Федеральное государственное автономное образовательное учреждение высшего образования "Национальный исследовательский университет ИТМО"

Лабораторная работа №2

по дисципение "Информационная безопасность"

Атака на алгоритм шифрования RSA методом повторного шифрования

Вариант 10

Выполнил: студент Саржевский И.А.

Группа: P3402

Преподаватель: к.т.н., доцент
Маркина Т.А.

г. Санкт-Петербург

2021 г.

Лабораторная работа №2

Атака на алгоритм шифрования RSA методом повторного шифрования

Цель работы

Изучить атаку на алгоритм шифрования RSA методом повторного шифрования.

Задание

Алгоритм шифрования RSA состоит из следующих шагов:

1. Выбираются простые числа p и q, вычисляется n = p * q;
2. $\phi(n) = (p-1)(q-1)$;
3. Находится число e, взаимно простое $\phi(n)$;
4. Вычисляется d, такое, что de эквивалентно единице по модулю $\phi(n)$.

(n, e) - публичный ключ. Для шифрования сообщение разбивается на блоки $t(< n)$, зашифрованный текст: $c = t^e mod n$.

Для дешифрования используется приватный ключ (n, d): $t = c^d \ mod \ n$.

В данной лабораторной работе рассматривается атака с помощью повторного шифрования. Метод заключается в решении уравнения $y = x^e \ mod \ n$. Для этого строим последовательность, где $y_1 = y$, каждый следующий $y_i = y_{i-1}^e \ mod \ n$. Если текущий y_i равен изначальному C, то y_{i-1} является решением уравнения $y = x^e \ mod \ n$, и, следовательно, открытым текстом.

Разработанная программа принимает путь к yaml-файлу, в котором описаны исходные данные: N, e и блоки данных C.

Исходные данные

```
N: 301916099393
e: 301319
C:
  - 300229084086
  - 103375119523
  - 47856681522
  - 299308768883
  - 259681434827
  - 155394796250
  - 203569645393
  - 81385593446
  - 153370193599
  - 11291771251
  - 297354725266
  - 71677781247
  - 298448677628
```

Листинг разработанной программы

main.rs

```
mod mod_ops;

extern crate byteorder;
use byteorder::{BigEndian, WriteBytesExt};
use std::mem;
use getopts::Options;
use yaml_rust::YamlLoader;
use std::fs;
use std::env;
use num_traits::cast::ToPrimitive;

fn main() {
  // --- Adding cmd line arguments: ------------------
  //      -f: Path to input data yaml-file
  let args: Vec<String> = env::args().collect();

  let mut opts = Options::new();
  opts.optopt("f", "file", "input file path", "input.yaml");

  let matches = match opts.parse(&args[1..]) {
    Ok(m) => { m }
    Err(f) => { panic!(f.to_string()) }
  };

  if !matches.opt_present("f") {
    println!("You must specify the input file path!");
    return;
  }
  // -------------------------------------------------
  // --- Parsing input yaml-file ---------------------
  let file_contents = fs::read_to_string(matches.opt_str("f")
                                         .unwrap()).unwrap();
  let docs = YamlLoader::load_from_str(&file_contents).unwrap();
  let doc = &docs[0];
  let n = doc["N"].as_i64().unwrap();
  let e = doc["e"].as_i64().unwrap();
  // -------------------------------------------------
  // --- Decoding every present chunk of data --------
  let mut in_c = Vec::new(); // initialte input data vector
  for c in doc["C"].as_vec().unwrap() {
    in_c.push(c.as_i64().unwrap());
  }

  println!("Parsed {}\nN: {}\ne: {}\n{} chunks of data",
  matches.opt_str("f").unwrap(), n, e, in_c.len());
  println!("Trying to compute the exponent rank...");
```

```
47
48    let mut res = in_c.clone(); // result vector
49    let mut i = 1;
50    // in this loop we calculate [last_res[i]^e mod n] for every element
51    // and update last_res on every iteration until the first element
52    // of vector will match the initial chunk of data. That would mean
53    // that we have the decoded text in res
54    loop {
55      let mut last_res = res.clone();
56      for j in 0..in_c.len() {
57        last_res[j] = mod_ops::mod_exp(&res[j], &e, &n).to_i64().unwrap();
58      }
59      if last_res[0] == in_c[0] {
60        break;
61      }
62      res = last_res;
63      i += 1;
64    }
65    // Print the answer
66    println!("Found the exponent rank: {}", i);
67    let mut bs = [0u8; mem::size_of::<i64>()];
68    let mut res_s = String::new();
69    for r in res {
70      bs.as_mut()
71        .write_i64::<BigEndian>(r)
72        .expect("Unable to write");
73      let encoder = encoding_rs::WINDOWS_1251;
74      let (s, _, _) = encoder.decode(&bs);
75      res_s.push_str(&s);
76    }
77    println!("Message: {}", res_s);
78  }
```

Результаты работы программы

```
keker (~/code/itmo-4th-year/infosec/part2/lab2/target/debug) master
↳ ./lab2 -f input.yaml
Parsed input.yaml
N: 301916099393
e: 301319
13 chunks of data
Trying to compute the exponent rank...
Found the exponent rank: 99450
Message: количество ошибок CRC хорошим кабельным тестером или
```

Рис. 1: Результат работы программы

Вывод

В результате выполнения данной лабораторной работы была изучена атака на алгоритм шифрования RSA методом методом повторного шифрования. Была реализована программа, позволяющая расшифровать сообщение, зашифрованное с помощью RSA.