Cryptology I (spring 2018) Dominique Unruh

Exercise Sheet 6

Out: 2018-03-30 Due: 2018-04-06

Problem 1: Hybrid encryption – implementations

(a) Implement a hybrid encryption using ElGamal and AES. You are allowed to use ready-made ElGamal and AES.

In the contributed file `hybrid.py` (lecture webpage), you find a prepared template in Python that already provides function for ElGamal and AES encryption as well as some utility functions and testing code that you might need. I recommend to use that code. If you wish to use another language, you will have to find your own ElGamal and AES routines.

You should check that `hybrid_decrypt(sk,hybrid_encrypt(pk,msg))` returns `msg`.

It is OK if you only allow encrypting messages whose length is a multiple of 16 bytes (blocklength of AES).

(b) **[Bonus problem]** The ElGamal implementation used in `hybrid.py` might leak whether the message `msg` is a quadratic residue. Using the methods developed in **??**, fix the functions `elgamal_encrypt` and `elgamal_decrypt` to avoid this leakage. (You need to make sure that `elgamal_decrypt(sk,elgamal_encrypt(pk,msg))` still returns `msg`.)

Problem 2: Encoding messages for ElGamal (bonus problem)

The message space of ElGamal (when using the instantation that operates modulo a prime $p > 2$ with $p \equiv 3 \bmod 4$ [1]) is the set $\mathrm{QR}_p = \{x^2 \bmod p : x = 0, \dots, p-1\}$.

The problem is now: if we wish to encrypt a message $m \in \{0,1\}^\ell$ (with $\ell \leq |p| - 2$), how do we interpret m as an element of QR_p?

One possibility is to use the following function $f : \{1, \dots, \frac{p-1}{2}\} \to \mathrm{QR}_p$:

$$f(x) := \begin{cases} x & \text{if } x \in \mathrm{QR}_p \\ -x \bmod p & \text{if } x \notin \mathrm{QR}_p \end{cases}$$

Once we see that f is a bijection and can be efficiently inverted, the problem is solved, because a bitstring $m \in \{0,1\}^\ell$ can be interpreted as a number in the range $1, \dots, \frac{p-1}{2}$ by simply interpreting m as a binary integer and adding 1 to it. (I.e., we encrypt $f(m+1)$.)

[1] You do not actually need to use this fact, but the hint that $-1 \notin \mathrm{QR}_p$ below is only true in this case.

We claim that the following function is the inverse of f:

$$g(x) := \begin{cases} x & \text{if } x = 1, \ldots, \frac{p-1}{2} \\ -x \bmod p & \text{if } x \neq 1, \ldots, \frac{p-1}{2} \end{cases}$$

We thus need to show the following: the range of f is indeed QR_p, and that $g(f(x)) = x$ for all $x \in \{1, \ldots, \frac{p-1}{2}\}$.

(a) Show that $f(x) \in \mathrm{QR}_p$ for all $x \in \{1, \ldots, \frac{p-1}{2}\}$.

Hint: You can use (without proof) that $-1 \notin \mathrm{QR}_p$ (this only holds in QR_p for p prime with $p \equiv 3 \bmod 4$). And that the product of two quadratic non-residues is a quadratic residue (this only holds in QR_p, but not in QR_n for n non-prime).

(b) Show that $g(f(x)) = x$ for all $x \in \{1, \ldots, \frac{p-1}{2}\}$.

(This then shows that f is injective and efficiently invertible. Bijectivity follows from injectivity because the domain and range of f both have the same size.)

Hint: Make a case distinction between $x \in \mathrm{QR}_p$ and $x \notin \mathrm{QR}_p$. Show that for $x \in \{1, \ldots, \frac{p-1}{2}\}$ it holds that $-x \bmod p \notin \{1, \ldots, \frac{p-1}{2}\}$.

Problem 3: Authentication in WEP

In the WEP-protocol (used for securing Wifi, now mostly replaced by WPA), messages are "encrypted" using the following procedure: First, a key k is established between the parties A and B. (We do not care how, for the purpose of this exercise we assume that this is done securely.) Then, to transmit a message m, A chooses an initialization vector IV (we do not care how) and sends IV and $c := \mathit{keystream} \oplus (m \| \mathit{CRC}(m))$. Here *keystream* is the RC4 keystream computed from IV and k (we do not care how).

The function CRC is a so-called cyclic redundancy check, a checksum added to the WEP protocol to ensure integrity. We only give the important facts about CRC and omit a full description. Each bit of $\mathit{CRC}(m)$ is the XOR of some of the message bits. Which messages bits are XORed into which bit of $\mathit{CRC}(m)$ is publicly known. (In other words, the i-th bit of $\mathit{CRC}(m)$ is $\bigoplus_{j \in I_i} m_j$ for a publicly known I_i.)

An adversary intercepts the ciphertext c. He wishes to flip certain bits of the message (i.e., he wants to replace m by $m \oplus p$ for some fixed p). This can be done by flipping the corresponding bits of the ciphertext c. But then, the CRC will be incorrect, and B will reject the message after decryption! Thus the CRC *seems* to ensure integrity of the message and to avoid malleability. (This is probably why the designers of WEP added it here.)

Show that the CRC does not increase the security! That is, show how the adversary can modify the ciphertext c such that c becomes an encryption of $m \oplus p$ and such that the CRC within c is still valid (i.e., it becomes the CRC for $m \oplus p$).

Hint: Think of how the i-th bit of $\mathit{CRC}(m \oplus p)$ relates to the i-th bit of $\mathit{CRC}(m)$. (Linearity!)

Problem 4: Collisions in Iterated Hash

Let E be a block cipher with n-bit keys and messages. Assume the following compression function:

$$F(x\|y) := E(y, x) \oplus x.$$

(y is used as the key for E.) This is a very slight variation of the Davies-Meyer compression function.

Let H be the Iterated Hash construction using compression function F.

Assume the designer of the standard happens to have chosen the initialization vector iv as $iv := D(z, 0)$ for random z. (Here D is the decryption corresponding to E.) The designer justified this with the fact that this basically leads to a random iv.

Describe how to (efficiently) find a collision for H.

Note: You can assume that you know which z the designer used.

Hint: First explain how to efficiently construct x^* as describe in the lecture in the attack on Iterated Hash.