

Corso di Reti di Calcolatori

A.A. 2022-23

Progetto su IMUNES







Omar Daniel, 01664A

Autore: Omar Daniel

Ultima modifica: 15 settembre 2023 – versione 2.0

Prima modifica: 26 agosto 2023

Indice

-  [Introduzione](#)
-  [Spiegazione del lavoro](#)
-  [Topologia](#)
-  [Funzionamento generale](#)
-  [Walkthrough](#)
-  [Approfondimenti teorici](#)

Introduzione

Il seguente progetto è stato volto con l'utilizzo dell'applicativo IMUNES, un software che ci permette di simulare il funzionamento di dispositivi hardware come switch e router a livello software. Ad esempio, per l'implementazione e la gestione di switch software-oriented utilizziamo OpenVSwitch (Software Defined Networking – SDN).

Il software è utilizzato attraverso una macchina virtuale Ubuntu 64bit e il progetto implementato tramite script bash per favorirne la riproducibilità.

All'interno della cartella del progetto troviamo inoltre un file di configurazione per il server DNS e uno script bash per la generazione di una password unix-like (/etc/shadow).

Spiegazione del lavoro

Il progetto scelto è intitolato:

“Design and deployment of a network consisting of:

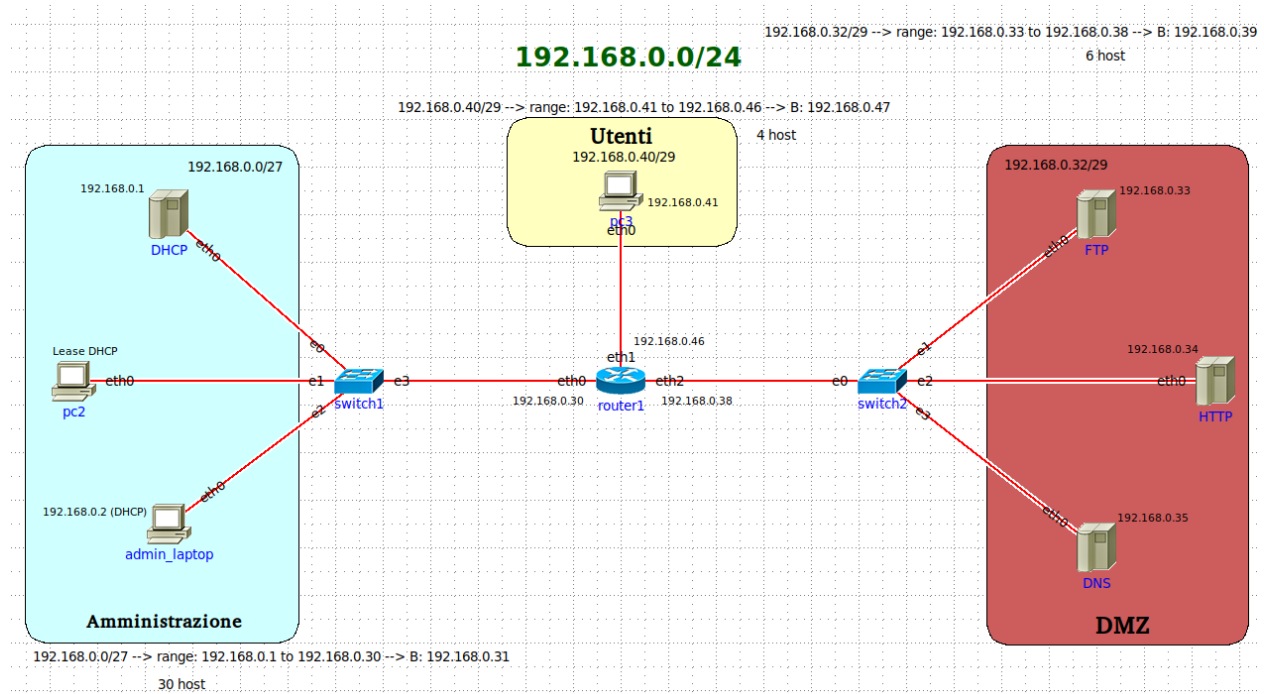
- *A DHCP server distributing IP addresses*
- *A DNS server translating domain names in IP addresses*
- *A choice of two: FTP server, HTTP server, SMTP server*
 - *An internal and an external user*
 - *MUNES”*

Come server sono stati scelti FTP e HTTP.

