# НТО 2022-2023 Информационная безопасность

Название команды: «; DROP ALL TABLES; --»

Номер команды: 5

## Отчёт по наступательной кибербезопасности

### *1) CRYPTO-1*

1. Устанавливаем библиотеку *sagemath* командой «pip install sagemath»
2. Изменяем формат файла «medium_task» с *.sage* на *.py*; изучаем его.
3. Работаем в полученном .py-файле: удаляем строку `from flag import flag` и создаём список *m*. Присваиваем ему содержимое файла «hashed.txt».
4. Далее удаляем эту часть кода:

```
if __name__ == "__main__":
        dihedral = DihedralCrypto(1337)
        answer = dihedral.hash(flag)
        with open('hashed','w') as f:
                f.write(str(answer))
```

   И вместо него пишем данный код:

```
if __name__ == "__main__":

        dihedral = DihedralCrypto(1337)
        res = ''
        for i in m:
                for j in range(10000):
                        answer = dihedral.hash([j])
                        if answer == [i]:
                                res += str(j) + ' '
                                break
        print(res)
```

   Этот код методом перебора составляет строку *res* из элементов закодированного флага.

5. С помощью *CyberChef* декодируем последовательность десятичных чисел *res* операцией «From Decimal» и тем самым получаем флаг.

### *2) CRYPTO-2*

1. По коду, приведённому в данном веб-файле, мы можем увидеть, что в зависимости от запрашиваемого бита выполняется либо функция *pow()*, либо функция *randint()*. Эти функции имеют разную сложность: *pow()* выполняется дольше, чем *randint()*.
2. По времени, за которое отвечает сервер, определяем, какая функция вызывается и в соответствии с этим определяем значение запрашиваемого бита. Проделываем эту операцию со всеми битами.
3. С помощью *CyberChef* декодируем полученные биты операцией «From Binary» и получаем флаг.

### *3) REVERSE-1*

1. Проанализируем через *Ghidra* данный .exe файл («hello_ZmDKPvB.exe»): обнаруживаем в конце главной функции цикл, отвечающий за вывод текста. Внутри цикла находится прерывание BIOS (int 0x15, 0x86), которое заставляет процессор останавливаться на указанное количество времени.
2. Скачиваем с *github.com* исходный код *DOSBox*.
3. В исходном коде меняем функцию, обрабатывающую прерывание BIOS так, чтобы время ожидания было равно нулю.
4. Компилируем и запускаем *DOSBox*, запускаем в нём данный .exe файл и получаем флаг.

### *4) WEB-2*

1. Используя уязвимость «*HTTP request smuggling*», указываем внутри *username* GET-запрос на второй сервис.
2. Переходим на корневую страницу первого сервиса. Замечаем, что от второго сервиса вернулись ответы на запрос от первого сервиса и на запрос из username. Теперь найдем способ передать флаг из заголовка запроса обратно.
3. Дописываем в конец *username* http-заголовок «X-Forwarded-For» (XFF) так, чтобы флаг попал не в cookie, а в этот заголовок. Теперь второй сервис вернёт ответ о неудаче вместе с флагом в точно таком же заголовке.

# Отчёт по расследованию инцидента

## 1) Подготовка

1. Скачан установочный образ Xubuntu (Live) [http://mirror.yandex.ru/ubuntu-cdimage/xubuntu/releases/22.10/release/](http://mirror.yandex.ru/ubuntu-cdimage/xubuntu/releases/22.10/release/)

2. VM загружена с этого образа

## 2) Куда logkeys пишет логи?

1. При изучении содержимого папки /var/log диска машины обнаружен файл logkeys.log, содержащий логи нажатий клавиш на клавиатуре

