Solutions: Sheet 4

1. Confusion and Diffusion

Explain confusion and diffusion as two essential basic principles of cryptosystems. Which operations are used to achieve confusion and diffusion?

Confusion: ... -> substitution are used

Diffusion: ... -> permutations are used (change order)

Diffusion relations between function and ciphertxt???

Confussion realation between key and cipertext

2. Block Cipher Modes

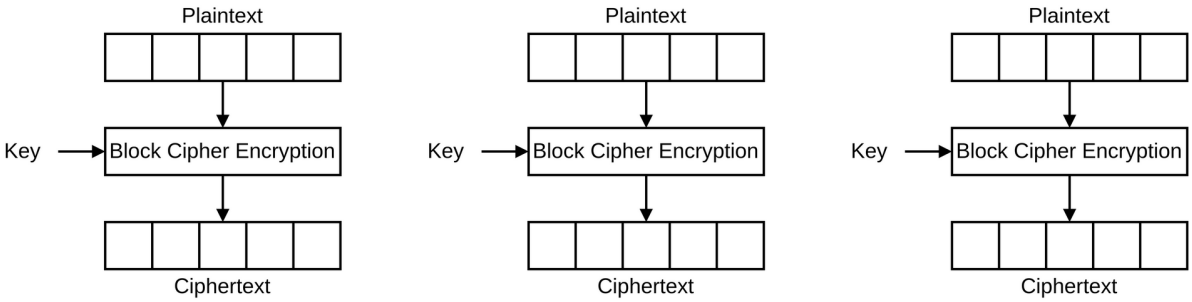
(a) Describe the purpose and basic functionality of operating modes for block ciphers.

Block ciphers encrypt plaintexts of a certain block size. For longer plaintexts operatings modes are used. There the text is broken down into blocks and the last block is filled with a *PADDING*. The operating modes describe how the blocks organize when encrypting.

```
001010....... 101001....... ............. 101111PADDING
\___________/ \___________/               \___________/
   Block 1       Block 2                     Block 8
   128 bit       128 bit                     128 bit
```

Different operating modes exist: *Electronic Code Book (ECB)*, *Cipher Block Chaining (CBC)*, *Counter Mode (CTR)*

Electronic Code Book (ECB):



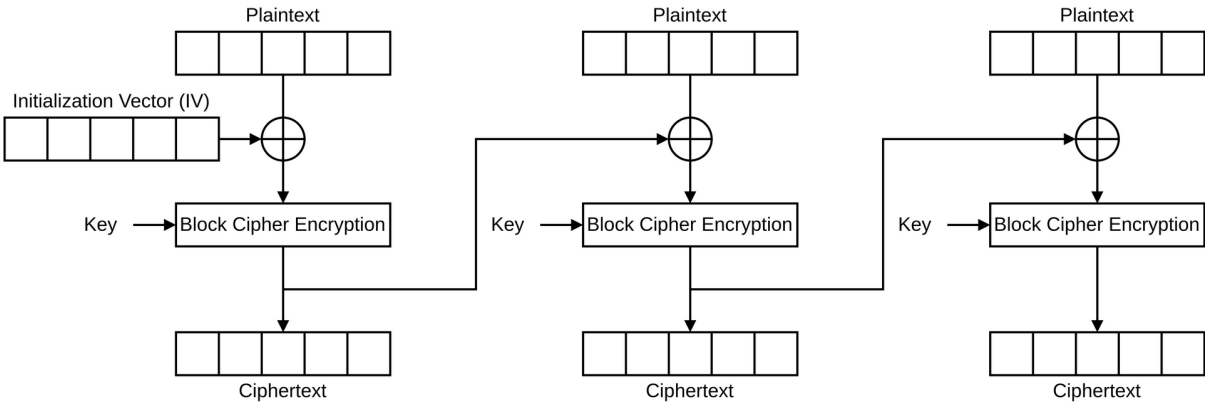
1. Plaintext is broken down into blocks

2. Block by block is encrypted

Problem: Same plaintext blocks are mapped on same ciphertext block, which can be used to recognize a pattern in the ciphertext. $\rightarrow$ ciphertext should not depend on key and plaintext but also on other parameter

Cipher Block Chaining (CBC):

For Encryption:

