**Task 5.1 (theoretical): Block Cipher Based MACs**

**Part(a)**

**ECB**

Is it suitable as a MAC?

* No.

Why not?

* Because each block is encrypted separately there is no relation and influence between the encrypted blocks.

**CBC**

Is it suitable as a MAC?

* Yes.
* Because the last block of cipher text contains the influence of all the previous blocks so, loss or change of any block will reflect on the last block of cipher text.

**CTR**

Is it suitable as a MAC?

* No.

Why not?

* Because each block of plaintext is encrypted independently there is no influence of one block to another.

**Part (b)**

**ECB**

Is it still suitable when dealing with messages of variable length?

* Yes.

Because each block of message is encrypted independently. Block that cannot match the necessary size can be padded with zeros.

**CBC**

Is it still suitable when dealing with messages of variable length?

* No.

Because, CBC uses chaining of XOR to produce cipher text hash.

Thus, when the final cipher block of one message is XORed with the first plaintext block of another message and produced a hash then concatenate this result to the hash result of the first message this will cancel out the effect of the first message because of XOR property.

Let, m=<m1,m2,m3> a message and mn be the final cipher block of m.

Let ,n be another message with final cipher block nn.

Then, if we create a new message such that, O=<(m1^nn),m2,m3>

Use hash function H().

The result will be , H(n||O)=H(m) . Means, the effect of n is totally cancel out.

**CTR**

Is it still suitable when dealing with messages of variable length?

* Yes.

Because, each block of message is encrypted independently so even if the length changes it will be processed separately. Block that cannot match the necessary size can be padded with zeros.

**References:**

<http://www.tutorialspoint.com/cryptography/block_cipher_modes_of_operation.htm>

<https://en.wikipedia.org/wiki/CBC-MAC>

<http://crypto.stackexchange.com/questions/18538/aes256-cbc-vs-aes256-ctr-in-ssh>

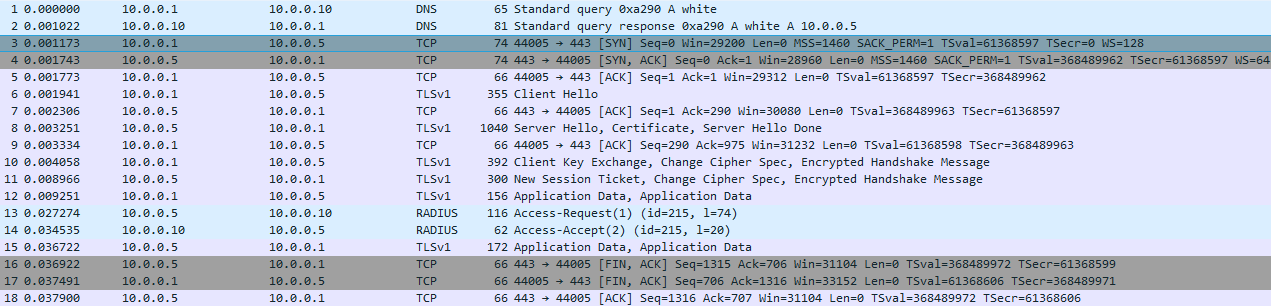
<http://stackoverflow.com/questions/1220751/how-to-choose-an-aes-encryption-mode-cbc-ecb-ctr-ocb-cfb>

<http://johnx.blogspot.de/2010/10/aes-cbc-or-aes-ctr-mode.html>

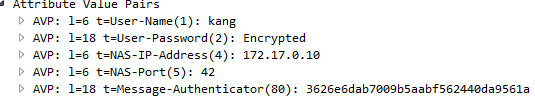
**Task 5.2 (theoretical): RADIUS**

Which hosts of the SecLab are involved in the authentication process?

* User IP= 10.0.0.1
* RADIUS Client IP = 10.0.0.5 (White)
* RADIUS Server IP = 10.0.0.10



1. In first 2 steps the user queries the DNS server for the name of the RADIUS client and the DNS server returns the name of the NAS as “white” that have IP=10.0.0.5. (the DNS and RADIUS server have same IP because they are in the same machine)
2. In step 3, 4 and 5 the user completes the 3 way TCP hand shake with the RADIUS clients. We can see that in the first 3 steps of this packet.
3. In step 6 the user sends the “Client Hello” using a signature to verify it self to the NAS. This step uses TLSv1 protocol.
4. In step 7 the NAS sends the acknowledgement of that “Client Hello”.
5. Step 8 the NAS sends its certificate to the user to verify itself with “Server Hello” and finally the message “Server Hello Done”. This step uses TLSv1 protocol.
6. In step 9 the user sends verification acknowledgement of the certificate to the RADIUS client/NAS.
7. In step 10 the user exchange the shared secret key with the NAS. Sends “Change Cipher Spec” message to negotiate the usage of this same CipherSpec. Finally sends the first encrypted message using that specific algorithm and shared secret key. This step uses TLSv1 protocol.
8. In step 11 the NAS sends the user the “New session ticket”, aggress on the usage of the same CipherSpec and also sends it first encrypted message using that specific algorithm and shared secret key to the user. This step uses TLSv1 protocol.
9. In step 12 the client sends its identity credentials in encrypted form using the negotiated CipherSpec as “Application Data”. This step uses TLSv1 protocol.
10. In step 13 the NAS sends the “Access-Request(1)” with all the necessary encrypted data and user credentials to the RADIUS server. In the attribute field of this packet we can see the user name “kang” and all other details.



This step uses RADIUS protocol.

1. In step 14 the RADIUS server sends “Access-Accept” to the NAS after checking its submitted data. This step uses RADIUS protocol.
2. In step 15 the RADIUS client confirms the user about successful access by returning its credentials. This step uses TLSv1 protocol.
3. Now NAS and the user communicates normally using TCP.

In the sketch I draw from step 6 to simplify and I used pen and paper because you asked for sketch if I am not wrong.