Durante l'esecuzione c'è un'evoluzione della situazione, in quanto inizialmente il pc3 è funzionante e riesce ad accedere a tutti i servizi, ma successivamente verrà disabilitato l'accesso al servizio FTP da quel pc a seguito di una violazione del sistema da parte di un utente malevolo.

Topologia

La topologia scelta per lo scenario è la seguente:



Come indirizzo di partenza è stato scelto l'indirizzo di classe C 192.168.0.0/24 e successivamente subnettato in 3 sottoreti tramite la tecnica VLSM (Variable Length Subnet Mask)

La topologia fisica vede l'utilizzo di due switch collegati mediante un router. Da un lato troviamo la subnet Amministrazione composta da admin_laptop, pc2 e il server DHCP che assegna gli IP in modo dinamico agli utenti della subnet. Subnet composta da 30 host e che quindi richiede subnet mask CIDR /27. La dimensione è stata scelta tenendo conto di un possibile incremento del numero di host.

Dall'altro lato vediamo la subnet DMZ (Demilitarized Zone) contenente i server HTTP, DNS e FTP. Subnet composta da 6 host e che quindi richiede subnet mask CIDR /29. La dimensione è stata scelta tenendo conto di un possibile incremento del numero di server.

In alto, invece, troviamo la subnet **Utenti** che contiene pc3. Anche in questo caso la subnet mask CIDR necessaria è /29 in quanto è stata scelta una dimensione di 4 host.

Subnet Name	Needed Size	Allocated Size	Address	Mask	Dec Mask	Assignable Range	Broadcast
Amministrazione	30	30	192.168.0.0	/27	255.255.255.224	192.168.0.1 - 192.168.0.30	192.168.0.31
DMZ	6	6	192.168.0.32	/29	255.255.255.248	192.168.0.33 - 192.168.0.38	192.168.0.39
Utenti	4	6	192.168.0.40	/29	255.255.255.248	192.168.0.41 - 192.168.0.46	192.168.0.47

Funzionamento generale

Dopo aver impostato i vari indirizzi **IP statici**, i server **DHCP**, **DNS**, **HTTP** ed **FTP**, le **rotte statiche** e aver fatto diversi **test** per verificarne il corretto funzionamento, effettuiamo delle **richieste HTTP** per interagire con il servizio web.

Successivamente proviamo ad effettuare le richieste al server web tramite il nome di dominio e quindi coinvolgendo anche il **server DNS**.

Poi creiamo un **utente FTP** e vengono utilizzate alcune funzioni del servizio come LS, GET, PUT e DEL.

Ad un certo punto simuliamo **l'accesso malevolo** al pc3 della subnet Utenti. Ipoteticamente l'attaccante è riuscito ad accedere al servizio FTP comportando alcuni danni; quindi la rete di Amministrazione decide di **bloccare** l'accesso FTP ai pc della subnet Utenti per effettuare delle operazioni di mitigation e remediation e per cercare di capire come l'attaccante sia riuscito ad ottenere l'accesso.

Walkthrough

Iniziamo con la creazione della topologia su IMUNES e tramite l'utilizzo degli strumenti grafici scriviamo alcune note importanti come IP, subnet, nomi... per maggiore chiarezza. Disabilitiamo l'autoassegnamento degli indirizzi IPv4 e IPv6

Successivamente decido un indirizzo IP di partenza per la rete (nel nostro caso 192.168.0.0/24) in base alle esigenze e mi occupo del subnetting. Utilizzo la tecnica VLSM per ottimizzare al meglio il numero di indirizzi IP utilizzati e diminuire il numero di indirizzi IP inutilizzati.

```
Major Network: 192.168.0.0/24
Available IP addresses in major network: 254
Number of IP addresses needed: 40
Available IP addresses in allocated subnets: 42
About 19% of available major network address space is used
About 95% of subnetted network address space is used
```

