CAA 24-25

Exercise Sheet on Symmetric Crypto 1
Solutions

1 Modes of Operation

For the birthday attacks:

- In CBC: if two ciphertext blocks c_i and c_j are equal, then, $c_{i-1} \oplus c_{j-1} = m_i \oplus m_j$.
- In CFB: if two ciphertext blocks c_i and c_j are equal, then, $c_{i+1} \oplus c_{j+1} = m_{i+1} \oplus m_{j+1}$.

Mode of operation	**stream**	**parallel**	**partial enc/dec**	**Padding**	**Only encryption?**
ECB	No	Yes	Yes	Yes	No
CBC	No	Yes (decryption)	Yes(decryption)	Yes	No
CFB	No	Yes (decryption)	Yes(decryption)	No	Yes
OFB	Yes	No	No	No	Yes
CTR	Yes	Yes	Yes	No	Yes

Mode of operation	**IV reuse**	**Error propagation** (one bit change in ciphertext)	**Security issues**
ECB	No IV	total change in current block of plaintext	Do not use: same plaintext block implies same ciphertext block
CBC	If first blocks are similar, leaks	total change in current block of plaintext, one bit change in next	After $2^{n/2}$ blocks (birthday attack), repetition of ciphertext block implies IV reuse. Padding oracle attack. Needs a random IV and not a counter.
CFB	XOR of first different block ciphertext = XOR first different block plaintext	one bit change in current block of plaintext, total change in the next block	After $2^{n/2}$ blocks (birthday attack), repetition of ciphertext implies that XOR of next plaintexts = XOR of next ciphertexts
OFB	Stream cipher: XOR of plaintext = XOR of ciphertext	one bit change in plaintext	After $2^{n/2}$ blocks cycle in stream. Total repetition of following keystream. Strictly no integrity protection.
CTR	Stream cipher: XOR of plaintext = XOR of ciphertext	one bit change in plaintext	Strictly no integrity protection. Smaller nonce size than other construction. Beware of collisions if random nonces are used. Beware of counter overflow.

Finally, regarding predictable IVs, only CBC is vulnerable. Let's suppose that you want to guess the value of a plaintext block (e.g. while doing a bruteforce). For instance, let's say that the block is either `yes` or `no`. The ciphertext block is then $c = \mathsf{AES}_K(m \oplus \mathsf{IV}_1)$, where m is either `yes` of `no`. Since the IV is predictable, you know that the next IV is going to be IV_2. Hence, to guess `yes`, you can submit $\texttt{yes} \oplus \mathsf{IV}_1 \oplus \mathsf{IV}_2$ to the encryption oracle. If your guess is correct, then you will obtain c. This attack was used against TLS 1.0 in the BEAST attack.

2 Forensics on SHA-3

In the RAM, you were able to find the outer **and** the inner state. Hence, it is rather easy to go backwards. We start with the final state and we apply on it the function f^{-1}. This function can be deduced from the definition of f, is given in the specification, and is even already part of the keccak tools package: `https://keccak.team/software.html`.

If you look now at the outer state, you have the password, since 70 chars = 560 bits which is smaller than any possible rate in SHA-3.