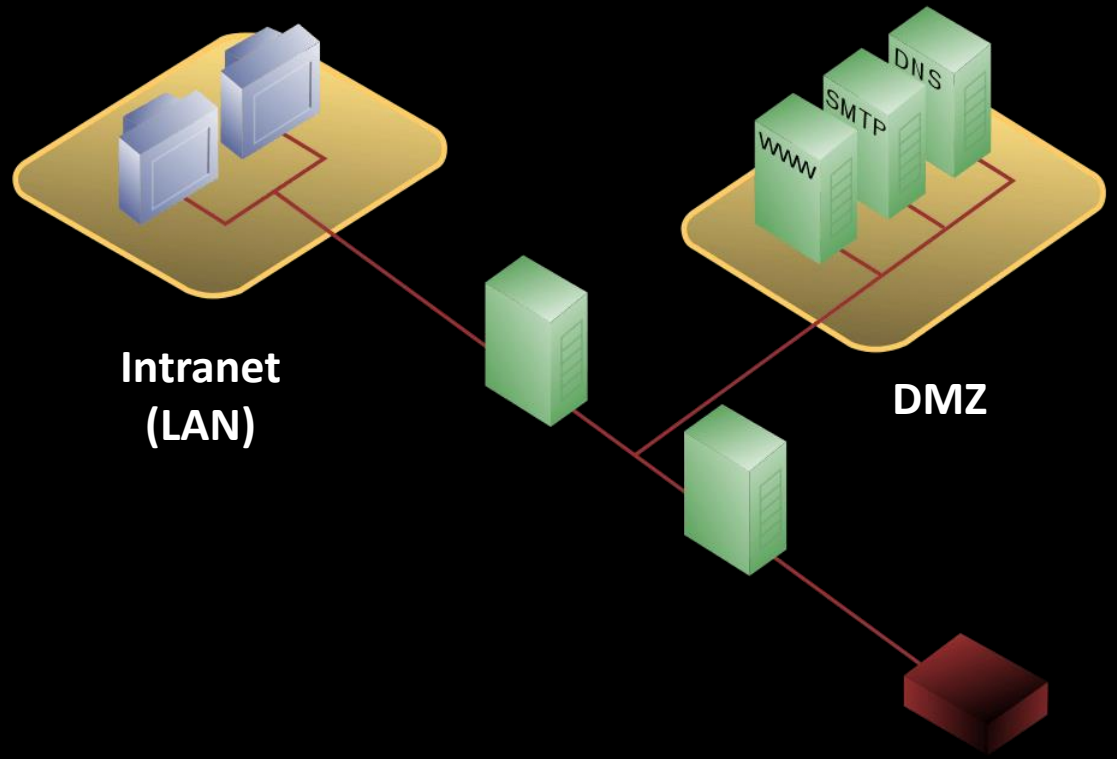
Progettazione e Configurazione di una rete aziendale protetta da due Firewall e dotata di DMZ

Progetto di Network Security

Emma Melluso matr. M63001176
Carmine Pio D'Antuono matr. M63001224
Pasquale Gaviglia matr. M63001188

