Министерство образования и науки Российской Федерации

Санкт-Петербургский Политехнический Университет Петра Великого

—

Высшая школа кибербезопасности

К У Р С О В А Я Р А Б О Т А

«Подходы к созданию CTF»

1. по дисциплине «Технологии реверс-инжиниринга программного обеспечения»
2. Выполнил
3. студент гр. 5151004/10101 Веремейчук Я.Ю.

<подпись>

1. Преподаватель
2. асс. преподавателя Овасапян Т.Д.

<подпись>

1. Санкт-Петербург
2. 2023

СОДЕРЖАНИЕ

|  |  |
| --- | --- |
| **ВВЕДЕНИЕ...……………………………………………………………………** | **3** |
| **1. ОПРЕДЕЛЕНИЕ И ВИДЫ CTF.…………………………………………...** | **5** |
| 1.1. Основные виды CTF……………...……………………………………... | 5 |
| 1.2. Направления Task-Based CTF………………………………………….. | 8 |
| **2. АНАЛИЗ И РЕШЕНИЕ ЗАДАЧ CTF.…………..…………………………** | **12** |
| 2.1. Решение Crypto-задач…...………………………………………….…… | 12 |
| 2.2. Решение Web-задач……………………………………………….…….. | 15 |
| **3. РАЗРАБОТКА СОБСТВЕННЫХ ЗАДАЧ CTF...……...…………………** | **18** |
| 3.1 Структура формата PNG……………………….....…………….…….…. | 18 |
| 3.2 Сокрытие информации заменой красного оттенка пикселей…….…… | 20 |
| 3.3 Сокрытие информации в метаданных…………………………….……. | 26 |
| 3.4 Сокрытие файлов……………………………………………….………... | 30 |
| 3.5 PNG-наполнение за границами картинки…………………….………… | 31 |
| **ЗАКЛЮЧЕНИЕ………………………………………………………………...** | **33** |
| **СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ….………………………..** | **34** |
| **Приложение А…………………………………………………………………..** | **37** |
| **Приложение Б…………………………………………………………………..** | **38** |
| **Приложение В…………………………………………………………………..** | **40** |
| **Приложение Г…………………………………………………………………..** | **43** |
| **Приложение Д…………………………………………………………………..** | **44** |
|  |  |
|  |  |

ВВЕДЕНИЕ

В современном мире информационные технологии играют ключевую роль практически во всех сферах деятельности человека. С каждым годом количество информации, передаваемой через интернет, увеличивается в геометрической прогрессии. Это, в свою очередь, ведет к росту числа кибератак и угроз информационной безопасности.

Актуальность темы данной курсовой работы обусловлена необходимостью обеспечения защиты информации от несанкционированного доступа и анализа возможных уязвимостей информационных систем. Одним из эффективных методов обучения и тренировки специалистов в области информационной безопасности является проведение соревнований по решению задач на безопасность информации (CTF – Capture the Flag).

**Цель работы**

Целью данной работы является изучение подходов к созданию и проведению соревнований CTF, а также анализ существующих проблем и возможностей их решения.

**Задачи**

Для достижения данной цели были поставлены следующие задачи:

1. Изучить понятие CTF и различные виды задач, используемых в этих соревнованиях.
2. Решить несколько задач различных видов для лучшего понимания их особенностей.
3. Разработать собственные задачи для CTF в соответствии с выбранным видом.

Несмотря на популярность и широкое распространение CTF-соревнований, существует ряд проблем, которые требуют дополнительного внимания. Это включает в себя отсутствие единого стандарта для оценки сложности задач, недостаточную автоматизацию процесса создания CTF и неэффективные методы обучения и подготовки участников. Решение данных проблем является ключевым направлением дальнейших исследований в области CTF и информационной безопасности в целом.

1. ОПРЕДЕЛЕНИЕ И ВИДЫ CTF

* 1. Основные виды CTF

CTF (Capture The Flag, «Захват флага») — это формат соревнований, где команды или индивидуальные участники пытаются решать различные задачи в области информационной безопасности. Целью участников является поиск и эксплуатация уязвимостей в системах, поиск секретной информации и захват «флагов» — меток, которые свидетельствуют об успешном решении задачи [1].

Существует два основных вида CTF: Attack-Defence и Task-Based CTF. Ниже можно увидеть сравнительную таблицу этих видов CTF [4].

Таблица – Сравнительная таблица основных видов CTF: Attack-Defence и Task-Base

|  |  |  |
| --- | --- | --- |
| Вид | Преимущества | Недостатки |
| Attack-Defence | Наиболее приближен к  жизненным реалиям. | Невозможность  продолжения участия в  соревновании, если появились  «нерешаемые» проблемы для  команды;  Сложность разработки. |
| Task-Based | Удобство для новичков;  Возможность выбора  командами решаемых  заданий. | Невозможность  отслеживания обмена  информацией между  соперничающими командами. |

Attack-Defence признается классическим видом CTF, поскольку придерживается основных правил игры "Захват флага". Организаторы предоставляют командам определенное программное окружение, чаще всего в виде виртуальной машины. В предоставленной среде разворачивается операционная система, а в ней - несколько сервисов, реализованных на различных языках программирования. Эти сервисы специально создаются с уязвимостями.

Участники обычно обладают полными правами в предоставленном окружении. Основной целью CTF является поиск уязвимостей в своей системе и попытка атаки на системы других команд.

Несмотря на то, что все сервисы должны быть написаны на разных языках программирования, они обладают общей характеристикой – доступностью с внешних источников. Например, это может быть веб-сервер или же скомпилированный бинарный файл, который прослушивает определенный порт и выполняет специфические задачи.