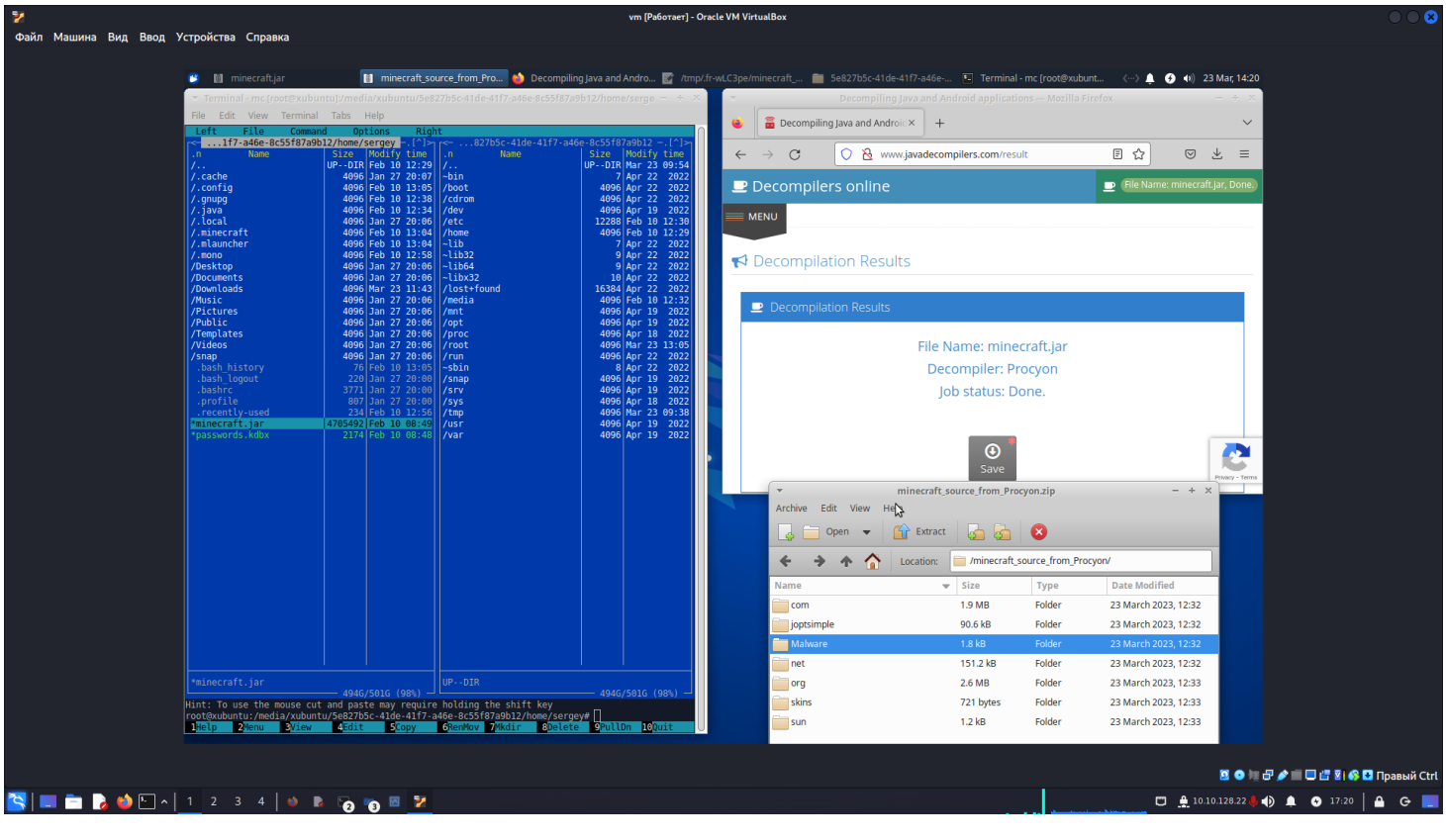
   Ответ: /var/log/logkeys.log

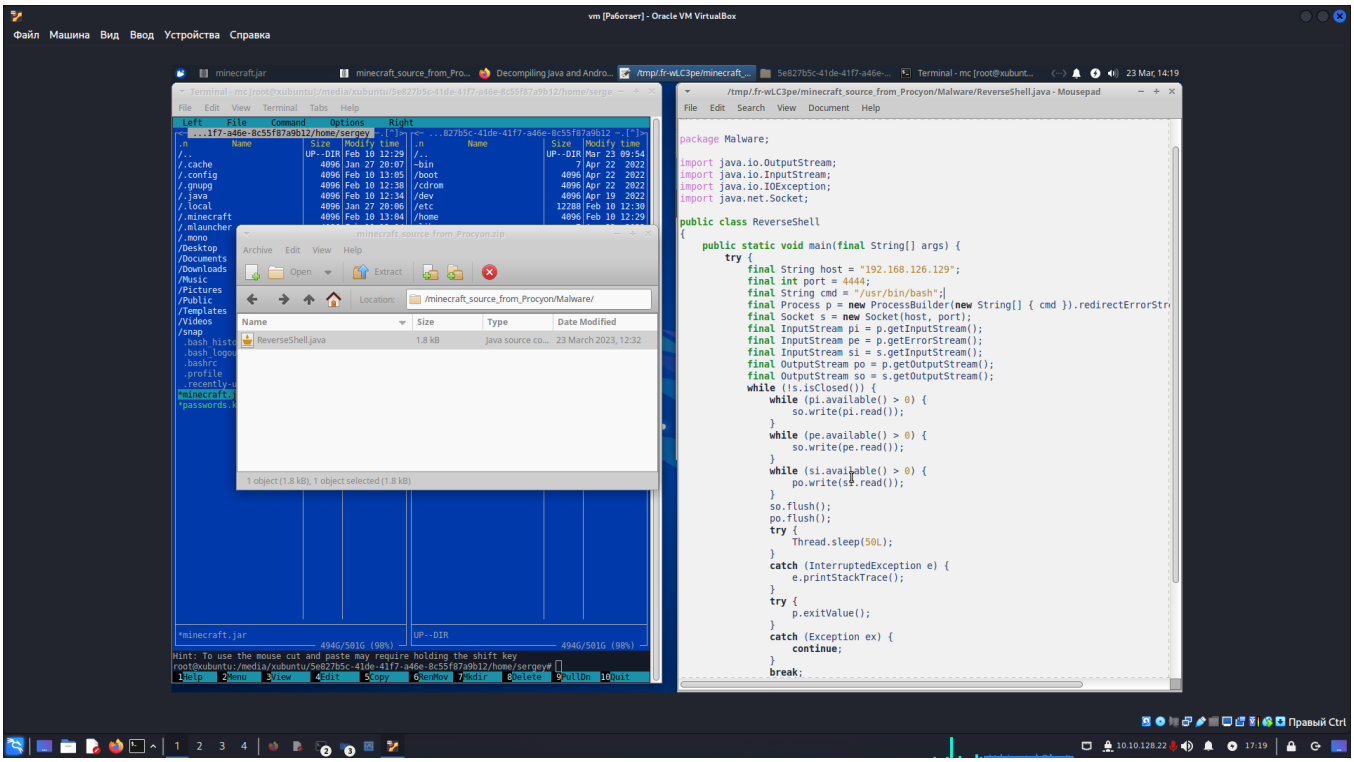
## 3) Пароль от чего лежит в passwords.kdbx?

1. Через консольную утилиту find находим файл passwords.kdbx Он расположен по пути /home/sergey/passwords.kdbx
2. Воспользовавшись поиском в интернете находим, что расширение .kdbx используется программой KeePass.
3. Учитывая то, что на машине работал кейлоггер, выполняем поиск в нём названия программы без учёта регистра Находим ввод: keepass2 После него - длинная строка - пароль от защищённой БД с паролями.
4. Устанавливаем эту программу sudo apt install keepassxc
5. Открываем через неё файл БД менеджера паролей
6. Вводим пароль от БД, пользуясь сохранёнными кейлоггером данными Пароль: 1_D0N7_N0W_WHY_N07_M4Y83_345Y
7. Видим сохранённый пароль, подписанный windows_rdp Логин: Administrator Пароль: SecretP@ss0rdMayby_0rNot&

   Ответ: в указанном файле лежит пароль от учётной записи с административными правами для доступа по протоколу удалённого рабочего стола.