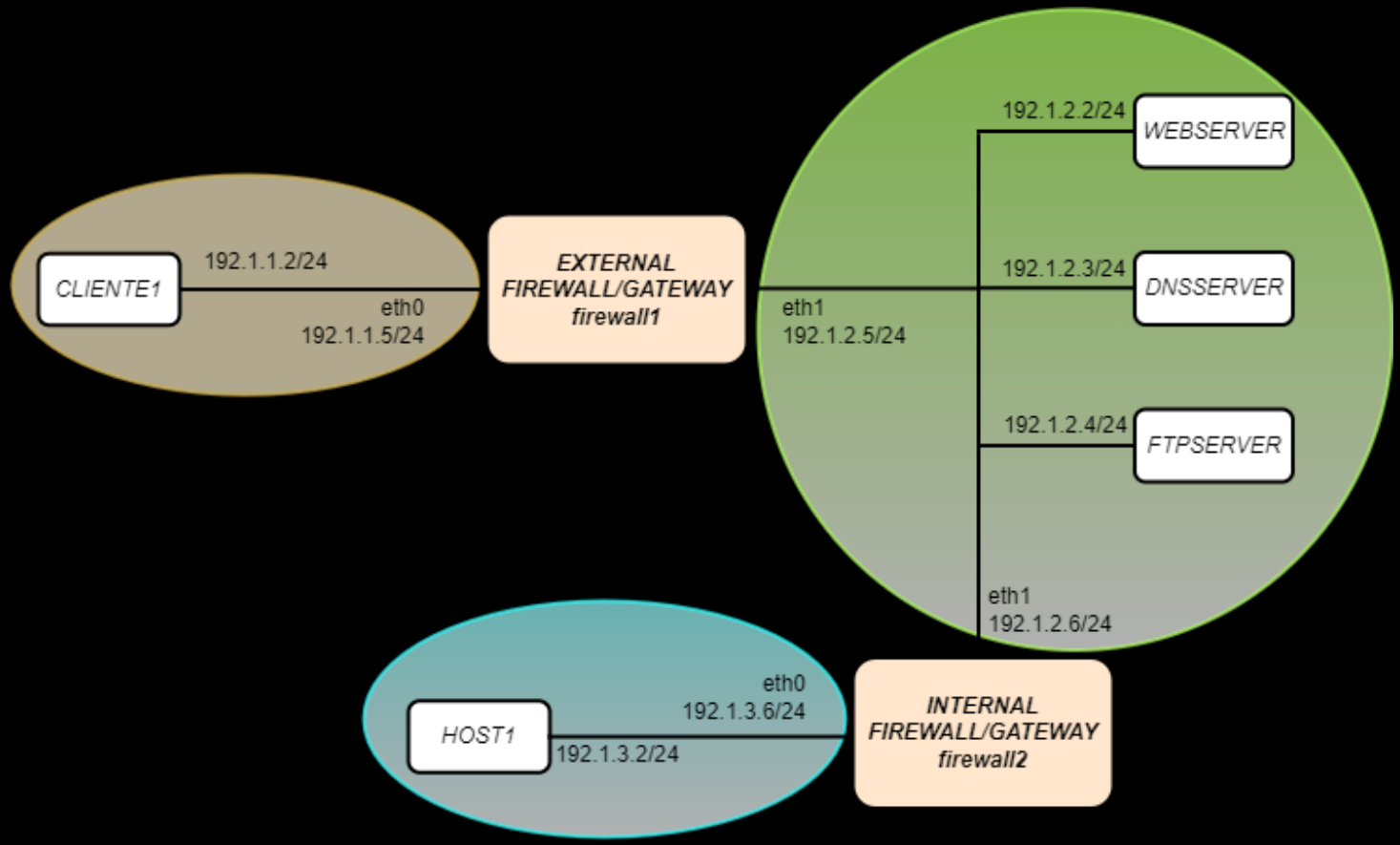
OBIETTIVI

- Realizzazione in ambiente controllato di un testbed di rete che emula, in maniera realistica, la configurazione di una rete aziendale protetta da ***firewall*** e dotata di ***DMZ - DeMilitarized Zone***.



PROGETTAZIONE DELLA RETE

- Architettura su due livelli in modo da:
  - Evitare situazioni di *single point of failure.*
  - *Gestione più efficiente ed agevole della rete stessa.*
  - Separazione netta tra rete esterna e rete interna.



CONFIGURAZIONE RETE CON DOCKER

In che modo?

- Creazione di *Dockerfile* appositi per avere immagini customizzate in locale.

- Upload delle immagini su Docker Hub.

- Creazione della rete sfruttando le funzionalità offerteci dalla piattaforma Docker.

Immagini realizzate:

1. ***hostubuntu*** → utilizzata per cliente esterno e per host interno. Parte da una semplice immagine Ubuntu a cui sono stati integrati tool di utilità per il testing della rete.
2. ***firewall_ulogd2*** → utilizzata per i due firewall. Parte da una semplice immagine Ubuntu a cui sono stati integrati vari tool tra cui **iptables** e **ulogd2.**

Immagini di default:

1. ***web_server***
2. ***dns_server***
3. ***ftp_server***

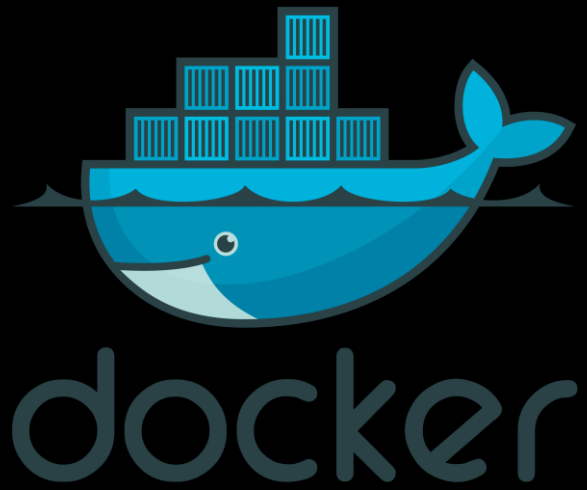
Dockerfile Firewall

```
FROM ubuntu:latest

LABEL maintainer emme "emmamelluso@libero.it"

RUN apt-get update && apt-get install -y \
bridge-utils \
net-tools \
iptables \
ulogd2 \
nano

CMD echo "Dockerfile Firewall. Iptables : " && \
iptables -L
```