Иногда участникам даже приходится создавать свои собственные сервисы с нуля. Это отклонение от классической модели CTF, однако такие ситуации также встречаются. В данном случае на определенный порт поступают определенные сетевые пакеты, и участники должны разработать обработчик для этих данных.

Task-Based CTF – это формат соревнований, который пользуется особой популярностью благодаря своей относительной доступности и простоте в организации. Task-Based CTF основан на решении задач, ранжированных по сложности, в различных направлениях: администрирование серверов, криптография и стеганография, нахождение веб-уязвимостей, реверсный инжиниринг и других. Участники могут решать их в любом порядке и получать очки в зависимости от сложности. Рейтинг участников ведется по набранному количеству очков [4].

Существуют CTF, которые объединяют черты attack-defence и task-based: командам требуется не только защищать свои сервисы и атаковать чужие, но и решать задания.

King of the hill – еще один молодой вид CTF соревнований. Целью участников в соревновании вида «Царь горы» или «KotH» является поиск уязвимостей архитектуры системы и несанкционированное проникновение в неё. На следующем этапе в системе необходимо закрепиться и не позволить другим участникам перехватить управление. Основное отличие от других видов CTF соревнований заключается в том, что через установленные администрацией промежутки времени веб-серверы возвращаются в исходное состояние и соревнование начинается заново. Такой подход к организации соревнований полностью соответствует духу известной модели жизненного цикла киберугроз под названием Cyber-Kill Chain. На рисунке ниже можно увидеть общую топологию сети в соревновании «KotH» [7].

Изображение выглядит как диаграмма, зарисовка, линия, План

Автоматически созданное описание

Рисунок – Общая топология сети в соревновании "KotH"

* 1. Направления Task-Based CTF

Reverse. Реверс-инжиниринг – это процесс анализа и понимания работы программного обеспечения или аппаратного устройства путем извлечения исходного кода, создания схем и диаграмм, а также исследования внутренних компонентов. В контексте CTF (capture the flag), реверс-инжиниринг часто используется для анализа и взлома защищенных приложений или систем с целью получения «флагов». Одним из популярных видов CTF на основе реверс-инжиниринга является анализ вредоносных программ. Злоумышленники часто используют различные виды вредоносного ПО для обхода систем защиты, поэтому для хакеров важно уметь анализировать и понимать работу таких программ, чтобы найти уязвимости и предотвратить их использование.

Участникам необходимо разгадать логику программы, извлечь информацию или даже создать кейгены.

Exploit. Это категория задач в CTF, связанная с поиском и использованием уязвимостей в программном обеспечении для получения флагов. Уязвимости могут быть различными, включая переполнение буфера, использование форматной строки, SQL-инъекции и другие. Здесь часто используются двоичные файлы для операционных систем Linux, которые содержат уязвимости. Для решения Exploit-задач необходимо иметь глубокие знания в области компьютерной безопасности, программирования и анализа двоичного кода. Кроме того, необходимо уметь использовать различные инструменты для анализа уязвимостей, такие как отладчики, дизассемблеры и другие [6].

Web. Включает в себя задания, связанные с веб-безопасностью. Уязвимости, которые могут быть в этих заданиях, включают в себя SQL Injection, XXE, RCE. Для усложнения задач могут использоваться WAF (Web Application Firewall) - механизмы защиты, которые блокируют атаки на веб-приложения.

Для выполнения заданий в этой категории необходимо обладать знаниями в области веб-безопасности, знать, как работают веб-серверы, HTTP-протокол, как происходит обработка запросов, как работать с инструментами для анализа веб-приложений, такими как Burp Suite или OWASP ZAP.

Crypto. Криптография – это область науки, посвященная методам сокрытия информации таким образом, чтобы она оставалась доступной только тем, кому предназначена. Основной целью криптографии является обеспечение конфиденциальности, целостности и подлинности передаваемых данных [8].

В данной категории участники анализируют уязвимости криптографических алгоритмов. Алгоритмы часто представлены в виде самописных, но могут содержать уязвимости, основанные на ранее найденных уязвимостях других криптографических методов.

Stegano. Стеганография – это искусство и наука о способах передачи (хранения) скрытой информации, при которых скрытый канал организуется на базе и внутри открытого канала с использованием особенностей восприятия информации. На современном этапе стеганография претерпевает интенсивное развитие и активно применяется в информационных системах. Она занимает существенное место в области информационной безопасности, где главное ее преимущество заключается в том, что она может не только замещать, но и дополнять, так давно применяющуюся, криптографию. Например, сокрытие факта передачи зашифрованного сообщения значительно снижает вероятность несанкционированного получения информации, так как прежде чем приступить к анализу применяемого шифра, необходимо найти само зашифрованное сообщение.

Стеганографию возможно применять к изображениям, аудио-видео файлам и многим другим форматам передачи информации. Участникам CTF обычно предлагается найти информационное сообщение (флаг), к которому применяются как стеганографические методы защиты информации, так и криптографические [2, 3].

Forensic. Выделяют следующие виды форензики: компьютерная, мобильная, сетевая и исследование баз данных. Задачи в этой категории имитируют расследование инцидентов и анализ образов и файлов. Они могут включать восстановление удаленных файлов, анализ самописных вирусов и анализ дампов сетевого трафика. В процессе расследования инцидента используются средства блокирования записи (программные и аппаратные блокираторы), инструменты клонирования информационного содержимого накопителей информации (AccessData FTK Imager, Guymager), средства дампинга оперативной памяти (Belkasoft RAM Capturer, LiME), файловые утилиты (File, Head), HEX-редакторы, сетевые утилиты (Wireshark, Scapy), средства работы с БД (sqlite3), многофункциональные дистрибутивы (Kali Linux, Parrot Security OS, CAINE, SIFT) и другие (в зависимости от частных задач) [5].

