CHALMERS TEKNISKA HÖGSKOLA
Datavetenskap
Björn von Sydow INN150/TDA350/TDA351

# Solutions to exam in Cryptography 071220

1. (a) An $n$ bit LFSR has $2^n$ possible states. The output depends only on the state, so whenever the device returns to a previous state, we will have completed one period. Thus the period is at most $2^n$. However, the all zero state cannot appear in a maximal period sequence, since it would generate an all zero output.

   (b) If an LFSR has an odd number of taps and enters the state with all ones, then the bit to be shifted in is the xor of an odd nummber of ones and hence one. So, the device is stuck in this state.

2. (a) $M_0$ is retrieved by xoring the first equation with $IV$, so $M_0 = C_0 \oplus IV$. For the remaining blocks, one first decrypts $C_i$ and then xors to solve for $M_i$:

$$M_i \quad = \quad D_K(C_i) \oplus M_{i-1} \oplus C_{i-1}, \quad i = 1,2 \ldots n.$$

   (b) In ordinary CBC, a bit error in block $C_i$ during transmission causes block $M_i$ to be corrupted and a one bit error in $M_{i+1}$. In propagating CBC, the corrupted block $M_i$ is also xored with $M_{i+1}$, so also this, and all subsequent blocks, will be garbage.

3. Blinding means that the ciphertext is replaced by $c' = r^e \cdot c$. This is decrypted using ordinary RSA decryption:

$$m' = c'^d = (r^e m^e)^d = r \cdot m,$$

   so the final steps are to compute $r^{-1}$ (using the extended Euclidean algorithm) and then $m = r^{-1} \cdot m'$.

4. (a) Let the *toAccount* of $m$ be block $M_2$ and the adversary's account block $M_2'$. The adversary then replaces $C_2$ by

$$C_2' = C_2 \oplus M_2 \oplus M_2' = (M_2 \oplus K_2 \oplus M_2 \oplus M_2') = K_2 \oplus M_2',$$

   so when the message $C_1 C_2' C_3$ is decrypted it will show the adversary's account as *toAccount*.

   (b) The message could be authenticated by adding $MAC_{K_M}(M_1 M_2 M_3)$ after the message before encryption. If the receiver checks the MAC before accepting the message, the attack will be discovered.

   (c) We need to use the key to generate a keystream for xoring with the plaintext. One way to do this is to let $K_i = H(K||i)$. Then messages are split into blocks of the size of hash values: $m = M_1 M_2 \ldots M_n$ and $c = C_1 C_2 \ldots C_n$ where $C_i = M_i \oplus K_i$. Decryption is the same as encryption (so the receiver generates the same keystream, using the common key $K$).

5. (a) We want that $(m^e)^d = m^{ed} = m \in \mathbb{Z}_p^*$. For this we need $ed = 1$ in $\mathbb{Z}_{\Phi(p)}^* = \mathbb{Z}_{p-1}$. So, we compute $d = e^{-1}$ in $\mathbb{Z}_{p-1}^*$ using the extended Euclidean algorithm, which is possible if $\gcd(e, p-1) = 1$.

   (b) The difference is that in Pohlig-Hellman $\Phi(p)$ is known for everyone, so anyone who knows $e$ can compute $d$. In RSA, $\Phi(N) = (p-1)(q-1)$, so you must know the factorization of $N$ to compute $d$ from $e$.

6. (a) $A$ believes that she shares $K'_{AB}$ with $B$ since her nonce came back in message 2 encrypted with a key known only to $B$ (and $A$).

   $B$ believes that he shares $K'_{AB}$ with $A$ since $N_A$ was encrypted with $K'_{AB}$, which could only be retrieved from message 2 by someone who knows $K_{AB}$ (and this is known only by $A$ and $B$).

   $A$ believes that $K'_{AB}$ is fresh since it is included in message 2 together with $N_A$ (and hence message 2 must have been constructed after message 1 was sent).

   $B$ believes (indeed, knows) that $K'_{AB}$ is fresh since he chose it himself.

   (b) We consider the following interleaved runs of the protocol:

$$
\begin{array}{lll}
1. & A \longrightarrow C(B): & A, N_A. \\
1'. & C(B) \longrightarrow A: & B, N_A. \\
2'. & A \longrightarrow C(B): & \{N_A, K'_{AB}\}_{K_{AB}}. \\
2. & C(B) \longrightarrow A: & \{N_A, K'_{AB}\}_{K_{AB}}. \\
3. & A \longrightarrow C(B): & \{N_A\}_{K'_{AB}}.
\end{array}
$$

   $C$ cannot encrypt $A$:s nonce, so he needs to get help with message 2. He therefore starts a new run with $A$, letting $A$ do the encryption and reflecting the reply back. $A$ will accept the unprimed protocol run and believe that $B$ is present. (Of course, $C$ does not learn the session key, but that is another story.)

   (c) To prevent the attack, we need to be more explicit in the messages, e.g. by changing message 2 to include the sender and receiver (in this order), i.e. to be $\{A, B, N_A, K'_{AB}\}_{K_{AB}}$.

7. (a) To decrypt, the receiver first parses the ciphertext as $Y||C||T$. She then computes $Y^x = K$, and then $k_1||k_2 = H(K||Y)$. She can now check whether $T = MAC_{k_2}(C)$; if not the ciphertext is rejected. If the MAC is ok, the final step is to decrypt the message as $m = E_{k_1}^{-1}(C)$.

   (b) A cipher is CCA2 resistant (resistant against adaptive chosen ciphertext attacks) if an efficient adversary has no more than a negligible advantage over guessing in the following game against Alice:

   - The adversary chooses two ciphertext $m$ and $m'$ and gives them to Alice.
   - Alice chooses one of these at random and encrypts it, giving $c$.
   - The adversary chooses a number of ciphertexts (except $c$) and gets them decrypted by Alice. Finally, the adversary guesses whether $m$ or $m'$ was encrypted.

   (c) Instead of a subgroup of $\mathbb{Z}_p^*$ one chooses an elliptic curve group of prime order $q$ and with generator $g$. The secret key $x$ and the random $y$ is chosen in the same way as before, but modular exponentiation is replaced by multiplication with a constant. $X, Y$ and $K$ will be group elements, represented by their coordinates, but this does not prevent $K||Y$ from being hash function argument.

   The expected gain is increased efficiency because of smaller key length.