Dockerfile Host Ubuntu

```
FROM ubuntu:latest

LABEL maintainer emme "emmamelluso@gmail.com"

RUN apt-get update && apt-get install -y \
bridge-utils \
net-tools \
iputils-ping \
nmap \
hping3 \
ftp

CMD echo "Ubuntu Host"
```

CREAZIONE DELLA RETE: script Bash

Sono state create le tre seguenti sottoreti:

- Rete interna con indirizzo 192.1.3.0/24, a cui è stato connesso il container *host1.*

```
docker network create --driver bridge --subnet=192.1.3.0/24 rete_interna
docker run --privileged --network=rete_interna --ip 192.1.3.2 -td --name=host1 hostubuntu bash
```

- Rete esterna con indirizzo 192.1.1.0/24, a cui è stato connesso il container *cliente1.*

```
docker network create --driver bridge --subnet=192.1.1.0/24 rete_esterna
docker run --privileged --network=rete_esterna --ip 192.1.1.2 -td --name=cliente1 hostubuntu bash
```

- Rete DMZ con indirizzo 192.1.2.0/24, a cui sono stati connessi i container di *web-server, dns-server* ed *ftp-server.*

```
docker network create --driver bridge --subnet=192.1.2.0/24 dmz
docker run --privileged --network=dmz --ip 192.1.2.2 -p80:80 -p443:443 -tdi --name=webserver
    linode/lamp bash
docker exec --privileged -t webserver service apache2 start
docker run --privileged --network=dmz --ip 192.1.2.3 -p53:53/udp -tdi --name=dnsser
    cosmicq/docker-bind:latest bash
docker run --privileged --network=dmz --ip 192.1.2.4 -p20:20 -p21:21 -tdi --name=ftpserver
    ftpser bash
```

CREAZIONE DELLA RETE: script Bash

È stato runnato il container del *firewall1*:

- Abilitando l'ip forwarding
- Connettendolo alla rete esterna
- Riavviando il demone ***ulogd2***
- Infine connettendolo alla dmz

```
docker run --privileged --sysctl net.ipv4.ip_forward=1 --network=rete_esterna --ip 192.1.1.5
    -td --name=firewall1 emmame/firewall_ulogd2 bash
docker exec --privileged -t firewall1 service ulogd2 restart
docker network connect --ip 192.1.2.5 dmz firewall1
```

Sono stati eseguiti comandi analoghi per il *firewall2*, il quale è stato collegato a **rete interna** e **DMZ**.

CREAZIONE DELLA RETE: script Bash

Inserimento delle regole di routing nei container della rete.

```
docker exec cliente1  route add default   gw firewall1
docker exec host1     route add default   gw firewall2
docker exec webserver route add 192.1.1.2 gw firewall1
docker exec webserver route add 192.1.3.2 gw firewall2
docker exec dnsser    route add 192.1.1.2 gw firewall1
docker exec dnsser    route add 192.1.3.2 gw firewall2
docker exec ftpserver route add 192.1.1.2 gw firewall1
docker exec ftpserver route add 192.1.3.2 gw firewall2
```

Per i servizi della DMZ è stata pensata la seguente policy di instradamento:

- Default Gateway = firewall1, per comunicazioni con rete esterna.
- Default Gateway = firewall2, per comunicazioni con rete interna.

Cliente esterno ed host interno non possono in alcun modo comunicare, dato che non è stata prevista una policy di instradamento che li collega.

CREAZIONE DELLA RETE: docker-compose

- La configurazione precedente è stata ulteriormente riprodotta mediante il *tool* docker-compose.

(*Nella foto è stato riportato parte del file docker-compose.yml)*

```
version: "3"
services:
    cliente1:
        image: emmamc/simpleubuntu
        container_name: c1-esterno
        tty: true
        stdin_open: true
        privileged: true
        command: bash -c "route add default gw f1-esterno && bash"
        networks:
            net-esterna:
                ipv4_address: 192.1.1.2
    firewall1:
        image: emmamc/firewall_ulogd2
        container_name: f1-esterno
        tty: true
        stdin_open: true
        privileged: true
        command: bash -c "service ulogd2 restart && bash"
        sysctls:
            - net.ipv4.ip_forward=1
        networks:
            net-esterna:
                ipv4_address: 192.1.1.5
            net-dmz:
                ipv4_address: 192.1.2.5
```

IPTABLES

- Implementazione **Firewall Packet Filtering** mediante il tool iptables.

Il tool gestisce delle tabelle tra cui:
- Filter
- Nat

- Nella tabella filter possono essere istanziate 3 tipi di catene (lista di regole):

- Input
- Forward
- Output

Per come è stato implementato il nostro progetto abbiamo inserito regole nella sola catena di **FORWARD**

ULOGD2

- Per evitare possibili attacchi verso l'Host che ospita il container, la regola *–j LOG* è disabilitata. Per questo motivo è stato necessario configurare il daemon **ulogd2**.