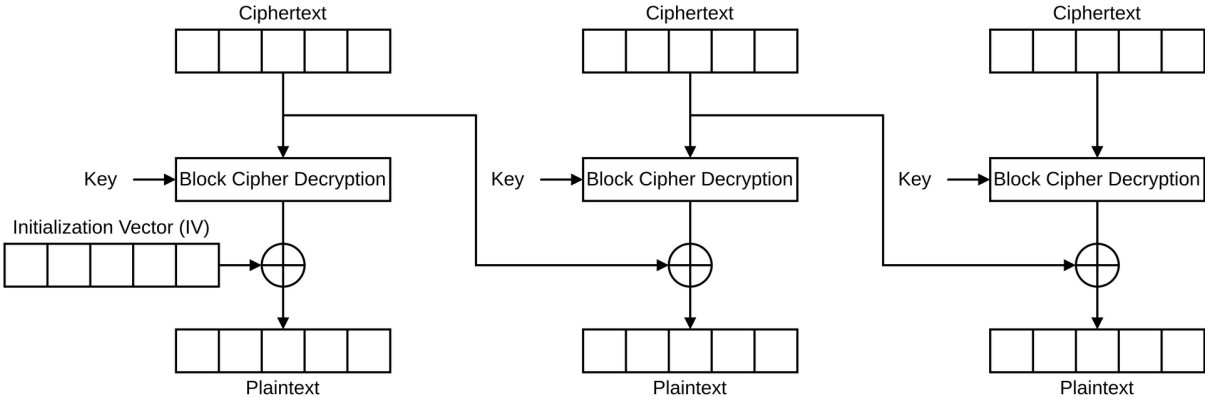


1. An random initilization vector is linked with the first block of plaintext and encrypted

2. The generated ciphertext is then linked via XOR with the next ciphertext and encrypted and so on

- In general we can say i-th plaintext m_i is linked with the previous ciphertext c_{i-1} via XOR and then encrypted

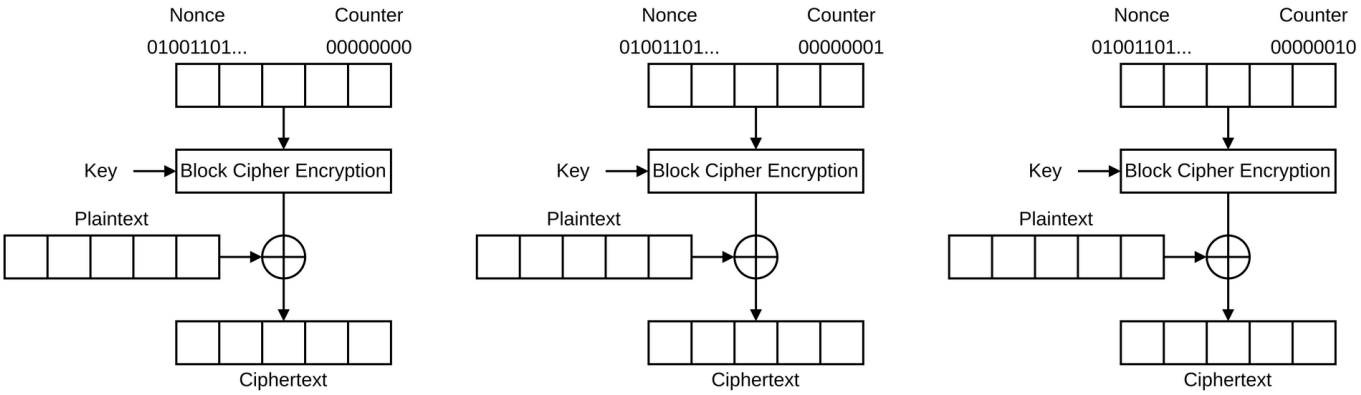
For Decryption:



- In general we can say i-th plaintext c_i is decrypted and then linked to the $(i-1)$-th cipertext c_{i-1} via XOR

Counter Mode (CTR):

For Encryption:



(b) How does the Electronic Codebook Mode (ECB) perform in terms of execution in parallel and error propagation?

The encryption in ECB can be executed parallel as the encryption of each block is independent of the encryption of another block. Thus if an error occurs in one block it's not in the next block. (no error propagation)

(c) Why is the Electronic Codebook Mode (ECB) insecure?

Identical plaintext blocks is always encrypted on identical ciphertexts. This allows patterns to be found in the encrypted ciphertext and conclusions to be drawn about the plaintext.

(d) Name two important differences in the construction between the Electronic Codebook Mode (ECB) and the Cipher Block Chaining Mode (CBC).

1. While ECB takes the key and plaintext as input for encryption, CBC adds one more parameter, a random initialization vector for the first block and the previous output (cipertext) for encrypting the next blocks.

