МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования «Балтийский федеральный университет имени Иммануила Канта»

ОНК «Институт высоких технологий»

ОТЧЁТ

о прохождении учебной практики по получению первичных

профессиональных умений и навыков, в том числе первичных умений и навыков научно-исследовательской деятельности

на базе Высшей школы компьютерных наук и прикладной математики образовательно-научного кластера "Институт высоких технологий"

Выполнил Затирахин Кирилл Сергеевич \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

студент очной формы обучения 3 курса

специальности 10.05.01 Компьютерная безопасность

специализация «Математические методы защиты информации»

Руководитель практики от университета

доцент ОНК «ИВТ» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Киршанова Е.А.

г. Калининград 2023 г.

**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ 3

ГЛАВА 1. ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ НА ПРАКТИКУ 4

ГЛАВА 2. ВЫПОЛНЕНИЕ ЗАДАНИЙ НА ПРАКТИКУ 5

2.1 ТЕОРЕТИЧЕСКОЕ ОБОСНОВАНИЕ 5

2.2 ПРАКТИЧЕСКАЯ СТОРОНА РЕШЕНИЯ 5

2.3 ВЕРСТКА ПРЕЗЕНТАЦИИ В СИСТЕМЕ LATEX 6

ЗАКЛЮЧЕНИЕ 8

СПИСОК ЛИТЕРАТУРЫ 9

**ВВЕДЕНИЕ.**

Вид практики – учебная практика по получению первичных профессиональных умений и навыков, в том числе первичных умений и навыков научно-исследовательской деятельности, далее Учебная практика.

Цель учебной практики: Сформировать и укрепить навыки владения системой верстки документов LaTeX, системой управления версиями git. Также укрепить навыки владения языком программирования python при решении поставленной задачи, укрепить навыки публичного выступления и защиты работы.

Помимо целей, представленных выше, важной задачей является сформировать умение применять полученные теоретические знания для решения практических задач.

Для того, чтобы достичь поставленных целей, были поставлены задачи:

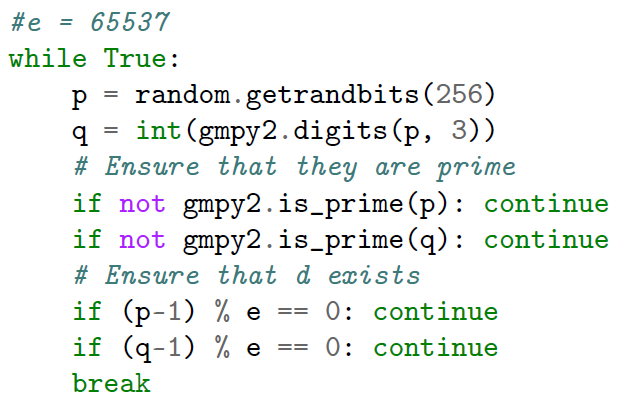
* Решить любую из задач в разделе CTF Archive на сайте cryptohack.org, используя любой удобный язык программирования, а также, основываясь на теоретических знаниях, полученных за период обучения.
* Сформировать презентацию в системе верстки документов LaTeX с подробным описанием сути задачи и методах ее решения.
* Создать открытый git-репозиторий с исходными кодами презентации и самого решения.

**ГЛАВА 1. ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ НА ПРАКТИКУ.**

**Задание 1.** Решить задачу с сайта [cryptohack.org](https://cryptohack.org/) и проверить решение:

Дана программа, шифрующая исходное сообщение с помощью алгоритма RSA. Также дан файл с выводом представленной программы. Конкретно: модуль по которому ведутся вычисления *n*, экспонента шифрования *e*, зашифрованное сообщение *c*. Необходимо расшифровать исходное сообщение.

Ниже представлен фрагмент исходного кода задачи, где генерируются простые *p* и *q*.



**Задание 2.** С помощью системы LaTeX составить презентацию.

**Задание 3.** Загрузить в свой git-репозиторий исходные коды презентации и решения.

**ГЛАВА 2. ВЫПОЛНЕНИЕ ЗАДАНИЙ НА ПРАКТИКУ.**

**2.1 ТЕОРЕТИЧЕСКОЕ ОБОСНОВАНИЕ.**

Число *n* зависит от простых чисел *p* и *q*. В свою очередь, число *q* напрямую зависит от *p*. Число *p* генерируется длиной 256 бит, то оно находится в промежутке от до . Таким образом, задача сводится к поиску такого простого числа *p*, что полученный модуль совпадет с исходным. Если это произойдет, можно легко вычислить функцию Эйлера *φ* и найти обратный элемент по отношению к экспоненте *e* и расшифровать сообщение.

