SOLUTION NOTES

Introduction to Security 2002 Paper 3 Question 2 (MGK)

(*a*) (*i*) Collision resistance of a hash function h means that it is computationally infeasible to find a pair of inputs x and y such that $h(x) = h(y)$. In a digital signature application, an attacker could search for such a pair of texts (for example in a brute-force search by substituting synonyms), present one of them to the signer as a proposed contract, and present the other to the verifier with the claim that this is the signed contract.

(*ii*) First pad the message M in a reversible way to the next multiple of the block size, for instance by appending a one bit followed by between 0 and 63 zero bits. Cut the resulting padded text into 64-bit blocks $M_1, \ldots, M_n$. Set $H_0 = 0$ and $H_{i+1} = E_{M_i}(H_i) \oplus H_i$, where E is the DES encryption function and $\oplus$ is the 64-bit XOR operator. Use H_n as the hash value. [There are several other possible constructions.]

According to the Birthday Paradox, there is a chance of around $\frac{1}{2}$ to find a 64-bit hash collision in $\sqrt{2^{64}} = 2^{32}$ messages, which can be searched within a few hours on a low-cost personal computer. Therefore, no 64-bit hash function can be considered collision resistant.

(*b*) (*i*) electronic code book: 32 bits in P_3

(*ii*) cipher block chaining: 32 bits in P_3, 1 bit in P_4

(*iii*) 64-bit cipher feedback: 1 bit in P_3, 32 in P_4

(*iv*) 64-bit output feedback: 1 bit in P_3

(*v*) counter: 1 bit in P_3

(*b*) (*i*) The Feistel principle is a technique for constructing an invertible function (blockcipher) from non-invertible building blocks. The input block is split in two halves. The left half is fed into one of the non-invertible functions and the result XORed onto the right half. Then the left half and the new right half are swapped and the process applied again. After three rounds, all bits depend on all other bits, more rounds can be applied to add to the security of a cipher. The process can be reversed starting from the last round (for decryption).

(*ii*) Split the 300-bit input X of function P into three 100-bit blocks $X = X_1 \| X_2 \| X_3$. Then apply the following sequence of assignments:

$$\begin{aligned} X_2 &:= X_2 \oplus F(X_1) \\ X_3 &:= X_3 \oplus F(X_2) \\ X_1 &:= X_1 \oplus F(X_3) \\ X_2 &:= X_2 \oplus F(X_1) \\ X_3 &:= X_3 \oplus F(X_2) \end{aligned}$$

The concatenation $X_1 \| X_2 \| X_3$ of the result is the output of P.

[*This question refers mainly to the course chapters on "block ciphers", "Feistel network", "modes of operation", "hash functions", and "digital signatures".*]