Indice

Introduzione **iii**

1 Panoramica sui problemi di sicurezza **1**
1.1 Il concetto di sicurezza 1
1.2 Il protocollo TCP/IP 2
1.3 Il firewall 3
1.3.1 I vantaggi 3
1.3.2 I punti deboli 4
1.3.3 Tipologie di firewall 4

2 IPTABLES: struttura e funzionalità principali **7**
2.1 Le tabelle 7
2.1.1 FILTER 8
2.1.2 NAT 8
2.1.3 MANGLE 8
2.2 Le catene e le regole 9
2.3 La macchina a stati 10
2.4 Le catene speciali 13
2.5 Le policy 14
2.6 Le funzionalità avanzate 14

3 Firewall-Config: struttura e utilizzo **17**
3.1 Scopo del programma 17
3.2 Tipologie di rete supportate 18
3.2.1 Workstation 18
3.2.2 Rete LAN 18
3.2.3 Rete DMZ 19
3.2.4 Reti LAN e DMZ 20
3.2.5 Topologie complesse 20
3.3 Struttura del software 21
3.4 Analisi dei requisiti 22

3.5 Opzioni attivabili da linea di comando 23
3.6 Utilizzo del programma 24
3.6.1 *Device Management* 24
3.6.2 Definizione dei servizi e delle reti 25
3.6.3 *Services Definitions* 25
3.6.4 *Networks Definitions* 25
3.6.5 *Interfaces Configuration* 25
3.6.6 *NAT Rules* 27
3.6.7 *Public IP Mapping* 27
3.6.8 *Enable Services* 28
3.6.9 Le Interconnessioni 29
3.6.10 *Interconnections Configuration* 30
3.6.11 *Firewall Options* 30
3.6.12 *Blacklist* 33
3.6.13 *Show Actual Configuration* 33
3.6.14 *Save Configuration* 33
3.6.15 *Apply Saved Configuration* 34

4 Firewall-Config: Implementazione **35**
4.1 Strutture dati 36
4.2 Il file di configurazione 40
4.2.1 *programs* 40
4.2.2 *create_interface* 41
4.2.3 *interface* 41
4.2.4 *interconnection* 42
4.2.5 *definitions* 42
4.2.6 *services* 43
4.2.7 *blacklist* 43
4.2.8 *options* 43
4.3 Lo script di inizializzazione 44

Conclusioni **46**

Appendice A: esempio di file di configurazione **47**

Appendice B: esempio di script di inizializzazione **51**

Appendice C: alcune schermate **62**

Bibliografia **63**

Introduzione

Il mondo attuale delle reti e dei computer può presentare diverse minacce per quanto riguarda la sicurezza. Un utente irresponsabile è in grado di provocare ad un sistema non adeguatamente protetto danni enormi e l'interruzione dei servizi offerti.

La sicurezza riguardante le reti informatiche sta necessariamente assumendo un ruolo ed un'importanza predominante dovuta all'aumento vertiginoso degli utenti che hanno a disposizione un collegamento ad Internet. Per proteggere una qualsiasi rete accessibile dall'esterno, un amministratore di rete dovrà necessariamente pianificare adeguatamente la struttura da utilizzare e adottare tutte le misure di sicurezza atte a garantire l'integrità dei dati presenti sulle proprie macchine e l'accessibilità ai servizi offerti.

Molti di questi obiettivi possono essere raggiunti utilizzando un sistema di firewalling che realizza due funzioni: una implementa una politica ben precisa nei confronti delle diverse connessioni tra la propria rete e quella esterna, l'altra memorizza tutte le informazioni di interesse attraverso i file di log.

Uno dei sistemi software più utilizzati per la realizzazione di un firewall, è il programma IPTABLES, diffuso in ambiente GNU/Linux, che possiede, tra le tante altre caratteristiche, anche un sistema di filtraggio dei pacchetti (packet filtering) molto efficace.

Dato che la configurazione manuale di tale software spesso risulta essere estremamente laboriosa soprattutto per sistemi complessi, si è voluto realizzare uno strumento console-based in grado di semplificarla attraverso un'intuitiva interfaccia utente.