- Ulogd2 è uno userspace logging daemon per il logging dei pacchetti in transito nella rete mediante netfilter/iptables.

- Sintassi dei log risulta modificata rispetto a quella base: si passa da *–j LOG* a *–j NFLOG*

ULOGD2: utilizzo

- Nel nostro progetto i LOG sono stati utilizzati per il debug delle regole di sicurezza.
- È stata inserita una regola di log prima e dopo il set di regole da testare.
- Ci accorgiamo che il test va a buon fine (le regole inserite impediscono la riuscita dell'attacco) nel momento in cui nei log compare solo la prima tra le due regole

```
# Protezione Ping of Death Attack
docker exec --privileged -t firewall1 iptables -A FORWARD -j NFLOG --nflog-prefix="FORWARD Log pre-regola: "
docker exec --privileged -t firewall1 iptables -N PING_OF_DEATH
docker exec --privileged -t firewall1 iptables -A FORWARD -p icmp -j PING_OF_DEATH
# Accetto tutte le richieste se rispettano i limiti prefissati
docker exec --privileged -t firewall1 iptables -A PING_OF_DEATH -p icmp --icmp-type echo-request -m limit --limit 1/s -j RETURN
# Se non ho un match con la regola di sopra il pacchetto va necessariamente scartato
docker exec --privileged -t firewall1 iptables -A PING_OF_DEATH -p icmp --icmp-type echo-request -j DROP
docker exec --privileged -t firewall1 iptables -A FORWARD -j NFLOG --nflog-prefix="FORWARD Log post-regola: "
```

- Inoltre i log potrebbero essere eventualmente analizzati in modo da tenere traccia della tipologia di traffico che attraversa la rete

ULOGD2: ulogd.conf

- È stato definito il path del main logfile

```
# logfile for status messages
logfile="/var/log/ulog/ulogd.log"
```

- Sono stati decommentati i plugin necessari al corretto funzionamento del log tra cui:
- *ulogd_raw2packet_BASE.so*, forse quello più importante, il quale permette di interpretare gli header di svariate tipologie di pacchetto
- *ulogd_output_LOGEMU.so*, plugin di output il quale emula il target standard LOG e permette il salvataggio dei pacchetti in un file.

- È stata lasciata la configurazione di default

```
# this is a stack for logging packet send by system via LOGEMU
stack=log1:NFLOG,base1:BASE,ifi1:IFINDEX,ip2str1:IP2STR,print1:PRINTPKT,emu1:LOGEMU
```

- È stato definito il file su cui verranno loggati i pacchetti

```
[emu1]
file="/var/log/ulog/syslogemu.log"
sync=1
```

IPTABLES: Regole (1)

- Vengono resettate le catene standard (e non) nel firewall

```
# Eliminazione catene standard #
docker exec --privileged -t firewall1 iptables -F
docker exec --privileged -t firewall1 iptables -F -t nat

# Eliminazione catene non standard vuote #
docker exec --privileged -t firewall1 iptables -X
```

- Di policy di default (ACCEPT) di iptables è stata settata a DROP

```
# Policy di base #
docker exec --privileged -t firewall1 iptables -P INPUT DROP
docker exec --privileged -t firewall1 iptables -P OUTPUT DROP
docker exec --privileged -t firewall1 iptables -P FORWARD DROP
```

IPTABLES: Regole (2)

- Sono state impostate le regole per permettere il corretto istradamento dei pacchetti nella rete:

```
docker exec --privileged -t firewall1 iptables -t filter -A FORWARD -i eth0 -o eth2
    -m state --state NEW,ESTABLISHED,RELATED -j DROP
```

- I pacchetti dalla rete esterna verso la DMZ sono accettati:

```
docker exec --privileged -t firewall1 iptables -t filter -A FORWARD -i eth0 -o eth1
    -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
docker exec --privileged -t firewall1 iptables -t filter -A FORWARD -i eth1 -o eth0
    -m state --state ESTABLISHED,RELATED -j ACCEPT
```

- I pacchetti dalla rete interna verso la DMZ sono accettati:

```
docker exec --privileged -t firewall1 iptables -t filter -A FORWARD -i eth2 -o eth1
    -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
docker exec --privileged -t firewall1 iptables -t filter -A FORWARD -i eth1 -o eth2
    -m state --state ESTABLISHED,RELATED -j ACCEPT
```

SECURITY RULES (1)

PACCHETTI FRAMMENTATI

- Inserimento regola per effettuare il DROP dei pacchetti IP frammentati

```
docker exec --privileged -t firewall1 iptables -A FORWARD -p ip -f -j DROP
```

Lo scopo di tale regola è evitare attacchi come il *Tiny fragment attack* che potrebbero causate DoS