2. CBC based on previous block (chaining), ECB not

3. Simple Block Cipher Example

In the following, we consider a symmetric cipher with a block size of four bits and key `0110`.

For ECB, we obtain the following:

Plaintext	Ciphertext	Plaintext	Ciphertext
`0000`	`1110`	`1000`	`1001`
`0001`	`1100`	`1001`	`0010`
`0010`	`1010`	`1010`	`1101`
`0011`	`0011`	`1011`	`0001`
`0101`	`1011`	`1101`	`1111`
`0100`	`1000`	`1100`	`0101`

Plaintext	Ciphertext	Plaintext	Ciphertext
`0110`	`0100`	`1110`	`0110`
`0111`	`0000`	`1111`	`0111`

(a) Complete the following table for ECB with key `0110`:

Ciphertext	Plaintext	Ciphertext	Plaintext
`0000`	`0111`	`1000`	`0100`
`0001`	`1011`	`1001`	`1000`
`0010`	`1001`	`1010`	`0010`
`0011`	`0011`	`1011`	`0101`
`0101`	`1100`	`1101`	`1010`
`0100`	`0110`	`1100`	`0001`
`0110`	`1110`	`1110`	`0000`
`0111`	`1111`	`1111`	`1101`

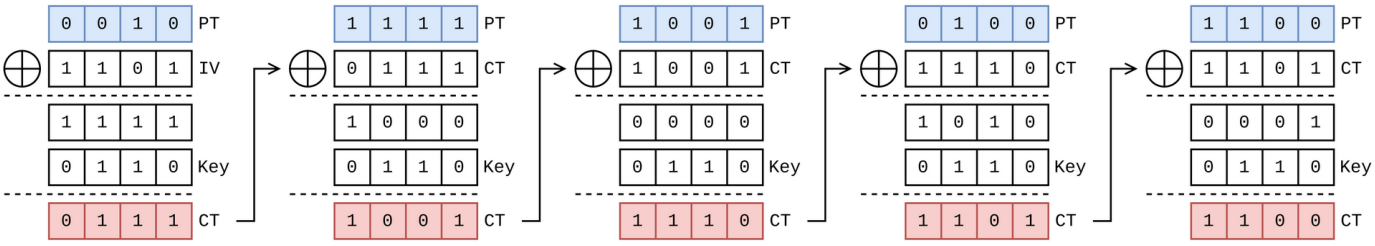
(b) In ECB, what would be the encryption for `0010 1111 1001 0100 1100` with key `0110`?

```
enc(0010 1111 1001 0100 1100, 0110) = 1010 0111 0010 1000 0101
```

(c) In ECB, what would be the decryption for `1101 0101 0110 0001 0110` with key `0110`?

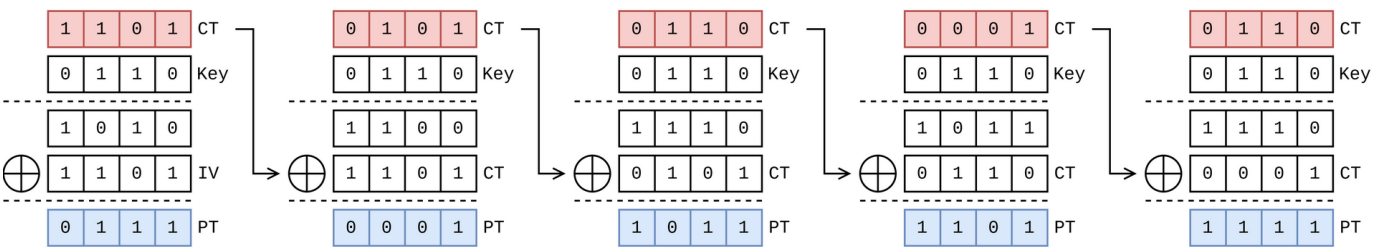
```
dec(1101 0101 0110 0001 0110) = 1010 1100 1110 1011 1110
```

(d) For CBC, what would be the encryption for `0010 1111 1001 0100 1100` with IV `1101` and key `0110`?