PPC. Эта категория объединяет участников в решении задач, которые требуют не только навыков программирования, но и практического применения этих навыков для достижения конкретных целей. В рамках PPC-задач, участники сталкиваются с различными аспектами программирования, такими как алгоритмы и структуры данных, брутфорс и оптимизация, работа с сетью, разработка ботов, парсинг данных, обратная разработка.

OSINT. Разведывательная дисциплина, включающая в себя поиск, выбор и сбор разведывательной информации из общедоступных источников, а также ее анализ. В разведывательном сообществе термин «открытый источник разведывательных данных» указывает на общедоступность источника. Участники оттачивают навыки выявления уязвимостей, предотвращения кибератак, расследования инцидентов и предупреждения социальной инженерии [9]. Активное использование OSINT способствует повышению кибергигиены и обеспечивает эффективную защиту от киберугроз в цифровой среде.

PWN. Это категория задач, где участники получают флаг, эксплуатируя уязвимость в исполняемом файле с целью вызвать нарушение памяти и получить доступ к удаленному компьютеру. Обычно задачи PWN представляют собой исполняемые программы, написанные на C/C++, которые выполняются на сервере. Владелец взаимодействует с сервером через сеть. Автор оставляет уязвимости в программе, и злоумышленник может эксплуатировать программу, отправляя вредоносные данные на удаленный сервер, что приводит к выполнению кода, необходимого злоумышленнику, и затем получению флага на удаленном сервере.

Основной принцип PWN – это понимание работы программ через обратную разработку. Задачи включают в себя поиск и использование уязвимостей в бинарных файлах. Понимание ассемблера – ключевой шаг для успешного вхождения в PWN. Это помогает узнать, как уязвимость приводит к выполнению злоумышленником кода. Задачи PWN также требуют знания операционных систем и архитектур компьютеров [10].

Misc. В эту категорию попадают задачи, которые не подпадают под другие группы. Здесь могут быть как конкурентные задания, такие как разведка, и развлекательные задачи.

2. АНАЛИЗ И РЕШЕНИЕ ЗАДАЧ CTF

После теоретического ознакомления с CTF были выбраны некоторые виды задач и решены их примеры. Существует множество онлайн-платформ, на которых можно оттачивать навыки и тренироваться в решении задач. Один из таких ресурсов – Хакердом [11].

2.1. Решение Crypto-задач

Древнеримская замена. Условие задачи: Мы получили это от нашего древнеримского знакомого: vbqwyivxgshofjeiycfbu.

Название задачи подсказывает об использовании в нем шифра Цезаря.

Описание алгоритма. Если сопоставить каждому символу алфавита его порядковый номер (нумеруя с 0), то шифрование и дешифрование можно выразить формулами:

*,*

где x – символ открытого текста; y – символ шифрованного текста; n – мощность алфавита (кол-во символов); k – ключ (Рисунок 2).

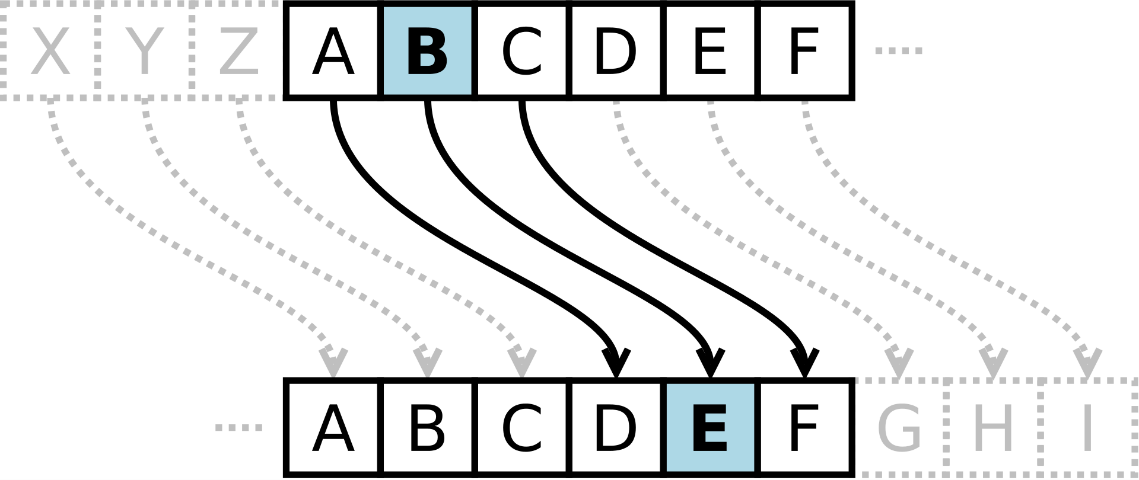


Рисунок – Шифр Цезаря со сдвигом на 3

При шифровании отдельные части исходного текста с помощью ключа заменяются на какие-либо другие буквы, числа, символы и т. п. Если к шифрованному символу вновь применить операцию с тем же ключом, то результатом будет исходный символ [12].

Для решения задачи был реализован алгоритм дешифрования шифра Цезаря с перебором возможных смещений. Код реализации алгоритма можно увидеть в Приложении А.