Per quanto riguarda lo script, si inizia assegnando gli indirizzi IP statici ai vari server della rete

```
#####
##### IMPOSTO INDIRIZZI IP STATICI #####
#####

do_cmd "sudo himage DHCP ip addr add 192.168.0.1/27 dev eth0" "Assegno
l'indirizzo 192.168.0.1 al server DHCP"
do_cmd "sudo himage router1 ip addr add 192.168.0.30/27 dev eth0" "Assegno
l'indirizzo 192.168.0.30 al router sull'interfaccia eth0"

do_cmd "sudo himage FTP ip addr add 192.168.0.33/29 dev eth0" "Assegno
l'indirizzo 192.168.0.33 al server FTP"
do_cmd "sudo himage HTTP ip addr add 192.168.0.34/29 dev eth0" "Assegno
l'indirizzo 192.168.0.34 al server HTTP"
do_cmd "sudo himage DNS ip addr add 192.168.0.35/29 dev eth0" "Assegno
l'indirizzo 192.168.0.35 al server DNS"
do_cmd "sudo himage router1 ip addr add 192.168.0.38/29 dev eth2" "Assegno
l'indirizzo 192.168.0.38 al router sull'interfaccia eth2"

do_cmd "sudo himage pc3 ip addr add 192.168.0.41/29 dev eth0" "Assegno
l'indirizzo 192.168.0.41 al pc3"
do_cmd "sudo himage router1 ip addr add 192.168.0.46/29 dev eth1" "Assegno
l'indirizzo 192.168.0.46 al router sull'interfaccia eth1"
```

Successivamente setto il **server DHCP**, scrivo i relativi file di configurazione e faccio partire il servizio (**daemon**) sia sul server (**DHCPD**) che sui vari client (**dhclient**).

Per i file di configurazione prima cancello il contenuto di **/etc/dhcp/dhcpd.conf** e poi lo riscrivo con i miei statement.

```
#####
#### IMPOSTO IL SERVER DHCP E FILE DI CONFIGURAZIONE ####
#####

do_cmd "sudo himage DHCP sh -c 'echo > /etc/dhcp/dhcpd.conf'" "Svuoto il file di
configurazione /etc/dhcp/dhcpd.conf"

do_cmd "sudo himage DHCP sh -c 'cat > /etc/dhcp/dhcpd.conf << EOF
option domain-name-servers 192.168.0.35;
option subnet-mask 255.255.255.224;
option routers 192.168.0.30;

subnet 192.168.0.0 netmask 255.255.255.224 {
    range 192.168.0.3 192.168.0.30;
    host admin_laptop {
        hardware ethernet 42:00:aa:00:00:02;
        fixed-address 192.168.0.2;
    }
}
EOF'" "Configuro il server DHCP"

do_cmd_pid "sudo xterm -e 'himage DHCP dhcpd -d'" "Avvio il server DHCP" pid1

do_cmd "sudo himage pc2 dhclient eth0" "Avvio il client DHCP su pc2"

do_cmd "sudo himage admin_laptop dhclient eth0" "Avvio il client DHCP su
admin_laptop"
```


Imposto le varie **rotte statiche** per permette alle diverse subnet di comunicare tra di loro tramite il router intermedio.

```
#####
#### IMPOSTO LE ROTTE VERSO LE ALTRE SUBNET ####
#####

do_cmd "sudo himage pc2 ip route add 192.168.0.40/29 via 192.168.0.30 dev eth0"
"Imposto la rotta da Amministrazione verso Utenti per pc2"
do_cmd "sudo himage pc2 ip route add 192.168.0.32/29 via 192.168.0.30 dev eth0"
"Imposto la rotta da Amministrazione verso DMZ per pc2"

do_cmd "sudo himage admin_laptop ip route add 192.168.0.40/29 via 192.168.0.30
dev eth0" "Imposto la rotta da Amministrazione verso Utenti per admin_laptop"
do_cmd "sudo himage admin_laptop ip route add 192.168.0.32/29 via 192.168.0.30
dev eth0" "Imposto la rotta da Amministrazione verso DMZ per admin_laptop"

do_cmd "sudo himage pc3 ip route add 192.168.0.0/27 via 192.168.0.46 dev eth0"
"Imposto la rotta da Utenti verso Amministrazione per pc3"
do_cmd "sudo himage pc3 ip route add 192.168.0.32/29 via 192.168.0.46 dev eth0"
"Imposto la rotta da Utenti verso DMZ per pc3"

do_cmd "sudo himage FTP ip route add 192.168.0.40/29 via 192.168.0.38 dev eth0"
"Imposto la rotta da DMZ verso Utenti per FTP"
do_cmd "sudo himage FTP ip route add 192.168.0.0/27 via 192.168.0.38 dev eth0"
"Imposto la rotta da DMZ verso Amministrazione per FTP"

do_cmd "sudo himage HTTP ip route add 192.168.0.40/29 via 192.168.0.38 dev eth0"
"Imposto la rotta da DMZ verso Utenti per HTTP"
do_cmd "sudo himage HTTP ip route add 192.168.0.0/27 via 192.168.0.38 dev eth0"
"Imposto la rotta da DMZ verso Amministrazione per HTTP"

do_cmd "sudo himage DNS ip route add 192.168.0.40/29 via 192.168.0.38 dev eth0"
"Imposto la rotta da DMZ verso Utenti per DNS"
do_cmd "sudo himage DNS ip route add 192.168.0.0/27 via 192.168.0.38 dev eth0"
"Imposto la rotta da DMZ verso Amministrazione per DNS"
```