Для поиска числа p используем алгоритм бинарного поиска. Двоичный поиск выполняется за логарифмическое время в худшем случае, производя сравнения, где *�n* - количество элементов в массиве.

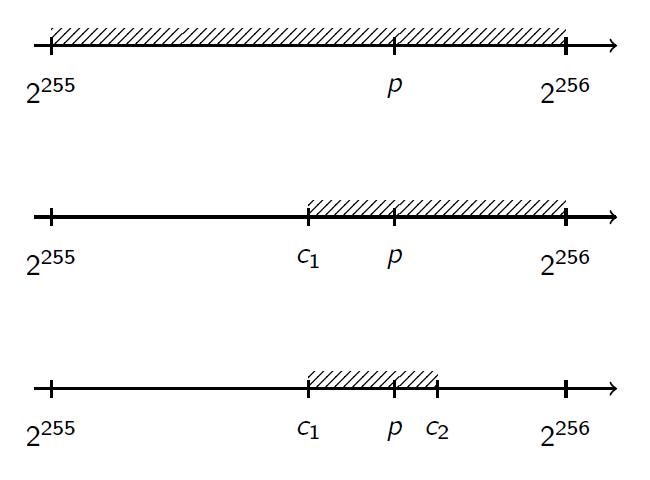


Рисунок 1. Визуализация бинарного поиска

**2.2 ПРАКТИЧЕСКАЯ СТОРОНА РЕШЕНИЯ.**

Для решения данной задачи был выбран язык Python, так как он наиболее простой в использовании и программа шифрующая исходное сообщение была написана на этом языке. Для решения задачи использовались библиотеки ***gmpy2*** для генерации простых чисел, а также ***Crypto.Util.number*** для преобразования числа в байты. Программа состоит из двух частей: первая – нахождение числа p с помощью алгоритма бинарного поиска, вторая – реализация криптосистемы RSA. (См. приложение 1).