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Автоматически созданное описание

Рисунок – Результат перебора ключей шифра Цезаря

Диффи-Хеллман. Условие задачи: по незащищенному от прослушивания каналу связи передаются числа g и q. Алиса и Боб генерируют по одному числу, a и b соответственно. И отправляют друг другу результаты вычисления остатка от деления на q числа, равному возведению числа g в степень сгенерированного числа. Также дан остаток от деления числа g в степени a\*b. Необходимо найти a и b, значения которых не превышают 1000.

Алгоритм Диффи-Хеллмана – один из первых алгоритмов асимметричного шифрования, обеспечивающий возможность обмена ключами по незащищенному каналу. Суть алгоритма заключается в следующем:

1. Имеются две стороны обмена, которые прежде всего договариваются об использовании некоторого конечного поля F(q), определяют порождающий элемент поля g с помощью специальных алгоритмов. Далее выбирают некоторое большое простое число p. Полученные числа p и q будут являться общими исходными данными для дальнейшего обмена.
2. Сторона А генерирует секретный ключ (а) и отправляет второй стороне обмена результат вычисления выражения
3. Сторона В генерирует секретный ключ (b) и отправляет второй стороне обмена результат вычисления выражения
4. Полученные значения обе стороны еще раз возводят в свою секретную степень

В результате обе стороны получают одинаковое значение – ключ обмена [13].

Путем итеративного вычисления a и b с использованием разных значений i и j, а затем сравнения остатков с заданными целевыми значениями, разработанный алгоритм находит значения a и b. Код реализации алгоритма можно увидеть в Приложении Б.

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Рисунок – Результат поиска секретных ключей

Xor 24. Условие задачи: есть файл, зашифрованный с помощью XOR, получите ключ. Длина ключа — 24 байта, алфавит исходного файла — «qwertyuiopasdfghjklzxcvbnm\_» (без кавычек) и символ перевода строки. Других символов в исходном файле не было.

В криптографии простой шифр XOR представляет собой тип аддитивного шифра, алгоритма шифрования, который работает в соответствии с принципами:

,

,

,

,

,

где ⊕ обозначает операцию исключающей дизъюнкции (XOR).

Для решения задачи на языке программирования C++ был реализован алгоритм вычисления ключа. Зашифрованный текст разделяется на группы, количество которых соответствует длине ключа. Каждая группа содержит символы, которые кодируются одним и тем же символом ключа. Затем для каждой группы анализируются частоты символов, чтобы найти символ, который чаще всего встречается. Этот символ считается наиболее вероятным символом, который был зашифрован с использованием соответствующего символа ключа. Далее наиболее часто встречающиеся символы в каждой группе XOR'ятся с символом, который наиболее часто встречается в алфавите. В рассматриваемом случае это пробел (нижнее подчеркивание). Результат XOR'а дает символ ключа для этой группы.

Код реализации алгоритма можно увидеть в Приложении B.

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Автоматически созданное описание

Рисунок – Результат вычисления ключа

2.2. Решение Web-задач

В данной работе были разобраны задачи на SQL-инъекции.

SQL-инъекция – это атака, при которой злоумышленником производится вставка вредоносного кода в строки, передающиеся на сервер системы управления базами данных (СУБД) для синтаксического анализа и выполнения. Успешная реализация данной атаки позволяет обойти систему безопасности приложения и получить доступ к конфиденциальной информации, которая содержится в базе данных (БД), а также к функциональным возможностям СУБД и в некоторых случаях – доступ к операционной системе сервера, на котором функционирует СУБД. Как следствие, злоумышленник получает доступ к конфиденциальной информации, содержащейся в БД, и доступ к командам операционной системы сервера СУБД, тем самым делая его плацдармом для последующих атак других серверов и приложений, расположенных в корпоративной сети организации. Как правило, SQL-инъекции рассматривают применительно к web-приложениям, на самом деле данным уязвимостям подвержены любые клиент-серверные и сервис-ориентированные приложения, работающие с СУБД.

В CTF заданиях на SQL-инъекции участникам обычно предлагается извлечь данные из базы данных, обойти механизмы аутентификации и авторизации, изменить данные, выполнить запросы с повышенными привилегиями, о пределить структуру базы данных.

Первое задание – запрос со всеми данными, которые нам заведомо известны.

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рисунок – Запрос к базе данных

Второе задание заключается в том, что входные данные не фильтруются и можно закрыть кавычку и открыть новую, вставив еще одну команду OR.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок – Инъекция с помощью дополнительных кавычек

Третья задача аналогична второй, только в запросе добавляется лимит на одну выводимую строчку. Чтобы обойти эту проверку, можно добавить комментарий, который «съедает» команду LIMIT.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок – Обход LIMIT

1. РАЗРАБОТКА СОБСТВЕННЫХ ЗАДАЧ CTF
   1. Структура формата PNG

Для написания собственных задач была выбрана категория «Стеганография». Перед началом работы была изучена структура PNG. Структура PNG в общем случае представляет собой подпись (сигнатуру формата) и некоторое количество блоков (чанков) [15].

Изображение выглядит как текст, Шрифт, снимок экрана, логотип

Автоматически созданное описание

Рисунок – Структура формата PNG

Чанк IHDR определяет основные параметры изображений и содержит следующие поля:

* ширина, 4 байта;
* высота, 4 байта;
* битовая глубина (bit depth), определяет количество бит на каждый сэмпл(не пиксель), 1 байт;
* тип цвета, состоит из 3 флагов: 1 (используется палитра), 2 (используется цвет, не монохромное изображение) и 3 (присутствует альфа-канал), 1 байт;
* метод сжатия. На данный момент доступно только значение 0 — сжатие по алгоритму deflate. Если значение отлично от 0, чанк считается нераспознанным, и декодер рапортует об ошибке, 1 байт;
* метод фильтрации. Так же, как и в случае сжатия, на данный момент может быть только нулем, 1 байт;
* interlace(переплетение) метод. Определяет порядок передачи данных. На данный момент доступно 2 значения: 0 (no interlace) и 1 (Adam7 interlace), 1 байт.