Per assicurarmi del corretto funzionamento della rete effettuo dei **test** tramite il comando **ping** tra gli host delle diverse subnet.

```
#####
##### FACCIO DEI TEST CON PING #####
#####

do_cmd "sudo himage pc2 ping -c 1 192.168.0.33" "Amministrazione comunica con DMZ"

do_cmd "sudo himage admin_laptop ping -c 1 192.168.0.41" "Amministrazione comunica con Utenti"

do_cmd "sudo himage pc3 ping -c 1 192.168.0.34" "Utenti comunica con DMZ"
```

Ora mi occupo del **server HTTP**. Faccio partire il servizio **lighttpd** sull'apposito server, ne controllo lo stato e poi effettuo dei test da diversi host per controllare la raggiungibilità e la disponibilità della pagina web di prova offerta dal servizio lighttpd tramite il comando **CURL** (siccome non abbiamo ancora impostato il server DNS, utilizzo l'IP per effettuare la richiesta GET).

Siccome il server si trova all'interno della **DMZ**, imposto una **policy di deny-all** sulla chain INPUT e setto delle regole per permettere la comunicazione verso questo server solo sulla porta 80 e solo da sorgenti ben definite.

```
#####
##### IMPOSTO E TESTO SERVER HTTP #####
#####

do_cmd "sudo himage HTTP iptables -t filter -P INPUT DROP" "Modifico la policy per fare un deny-all"

do_cmd "sudo himage HTTP iptables -t filter -I INPUT -s 192.168.0.0/27 -p tcp --dport 80 -j ACCEPT" "Imposto la rules per accettare il traffico verso HTTP dalla subnet Amministrazione"

do_cmd "sudo himage HTTP iptables -t filter -I INPUT -s 192.168.0.40/29 -p tcp --dport 80 -j ACCEPT" "Imposto la rules per accettare il traffico verso HTTP dalla subnet Utenti"

do_cmd "sudo himage HTTP iptables -L -n" "Mostro le rules della tabella filter"

do_cmd "sudo himage HTTP service lighttpd start" "Starto il server HTTP"
```

```
do_cmd "sudo himage HTTP service lighttpd status" "Controllo lo stato del servizio"

do_cmd "sudo himage pc2 curl 192.168.0.34" "Amministrazione raggiunge il sito web (PER ORA NO DNS, QUINDI USO IP)"

do_cmd "sudo himage pc3 curl -I 192.168.0.34" "Utenti raggiunge il sito web (NO DNS)"
```

Per quanto riguarda il **server DNS**, creo il file di configurazione **named.conf.local** e il file di zona **db.progetto.com** (uno scritto direttamente tramite lo script, l'altro importato). Poi ne controllo la correttezza tramite i due comandi **named-checkconf** e **named-checkzone** e faccio partire il servizio (daemon **Bind**).