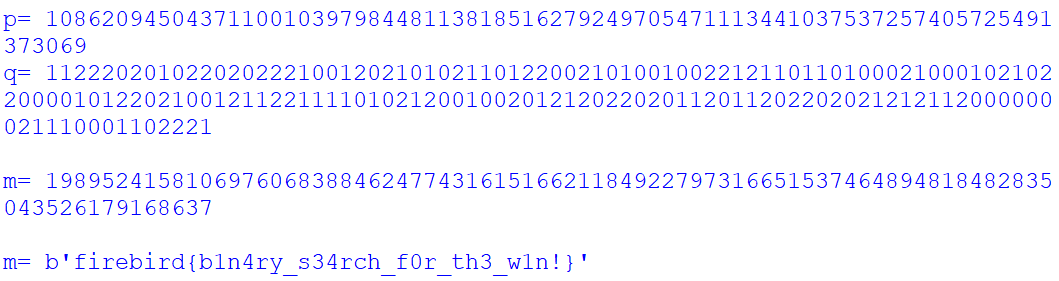


Рисунок 2. Вывод программы после выполнения

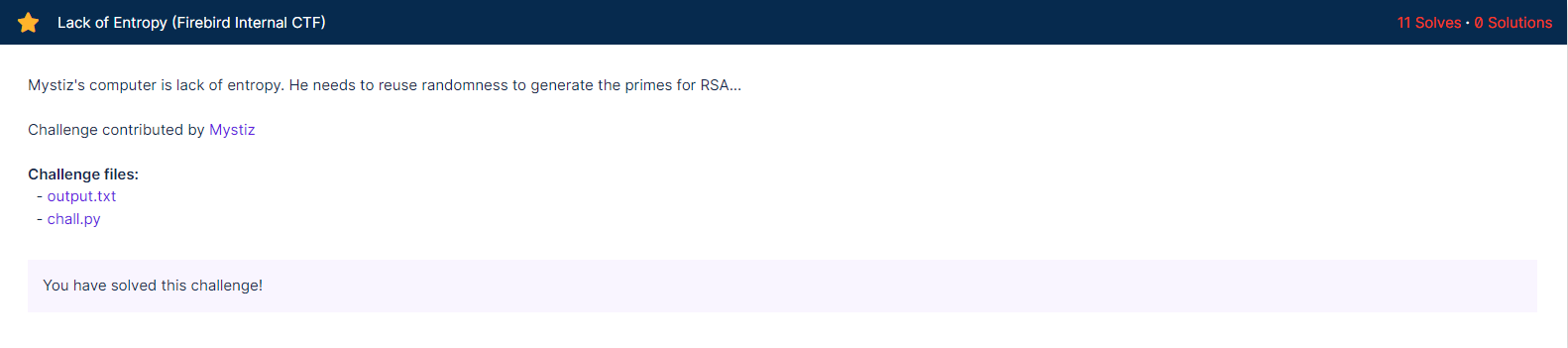


Рисунок 3. Решение принято системой cryptohack.org

**2.3 ВЕРСТКА ПРЕЗЕНТАЦИИ В СИСТЕМЕ LATEX.**

Для верстки презентации была выбрана платформа Overleaf по той причине, что она бесплатная и позволяет быстро работать в режиме online и не устанавливать программное обеспечение. Также на официальном сайте предоставлены подробные видео-уроки, позволяющие быстро изучить основы работы с LaTeX. (Исходный код презентации см. в приложении 2).