L'utilizzatore che fruirà di **Firewall-Config** non dovrà infatti essere necessariamente a conoscenza di tutti gli strumenti avanzati messi a disposizione da IPTABLES, ma dovrà solamente conoscere la tipologia strutturale della propria rete per essere poi guidato, attraverso dei semplici menu, alla configurazione di strutture anche complesse. Infatti, la novità introdotta da

questo programma, consiste proprio nel porsi su di un livello più alto rispetto ai molti script reperibili in rete su tale argomento, permettendo quindi all'amministratore una configurazione in pochi e semplici passi.

Il programma è stato scritto in Perl e, per la realizzazione dell'interfaccia utente, si è utilizzato il programma *dialog* che implementa molte delle funzionalità messe a disposizione dalle librerie *ncurses*.

La struttura di questo lavoro di tesi è la seguente: nel primo capitolo verrà presentata una panoramica sui problemi di sicurezza di rete, sul protocollo utilizzato per le comunicazioni (TCP/IP) e sui principali meccanismi di firewalling comunemente utilizzati. Il secondo capitolo verte sulla struttura e sull'utilizzo software IPTABLES. Infine, gli utilimi due capitoli illustrano **Firewall-Config**, la configurazione, l'utilizzo e i dettagli implementativi.

Capitolo 1

Panoramica sui problemi di sicurezza

1.1 Il concetto di sicurezza

All'interno di un mondo quale quello della comunicazione digitale in continua evoluzione e che presenta costi sempre più bassi di interconnessione, il concetto di sicurezza sta assumendo un ruolo sempre più importante, ed è diventato un requisito fondamentale per ogni software, sistema, rete semplice o complessa.

La sicurezza di rete sta assumendo giorno dopo giorno un ruolo predominante all'interno di tali problematiche. Difatti, ogni comunicazione che attraversa Internet o in generale una qualsiasi rete geografica, può toccare diversi nodi (*host*), dando quindi ad altri utenti l'opportunità di tracciare, intercettare, modificare i dati in transito. Un'altra problematica fondamentale consiste nell'evitare accessi non autorizzati alla propria rete da parte di malintenzionati in grado di appropriarsi, modificare o eliminare dati sensibili.

Un ipotetico attaccante potrebbe compromettere la sicurezza di una rete utilizzando diverse metodologie oramai note. Innanzitutto protrebbe, ad esempio, prendere il controllo di un host sfruttando un bug di qualche servizio che gira su tale macchina; successivamente, potrebbe esplorare il traffico della rete attraverso la tecnica del *packet sniffing* (cercando cioè di intercettare il più alto numero possibile di pacchetti che transitano attraverso il mezzo fisico) e successivamente utilizzare le macchine compromesse per attacchi di tipo DoS o DDoS in modo tale da occupare tutte le risorse messe a disposizione dai servizi della macchina vittima affinchè questa non sia più in grado di rispondere alle richieste legittime.

Per difendere la propria rete da tali attacchi e da molti altri è necessaria la conoscenza del protocollo che viene comunemente utilizzato per veicolare i pacchetti su Internet, il TCP/IP.

1.2 Il protocollo TCP/IP

Una rete può essere idealmente definita come una maglia di collegamenti. Ogni nodo di questa struttura è generalmente un elaboratore. I diversi host sono in grado di comunicare attraverso dei pacchetti che rappresentano il mezzo con cui i dati possono transitare attraverso la rete. In ogni pacchetto è specificato il mittente e il destinatario dello stesso e la sua struttura dipende dalla rete fisica utilizzata.

I pacchetti vengono inviati e ricevuti in base a delle regole definite da un protocollo di comunicazione. Principalmente esistono due tipologie di protocolli a secondo se esso sia in grado di assicurare una connessione virtuale mettendo a disposizione ad esempio strumenti quali conferma dell'invio di un messaggio, ritrasmissione in caso di errore, controllo di flusso e ricomposizione dell'ordine dei pacchetti o meno.

Il nome TCP/IP rappresenta un insieme di protocolli di comunicazione basati su IP (*Internet Protocol*), un protocollo che si colloca all'interno del terzo livello ISO/OSI, il Network Level.