SECURITY RULES (2)

PACCHETTI «NO-SENSE»

- Pacchetti TCP che non hanno alcuna funzionalità nella comunicazione
- Molte operazioni di scanning e/o attacchi partono dall'invio di questo tipo di pacchetti con flag settati ad arte per infastidire il sistema:

```
docker exec --privileged -t firewall1 iptables -A FORWARD -p tcp --tcp-flags ALL ACK,RST,SYN,FIN -j DROP
docker exec --privileged -t firewall1 iptables -A FORWARD -p tcp --tcp-flags ALL ALL -j DROP
docker exec --privileged -t firewall1 iptables -A FORWARD -p tcp --tcp-flags ALL NONE -j DROP
docker exec --privileged -t firewall1 iptables -A FORWARD -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP
docker exec --privileged -t firewall1 iptables -A FORWARD -p tcp --tcp-flags SYN,RST SYN,RST -j DROP
```

SECURITY TEST (2)

XMAS TREE ATTACK

- Invia un segmento TCP con i flag SYN, FIN, URG, PUSH alti

```
hping3 -d 120 --xmas -p 80 192.1.2.2
```

- Il risultato dei log evidenzia il successo delle regole inserite

```
Apr  6 09:24:34 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=32277 PROTO=TCP SPT=1377 DPT=80 SEQ=1549559875
Apr  6 09:24:35 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=2571 PROTO=TCP SPT=1378 DPT=80 SEQ=1446926347 A
Apr  6 09:24:36 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=63680 PROTO=TCP SPT=1379 DPT=80 SEQ=1016741717
Apr  6 09:24:37 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=60450 PROTO=TCP SPT=1380 DPT=80 SEQ=54359008 AC
Apr  6 09:24:38 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=47266 PROTO=TCP SPT=1381 DPT=80 SEQ=1161161160
Apr  6 09:24:39 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=39543 PROTO=TCP SPT=1382 DPT=80 SEQ=1698339583
Apr  6 09:24:40 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=9837 PROTO=TCP SPT=1383 DPT=80 SEQ=1487584536 A
Apr  6 09:24:41 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=4379 PROTO=TCP SPT=1384 DPT=80 SEQ=710394599 AC
Apr  6 09:24:42 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=31833 PROTO=TCP SPT=1385 DPT=80 SEQ=1790533112
Apr  6 09:24:43 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=56782 PROTO=TCP SPT=1386 DPT=80 SEQ=136256369 A
Apr  6 09:24:44 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=674 PROTO=TCP SPT=1387 DPT=80 SEQ=1031753968 AC
Apr  6 09:24:45 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=29567 PROTO=TCP SPT=1388 DPT=80 SEQ=625921798 A
Apr  6 09:24:46 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=31090 PROTO=TCP SPT=1389 DPT=80 SEQ=103078161 A
Apr  6 09:24:47 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=22409 PROTO=TCP SPT=1390 DPT=80 SEQ=602001873 A
Apr  6 09:24:48 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=45227 PROTO=TCP SPT=1391 DPT=80 SEQ=329039347 A
Apr  6 09:24:49 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=45530 PROTO=TCP SPT=1392 DPT=80 SEQ=1750627989
Apr  6 09:24:50 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=18810 PROTO=TCP SPT=1393 DPT=80 SEQ=265883916 A
Apr  6 09:24:51 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=4055 PROTO=TCP SPT=1394 DPT=80 SEQ=1075459961 A
Apr  6 09:24:52 0f9088e4e839 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.1.2 DST=192.1.2.2 LEN=160 TOS=00 PREC=0x00 TTL=63 ID=15179 PROTO=TCP SPT=1395 DPT=80 SEQ=684738708 A
```

SECURITY RULES (3)

IP SPOOFING

- È una tecnica di attacco che prevede l'utilizzo di un pacchetto IP nel quale viene falsificato l'indirizzo IP del mittente
- Introdotta per evitare che un cliente esterno possa fingersi un host della rete interna ed averne i suoi privilegi

```
docker exec --privileged -t firewall1 iptables -A FORWARD -s 192.1.3.0/24  -i eth0 -j DROP
```

SECURITY TEST (3)

IP SPOOFING

```
hping3 --rawip -d 120 --spoof 192.1.3.2 192.1.2.2
```

- Inoltro di pacchetti IP di dimensione 120 bytes
- Indirizzo IP forgiato ad hoc attraverso l'opzione –spoof

