Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет безопасности информационных технологий

Дисциплина:
«Разработка систем аутентификации и криптографии»

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1
«Алгоритмы криптографии и подпись приложений»

Выполнил:
Магистрант гр. №42514с
И. М. Гарипов
_____

Проверил:
Ассистент ФБИТ
Р. И. Фёдоров
_____

Санкт-Петербург
2020г.

СОДЕРЖАНИЕ

1 Цель работы

Часть 1: реализация алгоритма шифрования **Elgamal**.

Требования к реализации:

– необходимо реализовать сам алгоритм (процедуры генерации ключей, шифрования и дешифрования) без использования криптографических библиотек

– программа должна запускаться в среде Windows, исполняемый файл программы должен иметь расширение .EXE

Часть 2: подпись полученного в первой части файла .EXE

Требования к выполнению:

– необходимо подписать полученный файл .EXE с помощью команд Windows PowerShell (лучше использовать PKI Client)

В итоге, при открытии «Свойства» файла .EXE в разделе «Цифровые подписи» мы должны будем увидеть свою подпись.

2 Описание выбранных средств реализации и обоснования выбора

В качестве средства реализации алгоритма шифрования был выбран язык программирования Python, поскольку данный язык является кроссплатформенным. Это означает, что реализация кода будет доступна для работы независимо от операционной системы.

К тому же Python имеет довольно лёгкий синтаксис, прост в изучении и не требует больших ресурсов для его использования.

Также выбор данного инструмента разработки обусловлен личной симпатией автора.

3 Описание алгоритма

Схема Эль-Гамаля – криптосистема с открытым ключом, основанная на трудности вычисления дискретных логарифмов в конечном поле. Криптосистема включает в себя алгоритм шифрования и алгоритм цифровой подписи.

Первый этап алгоритма Эль-Гамаля заключается в генерации ключей. Этот этап включает следующую последовательность действий:

– генерируется случайное простое число p;

– выбирается целое число g — первообразный корень p;

– выбирается случайное целое число, взаимно простое с $(p-1), x$ такое, что $1 < x < p-1$;

– вычисляется $y \equiv g^x mod\ p$;

– открытым ключом является y, закрытым ключом — число x.

Второй этап алгоритма включает шифрование. Сообщение M должно быть меньше числа p. Сообщение шифруется следующим образом:

– выбирается сессионный ключ — случайное целое число, взаимно простое с $(p\ -\ 1)$, k такое, что $1 < k < p-1$;

– вычисляются числа $a \equiv g\hat{}k\ mod\ p$ и $b \equiv y^k M\ mod\ p$;

– Пара чисел (a, b) является шифротекстом.

Заключительный этап схемы Эль-Гамаля – это расшифрование. Зная закрытый ключ x, исходное сообщение M можно вычислить из шифртеста (a, b) по формуле $M = b(a^x)^{-1} mod\ p$

Исходный код реализации алгоритма шифрования по схеме Эль-Гамаля представлен в репозитории на *GitHub* по следующей ссылке:

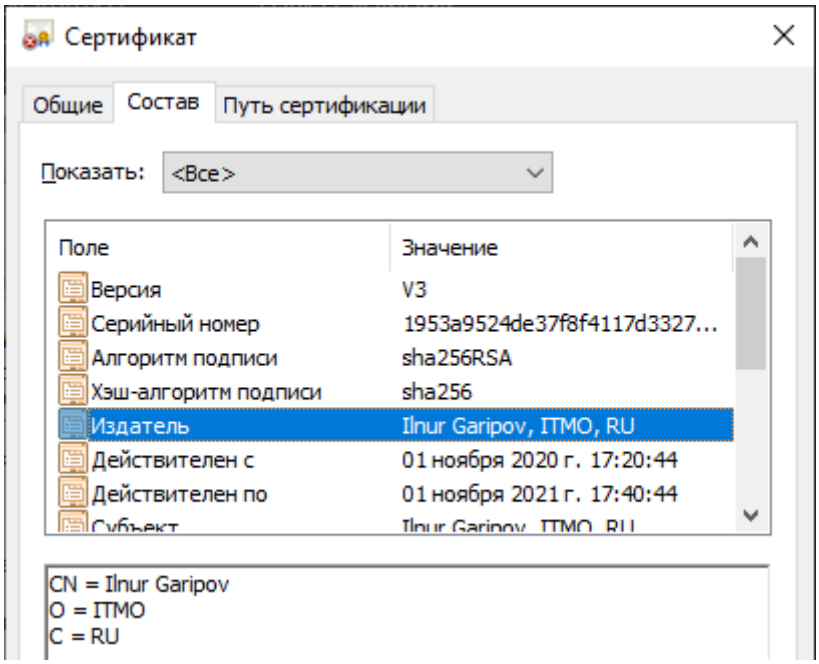
– https://github.com/BetterThanLeonid/El-Gamal

4 Подпись исполняемого файла

Подписать исполняемый файл самоподписанным сертификатом через PKIClient можно следующим образом:

1. Создать самоподписанный сертификат в Windows PowerShell:

```
New-SelfSignedCertificate -Type Custom -Subject "CN=Ilnur Garipov, O=ITMO, C=RU" -KeyUsage DigitalSignature -FriendlyName "Ilnur Garipov" -CertStoreLocation "Cert:\CurrentUser\cert" -TextExtension @("2.5.29.37={text}1.3.6.1.5.5.7.3.3", "2.5.29.19={text}")
```



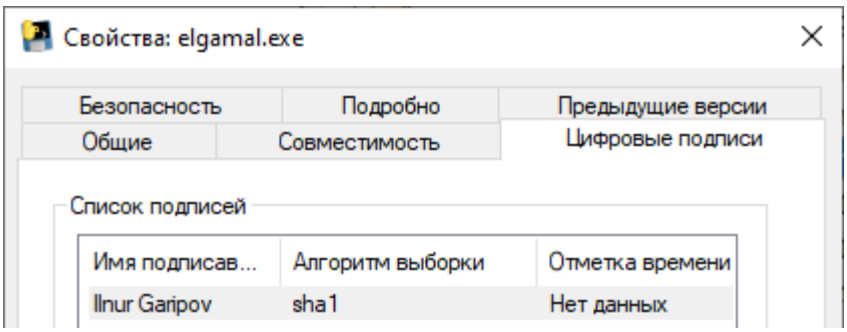
2. Задать переменной cert только что созданный сертификат:

```
$cert=Get-ChildItem -Path cert:\CurrentUser\my -CodeSigningCert
```

3. Подписать exe файл этим сертификатом командой:

```
Set-AuthenticodeSignature elgamal.exe $cert
```

4. Файл подписан.



5 Выводы

В ходе выполнения данной лабораторной работы был изучен алгоритм шифрования по схеме Эль-Гамаля, а также были изучены основные принципы при работе с подписью исполняемых файлов через Windows PowerShell. Была написана реализация изученного алгоритма на языке Python и преобразована в исполняемый файл, который был подписан самозаверяющим сертификатом посредством PKI Client.