Universität des Saarlandes
Lehrstuhl für Softwaretechnik
Prof. Andreas Zeller, Stephan Neuhaus
Design of Secure Software Systems
Exercise 4

Total number of points obtainable: 10, plus 0 bonus points.

1 Random Number Quality

Typical "random" quantities in cryptographic algorithms and protocols are cryptographic keys, IVs and nonces.

Let's say that a quantity is *ever-changing* if the probability to happen upon the same quantity twice in different contexts is negligible. For example, the quantity generated by $X_0 = 0$ and $X_{n+1} = (X_n + 1) \bmod 2^{64}$ is ever-changing because it repeats only after 2^{64} iterations.

Let's say that a quantity is *unpredictable* if the probability to guess the next element in a sequence is no better than random, even if you know all past elements of that sequence. For example, the sequence Y_n generated by $X_{n+1} = (X_n + 1) \bmod 2^{64}$, $Y_n = \text{MD5}(X_n)$ is unpredictable if we don't know X_0.

- **(2 points)** Prove that ever-changingness and unpredictability are independent properties by giving a sequence that is ever-changing but not unpredictable and a sequence that is unpredictable but not ever-changing.
- **(3 points)** Assume that a sequence is used to generate symmetric cryptographic keys. Does this sequence need to be ever-changing, unpredictable, or both?
- **(3 points)** Solve the previous exercise with initialization vectors (IVs) instead of keys.

Some cryptographic protocols require *nonces*, which are ever-changing quantities. However, some protocols also require unpredictability, others so not need this feature, so beware!

2 Message Digest Ciphers

(2 points) In the lecture, we have shown a way to turn a message digest into a block cipher. The idea was to use the fact that a message digest does not digest the entire message in one go, but rather in pieces, and therefore needs to keep some state from piece to piece.

More precisely, a hash function is defined as $h\colon \{0,1\}^k \times \{0,1\}^* \to \{0,1\}^k$, where its first argument is the state left over from the previous invocation, or a fixed initial state, respectively. To encrypt a message M, it is first split into chunks of k bits each: $M = (M_1, \ldots, M_n)$. Choose an arbitrarily long key K and an initialization vector *IV*. Then set

$$\begin{aligned} C_1 &= M_1 \oplus h(IV, K) \\ C_j &= M_j \oplus h(C_{j-1}, K) \qquad \text{for } 1 < j \le n, \end{aligned}$$

which is just like k-bit CFB mode. Prove that the recipient can recover the plaintext by computing

$$\begin{aligned} P_1 &= C_1 \oplus h(IV, K) \\ P_j &= C_j \oplus h(C_{j-1}, K) \qquad \text{for } 1 < j \le n. \end{aligned}$$