RUHR UNIVERSITÄT BOCHUM RUB

Lehrstuhl für Kryptologie und IT-Sicherheit
Prof. Dr. Alexander May
Elena Kirshanova

Hausübungen zur Vorlesung

Kryptanalyse I

SS 2015

Blatt 2 / 7. May 2015

Abgabe bis: 21. May 12:00 Uhr, Kasten NA/02

Aufgabe 1 (5 Punkte):
Why not to choose primes close to $\sqrt{N}$ for RSA.
Assume one of the RSA primes is close to $\sqrt{N}$: $|p - \sqrt{N}| < \sqrt[4]{N}$. Show how to factor N in polynomial time.
Hint. You might want yo use the following fact: for $N = pq$, $N = \left(\frac{p+q}{2}\right)^2 - \left(\frac{p-q}{2}\right)^2$. Note that the first summand is $\approx \sqrt{N}$, while the second one is small.

Aufgabe 2 (8 Punkte):
Why not to share N among several users in RSA. Give a probabilistic polynomial time algorithm that finds a non-trivial divisor of N, having as input an RSA key-pair (e, d).

Aufgabe 3 (5 Punkte):
Meet-in-the middle on El-Gamal. Given an El-Gamal ciphertext $(\alpha^r, \alpha^{rx}m)$ for the message m, where $\langle\alpha\rangle = \mathbb{Z}_p^*$, give a meet-in-the-middle type of attack on either r, x or m. Explain your choice and give a complexity estimate for your attack.

Aufgabe 4 (7 Punkte):
In this exercise, you will develop and analyze an algorithm to evaluate a polynomial $f(x)$, of degree less than $n = 2^k$ in n points $u_1, \ldots, u_n$ in $\mathcal{O}(n \log^2 n)$ time using $\widetilde{\mathcal{O}}(n)$ memory:

1. Show that $f(x) \mod (x - c) = f(c)$ for some constant c;

2. Let us define polynomials

$$P_{i,j} = \prod_{l=0}^{2^i-1} (x - u_{j\cdot 2^i + l}), \quad 0 < i < k,\ 0 < j < 2^{k-i}$$

   whence

$$P_{0,j} = (x - u_j), \quad 0 < j < k.$$

   Show that

$$P_{i+1,j} = P_{i,2j} \cdot P_{i,2\cdot j+1}.$$

   Show how to construct all $P_{i,j}$ in time $\mathcal{O}(\mathsf{Mul}(n) \log n)$, where $\mathsf{Mul}(n)$ is time to multiply two polynomials of degree n.

3. Using the construction from above and 1., devise a recursive algorithm that computes $f(u_1), \ldots, f(u_n)$. What is the running time?

Aufgabe 5 (10 Punkte)**:**

Programming assignment: Bleichenbacher attack. Here is another version of an adaptive CCA-attack on the RSA cryptosystem published in [1] on PKCS # 1. The weakness was hidden in the way the RSA formatted an input message : for a modulus $N < 2^{8k}$ of k bytes and a message $m < 2^{8k-11}$, the encryption block $EB = 00\|02\|\mathsf{padding}\|00\|m$ is formed, where padding has 8 bytes size. Decryption succeeds if and only if the underlying plaintext is of this special form (called PKCS conformed), otherwise the error is return.

Observe, that given a ciphertext c^* (assume it is a proper ciphertext and the corresponding plaintext is PKCS conformed), you can multiply it by any other ciphertext $c_0 = m_0^e \mod N$ and check whether $c_0 \cdot c^* \mod N$ is PKCS conform or not. Once you found c_0 s.t. $c_0 \cdot c^* \mod N$ is PKCS conformed, you can deduce some partial information on bytes of the challenge c^*.

In this homework, we simplify the task slightly, preserving the idea of the attack. Here, you are given an access to the oracle that checks the *Most Significant Bit* of the plaintext (for a given ciphertext) and answers 'Conform' if MSB(`dec(c)`) == 1. Note that now you are not allowed to query the decryption oracle, but the ability to extract just a bit of information is enough for the total break.

As in HW1, you will find N, e, c^* in 'params.txt'. The file 'dec.o' provides

```
bool IfConform (mpz_t c)
```

and return 1 if MSB(`m` = `dec(c)`) == 1, otherwise 0.

Your task is to find $m =$ `Dec(c*)`. You can follow the instructions from HW1. Submit your code!

Note: if you encounter numerical instabilities while getting *all* the bits, submit your code with a partial output.

Literatur

[1] Daniel Bleichenbacher, Chosen Ciphertext Attacks Against Protocols Based on the RSA Encryption Standard PKCS1, 1998