Scribe: Cryptography and Network Security (Class.29.A)

Vishal Gourav

25-Oct-2020

1 Introduction

We will be continuing our discussion on modes of block cipher used. We will start with **Output Feedback Mode**, then we will discuss **Counter Mode**, and then MAC, and finally ending with Merkle's Puzzle.

2 Output Feedback Mode

In Output Feedback Mode(OFB), the working can be explained with help of schematic diagram in Figure 1 as follows:-

- The basic equations at each block are:-

$$z_i = E_k(z_{i-1})$$
$$c_i = x_i \oplus z_i$$

- We take r bits of the message and XOR it with r bits of the string and the key string is generated by the encryption function.This creates the cipher text.
- Thus every time a output is fed back and a new string gets generated which is again XORed to r bits of the new output block.
- Something to be noted here is that though it is a block cipher mode but is closely works like a **stream cipher**.

Some properties of OFB are as follows:-

- Affecting one plain text block affects one cipher text block only.
- Changing a bit in the plain text changes a bit in the cipher text. This property leads to unforeseen issues in ensuring authentication.

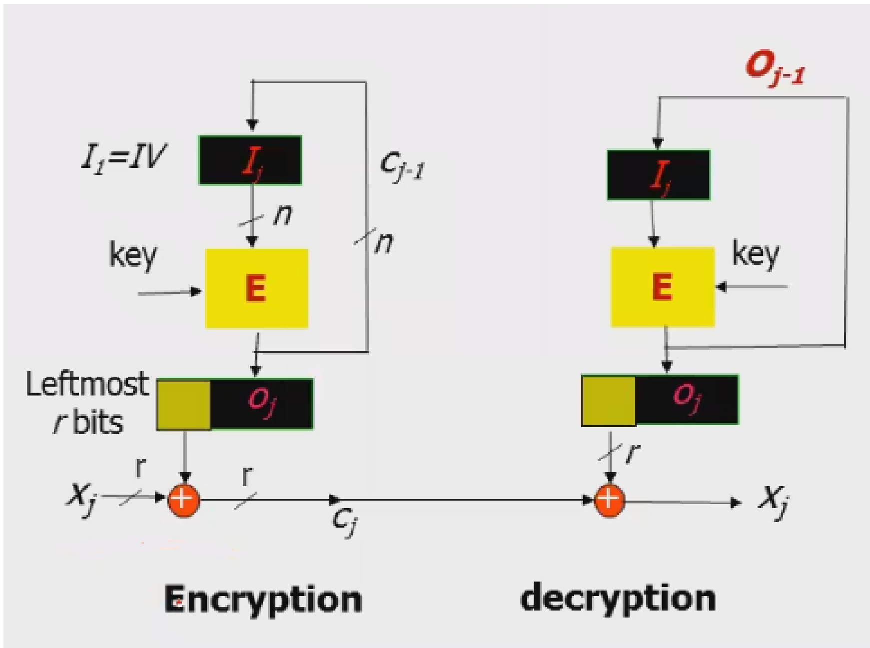

Figure 1: Output Feedback Mode

3 Counter Mode

A drawback in OFB mode which is lack of parallelism is corrected in Counter Mode. It is explained as follows:-

- The basic working of the counter mode is same as OFB with the difference that instead of waiting for c_{j-1} the inital variable IV is fed after concatenating it with a counter at every block while keeping in mind to increment the counter by 1 at every block.The updated equations are:-

$$T_i = (Count + i - 1) mod\ 2^n$$
$$y_i = x_i \oplus E_k(T_i)$$

- Thus the implementation can be made faster by employing parallelism. Therefore we can encrypt any jth location int the plain text without encrypting any previous location.

- Something that should always be kept in mind is never to reuse an IV because it will lead to same cryptogram.

- But, still the properties of **Authentication** and **Integrity** cannot be guaranteed.

- We instantly think of adding a hash value to help the integrity property hold but we can imagine a scenario where the attacker can just create a hash of his/her corrupted message and replace the whole thing. Thus we need to create a smarter defence system which led to MAC, which we will be discussing next.

4 MAC

The idea here is to use the encrypted message at each block to create the MAC which a basically a hash which is then concatenated to the final cipher text and sent to the recipient. The idea can be explained with the help of a schematic diagram in Figure 2 as follows:-

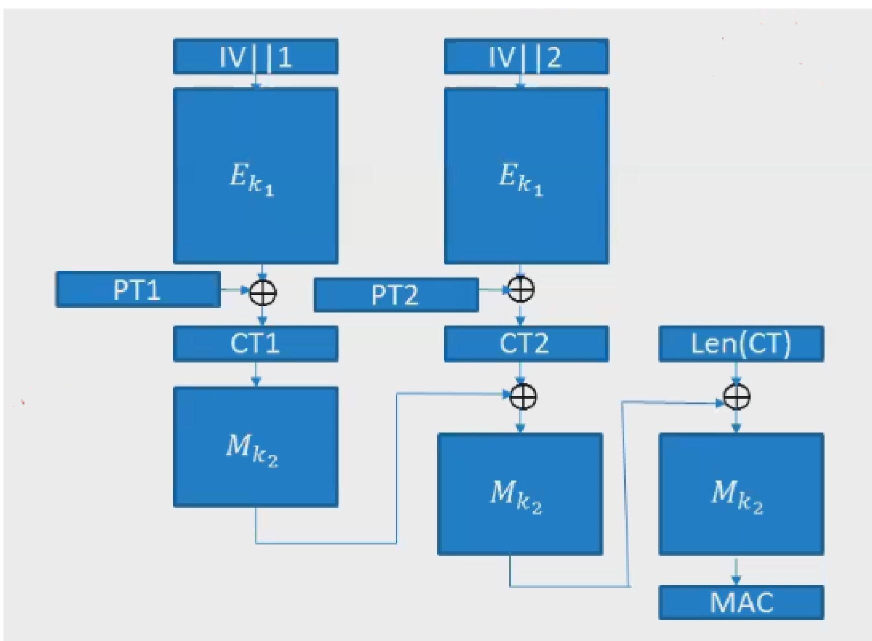

Figure 2: Galois MAC

- The upper part of the diagram is same as counter mode.
- What we do here is that, MAC of every cipher text block M_{k_i} is XORed with next cipher text block c_{i+1} to create $M_{k_{i+1}}$.
- Finally, the after the end of cipher text one more iteration of the same process is done with **Len(CT)** as one side of XOR instead of the cipher text to produce the final MAC.