- Il risultato dei log evidenzia il successo delle regole inserite

```
Apr  6 14:50:48 bc79ac0d5099 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.3.2 DST=192.1.2.2
Apr  6 14:50:48 bc79ac0d5099 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.3.2 DST=192.1.2.2
Apr  6 14:50:50 bc79ac0d5099 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.3.2 DST=192.1.2.2
Apr  6 14:50:50 bc79ac0d5099 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.3.2 DST=192.1.2.2
Apr  6 14:50:52 bc79ac0d5099 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.3.2 DST=192.1.2.2
Apr  6 14:50:52 bc79ac0d5099 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.3.2 DST=192.1.2.2
Apr  6 14:50:54 bc79ac0d5099 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.3.2 DST=192.1.2.2
Apr  6 14:50:54 bc79ac0d5099 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.3.2 DST=192.1.2.2
Apr  6 14:50:56 bc79ac0d5099 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.3.2 DST=192.1.2.2
Apr  6 14:50:56 bc79ac0d5099 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=192.1.3.2 DST=192.1.2.2
```

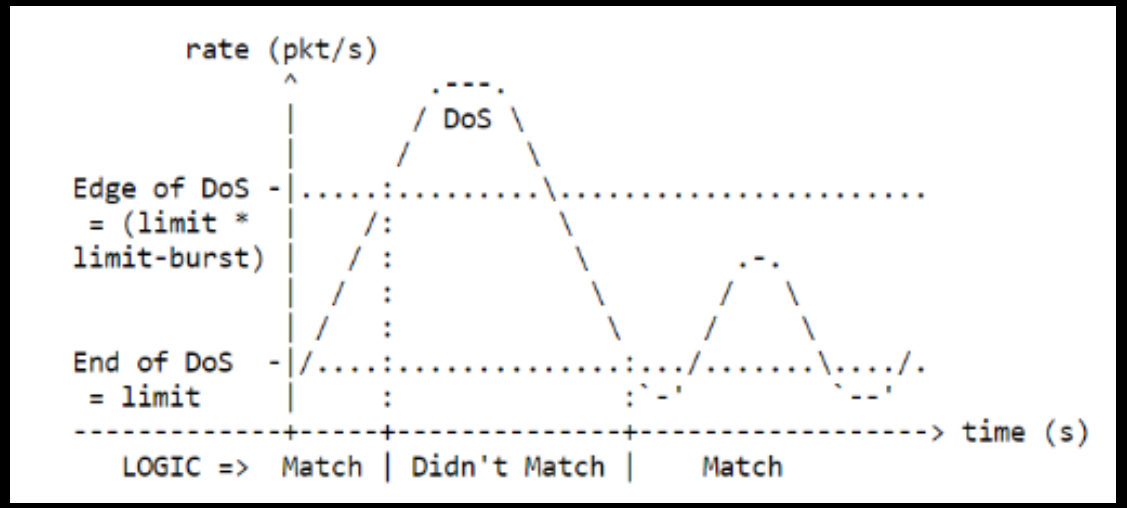
SECURITY RULES (4)

SYN FLOOD

- Attacco DoS basato sul protocollo TCP
- Attaccante manda una richiesta di connessione alla vittima (flag SYN alto), la vittima risponde, ma l'attaccante non replica mantenendo la connessione sempre attiva
- In questo caso si utilizza una catena creata ad hoc: SYN_FLOOD

```
docker exec --privileged -t firewall1 iptables -N SYN_FLOOD
docker exec --privileged -t firewall1 iptables -A FORWARD -p tcp --syn -j SYN_FLOOD
docker exec --privileged -t firewall1 iptables -A SYN_FLOOD -m limit --limit 1/s -j RETURN
docker exec --privileged -t firewall1 iptables -A SYN_FLOOD -j DROP
```

- L'estensione --limit nella catena limita il rate di pacchetti TCP in ingresso: non si può ricevere più di un pacchetto al secondo