Помимо прочего данный чанк отвечает за тип цвета, который у PNG может быть следующим:

* монохромное изображение;
* цветное изображение (RGB);
* цветное изображение с использованием палитры;
* монохромное изображение с альфа-каналом;
* цветное изображение с альфа-каналом [16].

Затем был подробнее изучен метод встраивания сообщения в графические файлы. Условие использования данного метода заключается в том, что объем зашифрованной информации не должен превышать размер файла-хранилища. Для этого используют программы, которые производят подмену определенных пикселей изображения. Это становится возможным благодаря тому, что каждый пиксель имеет размерность двоичного представления и может быть закодирован каким-то числом бит. Младший значащий бит (LSB) идеально подходит для изменения, так как он не заметен человеческому глазу и содержит меньше всего информации об изображении [14]. Кроме этого, одним из способов скрытия информации является изменение расширения файла. Изображение может оказаться архивом и чаще всего это видно, если картинка весит больше, чем должна весит при таких размерах. Архив обычно содержит в себе еще медиа-файлы, которые и хранят скрытое сообщение. Важно отметить значение метаданных изображения при использовании стеганографии. Например, на соревнованиях CTF можно встретить задачи, где флаг хранится в exif-файле изображения или в geo-тегах. Метаданные можно получить с помощью онлайн-сервисов или специализированных утилит. Недостатком применения стеганографии к изображениям является то, что не все форматы поддаются преобразованиям. Многие после модификаций теряют младшие биты. [2]

* 1. Сокрытие информации заменой красного оттенка пикселей

Был рассмотрен способ шифрования текста в изображении с помощью скрипта на языке программирования Python с помощью библиотеки PIL. Библиотека изображений Python добавляет возможности обработки изображений, обеспечивает обширную поддержку форматов файлов, эффективное внутреннее представление и мощные возможности обработки изображений. Основная библиотека изображений предназначена для быстрого доступа к данным, хранящимся в нескольких базовых пиксельных форматах, обеспечивает прочную основу для общего инструмента обработки изображений [17].

В разработанной программе каждый символ исходного текста переводится в число ASCII, генерируется кортеж случайных координат пикселей, в каждом из этих пикселей красный оттенок по RGB заменяется на код символа шифруемого текста, все ключи записываются в текстовый файл.

Код был протестирован на небольшом изображении (369 х 241) и сообщением «hello».



Рисунок – Исходное изображение



Рисунок – Изображение с зашифрованным сообщением "hello"

Изменения в файле незаметны. Но при попытке зашифровать большой текст (6250 символов), действия над изображением заметны.



Рисунок – Изображение с большим зашифрованным сообщением

Код для зашифрования и дешифрования текста можно увидеть в приложениях Г и Д.

Данный подход прост в реализации, но достаточно чувствителен к изменениям. Основная проблема – слабая стегостойкость. Так, например, существует интеллектуальное программное обеспечение, которое для выявления стеганографии проверяет области, состоящие из одного сплошного цвета [14].

Примером такого приложения может быть Beyond Compare – утилита, которая позволяет сравнивать содержимое директорий и файлов. При наличии исходного изображения можно выявить факт изменения некоторых пикселей.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок – Обнаружение сокрытия информации в файле

Также были рассмотрены утилиты, позволяющие выявлять аномалии в изображениях. Например, StegoSuit [18]. StegoSuit – это инструмент для извлечения секретных сообщений, скрытых в изображениях. Он имеет 5 основных функций, которые помогут пользователю проанализировать изображение.

Первый модуль – анализ уровня ошибок. Это выполняется в цифровых данных со сжатием с потерями, таких как JPEG. В этом модуле изображение подвергается сжатию и сравнивается с исходным изображением, которое затем умножается на коэффициент масштабирования, что дает нам область с различным соотношением шума в том месте, где редактируется изображение. Это может быть очень полезно при обнаружении поддельных изображений и обнаружении изображения, с которым были произведены манипуляции.

Второй модуль – пороговый анализ изображения. Это очень полезно при анализе изображений с более темной областью или оттенками. Это преобразует изображение в двоичное изображение, которое в основном представляет собой изображение с двумя цветами: черным, представленным 0, и белым, представленным 1. Это преобразует изображение в черно-белый формат и отображает только ту часть изображения, значение которой превышает указанное пороговое значение.

Третий модуль – обнаружение краев – используется для обнаружения всех мельчайших фрагментов изображения, которые не видны невооруженным глазом, и помогает определить основную структуру изображения. Он также выделяет все края изображения, поэтому исследователь может легко проверить наличие любых неровностей на изображении.

Четвертый модуль – это анализ метаданных, которые предоставляют все данные изображения, включая местоположение и устройство, используемое для щелчка по изображению, а также время, в которое был сделан щелчок по изображению.

Пятый модуль – это анализ строк, который помогает находить любой скрытый текст, присутствующий на изображении. При этом будет отображен весь читаемый текст, который инструмент найдет на изображении, и отобразится в текстовом поле present.

Для изображения, в которое была зашифрована информация, также был проведен анализ и выявлены аномалии (характерные черные точки).

