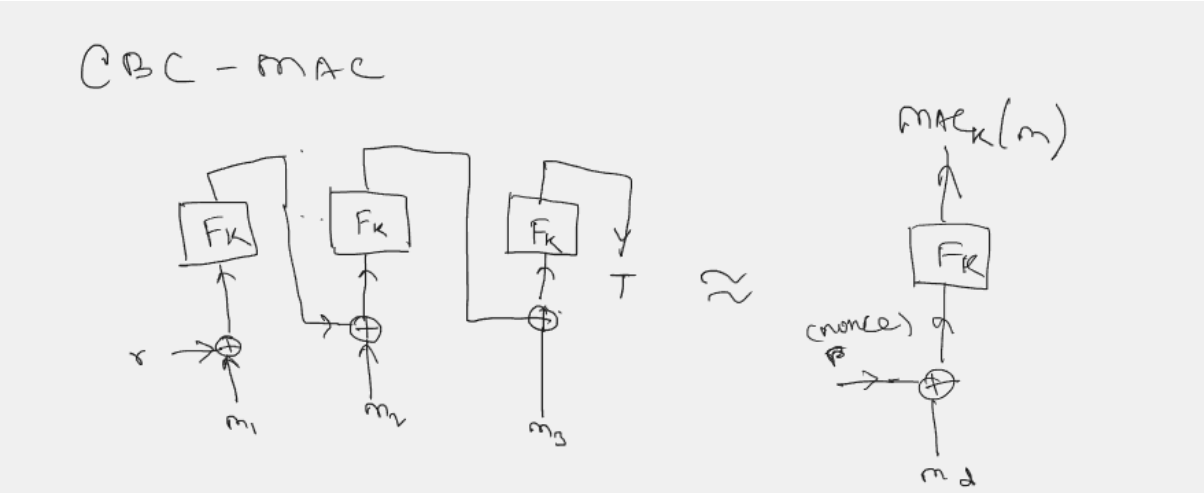
Soumodipta Bose| 2021201086

Use the PRF to build a secure MAC(Theory)

One of our main problems in CPA is what if the cipher text is tampered with, in certain use cases like finance these cipher text will be used to execute transactions. The attacker might cause harm by just changing a 1 to a zero which might cause serious damage to business due to the avalanche effect. Hence the need for a verification algorithm that verifies the authenticity of a cipher text.

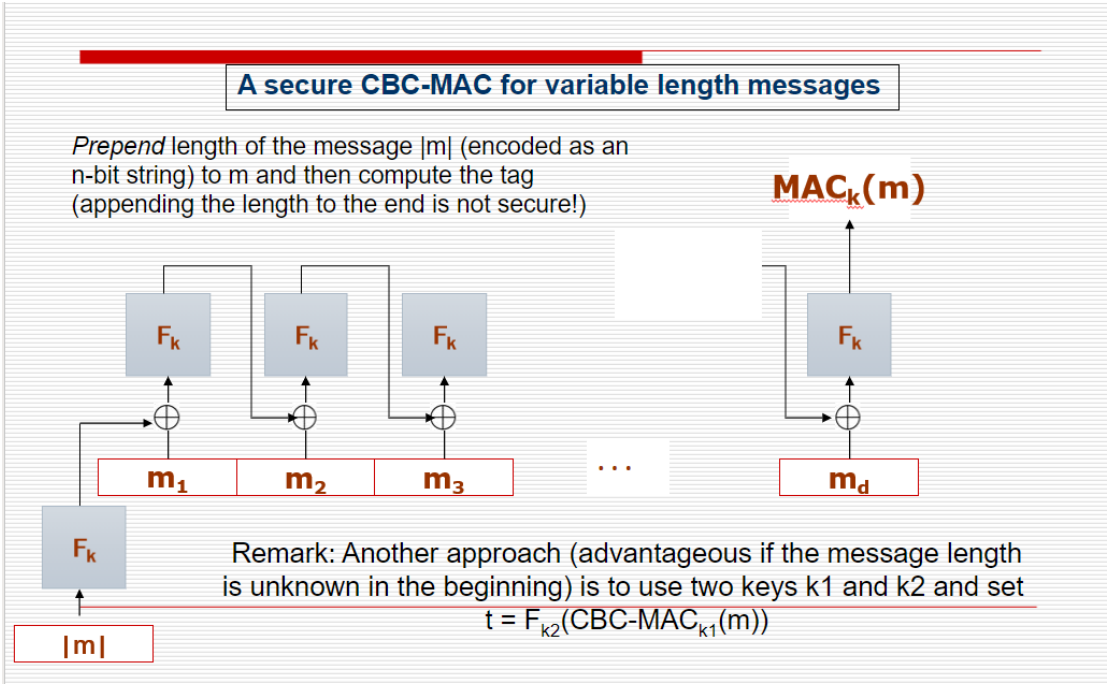
Sketches/Ideas:



Construction:

We had two options either creating a normal MAC which already takes care of the many attacks like replay attack, packet reordering, changing the tag and invalid message id's. Instead we chose CBC MAC to take care of our problem because it was relatively easy to construct and works hand in hand like the chaining in Encryption scheme. One problem it had was related to the length attack in which the attacker could use the tag of any stage and use that tag to propagate the next sequences and corrupt the message. This was mitigated by using the length and encoding it at the start of the message. Now let us come to its construction.

We will be reusing our Pseudo random Function F(k,x) from our crypto library. We have created two mechanism one to create the tag another to verify it. We will take a key k as input and feed it to the Pseudo random Function each time and the other input is our message. The length of the message will be encoded to the front of the message also ensuring it matches the key size. We generate the first tag using F(k,x). This tag will be used as an initialization vector.



Next we will create a propagation mechanism(loop) in which we are going to xor the tag and the message, the newly generated tag can be used in the next stage. We will keep on doing it for all the message fragments. The final tag generated will be used at the verification api.

In the verification algorithm we will simply call the CBC MAC creator to regenerate the tag from the message and then we will do an equality check. If it matches then we return True else False.