# Лабораторная работа № 3

# Факторизация на решётках

**К сдаче:** 25.04.2022

## 1 Алгоритм факторизации Шнорра

Лабораторная работа вдохновлена пре-принтом К.П. Шнорра [1], вызвавший большой резонанс [2, 3, 4]. В частности, аннотация статьи утверждает, что "This [атака] destroys the RSA cryptosystem." Суть этой лабораторной - попытаться факторизовать RSA модуль с помощью решёток. Вы увидите, что этот алгоритм, несмотря на вышеупомянутое утверждение, не несёт опасность современным параметрам криптосистемы RSA.

Алгоритм имеет длинную историю [5], мы будем придерживаться описания из [6]

В основе алгоритма (как и для продвинутых алгоритмов Number Field Sieive), лежит идея поиска пар $(x, y)$, удовлетворяющих

$$x^2 \equiv y^2 \bmod N. \tag{1}$$

Если $x \neq \pm y \bmod N$, то вычисление $\gcd(N, x + y)$ даст нетривиальный делитель $N$. Метод Шнорра ищет такие пары с доп. условие гладкости.

Число называется $B$-*гладким*, если все его простые делители меньше $B$. Обозначим $p_i$–$i$-ое простое число и зафиксируем некоторое целое $d > 1$. Основная вычислительная задача алгоритма Шнорра состоит в поиске четверок $(u, v, k, \gamma)$, таких что 1. $u, v, k$–$p_d$-гладкие; 2. $\gamma \geq 1$ – целое; 3. выполняется Диофантово уравнение

$$u = v + kN^{\gamma}.$$

Эти четверки будут находится с помощью коротких векторов решетки специального вида. А именно, рассмотри решетку, порожденную **столбцами** матрицы $A$:

$$A = \begin{pmatrix} \ln p_1 & 0 & \dots & 0 & 0 \\ 0 & \ln p_2 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & \ln p_d & 0 \\ C \ln p_1 & C \ln p_2 & \dots & C \ln p_d & C \ln N \end{pmatrix} \in \mathbb{R}^{d+1 \times d+1},$$

где $C$–некая (достаточно большая) константа. Заметьте, что базис решётки нецелочисленный. Для вектора $\mathbf{z} \in \mathbb{Z}^{d+1}$, справедливо

$$A\mathbf{z} = \begin{pmatrix} z_1 \ln p_1 \\ \vdots \\ z_d \ln p_d \\ C(\sum_i z_i \ln p_i + z_{d+1} \ln N). \end{pmatrix}$$

Если мы будем ассоциировать с вектором $\mathbf{z}$, элементы $u, k, \gamma$ следующим образом

$$u = \prod_{z_i > 0} p_i^{z_i}, \quad k = \prod_{z_i < 0} p_i^{-z_i} \quad \text{и} \quad \gamma = |z_{d+1}|,$$

то

$$\|A\mathbf{z}\|_1 = \sum_i^d |z_i| \ln p_i + C \left| \sum_i^d z_i \ln p_1 - |z_{d+1}| \ln N \right|,$$

и

$$\|A\mathbf{z}\|_1 = \ln u + \ln k + C \left| \ln u - \ln(kN^\gamma) \right|.$$

Отсюда, если $\|A\mathbf{z}\|_1$ – мала, то можно доказать следующее утверждение (см. [6] для доказательства):

Если $\|A\mathbf{z}\|_1 \leq 2C + 2\sigma \ln p_d - \gamma \ln N$, то $|u - kN^\gamma| < p_d^\sigma$.

Делаем вывод: чтобы найти четверку $(u, v, k, \gamma)$, необходимо найти короткий вектор в решетке, порожденной столбцами $A$. Как найти из такой четверки $(x, y)$, удовлетворяющие Сравнению (1)?

Найдем несколько различных четверок $(u, v, k, \gamma)$. Для каждой такой четверки положим $a_{i,j} = z_j$ для $z_j > 0$ (те $z_j$, что участвуют в записи $u_i$), а за $b_{i,j}$ обозначим степени в разложении $v_i = u_i - k_i N_i^\gamma = \prod_i p_i^{b_{i,j}}$ (для $i \leq d+1$). Обозначим далее вектора $\mathbf{a}_i = (a_{i,1}, \ldots, a_{i,d})$, $\mathbf{b}_i = (b_{i,1}, \ldots, b_{i,d})$. Их них, построим матрицу $M \in \mathbb{Z}^{d \times d}$, $i$-ый столбец которой есть $\mathbf{a}_i + \mathbf{b}_i$.Наша цель – построить матрицу $M \in \mathbb{Z}^{d \times d+1}$ полного ранга над $GF(2)$, то есть ранга $d$. Заметьте, что вам может понадобиться больше чем $d+1$ четверок $(u, v, k, \gamma)$.