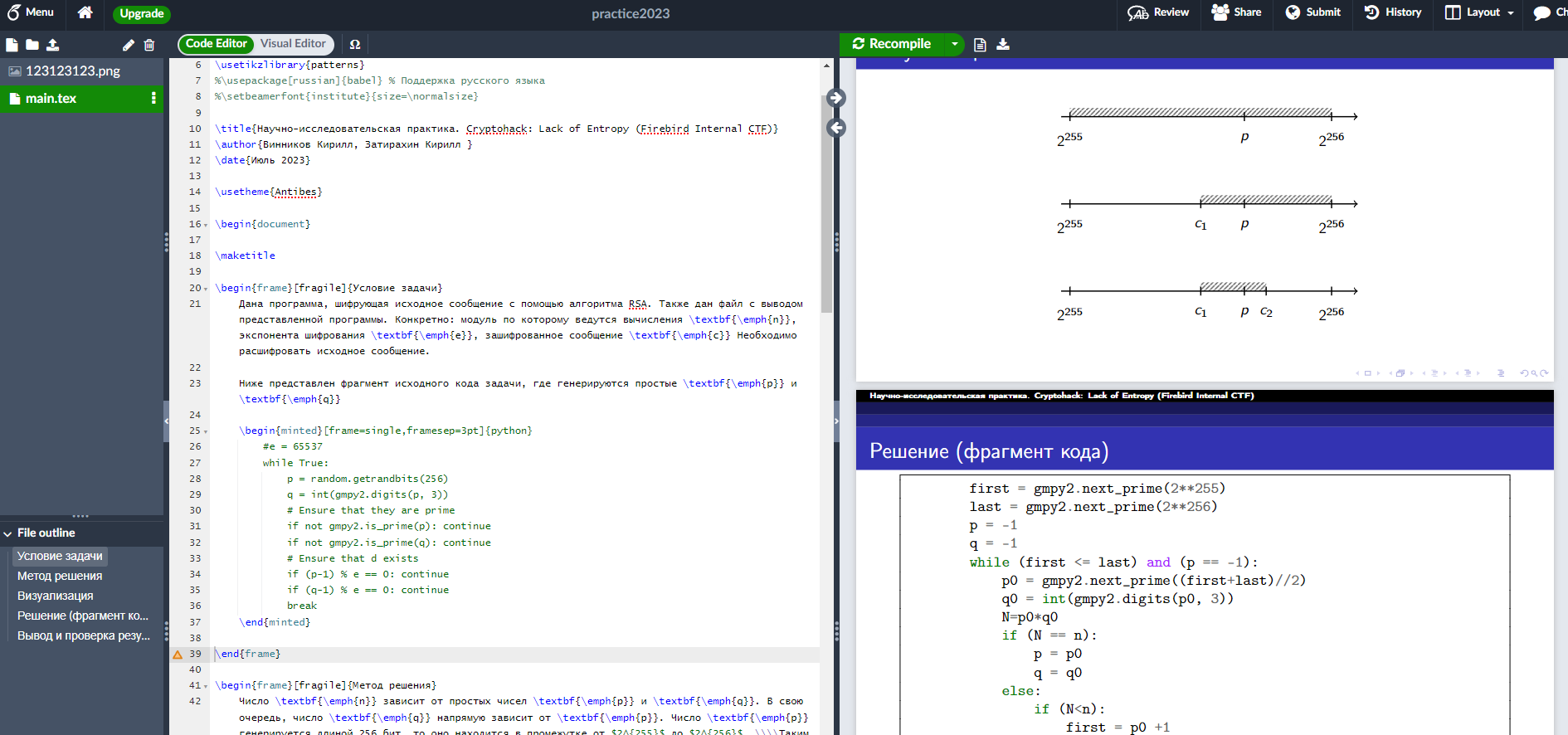


Рисунок 4. Интерфейс Overleaf

Теперь, когда презентация готова, осталось создать удаленный репозиторий с исходными кодами презентации и решения. В данный момент его можно открыть по ссылке: [https://github.com/ZatirahinKirill/practika2023](https://vk.com/away.php?to=https%3A%2F%2Fgithub.com%2FZatirahinKirill%2Fpractika2023&cc_key=) .

**ЗАКЛЮЧЕНИЕ.**

В течение учебной практики все поставленные задачи были выполнены, а цели достигнуты. Я укрепил свои навыки программирования на Python. Также улучшил навыки работы с LaTeX, Git и GitHub. Помимо этого, я отработал навыки исследовательской и научной деятельности, а также удалось применить теоретические знания на практике.

**СПИСОК ЛИТЕРАТУРЫ.**

**Перечень ресурсов информационно-телекоммуникационной сети «Интернет»**

1. Официальный сайт Overleaf с видео-уроками - <https://www.overleaf.com/learn/latex/Beamer_Presentations%3A_A_Tutorial_for_Beginners_(Part_1)%E2%80%94Getting_Started>
2. Условие решаемой задачи - https://cryptohack.org/challenges/ctf-archive/
3. Официальный сайт git - [https://git-scm.com/](https://git-scm.com/%20)
4. Официальная документация git - <https://git-scm.com/docs>

**Приложения**

Приложение 1

import gmpy2

from Crypto.Util.number import long\_to\_bytes, bytes\_to\_long

n = 12189464288007059657184858632825479990912616419482466046617619319388181010121359489739982536798361842485210016303524715395474637570227926791570158634811951043352789232959763417380155972853016696908995809240738171081946517881643416715486249

e = 65537

c = 10093874086170276546167955043813453195412484673031739173390677430603113063707524122014352886564777373620029541666833142412009063988439640569778321681605225404251519582850624600712844557011512775502356036366115295154408488005375252950048742

first = gmpy2.next\_prime(2\*\*255)

last = gmpy2.next\_prime(2\*\*256)

p = -1

q = -1

while (first <= last) and (p == -1):

p0 = gmpy2.next\_prime((first+last)//2)

q0 = int(gmpy2.digits(p0, 3))

N=p0\*q0

if (N == n):

p = p0

q = q0

else:

if (N<n):

first = p0 +1

else:

last = p0 -1

phi=(p-1)\*(q-1)

d=pow(e,-1,phi)

m=int(pow(c,d,n))

t=long\_to\_bytes(m)

print('p=',p,'\nq=',q,'\n\nm=',m,'\n\nm=',t)

Приложение 2

\documentclass[10pt, aspectratio=169]{beamer}

\usepackage{minted}

\usepackage[T2A]{fontenc}

\usepackage{graphicx} % Required for inserting images

\usepackage{tikz}

\usetikzlibrary{patterns}

%\usepackage[russian]{babel} % Поддержка русского языка

%\setbeamerfont{institute}{size=\normalsize}

\title{Научно-исследовательская практика. Cryptohack: Lack of Entropy (Firebird Internal CTF)}

\author{Винников Кирилл, Затирахин Кирилл }

\date{Июль 2023}

\usetheme{Antibes}

\begin{document}

\maketitle

\begin{frame}[fragile]{Условие задачи}

Дана программа, шифрующая исходное сообщение с помощью алгоритма RSA. Также дан файл с выводом представленной программы. Конкретно: модуль по которому ведутся вычисления \textbf{\emph{n}}, экспонента шифрования \textbf{\emph{e}}, зашифрованное сообщение \textbf{\emph{c}} Необходимо расшифровать исходное сообщение.