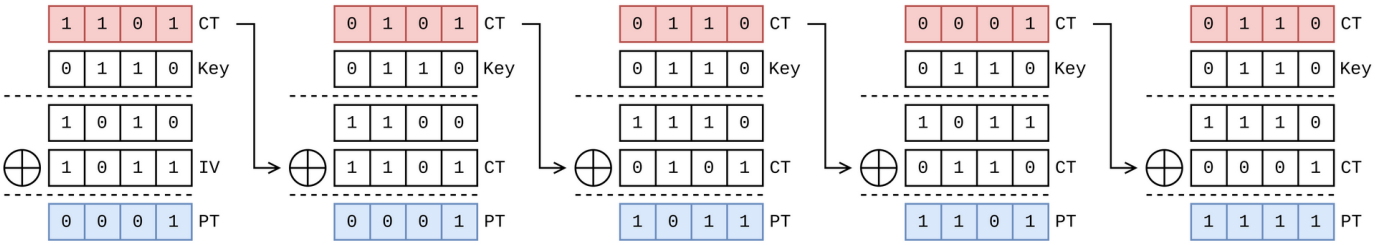


(e) For CBC, what would be the decryption for `1101 0101 0110 0001 0110` with IV `1101` and key `0110`? What is the decryption if you use the IV `1011`?

For IV `1101`:



For IV `1011`:



4. Hash Functions

1\ . Which properties that are required in cryptology should (cryptographic) hash functions have?

- input size of message possibly inifinite (possible)
- hash value always same fixed output size
- colission resistant
- preimage resitant
- efficently calculated

2. What are (cryptographic) hash functions used for? Give at least two examples.

- Digital signatures (and Certificate)
- Integrity check (input file and compare hash)
- Blockchain

3. Explain the term collision resistance in the context of cryptographic hash functions.

Preimage Resistance: Hash functions should be preimage resistant. This means that there should be no clues, such as a pattern in the hash value, that can be used to infer the original message.

Collision Resistance: Hash functions should be collision resistant. This means there should not be two different messages that generate the same hash value. In practice, however, collisions exit for every hash function since we map an infinite set of possible messages to a finite set of possible hash values, even it's practically impossible.

4. In an online shop, user passwords were stolen from the database. The webmaster claims that the data is not at risk in the hands of the thieves as only SHA1 hash values of the data are stored in the database. Do you agree with that? Prove to the webmaster that the data is not secure by decrypting the passwords with an appropriate tool if possible. The following twelve SHA1 bit strings are the hash values of the captured passwords:

It is not easily possible to find a solution for all SHA1 strings. To prove that the data is not secure, finding out eight of the twelve passwords is enough. Do some research on the internet for suitable tools.

```
(a) 5baa61e4c9b93f3f0682250b6cf8331b7ee68fd8
(b) 7c4a8d09ca3762af61e59520943dc26494f8941b
(c) d44ef90413033a0816ecbde55a69d912fcd410f7
(d) 7c222fb2927d828af22f592134e8932480637c0d
(e) 6367c48dd193d56ea7b0baad25b19455e529f5ee
(f) b1b3773a05c0ed0176787a4f1574ff0075f7521e
(g) ab87d24bdc7452e55738deb5f868e1f16dea5ace
(h) 19d5a2866d23792ff892a08ac0e65809fb331c94
(i) b7a875fc1ea228b9061041b7cec4bd3c52ab3ce3
(j) af8978b1797b72acfff9595a5a2a373ec3d9106d
(k) 3d4f2bf07dc1be38b20cd6e46949a1071f9d0e3d
(l) a2c901c8c6dea98958c219f6f2d038c44dc5d362
```

database with common passwords → https://md5decrypt.net/en/Sha1/

```
(a) 5baa61e4c9b93f3f0682250b6cf8331b7ee68fd8 = password
(b) 7c4a8d09ca3762af61e59520943dc26494f8941b = 123456
(c) d44ef90413033a0816ecbde55a69d912fcd410f7 = sicherespasswort+1234 (not
findable)
(d) 7c222fb2927d828af22f592134e8932480637c0d = 12345678
(e) 6367c48dd193d56ea7b0baad25b19455e529f5ee = abc123
(f) b1b3773a05c0ed0176787a4f1574ff0075f7521e = qwerty
(g) ab87d24bdc7452e55738deb5f868e1f16dea5ace = monkey
(h) 19d5a2866d23792ff892a08ac0e65809fb331c94 = mein+passwort (not findable)
(i) b7a875fc1ea228b9061041b7cec4bd3c52ab3ce3 = letmein
(j) af8978b1797b72acfff9595a5a2a373ec3d9106d = dragon
(k) 3d4f2bf07dc1be38b20cd6e46949a1071f9d0e3d = 111111
(l) a2c901c8c6dea98958c219f6f2d038c44dc5d362 = baseball
```

5. AES in CBC and ECB Mode You need a Linux installation in order to solve this exercise and also for the next exercise sheet.