Для всякого ненулевого вектора $\mathbf{c} \in \{0,1\}^{d+1}$, удовлетворяющего (т.к. $M$ полного ранга, такой $\mathbf{c}$ найдётся )

$$M \cdot \mathbf{c} \equiv 0 \bmod 2,$$

положим

$$x = \prod_{j=1}^d p_j^{\sum_{i=1}^{d+1} c_i (a_{i,j} + b_{i,j})/2} \bmod N \qquad y = \prod_{j=1}^d p_j^{\sum_{i=1}^{d+1} c_i a_{i,j}} \bmod N.$$

Если $x \neq \pm y \bmod N$, вычислим нетривиальный делитель $N$ как $\gcd(N, x+y)$.

### 1.1 Предложение Шнорра 2021

Алгоритм, описанный выше, требует поиска нескольких (минимум $d+1$) коротких векторов для решетки, порожденной матрицей $A$. Для этого, например, можно использовать алгоритм просеивания, возвращающий (почти) все короткие вектора решетки [7] (см. одну из предыдущих практик `https://crypto-kantiana.com/elena.kirshanova/teaching/lattices_2022/sieving.py`).

Шнорр в [1] предлагает рандомизировать $A$ следующим образом. Выбираем случайную перестановку $f : [1, d] \to [1, d]$ и строим $A'$

$$A' = \begin{pmatrix} CNf(1) & 0 & \ldots & 0 & 0 \\ 0 & CNf(2) & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & 0 \\ 0 & 0 & \vdots & CNf(d) & 0 \\ CN\ln p_1 & CN\ln p_2 & \ldots & CN\ln p_d & CN\ln N, \end{pmatrix}$$

для какого-то (большого) $C$, ответственного за корректность округления. Заметьте, что здесь все элементы матрицы умножаются на $CN$. Чтобы получить целочисленную матрицу, её элементы можно округлить для достаточно большого $C$ ($C > 2^{300}$).

Найдя короткий вектора этой решетки $A\mathbf{z}$, "уберем" скаляр $NC$, положив $\mathbf{v}' = A\mathbf{z}/\text{round}(NC)$ для всех коэффициентов $A\mathbf{z}$, кроме последнего (т.е. вектор $\mathbf{v}'$ размера $d$). Заметьте, что благодаря диагональной форме $A$, мы таким образом получим и (скалированный) вектор коэффициентов. Представим

$$u = \prod_{i:\mathbf{v}'_i > 0} p_i^{\mathbf{v}'_i} \qquad k = \prod_{i:\mathbf{v}'_i < 0} p_i^{-\mathbf{v}'_i}.$$

Из найденных значений $(u, v)$ строим $(x, y)$ аналогично процедуре выше, *если* $(u - kN)$ — $p_d$−гладкое число.

Преимущество метода Шнорр заключается в том, что если мы не нашли достаточное количество значений $(u, v)$, мы можем повторить всю процедуру для другой перестановки $f$.

### 1.2 Задание

Используя **любой** из подходов и их улучшений (смотрим ссылки), факторизовать 40-битный RSA-модуль $N$ (то есть $N = pq$, где $p \neq q$ – простые числа по $\approx 20$ бит каждое).

Комментарии к заданию:

- Для работы с простыми числами можете использовать функции randprime, nextprime библиотеки sympy, для реализации $f$ можете пользоваться функцией shuffle библиотеки random.
- Для факторизации 40-битного числа на практике хватает $d < 28$ простых для $< 20$ рандомизаций
- Для получения множества коротких векторов можете использовать алгоритм просеивания `https://crypto-kantiana.com/elena.kirshanova/teaching/lattices_2022/sieving.py` или алгоритм перечисления `https://crypto-kantiana.com/elena.kirshanova/teaching/lattices_2022/enumeration_example.py`

**Бонус.** Команде[1], успешно факторизовавшей 80-битный RSA-модуль $N$ алгоритмом Шнорра, положен +1 балл на экзамене.

## Список литературы


[1] Claus Peter Schnor. *Fast Factoring Integers by SVP Algorithms.* `https://eprint.iacr.org/2021/933`

[2] `https://twitter.com/inf_0_/status/1367376526300172288`

[3] `https://twitter.com/kennyog/status/1367132559117848583`

[4] `https://crypto.stackexchange.com/questions/88582/does-schnorrs-2021-factoring-method-show-that-t 88647#88647`

[5] `https://github.com/lducas/SchnorrGate`

[6] Antonio Vera. *A note on integer factorization using lattices* `https://arxiv.org/pdf/1003.5461.pdf`

[7] `https://github.com/fplll/g6k`


[1] Команда состоит из 1-2 человек в неизменном составе