An improvement that can be further made is the addition of **Auxiliary Data** to create the MAC. This auxiliary data may be **IP,Port number**,etc. This mode is therefore called **AEAD** or Authenticated Encryption with Auxiliary Data Mode. The following Figure 3 explains the working:-

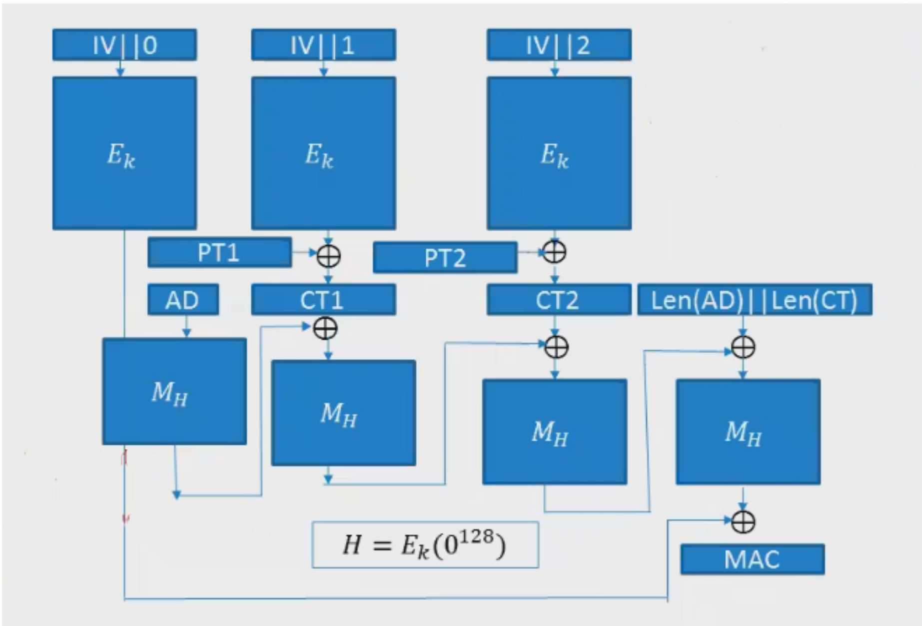

Figure 3: AEAD

- The working is same as Galois MAC as in Figure 2. The only differences are stated next.
- M_H is created by using the desired auxiliary data.
- At the final step instead of using only length of cipher text LEN(CT), concatenation of **LEN(AD)** i.e., length of auxiliary data and **LEN(CT)** is used to create the final MAC.

5 Merkle's Puzzle

Ralph Merkle proposed a method of symmetric key cryptography in 1978 which later became the basis of modern symmetric key cryptography. The protocol can be explained as follows:-

- The sender chooses l tuples $(k_1, s_1), ...(k_l, s_l)$.

- Then the sender prepares the following puzzle:-

$$\begin{aligned}
P'_1 &= (E(k_1, s_1), E(k_1, 1), E(k_1, 0)) \\
P'_2 &= (E(k_2, s_2), E(k_2, 2), E(k_2, 0)) \\
&\;.... \\
&\;.... \\
P'_l &= (E(k_l, s_l), E(k_l, l), E(k_l, 0))
\end{aligned}$$

  where the input s_i is applied upon by k_i is the first block, second block is encryption of index 1 by same key and similarly third block is encryption of index 0 by same keya s well.

- Now the sender permutes the puzzle randomly. For example,

$$\begin{aligned}
P'_1 = P'_5 &\quad = (E(k_1, s_1), E(k_1, 1), E(k_1, 0)) \\
P'_2 = P'_4 &\quad = (E(k_2, s_2), E(k_2, 2), E(k_2, 0)) \\
.... \\
.... \\
P'_l = P'_1 &\quad\quad = (E(k_l, s_l), E(k_l, l), E(k_l, 0))
\end{aligned}$$

- The sender sends this to the receiver.

- Now the receiver chooses a block randomly and tries to solve for k such that $D(k, E(k_i, 0)) = 0$ exhaustively using brute force. Now using the key receiver also solves the first 2 blocks i.e., $E(k_i, s_i)$ and $E(k_i, i)$ and discovers that the shared secret key is s_i. The receiver then sends the sender i to tell sender which key he/she has chosen.

- Now the attacker can only see the message i being sent from receiver to sender and he/she has to do a brute force on the whole message sent from sender to receiver to find what which secret key s_i, i corresponds to. This will increase attacker's breaking time to square of that of receiver i.e. if for receiver the time complexity is $O(n)$, then for attacker the time complexity would be $O(n^2)$.

- This is due to the permutation done by the sender.

The Merkle's puzzle does not have a very high confusion for an attacker but it served as basis for future improvements.

6 Conclusion

In the discussion we learned about the modes of Block Cipher and also took a detour to Merkle's Puzzle and realized how it was an idea that would inspire future excellency in field of cryptography .