Preparation: Installation of Linux (if not already available)

The goal of this task is to visualize the encryption with CBC and ECB in comparison. Research the function and format of headers for typical file formats because these must not be encrypted in order to get the correct result. Document all your steps.

(a) Encrypt the image Tux.ppm that can be found in Moodle with AES in ECB mode using openssl without the header.

1. Seperate body and header

```
$ head -n 4 Tux.ppm > header.txt
$ tail -n +5 Tux.ppm > body.bin
```

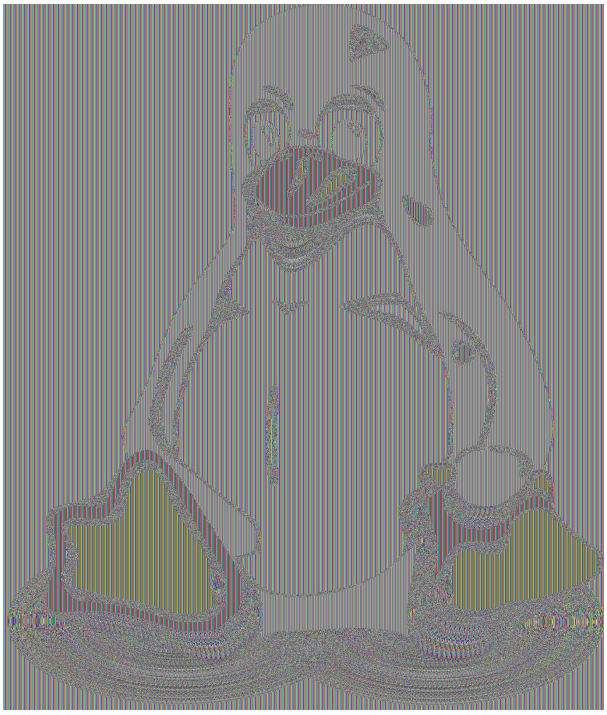
2. Encrypt via ECB

```
$ openssl aes-128-ecb -in body.bin -out body.bin.enc
enter AES-256-ECB encryption password: itsec
Verifying - enter AES-256-ECB encryption password: itsec
*** WARNING : deprecated key derivation used
```

3. Reassamble

```
$ cat header.txt body.bin.enc > Tux.ecb.ppm
```

4. Output:



(b) Afterwards encrypt the image with AES in CBC mode.

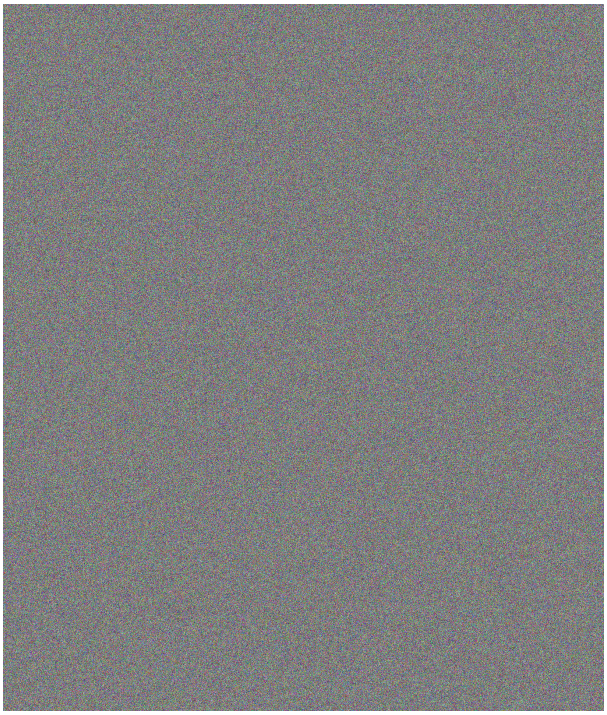
1. Encrypt via CBC

```
$ openssl aes-128-cbc -in body.bin -out body.bin.enc
enter AES-256-ECB encryption password: itsec
Verifying - enter AES-256-ECB encryption password: itsec
*** WARNING : deprecated key derivation used
```

3. Reassamble

```
$ cat header.txt body.bin.enc > Tux.cbc.ppm
```

4. Output



(c) Compare the two results and explain the effect you can see.

ECB: Encrypted block by block $\rightarrow$ pattern visible

CBC: Block encrypted with ciphertext of previous block or initialization vector $\rightarrow$ no pattern visible