Изображение выглядит как текст, кот, Мультимедийное программное обеспечение, снимок экрана

Автоматически созданное описание

Рисунок – Выявление аномалий

Еще одна утилита для анализа – StegSolve [19]. С помощью различных манипуляций над форматом открытия изображения утилита позволяет визуально выделить аномалии (характерные красные точки) в изображении.

Изображение выглядит как кот, текст, Мелкие и средние кошки, Кошачьи

Автоматически созданное описание

Рисунок – Выявление аномалий

Также существует множество утилит для сокрытия текста в картинках, например, Anubis, DeEgger Embedder, DeepSound, Hallucinate и другие.

Была подробнее рассмотрена утилита Hallucinate, написанная на Java. Интерфейс у Hallucinate прост и функционален. Требуется выбрать контейнер, указать скрываемый в нем файл и желаемую степень качества итоговой картинки. Доступны восемь вариантов. Чем сильнее огрубляется исходное изображение, тем больше в нем можно спрятать, но тем заметнее становятся артефакты.

Изображение выглядит как текст, снимок экрана, число, дисплей

Автоматически созданное описание

Рисунок – Интерфейс утилиты Hallucinate

Изображение выглядит как текст, кот, снимок экрана, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок – Сравнение файлов после сокрытия данных

* 1. Сокрытие информации в метаданных

Метаданные изображения представляют собой дополнительную информацию о файле изображения, которая может включать различные детали о нем. Эта информация обычно сохраняется внутри файла изображения и может быть использована для идентификации, классификации или улучшения работы с изображением.

Для изменения метаданных существует множество программ и утилит. Одна из них – TweakPNG. Ниже можно увидеть ее интерфейс при открытии изображения.



Рисунок – Исходное изображение

Изображение выглядит как текст, программное обеспечение, Значок на компьютере, веб-страница

Автоматически созданное описание

Рисунок – Интерфейс утилиты TweakPNG

Можно изменить поле tEXt, добавив в него текст. Для усложнения задачи можно закодировать флаг, например, с помощью Base64, отличающийся содержанием лишь цифр, латинских букв в верхнем и нижнем регистре и знаков «+» и «/».

Изображение выглядит как текст, программное обеспечение, снимок экрана, Значок на компьютере

Автоматически созданное описание

Рисунок – Вставка закодированного флага

Вид изображения при этом не изменился. Тогда любая утилита, работающая с метаданными изображения, отобразит полученный результат. Например, можно воспользоваться онлайн-сервисом METADATA2GO. Затем с помощью сайта www.base64encode.org можно раскодировать сообщение.

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

Рисунок – Отображение измененных метаданных

Изображение выглядит как текст, снимок экрана, веб-страница, программное обеспечение

Автоматически созданное описание

Рисунок – Получение флага

3.4. Сокрытие файлов

Еще один способ сокрытия информации в изображении – помещение архива в это изображение. Исходное изображение не может быть открыто с помощью утилиты WinRAR. Но с помощью команды copy в командной строке можно «склеить» картинку и архив. После этого получившаяся картинка будет открываться архиватором, где можно обнаружить данные с флагом.

Изображение выглядит как текст, снимок экрана, собака

Автоматически созданное описание

Рисунок – Невозможность открыть исходное изображение в архиваторе

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок – "Склеивание" файлов

Изображение выглядит как снимок экрана, программное обеспечение, текст, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок – Открытие получившегося изображения в архиваторе

3.5. PNG-наполнение за границами картинки

Для реализации этого метода необходимо понимать структуру вышеупомянутого чанка IHDR, который содержит основную информацию о изображении. Первые 8 байт указывают парсеру, который будет открывать изображение, сколько пикселей по ширине и высоте нужно отобразить. При уменьшении этих значений можно сохранить исходное изображение, но отображать только его часть. Ниже будет рассмотрено исходное изображение и его отображение в HEX-редакторе.



Рисунок – Исходное изображение

Изображение выглядит как текст, Шрифт, число, снимок экрана

Автоматически созданное описание

Рисунок – Открытие изображения в HEX-редакторе и поиск байтов, отвечающих за ширину (1024 пикселей) и высоту (620 пикселей)

Если изменить значение ширины с 02 6C на 02 1E, то отображающееся изображение будет меньше. При этом при обратной операции будет получен доступ к информации о невидимом участке изображения.

Изображение выглядит как текст, Шрифт, программное обеспечение, снимок экрана

Автоматически созданное описание

Рисунок – Изменение поля чанка IHDR



Рисунок – Получившееся изображение

ЗАКЛЮЧЕНИЕ

В заключении данной курсовой работы можно подчеркнуть важность и актуальность проведения соревнований по решению задач CTF в современном мире, где роль информационных технологий становится все более значимой. Защита информации от кибератак и обеспечение информационной безопасности представляют собой приоритетные задачи, требующие постоянного совершенствования методов и подходов.

В ходе работы были исследованы виды CTF, рассмотрены некоторые задачи по направлениям «Криптография» и «Web», а также разработана собственная задача по направлению «Стеганография».

Таким образом, проведенное исследование подчеркивает актуальность CTF-соревнований в контексте обучения и тренировки специалистов по информационной безопасности. Внесение улучшений в стандарты оценки, автоматизацию и методику обучения позволит эффективнее подготавливать специалистов, способных эффективно справляться с вызовами современной киберсреды.

**СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

1. Mohammed Atiquzzaman An Empirical Survey of Functions and Configurations of Open-Source Capture the Flag (CTF) Environments // Journal of Network and Computer Applications. - 2020. - №151
2. Львова А. П., Калашникова В. А. АНАЛИЗ МЕТОДОВ ЗАЩИТЫ ИНФОРМАЦИИ НА ОСНОВЕ ИСПОЛЬЗОВАНИЯ СТЕГАНОГРАФИИ // Сборник статей XII международной научно-практической конференции. - М.: Научно-издательский центр «Актуальность.РФ», 2017. - С. 109-110.
3. Abbas Cheddad, Joan Condell, Kevin Curran, Paul Mc Kevitt Digital image steganography: Survey and analysis of current methods // Signal Processing. - 2010. - №3. - С. 727-752.
4. Горбунов К.С., Семакин А.Е., Зулькарнеев И.Р. ОРГАНИЗАЦИЯ ВНУТРИВУЗОВСКИХ СОРЕВНОВАНИЙ ПО ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ В ФОРМАТЕ CTF // Сборник материалов XV Всероссийской научно-практической конференции студентов, аспирантов и молодых ученых. - Курган: 2016. - С. 11-14.
5. Ульянова М.А. ФОРЕНЗИКА КАК ЧАСТЬ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ: ЗАДАЧИ, СРЕДСТВА, МЕТОДЫ, ПРИМЕРЫ ПРАКТИКИ // ВСЕРОССИЙСКАЯ СТУДЕНЧЕСКАЯ КОНФЕРЕНЦИЯ «СТУДЕНЧЕСКАЯ НАУЧНАЯ ВЕСНА», ПОСВЯЩЕННАЯ 85-ЛЕТИЮ Ю. А. ГАГАРИНА. - М.: Издательский дом "Научная библиотека", 2019. - С. 39-40.
6. Capture the Flag (CTF): Website Tutorial to Boost Cybersecurity Training // PRCR URL: https://prcrepository.org/handle/20.500.12475/316 (дата обращения: 16.10.2023).
7. Лапонина О.Р., Матошенко В.А. Сравнительный анализ CTF-платформ для обучения кибербезопасности // International Journal of Open Information Technologies. - 2022. - №10. - С. 31-44.
8. George Sharkov, Christina Todorova Capture the Flag for Cyber-Resilience Exercising through Cryptographic Puzzles and Collaborative Problem-Solving // Information & Security: An International Journal. - 2017. - №37. - С. 95-102.
9. The Development of an Open Source Intelligence Gathering Exercise for Teaching Information Security & Privacy // AIS Electronic Library URL: https://aisel.aisnet.org/mwais2018/3/ (дата обращения: 22.10.2023).
10. Nu1L Team Handbook for CTFers. - Shanghai, China: Consortia, 2022. - 796 с.
11. Тренировки, сезон 2019-2020 // Хакердом URL: https://2019.hackerdom.ru/board/ (дата обращения: 16.10.2023).
12. Реализация и анализ быстродействия алгоритма шифрования и дешифрования шифра Цезаря // КиберЛенинка URL: https://cyberleninka.ru/article/n/realizatsiya-i-analiz-bystrodeystviya-algoritma-shifrovaniya-i-deshifrovaniya-shifra-tsezarya/viewer (дата обращения: 23.10.2023).
13. Герасимов А.А., Пиунова А.П. CРАВНИТЕЛЬНЫЙ АНАЛИЗ РЕАЛИЗАЦИИ ПРОТОКОЛОВ ДИФФИ-ХЕЛЛМАНА И ЭЛЬ-ГАМАЛЯ В ЭЛЛИПТИЧЕСКОЙ КРИПТОГРАФИИ // ПРОБЛЕМЫ СОВРЕМЕННОГО МИРА ГЛАЗАМИ МОЛОДЕЖИ. - М.: Российский университет дружбы народов (РУДН), 2015. - С. 13-23.
14. Вахаб А., Романенко Д.М. Методы цифровой стеганографии на основе модификации цветовых параметров изображения // Труды БГТУ. Серия 3: Физико-математические науки и информатика. - 2018. - №1. - С. 94-98.
15. Дронов Д.Р., Никишова А.В. ВСТРАИВАНИЕ ДАННЫХ В PNG-ИЗОБРАЖЕНИЯ // ИЗВЕСТИЯ ВОЛГОГРАДСКОГО ГОСУДАРСТВЕННОГО ТЕХНИЧЕСКОГО УНИВЕРСИТЕТА. - 2018. - №5. - С. 85-90.
16. Portable Network Graphics (PNG) Specification (Third Edition) // W3C Candidate Recommendation Snapshot URL: https://www.w3.org/TR/png/#abstract (дата обращения: 05.11.2023).
17. Pillow (PIL Fork) 10.2.0 documentation // pillow URL: https://pillow.readthedocs.io/en/stable/index.html (дата обращения: 05.11.2023).
18. StegoSuit // GitHub URL: https://github.com/rajan98/StegoSuit (дата обращения: 20.01.2024).
19. Stegsolve // GitHub URL: https://github.com/eugenekolo/sec-tools/tree/master/stego/stegsolve/stegsolve (дата обращения: 20.01.2024).