## 4) Как злоумышленник попал на машину?





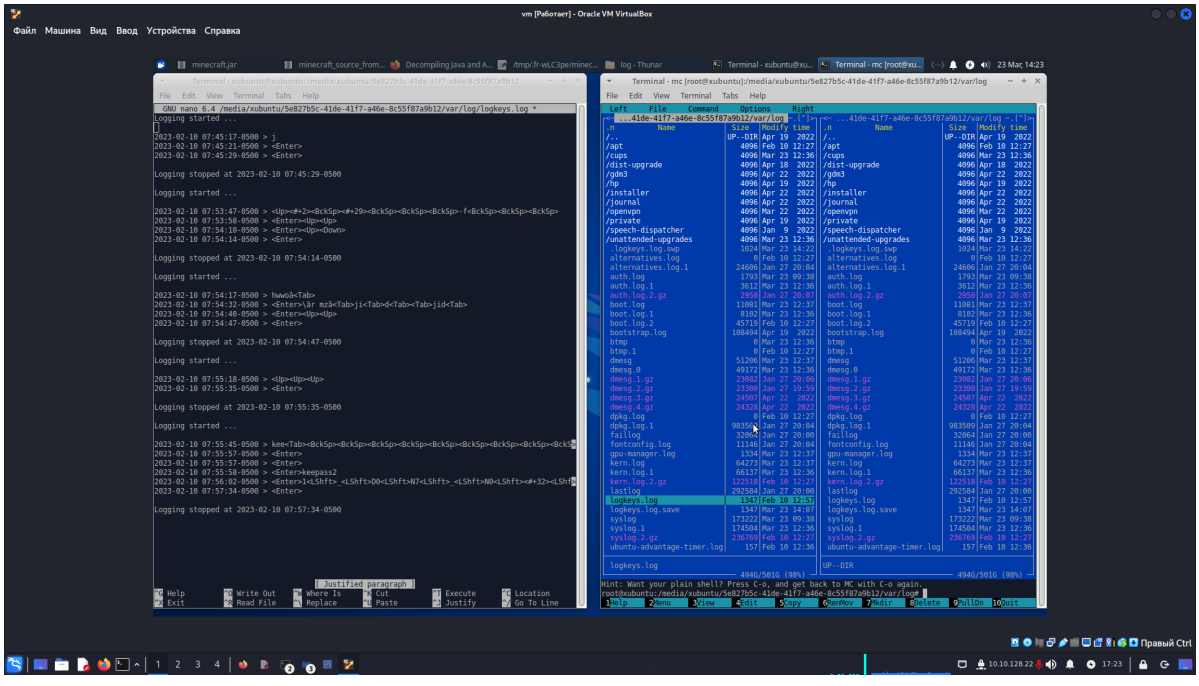

1. Просмотрим историю bash пользователя /home/sergey/.bash_history В нём видим, что пользователь запускал сторонний файл minecraft.jar.
2. Проведём декомпиляцию этого файла с помощью декомпилятора Procyon обнаруживаем внутри jar каталог Malware и ReverseShell.java

   Ответ: злоумышленник попал на машину с помощью ведоносного ПО в файле minecraft.jar, который был запущен пользователем.

## 5) Как злоумышленник узнал пароль от passwords.kdbx?

Записи о нажатых при вводе пароля клавишах хранятся в /var/log/logkeys.log (был найден ранее).

Ответ: злоумышленник узнал пароль от passwords.kdbx с помощью кейлоггера.



## 6) Как повысил свои права?

При помощи скрипта linpeas.sh в папке Downloads, злоумышленник повысил свои права, поставил SUID флаг для /usr/bin/find.