1.1 CBC Mode with predictable IV is not IND-CPA secure

Assuming the Advisory knows the following: IV is always incremented by one by the challenger or the oracle, IV for any ciphertext is appended in plaintext to the ciphertext, then, the following strategy will work to ensure the Advisory wins 100% of the time.

1) The advisory creates two plain text m_0 & m_1 which are equal in length.
2) Advisory sends both plaintexts to the challenger
3) Challenger chooses a uniform bit, b ∈ {0, 1} and encrypts m_b as $c = (IV_i = i, E_k(m_b \oplus IV_i))$
4) Challenger returns c to the advisory and increments i by one, this is the IV that is going to be used next either by the oracle or challenger. Let's call this IV_o
5) Since IV_i is given in plaintext in c, advisory can deduce $IV_{next} = IV_i + 1$
6) Advisory now sends the plaintext $p = (m_0 \oplus IV_i \oplus IV_{next})$ to the oracle
7) The oracle returns $p' = (IV_o = i, E_k(p \oplus IV_o)$. Note, at this stage $IV_o = IV_{next}$
8) When we simplify $E_k(p \oplus IV_o)$, we get $E_k(p \oplus IV_o) = E_k(m_0 \oplus IV_i \oplus (IV_{next} \oplus IV_o)) = E_k(m_0 \oplus IV_i \oplus 0) = E_k(m_0 \oplus IV_i)$
9) Therefore, $p' = (IV_o = i, E_k(m_0 \oplus IV_i))$
10) Advisor now has p'
11) If p' is the same as c then we can conclude that $b'\ must\ be\ 0$ otherwise $b'\ must\ be\ 1$

We can say the advisory wins the game as the probability of b' being correct is 1 as either way we can always output b'.

This strategy works by exploiting the fact the IV is predictable as it is incremented by one each time it is used. The fact that the IV is given as plaintext in the ciphertext lets us easily figure out what the next IV is going to be, thus letting us negate the IV for the next message we send to the oracle. By being able to predict the IV it lets us easily compute b' every time therefore, proving that CBC-mode is not IND-CPA secure when the IV is easily predictable.

1.2 HMAC

The new proposed HMAC schemes increases performance by half for larger messages however it is not secure under the existential forgeability model. I have come up with the experiment below to prove the scheme is not secure.

Definitions:

$Let\ m' = m||0\ if\ |m| = odd; otherwise\ m' = m$

$Mac_k(m) = \mathrm{H}\left((\mathrm{k} \oplus \mathrm{a})||H((k \oplus b)||(m'_L \oplus m'_R))\right)$, *where the inputs are* $m \in \{0,1\}^*$, *key* k

$Vrfy_k(\mathrm{m}, \mathrm{t}) = b$, with b = 1 meaning valid and b = 0 meaning invalid

Query Phase:

1) Advisory chooses the message m_1
2) Advisory sends m_1 to the oracle and returns tag t such that $t = Mac_k(m_1)$.
3) The advisory now has a valid pair (m_1, t) where $Vrfy_k(m_1, t) = 1$
4) Note that $m_1 \in Q$ or $Q = \{m_1\}$

Forge Phase:

5) Advisory can create m_2 such that $m_2 = m'_{1R} \oplus m'_{1L}$
6) Advisory can now use m_2 with previous tag t and still pass verification. $Vrfy_k(m_2, t) = 1$

Therefore, advisory can successfully forge the tag for m_2 which is not in Q and thus the scheme is not secure.

This strategy works because in the new proposed HMAC scheme the message is split into two halves and exclusive-ORed with each other, which implies the order of each half of the message does not matter. However, this is not true since $m_L||m_R \neq m_R||m_L$. A secure HMAC function should create completely different hash for both $m_1 \& m_2$ whereas the new proposed HMAC scheme does not.