**ПРИЛОЖЕНИЕ А**

Решение задания «Древнеримская замена»

#define ALPHABET\_SIZE 26

void **RomanDecrypt**(*const* std::string& roman\_message)

{

*for* (size\_t i = 0; i < ALPHABET\_SIZE; i++)

{

*for* (char c : roman\_message)

{

char decryptedChar = 'a' + (c - 'a' - i + ALPHABET\_SIZE) % ALPHABET\_SIZE;

std::cout << decryptedChar;

}

std::cout << std::endl;

}

}

ПРИЛОЖЕНИЕ Б

Решение задания «Диффи-Хеллман»

void Diffie\_Hellman()

{

size\_t q = 1357, g = 10, tmp\_a = 0, tmp\_b = 0, tmp\_ab = 0;

*for* (size\_t i = 0; i < 1000; i++)

{

tmp\_a = ModPow(g, i, q);

*if* (tmp\_a == 419)

{

*for* (size\_t j = 0; j < 1000; j++)

{

tmp\_b = ModPow(g, j, q);

*if* (tmp\_b == 34)

{

tmp\_ab = ModPow(g, i \* j, q);

*if* (tmp\_ab == 33)

{

std::cout << "a = " << i << std::endl;

std::cout << "b = " << j << std::endl;

*return*;

}

}

}

}

}

}

size\_t ModPow(size\_t base, size\_t exponent, size\_t mod)

{

size\_t result = 1;

base = base % mod;

*while* (exponent > 0) {

*if* (exponent % 2 == 1) {

result = (result \* base) % mod;

}

exponent = exponent >> 1;

base = (base \* base) % mod;

}

*return* result;

}

ПРИЛОЖЕНИЕ В

Решение задания «Xor 24»

#include <iostream>

#include <fstream>

#include <vector>

#include <map>

*//* *Функция* *для* *чтения* *зашифрованного* *текста* *из* *файла*

std::vector<char> **ReadEncryptedText**(*const* std::string& filename) {

std::ifstream encryptedFile(filename, std::ios::binary);

*if* (!encryptedFile.is\_open()) {

std::cerr << "Не удается открыть файл с зашифрованным текстом." << std::endl;

exit(1);

}

std::vector<char> encryptedText;

char ch;

*while* (encryptedFile.get(*ch*)) {

encryptedText.push\_back(ch);

}

encryptedFile.close();

*return* encryptedText;

}

*//* *Функция* *для* *нахождения* *ключа* *по* *зашифрованному* *тексту*

std::vector<char> **FindXORKey**(*const* std::vector<char>& encryptedText, int keyLength) {

std::vector<std::vector<char>> groups(keyLength);

*for* (size\_t i = 0; i < encryptedText.size(); i++) {

groups[i % keyLength].push\_back(encryptedText[i]);

}

char mostCommonChar = '\_';

std::vector<char> key(keyLength);

*for* (int i = 0; i < keyLength; i++) {

std::map<char, int> charFrequency;

*for* (char symbol : groups[i]) {

charFrequency[symbol]++;

}

char mostFrequentChar = ' ';

int maxFrequency = 0;

*for* (*const* *auto*& pair : charFrequency) {

*if* (pair.first != '\n' && pair.second > maxFrequency) {

maxFrequency = pair.second;

mostFrequentChar = pair.first;

}

}

key[i] = mostFrequentChar ^ mostCommonChar;

}

*return* key;

}

*//* *Функция* *для* *дешифровки* *текста* *с* *использованием* *ключа*

void **DecryptText**(*const* std::vector<char>& encryptedText, *const* std::vector<char>& key, *const* std::string& outputFilename) {

std::ofstream decryptedFile(outputFilename, std::ios::binary);

*if* (!decryptedFile.is\_open()) {

std::cerr << "Не удается создать файл для записи расшифрованного текста." << std::endl;

exit(1);

}

*for* (size\_t i = 0; i < encryptedText.size(); i++) {

char decryptedChar = encryptedText[i] ^ key[i % key.size()];

decryptedFile.put(decryptedChar);

}

decryptedFile.close();

std::cout << "Расшифрованный текст записан в " << outputFilename << std::endl;

}

int **main**() {

std::string encryptedFilename = "encrypted.txt";

std::string decryptedFilename = "decrypted.txt";

int keyLength = 24;

std::vector<char> encryptedText = ReadEncryptedText(encryptedFilename);

std::vector<char> key = FindXORKey(encryptedText, keyLength);

std::cout << "key: ";

*for* (char symbol : key) {

std::cout << symbol;

}

std::cout << std::endl;

DecryptText(encryptedText, key, decryptedFilename);

*return* 0;

}

ПРИЛОЖЕНИЕ Г

Исходный код шифрования текста в изображении

from PIL import Image, ImageDraw  
from random import randint  
  
  
def stega\_encrypt():  
 keys = [] # сюда будут помещены ключи  
 img = Image.open(input("path to image: ")) # создаём объект изображения  
 draw = ImageDraw.Draw(img) # объект рисования  
 width = img.size[0] # ширина  
 height = img.size[1] # высота  
 pix = img.load() # все пиксели тут  
 f = open('keys.txt', 'w') # текстовый файл для ключей  
  
 for elem in ([ord(elem) for elem in input("text here: ")]):  
 key = (randint(1, width - 10), randint(1, height - 10))  
 g, b = pix[key][1:3]  
 draw.point(key, (elem, g, b))  
 f.write(str(key) + '\n')  
  
 print('keys were written to the keys.txt file')  
 img.save("newimage.png", "PNG")  
 f.close()  
  
  
stega\_encrypt()

ПРИЛОЖЕНИЕ Д

Исходный код дешифрования текста в изображении

from PIL import Image  
from re import findall  
  
  
def stega\_decrypt():  
 a = []  
 keys = []  
 img = Image.open(input("path to image: "))  
 pix = img.load()  
 f = open(input('path to keys: '), 'r')  
 y = str([line.strip() for line in f])  
  
 for i in range(len(findall(r'\((\d+)\,', y))):  
 keys.append((int(findall(r'\((\d+)\,', y)[i]), int(findall(r'\,\s(\d+)\)', y)[i])))  
 for key in keys:  
 a.append(pix[tuple(key)][0])  
 return ''.join([chr(elem) for elem in a])  
  
  
print("you message: ", stega\_decrypt())