Similar plaintext blocks produce similar ciphertext blocks (not good!)

AES in CBC mode

[Picture due to Bart Preneel]
Initialization Vector Dangers

Initialization vector (supposed to be random)

plaintext

DES

DES

DES

DES

key

key

key

key

+ + + +

DES

DES

DES

DES

ciphertext

Found in the source code for Diebold voting machines:

DesCBCEncrypt((des_c_block*) tmp, (des_c_block*) record.m_Data, totalSize, DESKEY, NULL, DES_ENCRYPT)
Counter Mode (CTR): Encryption

- Identical blocks of plaintext encrypted differently
- Still does not guarantee integrity; Fragile if ctr repeats
Counter Mode (CTR): Decryption

Initial ctr

ct $\oplus$ ct $\oplus$ ct $\oplus$ ct $\oplus$ ct

Key

K

pt pt pt pt
Ok, so what mode do I use?

• Don’t choose a mode, use established libraries 😊

• Good modes:
  • GCM - Galois/Counter Mode
  • CTR (sometimes)
  • Even ECB is fine in ‘the right circumstance’

CBC is fine ..
When is an Encryption Scheme “Secure”? 

• Hard to recover the key?  
  • What if attacker can learn plaintext without learning the key?  

• Hard to recover plaintext from ciphertext?  
  • What if attacker learns some bits or some function of bits?
How Can a Cipher Be Attacked?

• Attackers knows ciphertext and encryption algthm
  • What else does the attacker know? Depends on the application in which the cipher is used!

• Ciphertext-only attack

• KPA: Known-plaintext attack (stronger)
  • Knows some plaintext-ciphertext pairs

• CPA: Chosen-plaintext attack (even stronger)
  • Can obtain ciphertext for any plaintext of choice

• CCA: Chosen-ciphertext attack (very strong)
  • Can decrypt any ciphertext except the target
Chosen Plaintext Attack

Crook #1 changes his PIN to a number of his choice

PIN is encrypted and transmitted to bank

cipher(key, PIN)

Crook #2 eavesdrops on the wire and learns ciphertext corresponding to chosen plaintext PIN

... repeat for any PIN value
Very Informal Intuition

• Security against chosen-plaintext attack (CPA)
  • Ciphertext leaks no information about the plaintext
  • Even if the attacker correctly guesses the plaintext, he cannot verify his guess
  • Every ciphertext is unique, encrypting same message twice produces completely different ciphertexts
    • Implication: encryption must be randomized or stateful

• Security against chosen-ciphertext attack (CCA)
  • Integrity protection – it is not possible to change the plaintext by modifying the ciphertext

Minimum security requirement for a modern encryption scheme