Ниже представлен фрагмент исходного кода задачи, где генерируются простые \textbf{\emph{p}} и \textbf{\emph{q}}

\begin{minted}[frame=single,framesep=3pt]{python}

#e = 65537

while True:

p = random.getrandbits(256)

q = int(gmpy2.digits(p, 3))

# Ensure that they are prime

if not gmpy2.is\_prime(p): continue

if not gmpy2.is\_prime(q): continue

# Ensure that d exists

if (p-1) % e == 0: continue

if (q-1) % e == 0: continue

break

\end{minted}

\end{frame}

\begin{frame}[fragile]{Метод решения}

Число \textbf{\emph{n}} зависит от простых чисел \textbf{\emph{p}} и \textbf{\emph{q}}. В свою очередь, число \textbf{\emph{q}} напрямую зависит от \textbf{\emph{p}}. Число \textbf{\emph{p}} генерируется длиной 256 бит, то оно находится в промежутке от $2^{255}$ до $2^{256}$. \\\\Таким образом, задача сводится к поиску такого простого числа \textbf{\emph{p}}, что полученный модуль совпадет с исходным. Если это произойдет, можно легко вычислить функцию Эйлера $\varphi$ и найти обратный эелемент по отношению к экспоненте \textbf{\emph{e}} и расшифровать сообщение.

\end{frame}

\begin{frame}{Визуализация}

\centering

\begin{tikzpicture}[mydrawstyle/.style={draw=black, thick}, x=1mm, y=1mm, z=1mm]

\draw[mydrawstyle, ->](-2,50)--(66,50);

\draw[mydrawstyle](0,49)--(0,51) node[below=10]{$2^{255}$};

\draw[mydrawstyle](40,49)--(40,51) node[below=10]{$p$};

\draw[mydrawstyle](60,49)--(60,51) node[below=10]{$2^{256}$};

\fill[pattern=north east lines] (0,52) rectangle (60,50);

\draw[mydrawstyle, ->](-2,30)--(66,30);

\draw[mydrawstyle](0,29)--(0,31) node[below=10]{$2^{255}$};

\draw[mydrawstyle](30,29)--(30,31) node[below=10]{$c\_1$};

\draw[mydrawstyle](40,29)--(40,31) node[below=10]{$p$};

\draw[mydrawstyle](60,29)--(60,31) node[below=10]{$2^{256}$};

\fill[pattern=north east lines] (30,32) rectangle (60,30);

\draw[mydrawstyle, ->](-2,10)--(66,10);

\draw[mydrawstyle](0,9)--(0,11) node[below=10]{$2^{255}$};

\draw[mydrawstyle](30,9)--(30,11) node[below=10]{$c\_1$};

\draw[mydrawstyle](45,9)--(45,11) node[below=10]{$c\_2$};

\draw[mydrawstyle](40,9)--(40,11) node[below=10]{$p$};

\draw[mydrawstyle](60,9)--(60,11) node[below=10]{$2^{256}$};

\fill[pattern=north east lines] (30,12) rectangle (45,10);

\end{tikzpicture}

\end{frame}

\begin{frame}[fragile]{Решение (фрагмент кода)}

\begin{minted}[frame=single,framesep=3pt]{python}

first = gmpy2.next\_prime(2\*\*255)

last = gmpy2.next\_prime(2\*\*256)

p = -1

q = -1

while (first <= last) and (p == -1):

p0 = gmpy2.next\_prime((first+last)//2)

q0 = int(gmpy2.digits(p0, 3))

N=p0\*q0

if (N == n):

p = p0

q = q0

else:

if (N<n):

first = p0 +1

else:

last = p0 -1

\end{minted}

\end{frame}

\begin{frame}{Вывод и проверка результата}

\begin{figure}[h]

\centering

\includegraphics[width=1\linewidth]{123123123.png}

\label{fig:mpr}

\end{figure}

\end{frame}

\end{document}