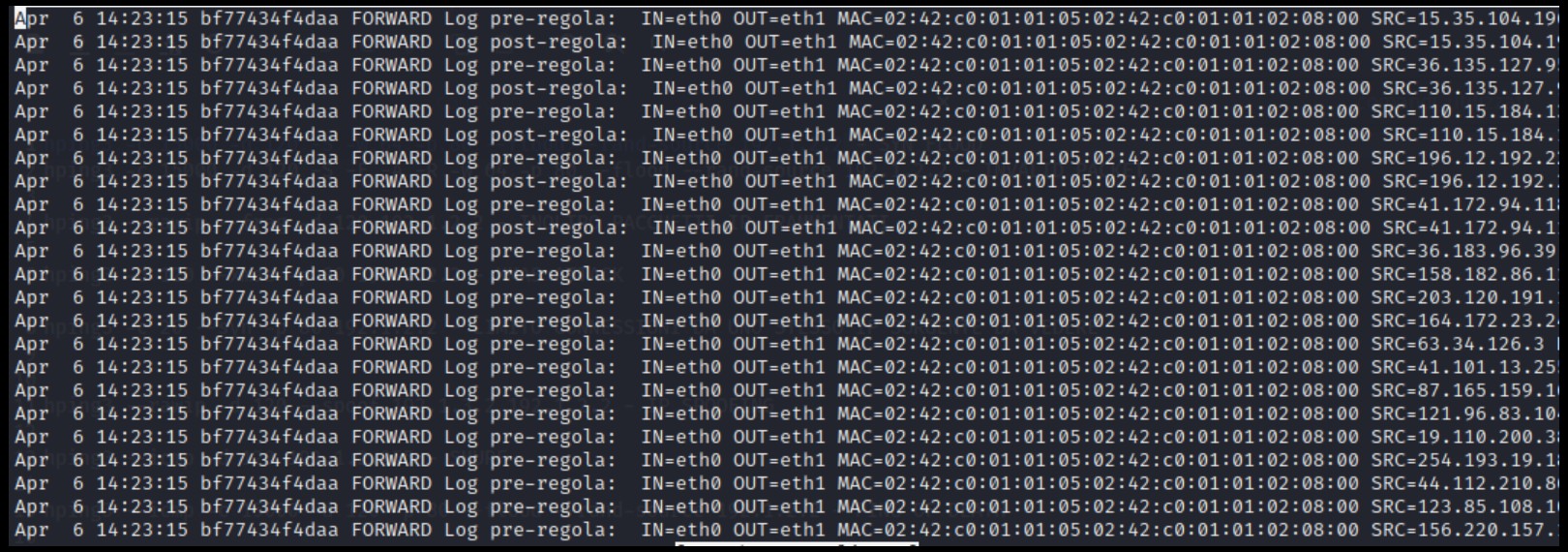
SECURITY TEST (4)

SYN FLOOD ATTACK

- Tramite il tag –flood il rate di pacchetti inoltrati è impostato al massimo rate che la macchina riesce a raggiungere
- --rand-source permette di inviare i pacchetti con diversi IP sorgente, mascherando il reale indirizzo IP dell'attaccante

```
hping3 -c 15000 -d 120 -S -w 64 -p 80 --flood --rand-source 192.1.2.2
```

- Il risultato dei log evidenzia il successo delle regole inserite



SECURITY RULES (5)

PING OF DEATH

- Attacco Dos in cui l'attaccante invia un pacchetto ICMP di grandi dimensioni verso un sistema vittima.
- Tale pacchetto viene frammentato e una volta a destinazione provocherà un buffer overflow sul sistema a causa del superamento della dimensione consentita, generando così un DoS.

```
docker exec --privileged -t firewall1 iptables -N PING_OF_DEATH
docker exec --privileged -t firewall1 iptables -A FORWARD -p icmp -j PING_OF_DEATH
docker exec --privileged -t firewall1 iptables -A PING_OF_DEATH -p
    icmp --icmp-type echo-request -m limit --limit 1/s -j RETURN
docker exec --privileged -t firewall1 iptables -A PING_OF_DEATH -p
    icmp --icmp-type echo-request -j DROP
```

SECURITY TEST (5)

PING OF DEATH

```
hping3 --icmp -c 15000 -d 120 -p 80 --flood --rand-source 192.1.2.2
```

- Il tag --icmp per l'inoltro della corretta tipologia di pacchetti
- Il risultato dei log evidenzia il successo delle regole inserite

```
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=72.217.244.130 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=13157 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=0 MA
Apr  6 09:40:21 2929ccf16d07 FORWARD Log post-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=72.217.244.130 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=13157 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=0 M
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=209.130.253.235 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=34993 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=256
Apr  6 09:40:21 2929ccf16d07 FORWARD Log post-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=209.130.253.235 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=34993 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=25
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=181.52.25.195 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=26276 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=512 M
Apr  6 09:40:21 2929ccf16d07 FORWARD Log post-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=181.52.25.195 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=26276 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=512 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=83.134.242.211 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=35191 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=768 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log post-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=83.134.242.211 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=35191 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=768
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=69.240.25.76 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=14120 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=1024 M
Apr  6 09:40:21 2929ccf16d07 FORWARD Log post-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=69.240.25.76 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=14120 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=1024 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=176.138.31.55 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=39268 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=1280 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=130.155.77.185 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=65341 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=1536 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=200.74.227.76 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=11792 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=2048 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=171.176.200.224 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=28053 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=230
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=120.146.201.76 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=36270 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=2816 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=252.25.110.11 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=31068 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=3072 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=89.79.54.139 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=60392 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=3328 M
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=117.64.109.147 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=34164 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=3584 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=5.242.55.78 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=40068 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=3840 MA
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=62.173.216.55 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=11195 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=4096 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=84.128.188.211 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=35432 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=4608 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=0.77.32.214 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=21668 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=4864 MA
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=109.41.145.130 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=11868 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=5120 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=156.235.232.27 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=62961 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=5376 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=122.31.203.16 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=1682 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=5632 M
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=186.78.75.213 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=5607 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=6144 M
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=168.66.219.35 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=23517 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=6656 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=133.142.96.169 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=9858 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=6912 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=15.90.198.232 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=4412 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=7168 M
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=164.219.18.126 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=20679 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=7424 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=242.217.148.78 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=32701 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=7680 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=155.198.64.195 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=17570 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=7936 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=79.117.158.54 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=55672 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=8192 
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=196.206.136.127 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=58977 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=844
Apr  6 09:40:21 2929ccf16d07 FORWARD Log pre-regola:  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=215.30.181.251 DST=192.1.2.2 LEN=148 TOS=00 PREC=0x00 TTL=63 ID=52761 PROTO=ICMP TYPE=8 CODE=0 ID=5888 SEQ=8704
```

