Problems

12.1. As we have seen, MACs can be used to authenticate messages. With this problem, we want to show the difference between two protocols—one with a MAC, one with a digital signature. In the two protocols, the sending party performs the following operation:

1. Protocol A:
$$y = e_{k_1}[x||h(k_2||x)]$$
where x is the message, $h()$ is a hash function such as SHA-1, e is a private-key encryption algorithm, "||" denotes simple concatenation, and k_1, k_2 are secret keys which are only known to the sender and the receiver.
2. Protocol B:
$$y = e_k[x||sig_{k_{pr}}(h(x))]$$

Provide a step-by-step description (e.g., with an itemized list) of what the receiver does upon receipt of y. You may want to draw a block diagram for the process on the receiver's side, but that's optional.

12.2. For hash functions it is crucial to have a sufficiently large number of output bits, with, e.g., 160 bits, in order to thwart attacks based on the birthday paradox. Why are much shorter output lengths of, e.g., 80 bits, sufficient for MACs?

For your answer, assume a message x that is sent in clear together with its MAC over the channel: $(x, MAC_k(x))$. Exactly clarify what Oscar has to do to attack this system.

12.3. We study two methods for integrity protection with encryption.

1. Assume we apply a technique for combined encryption and integrity protection in which a ciphertext c is computed as
$$c = e_k(x||h(x))$$
where $h()$ is a hash function. This technique is not suited for encryption with stream ciphers if the attacker knows the whole plaintext x. Explain *exactly* how an active attacker can now *replace* x by an arbitrary x' of his/her choosing and compute c' such that the receiver will verify the message correctly. Assume that x and x' are of equal length. Will this attack work too if the encryption is done with a one-time pad?
2. Is the attack still applicable if the checksum is computed using a keyed hash function such as a MAC:
$$c = e_{k_1}(x||MAC_{k_2}(x))$$
Assume that $e()$ is a stream cipher as above.

12.4. We will now discuss some issues when constructing an efficient MAC.

1. The messages X to be authenticated consists of z independent blocks, so that $X = x_1||x_2||\ldots||x_z$, where every x_i consists of $|x_i| = 8$ bits. The input blocks are consecutively put into the compression function

$$c_i = h(c_{i-1}, x_i) = c_{i-1} \oplus x_i$$

   At the end, the MAC value

$$MAC_k(X) = c_z + k \bmod 2^8$$

   is calculated, where k is a 64-bit long shared key. Describe how exactly the (effective part of the) key k can be calculated with only one known message X.
2. Perform this attack for the following parameters and determine the key k:

$$\begin{aligned} X &= \texttt{HELLO ALICE!} \\ c_0 &= 11111111_2 \\ MAC_k(X) &= 10011101_2 \end{aligned}$$

3. What is the effective key length of k?
4. Although two different operations ($[\oplus, 2^8]$ and $[+, 2^8]$) are utilized in this MAC, this MAC-based signature possesses significant weaknesses. To which property of the design can these be ascribed, and where should one take care when constructing a cryptographic system? This essential property also applies for block ciphers and hash functions!

12.5. MACs are, in principle, also vulnerable against collision attacks. We discuss the issue in the following.

1. Assume Oscar found a collision between two messages, i.e.,

$$\mathrm{MAC}_k(x_1) = \mathrm{MAC}_k(x_2)$$

   Show a simple protocol with an attack that is based on a collision.
2. Even though the birthday paradox can still be used for constructing collisions, why is it in a practical setting much harder to construct them for MACs than for hash functions? Since this is the case: what security is provided by a MAC with 80-bit output compared to a hash function with 80-bit output?