Introduction to cryptology
TD#2

2018-W05

Exercise 1: ECB, toy modes

Let $\mathscr{E} : \{0,1\}^\kappa \times \{0,1\}^n \rightarrow \{0,1\}^n$ be a block cipher. The ECB encryption of a message $m = m_0||m_1||\ldots$ (where all of the m_is are n-bit long) with $\mathscr{E}$ and a key k is given by $\mathscr{E}(k, m_0)||\mathscr{E}(k, m_1)\ldots$.

Q. 1: Explain why ECB is not a good mode (in particular why it is not IND-CPA).

We modify ECB to the following toy mode, that uses *domain separation* to solve some of the issues of ECB: the encryption of a message $m = m_0||m_1||\ldots$ (where all of the m_is are $n-b$-bit long) with $\mathscr{E}$ and a key k is given by $\mathscr{E}(k, m_0||t_0)||\mathscr{E}(k, m_1||t_1)\ldots$, where the t_is are b-bit pairwise-distinct values (for instance one can take $t_0 = 0$, $t_1 = 1$, etc.).

Q. 2: Give an upper-bound for the maximum message length that can be securely encrypted with this toy mode before having to change the key.

Q. 3: Are messages encrypted as above authenticated?

We modify again the toy mode. The encryption of a message $m = m_0||m_1||\ldots$ (where all of the m_is are $n-b-r$-bit long) with $\mathscr{E}$ and a key k is given by $\mathscr{E}(k, m_0||t_0||0^r)||\mathscr{E}(k, m_1||t_1||0^r)\ldots$, where the t_is are b-bit pairwise-distinct values and 0^r is a string or r zeros.

Q. 4: What is the probability that a uniformly random ciphertext corresponds to a message encrypted with the above toy mode? Explain how this allows to perform some authentication of the ciphertexts. Give a trivial (but limited) attack that may still be performed by an adversary.

Exercise 2: Arithmetic in $\mathbf{Z}/2^8\mathbf{Z}$ and $\mathbf{F}_{2^8}$

Q. 1: Compute the following in $\mathbf{Z}/2^8\mathbf{Z}$:

- $153 + 221$
- $29 + 8$
- $64 + 31$

Q. 2: Compute the following in $\mathbf{F}_{2^8}$ (where a decimal representation is used for the field elements, i.e. the addition corresponds to the bitwise XOR):

- $153 + 221$
- $29 + 8$
- $64 + 31$

Q. 3: Under what condition on their operands are the additions in $\mathbf{Z}/2^8\mathbf{Z}$ and $\mathbf{F}_{2^8}$ equivalent? (Prove it.)

Exercise 3: CTR mode

Let $\mathscr{E} : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^n$ be a block cipher. The CTR encryption of a message $m = m_0||m_1||\ldots$ (where all of the m_is are n-bit long) with $\mathscr{E}$ and a key k is given by $m_0 \oplus \mathscr{E}(k, t_0)||m_1 \oplus \mathscr{E}(k, t_1)\ldots$, where the t_is are n-bit pairwise-distinct values (for instance one can take $t_0 = 0$, $t_1 = 1$, etc.). In other words, one is encrypting a message with a pseudo-random keystream generated by $\mathscr{E}$.

Q. 1 : Show that the keystream used to encrypt a message of 2^n blocks (that is $n2^n$-bit long) is not perfectly random, if it is generated with a single key.

Hint: Exploit the fact that $\mathscr{E}(k,\cdot)$ is invertible.

We may try to solve the problem of the previous question by defining $\mathscr{E}'(k,x) := \mathscr{E}(k,x) \oplus x$. This makes $\mathscr{E}'$ non-injective. One may then still encrypt a message $m = m_0||m_1||\ldots$ as $m_0 \oplus \mathscr{E}'(k,t_0)||m_1 \oplus \mathscr{E}'(k,t_1)\ldots$.

Q. 2 : Show that if the t_i values are public, then $\mathscr{E}'$ suffers from the same problem as $\mathscr{E}$ in Q. 1.

(However, it can be shown that if the t_is are secret and "random" enough (for instance $t_i = \mathscr{E}''(k, t'_i)$ where the t'_is are pairwise distinct), then $\mathscr{E}'$ does achieve better security than $\mathscr{E}$ in CTR mode.)

Exercise 4: Bit-vector arithmetic

Q. 1: Write a small "naïve" C function that computes the scalar product of two vectors of $\mathbf{F}_2^{32}$. This function must have the following prototype:

```
uint32_t scalar32_naive(uint32_t x, uint32_t y).
```

Q. 2: Write another implementation of the same function, of prototype

```
uint32_t scalar32_popcnt(uint32_t x, uint32_t y),
```

that uses a *bitwise and* instruction "`&`" and the *population count* function for 32-bit words "`__builtin_popcount()`".

Q. 3 Explain why in C, assuming that `x` is of type `uint32_t`, `x << 1` computes the multiplication of `x` by two in $\mathbf{Z}/2^{32}\mathbf{Z}$.

Q. 4 Explain why in C, assuming that `x` is of type `uint32_t`, `x >> 1` is equivalent to `x / 2`.

Q. 5 Write the matrix M of dimension 8 over $\mathbf{F}_2$ such that $M\mathbf{x} =$ `mul2(x)`, where `mul2` is defined as:

```
uint8_t mul2(uint8_t x)
{
  return ((x << 1) & 0xFF);
}
```

and $\mathbf{x}$ and `x` are in natural correspondence (with the encoding convention that $\mathbf{x} = \begin{pmatrix}\mathbf{x}_0 & \mathbf{x}_1 & \ldots \mathbf{x}_7\end{pmatrix}^t \mapsto \mathbf{x}_7 2^7 + \mathbf{x}_6 2^6 + \ldots + \mathbf{x}_0 2^0$). Is this matrix invertible?

Q. 6 What are the logical formulas computed by the following functions on their inputs?

```
uint32_t f1(uint32_t x, uint32_t y, uint32_t z)
{
  return ((x & y) | (~x & z));
}

uint32_t f2(uint32_t x, uint32_t y, uint32_t z)
{
  return ((x & y) | (x & z) | (y & z));
}

uint32_t f2(uint32_t x, uint32_t y, uint32_t z)
{
  return (z ^ (x & (y ^ z)));
}
```

Which of these functions can be computed as matrices?