SECURITY RULES (6)

UDP FLOOD

- Un UDP flood è un attacco DoS basato sul protocollo di trasporto UDP che consiste nell'inondare la vittima di datagrammi UDP.
- In questo caso si utilizza una catena creata ad hoc: UDP_FLOOD

```
docker exec --privileged -t firewall1 iptables -N UDP_FLOOD
docker exec --privileged -t firewall1 iptables -A FORWARD -p udp -j UDP_FLOOD
docker exec --privileged -t firewall1 iptables -A UDP_FLOOD -p udp -m limit --limit 1/s -j RETURN
docker exec --privileged -t firewall1 iptables -A UDP_FLOOD -j DROP
```

- Inoltre, è stato limitato il traffico UDP verso le altre destinazione che non siano il server DNS sulla porta 53

```
docker exec --privileged -t firewall1 iptables -t filter -A FORWARD -i eth0 -o eth1
    -p udp -d 192.1.2.3 --dport 53 -j ACCEPT
docker exec --privileged -t firewall1 iptables -t filter -A FORWARD -i eth1 -o eth0 -p udp -j ACCEPT
```

SECURITY TEST (6)

UDP FLOOD ATTACK

```
hping3 --udp -c 15000 -d 120 -p 53 --flood --rand-source 192.1.2.2
```

- Il risultato dei log evidenzia il successo delle regole inserite

```
Apr  6 14:58:34 38afb7a26df9 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=36.146.63.191 DST=192.1.2.3 LE
Apr  6 14:58:34 38afb7a26df9 FORWARD Log post-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=36.146.63.191 DST=192.1.2.3 
Apr  6 14:58:34 38afb7a26df9 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=241.219.32.161 DST=192.1.2.3 
Apr  6 14:58:34 38afb7a26df9 FORWARD Log post-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=241.219.32.161 DST=192.1.2.3
Apr  6 14:58:34 38afb7a26df9 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=160.223.205.34 DST=192.1.2.3 
Apr  6 14:58:34 38afb7a26df9 FORWARD Log post-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=160.223.205.34 DST=192.1.2.3
Apr  6 14:58:34 38afb7a26df9 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=39.182.66.4 DST=192.1.2.3 LEN
Apr  6 14:58:34 38afb7a26df9 FORWARD Log post-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=39.182.66.4 DST=192.1.2.3 LE
Apr  6 14:58:34 38afb7a26df9 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=123.194.201.81 DST=192.1.2.3 
Apr  6 14:58:34 38afb7a26df9 FORWARD Log post-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=123.194.201.81 DST=192.1.2.3
Apr  6 14:58:34 38afb7a26df9 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=99.212.36.203 DST=192.1.2.3 LE
Apr  6 14:58:34 38afb7a26df9 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=93.123.20.145 DST=192.1.2.3 LE
Apr  6 14:58:34 38afb7a26df9 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=82.81.182.107 DST=192.1.2.3 LE
Apr  6 14:58:34 38afb7a26df9 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=116.61.93.239 DST=192.1.2.3 LE
Apr  6 14:58:34 38afb7a26df9 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=125.30.172.179 DST=192.1.2.3 
Apr  6 14:58:34 38afb7a26df9 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=194.4.86.90 DST=192.1.2.3 LEN
Apr  6 14:58:34 38afb7a26df9 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=202.5.230.194 DST=192.1.2.3 LE
Apr  6 14:58:34 38afb7a26df9 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=190.54.210.91 DST=192.1.2.3 LE
Apr  6 14:58:34 38afb7a26df9 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=212.114.42.182 DST=192.1.2.3 
Apr  6 14:58:34 38afb7a26df9 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=113.163.16.123 DST=192.1.2.3 
Apr  6 14:58:34 38afb7a26df9 FORWARD Log pre-regola :  IN=eth0 OUT=eth1 MAC=02:42:c0:01:01:05:02:42:c0:01:01:02:08:00 SRC=154.210.30.202 DST=192.1.2.3 
```