Inoltre devo impostare il server DNS per pc3 tramite il file **/etc/resolv.conf** (per la subnet amministrazione non c'è bisogno in quanto se ne occupa il server DHCP). Siccome il server si trova all'interno della **DMZ**, imposto una **policy di deny-all** sulla chain INPUT e setto delle regole per permettere la comunicazione verso questo server solo sulla porta 53 e solo da sorgenti ben definite. Inoltre, duplico le regole sia per UDP che per TCP in quanto se il pacchetto supera la dimensione di 512 byte, utilizzerà il protocollo TCP, altrimenti UDP (normalmente usato dagli utenti).

```
#####
##### IMPOSTO E TESTO SERVER DNS #####
#####Co#####

do_cmd "sudo himage DNS iptables -t filter -P INPUT DROP" "Modifico la policy per fare un deny-all"
do_cmd "sudo himage DNS iptables -t filter -I INPUT -s 192.168.0.0/27 -p tcp --dport 53 -j ACCEPT" "Imposto la rules per accettare il traffico verso DNS dalla subnet Amministrazione"
do_cmd "sudo himage DNS iptables -t filter -I INPUT -s 192.168.0.40/29 -p tcp --dport 53 -j ACCEPT" "Imposto la rules per accettare il traffico verso DNS dalla subnet Utenti"
do_cmd "sudo himage DNS iptables -t filter -I INPUT -s 192.168.0.0/27 -p udp --dport 53 -j ACCEPT" "Imposto la rules per accettare il traffico verso DNS dalla subnet Amministrazione"
do_cmd "sudo himage DNS iptables -t filter -I INPUT -s 192.168.0.40/29 -p udp --dport 53 -j ACCEPT" "Imposto la rules per accettare il traffico verso DNS dalla subnet Utenti"
do_cmd "sudo himage DNS iptables -L -n" "Mostro le rules della tabella filter"
```

```
do_cmd "sudo himage DNS sh -c 'cat > /etc/bind/named.conf.local << EOF
zone \"progetto.com\" {
    type master;
    file \"/etc/bind/db.progetto.com\";
};
EOF'" "Imposto il file /etc/bind/named.conf.local"

do_cmd "sudo hcp dns/db.progetto.com DNS:/etc/bind/db.progetto.com" "Importo il
file /etc/bind/db.progetto.com dalla cartella dns"

do_cmd "sudo himage DNS named-checkconf /etc/bind/named.conf.local" "Controllo i
file di configurazione named.conf.local"
do_cmd "sudo himage DNS named-checkzone progetto.com. /etc/bind/db.progetto.com"
"Controllo i file di configurazione db.progetto.com"

do_cmd "sudo himage DNS named" "Starto il servizio DNS"

do_cmd "sudo himage pc3 sh -c 'echo nameserver 192.168.0.35 > /etc/resolv.conf'"
"Specifico il server DNS per pc3" #per pc2 e admin non c'è bisogno che lo fa il
DHCP.
```

Contenuto del file **db.progetto.com** (file di zona) importato (i numeri a riga 3-4 sono rispettivamente Numero Seriale del file di zona, Refresh 12 ore, Retry 4 ore, Scadenza 4 giorni, TTL negative cache 2 ore):

```
$ORIGIN progetto.com.
$TTL 1h
progetto.com. IN SOA ns.progetto.com. admin.progetto.com. ( 1 43200 1440
345600 7200 )
progetto.com. IN NS ns.progetto.com.
ns.progetto.com. IN A 192.168.0.35
http.progetto.com. IN A 192.168.0.34
ftp.progetto.com. IN A 192.168.0.33
pc3.progetto.com. IN A 192.168.0.41
admin.progetto.com. IN A 192.168.0.2
dhcp.progetto.com. IN A 192.168.0.1
www.progetto.com. IN CNAME http.progetto.com.
```

In questo file andiamo ad indicare diversi **record** come NS, A e CNAME.

Ora posso testare il funzionamento del server DNS tramite i comandi dig, nslookup e provando a richiedere la pagina web tramite nome di dominio con curl

```
do_cmd "sudo himage pc2 curl -I www.progetto.com" "Testo il server DNS mandando una richiesta al server HTTP tramite il nome di dominio"

do_cmd "sudo himage pc3 nslookup ftp.progetto.com" "Testo il server DNS tramite nslookup"

do_cmd "sudo himage admin_laptop dig http.progetto.com" "Testo il server DNS tramite dig"
```

Utilizzo il daemon ftpd per settare il server FTP. Creo un utente Omar che andrò ad aggiungere al gruppo sudo così da evitare problemi di permessi su alcuni file. Per creare la password dell'utente Omar uso un semplice script bash che genera la password secondo lo standard (password + salt e poi hash). Per testare poi il funzionamento del servizio creo un file di prova sulla macchina pc2.

```
#####
#### IMPOSTO E TESTO IL SERVER FTP ####
#####

do_cmd "sudo himage FTP useradd -m -p saq7YK0/y2eaI Omar" "Creo l'utente Omar"
do_cmd "sudo himage FTP usermod -aG sudo Omar" "Inserisco l'utente Omar nel gruppo sudo"

do_cmd "sudo himage pc2 touch test.txt" "Creo file di prova da utilizzare con FTP"
do_cmd "sudo himage pc2 sh -c 'cat > test.txt << EOF
FILE DI TESTO DI PROVA
PER SERVIZIO FTP
PROGETTO RETI DI CALCOLATORI
EOF'" "Riempio il file di prova"

do_cmd "sudo xterm -e 'himage pc2 ftp 192.168.0.33'" "Avvio servizio FTP"
```

