Advanced Cryptography: Algorithmic Cryptanalysis

DANIEL LOEBENBERGER, KONSTANTIN ZIEGLER

8. Exercise sheet
Hand in solutions until Saturday, 04 June 2011, 23:59h.

To estimate the average effort you put into solving the following exercises, please add after each exercise the amount of time you spent for it.

Exercise 8.1 (Gram-Schmidt orthogonalization). (8 points)

Consider the Gram-Schmidt orthogonalization from the lecture. There we constructed, given a basis $B \in \mathbb{R}^{n \times m}$ of the vector space $V := \text{span}(B)$, an orthogonal basis B^* by defining $b_1^* := b_1$, $b_i^* := b_i - \sum_{j<i} \mu_{i,j} b_j^*$ with $\mu_{i,j} := \frac{\langle b_i, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle}$.

(i) Show that for $i_1 \neq i_2$ the vectors $b_{i_1}^*$ and $b_{i_2}^*$ are orthogonal. [3]

(ii) Show that for $i < j$ the vectors b_i and b_j^* are orthogonal. [3]

(iii) Consider the vector space $V = \text{span}(B)$, spanned by the basis [2]

$$B := \begin{bmatrix} 2 & 1 & 2 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{bmatrix}.$$

Compute an orthogonal basis of V.

Exercise 8.2 (A note on the volume). (5 points)

Let $B \in \mathbb{R}^{n \times m}$ a basis of the lattice $L = B$ and let B^* be the Gram-Schmidt matrix of B. We have defined the determinant of the lattice as $\det(L) = \text{vol}(P(B)) = \sqrt{\det(B^T B)}$. Prove that $\det(L) = \prod_i \|b_i^*\|$. Hint: Use the fact that $B^* = TB$ for some lower triangular matrix T with $T_{i,i} = 1$ for all $i = 1 \ldots m$. [5]

Exercise 8.3 (Breaking the knapsack cryptosystem). (26+10 points)

(i) NTL is a high-performance, portable C++ library providing data structures and algorithms for manipulating signed, arbitrary length integers, and for vectors, matrices, and polynomials over the integers and over finite fields. It has a highly optimized built in basis reduction algorithm that is suitable for the following tasks we have in mind. To start, install NTL on your computer and get familiar with the NTL-API. Hints how [+5]

to install NTL and details on the API can be found on `http://www.shoup.net/ntl/doc/tour.html`. Now run the code `lll.cpp` from the course page. To compile it, call for example under UNIX (or Mac OS X) the compiler in the following way: `g++ -o lll lll.cpp -lntl -lm`. You might have to include the headers using the $-I$ flag and the library using the $-L$ flag. Details on that can be found in the `man` page of `g++`. Consider now the lattice spanned by the matrix (written in row notation)

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}.$$

Hand in the output of the supplied program.

Goal of the following tasks is to become acquainted with the knapsack cryptosystem, implement and brake it. Let $m = 437$ and $c = 204$. Bob's private key is $b = (2, 6, 10, 26, 68, 161)$.

2 (ii) Compute Bob's public key a.

Now Alice wants to send the string $x = (0, 1, 0, 1, 1, 0)$.

1 (iii) Encrypt x with Bob's public key obtaining y.

3 (iv) Describe in detail how Bob will decrypt the encrypted message y and do the decryption.

Now realize the following three functions:

Algorithm. Generate Key Pair.
Input: A positive integer n.
Output: The private key (b, m, c) and the public key a. The private key consists of a superincreasing sequence $b = (b_1, \ldots, b_n)$ with $b_i \in \left(\sum_{j<i} b_j, 2\sum_{j<i} b_j\right)$, a value $m \in \left(\sum_{j\leq n} b_j, 2\sum_{j\leq n} b_j\right)$ and a value $c \in \mathbb{N}$ with $\gcd(c, m) = 1$. The public key is a sequence $a = (cb_1 \pmod m, \ldots, cb_n \pmod m)$.

Algorithm. Encrypt.
Input: A message $x \in \{0, 1\}^n$. The public key a.
Output: The encrypted message $y = \sum_{i\leq n} x_i a_i$.

Algorithm. Decrypt.
Input: A message $y \in \mathbb{N}$. The private key (b, m, c).
Output: The decrypted message x.

In the lecture we have seen the following algorithm that computes (sometimes) a solution to the knapsack problem:

Algorithm. Short vectors for subset sums.

Input: Positive integers $a_0, a_1, \ldots, a_n$.
Output: $(x_1, \ldots, x_n) \in \mathbb{Z}^n$ or "failure".

1. Let $M = \lceil 2^{(n-1)/2} n^{1/2} \rceil$.
2. If $a_0 < \sum_{1 \le i \le n} a_i/2$ then $a_0 \longleftarrow \tilde{a}_0 = \sum_{1 \le i \le n} a_i - a_0$ else $\tilde{a}_0 = 0$.
3. For $0 \le i \le n$, let $b_i \in \mathbb{Z}^{n+1}$ be as follows:

$$b_0 = (a_0 M, 0, \ldots, 0),$$
$$b_i = (-a_i M, 0, \ldots, 0, 1, 0, \ldots, 0) \text{ with 1 in position } i, \text{ for } 1 \le i \le n.$$

4. Let $L \subseteq \mathbb{Z}^{n+1}$ be the lattice generated by $b_0, \ldots, b_n$. Run the basis reduction on this basis and return a short nonzero vector $v = (v_0, \ldots, v_n) \in L$.
5. If $\tilde{a}_0 = a_0$ then For $0 \le i \le n$, let $v_i \longleftarrow 1 - v_i$
6. If $v \in \{0,1\}^{n+1}$ then return $(v_1, \ldots, v_n)$ else return "failure".

Implement also this algorithm.

(v) Assume you have intercepted the message $y = 1147$. Bob's public key is now [20]

$$a = (465, 441, 417, 241, 330, 251).$$

Use your implementation to compute the message $x \in \{0,1\}^6$ that Alice sent to Bob using the above algorithm.

(vi) Take $n = 6$ and $B = 36238786559$. Run 100 examples with $a_1, \ldots, a_6 \xleftarrow{\$} \{1, \ldots, B\}$ and $x \xleftarrow{\$} \{0,1\}^6 \setminus \{(0, \ldots, 0)\}$. How often did your algorithm not succeed in finding x? [+5]