Отчёт по лабораторной работе №7

Дисциплина: Математические основы защиты информации и информационной безопасности

Полиенко Анастасия Николаевна, НПМмд-02-23

Содержание

1 Цель работы **4**

2 Задание **5**

3 Теоретическое введение **6**

4 Выполнение лабораторной работы **7**

5 Выводы **9**

Список литературы **10**

Список иллюстраций

4.1 ρ-метод Полларда. 8

1 Цель работы

Изучить дискретное логарифмирование в конечном поле.

2 Задание

Реализовать алгоритм p-метода Полларда.

3 Теоретическое введение

Задача дискретного логарифмирования применяется во многих алгоритмах криптографии с открытым ключом. Предложенная в 1976 году У. Дифии и М. Хеллманом для установления сеансового ключа, эта задача послежила основой для создания протоколов шифрования и цифровой подписи, доказательств с нулевым разглашением и других криптографических протоколов.

Обозначим $F_p = Z/pZ, p$ - простое целое число и назовем конечным полем из p элементов. Задача дискретного логарифмирования в конечном поле F_p формулируется так: для данных целых чисел a и $b, a > 1, b > p$, найти логарифм - такое целое число x, что $a^x \equiv b(mod\ p)$ (если такое число существует). По аналогии с вещественными числами используется обозначение $x = log_a b$.

Безопасность соответствующих криптосистем основана на том, что зная числа a, x, p вычислить $a^x(mod\ p)$ легко, а решить задачу дискретного логарифмирования трудно. Рассмотрим p-метод Полларда, который можно применить и для задач дискретного логарифмирования. При этом случайное отображение f должно обладать не только сжимающими свойствами, но и вычислимостью логарифма (логарифм числа $f(c)$ можно выразить через неизвестный логарифм x и $log_a f(c)$).

Более подробно см. в [1–6].

4 Выполнение лабораторной работы

Реализуем алгоритм алгоритм p-метода Полларда. (рис. 4.1)

```python
def f(x, u, v):
    if x < 53:
        return (10 * x) % 107, u + 1, v
    return (64 * x) % 107, u, v + 1

p = 107
a = 10
b = 64
r = 53
u = 2; v = 2
u_c = 2; v_c = 2
u_d = 2; v_d = 2

c = ((a ** u) * (b ** v)) % p
d = c

c, u_c, v_c = f(c, u_c, v_c)
d, u_d, v_d = f(f(d, u_d, v_d)[0], f(d, u_d, v_d)[1], f(d, u_d, v_d)[2])

while c % p != d % p:
    c, u_c, v_c = f(c, u_c, v_c)
    d, u_d, v_d = f(f(d, u_d, v_d)[0], f(d, u_d, v_d)[1], f(d, u_d, v_d)[2])
    print(c, u_c, v_c, d, u_d, v_d)

x = 1
while (u_c + v_c * x) % r != (u_d + v_d * x) % r:
    x += 1

print("x =", x)
```

```
79 4 2 56 5 3
27 4 3 75 5 5
56 5 3 3 5 7
53 5 4 86 7 7
75 5 5 42 8 8
92 5 6 23 9 9
3 5 7 53 11 9
30 6 7 92 11 11
86 7 7 30 12 12
47 7 8 47 13 13
x = 20
```

Рис. 4.1: p-метод Полларда.

5 Выводы

Изучила дискретное логарифмирование в конечном поле.

Список литературы

1. GNU Bash Manual [Электронный ресурс]. Free Software Foundation, 2016. URL: https://www.gnu.org/software/bash/manual/.
2. Newham C. Learning the bash Shell: Unix Shell Programming. O'Reilly Media, 2005. 354 с.
3. Zarrelli G. Mastering Bash. Packt Publishing, 2017. 502 с.
4. Robbins A. Bash Pocket Reference. O'Reilly Media, 2016. 156 с.
5. Таненбаум Э. Архитектура компьютера. 6-е изд. СПб.: Питер, 2013. 874 с.
6. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.