Script per la **generazione della password**:

```
password="progettoretil"
pass=$(perl -e 'print crypt($ARGV[0], "salt")' $password)
echo "$pass"
```

Come ultima cosa gestisco la **compromissione** dell'host pc3 tramite le regole di filtering **Netfilter IPTABLES**. Faccio vedere che inizialmente il servizio FTP è raggiungibile da pc3, poi imposto una **policy di tipo DROP all** sulla chain INPUT e permette il traffico solo dalla subnet Amministrazione sulla porta 21 (control connection) e 20 (data connection)

```
#####
##### IPTABLES E FILTERING #####
#####

do_cmd "sudo xterm -e 'himage pc3 ftp 192.168.0.33'" "Avvio servizio FTP su pc3 e
vedo che funziona correttamente"

do_cmd "Simuliamo la compromissione di un host, ad esempio pc3. L'amministrazione
insieme al SOC nota che sono state effettuate delle azioni sul server FTP, allora
bloccano l'accesso a tutti e lo permettono solo all'amministrazione per cercare
di risolvere il problema"

do_cmd "sudo himage FTP iptables -t filter -P INPUT DROP" "Imposto la policy su
deny-all"
do_cmd "sudo himage FTP iptables -t filter -I INPUT -s 192.168.0.0/27 -p tcp --
dport 21 -j ACCEPT" "Imposto la rules per accettare il traffico verso FTP solo
dalla subnet Amministrazione"
do_cmd "sudo himage FTP iptables -t filter -A INPUT -s 192.168.0.0/27 -p tcp --
dport 20 -j ACCEPT" "Imposto la rules per accettare il traffico verso FTP solo
dalla subnet Amministrazione"

do_cmd "sudo xterm -e 'himage pc3 ftp 192.168.0.33'" "Provo a connettermi al
server FTP dal pc compromesso e vedo che è bloccato"

do_cmd "sudo xterm -e 'himage admin_laptop ftp 192.168.0.33'" "Provo a
connettermi al server FTP da un pc dell'Amministrazione e vedo che funziona"

do_cmd "sudo himage FTP iptables -L -n" "Mostro le rules della tabella filter"
```

Approfondimenti teorici (SISTEMARE SEZIONE IPTABLES)

→ Netfilter e IPTABLES

Netfilter è il daemon Linux che permette l'intercettazione e la manipolazione di pacchetti. Funziona grazie al concetto di tabelle implementate a livello kernel (4 tabelle: filter, nat, mangle e raw).

Ogni tabella contiene delle catene (chain) che sono delle ACL (Access Control List) che a loro volta contengono delle regole composte da:

- 1) **Filtro** → proprietà che un pacchetto deve avere affinché la regola sia valida
- 2) **Target** → azione da compiere nel caso in cui il pacchetto corrisponda alle proprietà impostate nel filtro (match) (possono essere ACCEPT, DROP, QUEUE, RETURN, LOG...)

La tabella che serve a noi nel progetto è la tabella Filter, cioè la tabella che contiene regole di filtraggio dei pacchetti e permette quindi di bloccare o far passare determinati pacchetti.

Ha 3 chain di base:

- a) **INPUT** – tutti i pacchetti in arrivo destinati al sistema passano per questa catena
- b) **OUTPUT** – tutti i pacchetti generati dal sistema passano per questa catena
- c) **FORWARD** – tutti i pacchetti in arrivo (non generati dal sistema stesso) destinati ad un altro sistema passano per questa catena (es. tramite router)

Noi abbiamo utilizzato la chain **INPUT** in quanto abbiamo implementato le regole **IPTABLES** sui vari server della **DMZ**.

Per gestire la **DMZ** abbiamo impostato su tutti i server una **politica di DROP ALL** sulla chain **INPUT** così da bloccare qualsiasi tipo di pacchetto verso tutti i server. Successivamente abbiamo definito delle regole di **ACCEPT** solamente per certe subnet e su certe porte.