IT-Security Cryptography and Secure Communications

Exercise:	**Block Cipher Operation**
Lecturer:	*Prof. Dr. Michael Eichberg*
Version:	2023-10-19

1. Why is it important in CBC to protect the IV?

   Solution

   If the IV is sent as is, we may be able in certain scenarios to flip some bytes of the plaintext (of the first block) when we change the IV.

2. In which operation modes is padding necessary?

   Solution

   ECB and CBC (the input to the encryption is a full plaintext block).

3. What happens in case of a transmission error (single bit flip in the ciphertext) in ECB, CBC, CFB, OFB, CTR?

   Solution

ECB:	one block is affected (in case of DES and AES approx. 50% of the bits).
CBC:	in the next block we will have one flipped bit in the plaintext and approx. 50% in the current block.
CFB:	The flipped bit will affect the corresponding plaintext bit and all subsequent bits with a probability of approx. 50% as long the flipped bit is used as input to the encryption.
OFB, CTR:	In the plaintext one bit will be flipped.

4. Why does the IV in OFB has to be a nonce (i.e., unique to each execution of the encryption algorithm)?

Solution

The O_i only depend on the key and the IV, if the IV is reused with the same key and we happen to know a specific plaintext we may be able to decrypt a corresponding ciphertext in a different message.

5. You want to determine if a program for encrypting files uses ECB mode. What do you need to do?

One Solution

Use a document that consists of more than one block where each block has the size of the underlying cipher and each block has the same content. If ECB mode is used all blocks are encrypted in the same way.

6. A friend of yours invented a new block cipher. You are **very** skeptical. Think about some very simple tests to invalidate the cipher.
7. Use the OFB mode in combination with a Caesar cipher. The block size is a single character. The key is the number of characters you are going to shift a character - as before. The IV is some character. To make the XOR work we map every character to a value and extend the alphabet with the digits 1 to 3, "!", "?" and the "_". This way it is always possible to output a meaningful character.

Hence, we will have the following encoding:

Index	Character	Binary Representation
0	A	00000
1	B	00001
2	C	00010
3	D	00011
4	E	00100
5	F	00101
6	G	00110
7	H	00111
8	I	01000
9	J	01001
10	K	01010
11	L	01011
12	M	01100
13	N	01101
14	O	01110
15	P	01111
16	Q	10000
17	R	10001

18	S	10010
19	T	10011
20	U	10100
21	V	10101
22	W	10110
23	X	10111
24	Y	11000
25	Z	11001
26	1	11010
27	2	11011
28	3	11100
29	!	11101
30	?	11110
31	_	11111

Now encode some messages using your new cipher. What will you realize?

Solution

The same character is no longer (necessarily) mapped to the same target when it reappears in the original message; i.e, we have some diffusion.

Example

$IV = A, k = 3, M = AA$

$..1. I_1 = IV = A; E(I_1) = D; C_1 = A \oplus D = D$

$..2. I_2 = D; E(I_2) = G, C_2 = A \oplus G = G$