Московский авиационный институт (национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №1 по курсу «Криптография»

Студент: А. В. Тимофеев
Преподаватель: А. В. Борисов
Группа: М8О-307Б-19

Дата:
Оценка:
Подпись:

Москва, 2022

Вариант №1

Задача:

Разложить каждое из чисел n1 и n2 на нетривиальные сомножители. Ниже представлены 20 вариантов.
Вариант соответствует номеру по списку группы. Если номер >=20, то взять остаток от деления на 20.

1) n1=568441757721070712527092775639582616443280731324663783129363506250326239368337 3,
n2=330202295900030604612878387051742686153612741688512674640516339592111786182112102952
744205334221107472202527866641026738964833952954682296035526349373351638998847856316617 3
570125125320373840042467197427715570566783354934876492962376348888456514955245307546351 3
219900132801375737362944845497774380469071477460824975475747214155932801784175961183689 6
600544007093551688480761463093260025078609841325445558002874051585803157223276060698810 5
994915916825321411634327591,

1 Описание

Процесс разложения числа на его простые множетели называется факторизацией. Для решения этой задачи существует множество алгоритмов, позволяющих находить множетели, используя свойства простых чисел.

Так как преподаватель не ограничил нас в выборе средств для решения задачи, я решил прибегнуть к готовому решению, онлайн калькулятору [1]: «Integer factorization calculator». В этом калькуляторе реализован общий метод решета числового поля. Этот метод считается одним из самых эффективных современных алгоритмов факторизации. Он справился с поставленной задачей для 1-го числа примерно за 2 минуту, что является впечатляющим результатом.
Однако 2 число имеет более 400 квадратичных знаков, факторизация которого на обычном компьютере за разумное время практически невозможна ни одним из ныне существующих алгоритмов. Однако я узнал что один из множетелей этого числа определяется к наибольший общий делитель с одним из чисел другого варанта. Поэтому я быстро написал программу, пребирающую все числа других варантов, определяющую их НОД с числом моего варанта и выводящий его, если он > 1. Второе же число определяется как результат деления числа моего варанта на НОД (по свойству делителя). Для работы с большими числами и поиска делителя в этой программе я использовал библиотеку gmp.

2 Исходный код

Программа для подбора нетривиальных сомножителей для n2.

```
#include <iostream>
#include <string>
#include <vector>
#include <csignal>
#include <ctime>
#include <cmath>
#include <gmpxx.h>

std::vector<mpz_class> pars (std::string inputStr){
    std::vector<mpz_class> n;
    bool strN2 = false;
    mpz_class tmp;
    tmp = 0;

    for(std::string::iterator i = inputStr.begin(); i != inputStr.end(); ++i){
        if (*i == '='){
            tmp = 0;
            ++i;
            while (*i != ','){
                tmp = tmp * 10 + (*i) - '0';
                ++i;
            }
            n.push_back(tmp);
        }
    }
    return n;
}

using namespace std;

int main(){
    mpz_class number, result;
    number = "
        330202295900030604612878387051742686153612741688512674640516339592111786182112102952744205334221107472202527866641026738964833952954682296035526349373351638998847856316617357012512532037384004246719742771557056678335493487649296237634888845651495524530754635132199001328013757373629448454977743804690714774608249754757472141559328017841759611836896600544007093551688480761463093260025078609841325445558002874051585803157223276060698810599491591682532141163432759";
    vector<mpz_class> n;
    string tmpStr;
    string numStr("\0");

    while (cin >> tmpStr){
        numStr = numStr + tmpStr;
    }
```

```
46      n = pars(numStr);
47      for(unsigned i = 0; i < n.size(); ++i){
48          mpz_gcd(result.get_mpz_t(), n[i].get_mpz_t(), number.get_mpz_t());
49          if(result != 1 ){
50              cout << "number #1: " << result << endl;
51              cout << "number #2: " << number / result << endl;
52              break;
53          }
54      }
55      return 0;
56  }
```

3 Integer factorization calculator

Число n1 = 5 684417 577210 707125 270927 756395 826164 432807 313246 637831 293635 062503 262393 683373

Нетривиальные сомножители :
2057 135607 937604 146710 725543 839188 722261,

2763 268282 011636 642475 653121 908570 867193,

```
5 684417 577210 707125 270927 756395 826164 432807 313246 637831 293635 062503
262393 683373 (79 digits) = 2057 135607 937604 146710 725543 839188 722261
(40 digits) × 2763 268282 011636 642475 653121 908570 867193 (40 digits)
```

4 Консоль

Число n2 = 330202295900030604612878387051742686153612741688512674640516339592111786182112102952744205334221107472202527866641026738964833952954682296035526349373351638998478563166173570125125320373840042467197427715570566783354934876492962376348888456514955245307546351321990013280137573736294484549777438046907147746082497547574721415593280178417596118368966005440070935516884807614630932600250786098413254455580028740515858031572232760606988105994915916825321411634327591

Нетривиальные сомножители для числа n2:
162257839621427704998966167419999134594347010619931431934253553615815831406985847131688778726472637456749117458469024415248493278440213186501074153016037291,

203504679139351872837215704246919986254481679672284138291298627544405104865941989256402305951681049741031930392849775141978341152279985618745105556177146860359597378012647447994578017110821713231386426603711337676136999084536885356359762416967611287813112363996249168423996714598121772213953355806217264577079,

```
dude@DESKTOP-545VSUH:/mnt/d/education/education/Cripta/lab1$ g++ main.cpp -lgmpxx
-lgmp
dude@DESKTOP-545VSUH:/mnt/d/education/education/Cripta/lab1$ ./a.out <test1.txt
```

number #1: 16225783962142770499896616741999913459434701061993143193425355361581583140698584713168877872647263745674911745846902441524849327844021318650107415301603729
number #2: 20350467913935187283721570424691998625448167967228413829129862754440510486594198925640230595168104974103193039284977514197834115227998561874510555617714686035959737801264744799457801711082171323138642660371133767613699908453688535635976241696761128781311236399624916842399671459812177221395335580621726457707 9

5 Выводы

Выполнив вторую по счету (первую по порядку) лабораторную работу по курсу «Криптография», я познакомился с новой для себя темой - факторизацией больших чисел.

Эта работа сама по себе не является очень трудной для выполнения, потому что для ее выполнения можно было использовать любые инструменты, но время обработки чисел, особенно n2, довольно большое, поэтому пришлось искать оптимальные инструменты для подсчета сомножителей. Так как нетривиальные сомножители n2 в любом случае будут искаться долго (время может измеряться не в часах, а в днях или неделях), было решено воспользоваться подсказкой, что в одном из вариантов какое-то число является наибольшим общим делителем моего числа n2. Я спарсил все варианты, далее каждое число и мое n2 прогнал через gcd, нашел один нетривиальный сомножитель и поделил на него n2. Это произошло очень быстро, несравнимо быстро с "честным"поиском нетривиальных сомножителей.

Данная лабораторная работа показывает надежность алгоритмов шифрования, таких как RSA. Потому что "честный"поиск нетривиальных сомножителей занимает очень много времени и ресурсов, но когда у нас появляется хотя бы небольшая подсказка мы можем сильно сократить время поиска.

Список литературы

[1] *Онлайн калькулятор.*
URL: `https://www.alpertron.com.ar/ECM.HTM` (дата обращения: 16.04.2022).