Il TCP (*Trasmission Control Protocol*), l'UDP (*User Datagram Protocol*) e l'ICMP (*Internet Control Message Protocol*) sono tutti protocolli di livello superiore che utilizzano l'IP incapsulando i propri messaggi all'interno dei pacchetti di tale protocollo. A loro volta, altri protocolli quali l'SMTP, l'FTP, l'HTTP utilizzeranno il TCP, l'UDP, l'ICMP, ponendosi quini ad un livello superiore rispetto a questi, per la precisione a livello applicazione (livello 7 ISO/OSI corrispondente al livello 4 della pila Internet).

Il datagramma IP è costituito da un'intestazione (header del pacchetto) e dai dati veri e propri nei quali possono anche essere incapsulate le informazioni di protocolli di livello superiore.

Tipicamente nell'header vengono specificate diverse informazioni che verranno utilizzate da tutti i nodi della rete coinvolti nel transito del pacchetto per effetturare un corretto controllo di flusso, indirizzamento e controllo di integrità dei dati quali la lunghezza del datagramma, la versione del protocollo utilizzato, il TTL (*Time to Live*), alcuni bit di checksum, gli indirizzi di sorgente e destinazione. Tali indirizzi sono composti da una sequenza di 32 bit suddivisi convenzionalmente in quattro gruppi da 8.

Mentre protocolli quali l'UDP e l'ICMP si basano su un servizio di distribuzione dei pacchetti privo di connessione e non affidabile, il TCP fornisce una serie di meccanismi in grado di garantire queste caratteristiche creando una sorta di connessione virtuale point-to-point tra i due elaboratori coinvolti nella comunicazione. Tutto ciò viene realizzato secondo un meccanismo chiamato *positive acknowledgemente with retrasmission* e da un *three-way handshake* iniziale. Inoltre, sia in TCP che in UDP, è rilevante il concetto di *port*, utilizzata per distinguere le diverse connessioni tra due host. La comunicazione tra l'implementazione del protocollo sul sistema operativo e l'applicativo che si serve di tali procedure, avverrà proprio attraverso questo meccanismo.

1.3 Il firewall

Un firewall è un sistema o un gruppo di sistemi che realizza una metodologia di controllo degli accessi tra due o più reti.

Può essere schematizzato attraverso una coppia di meccanismi: uno che realizza il blocco del traffico, l'altro che lo accetta. Alcuni firewall ripongono maggiore enfasi sul primo aspetto, altri sul secondo.

La prerogativa fondamentale consiste nel decidere la politica di sicurezza da adottare prima della realizzazione fisica della rete.
È infatti di fondamentale importanza che l'amministratore sappia con assoluta precisione quali e quanti servizi offrire al mondo esterno attraverso la propria rete, quali privilegi concedere agli utilizzatori dei terminali interni e come strutturare fisicamente la rete stessa.

Spesso un sistema di protezione quale un firewall risulta complesso da configurare e richiede delle conoscenze avanzate che riguardano la struttura intrinseca e il funzionamento della rete, la conoscenza dei protocolli, i meccanismi di comunicazione tra le diverse tipologie di reti.

1.3.1 I vantaggi

Secondo un principio generale, un firewall viene realizzato sia per concedere alle postazioni interne un numero limitato di operazioni (sarà l'amministratore a dover valutare la fiducia che ripone nei proprio utenti), sia, soprattutto, per proteggere la rete interna da connessioni non autorizzate provenienti dal mondo esterno.

Un altro elemento di forza viene realizzato progettando una struttura di rete che porti il firewall ad essere l'unico punto di contatto tra la rete locale privata e la rete esterna. In tal modo sarà possibile mettere a disposizione un importante meccanismo di logging degli accessi e del transito delle informazioni oltre a permettere un effettivo controllo del traffico nelle due direzioni.

Un firewall mette inoltre a disposizione dell'amministratore di rete dei dati molto importanti quali il traffico generato dalla rete stessa e l'elenco dei tentativi di intrusione dall'esterno.

1.3.2 I punti deboli

Un sistema illustrato in questi termini, presenta però alcuni punti deboli da non sottovalutare: infatti, un firewall non è ovviamente in grado di proteggere da attacchi che non transitano attraverso di esso. Per esempio, se venisse istaurato un collegamento di tipo dial-up da un terminale presente nella rete interna verso l'esterno, il firewall non sarà più in grado di attuare le misure di sicurezza per le quali è stato creato dato che la connessione generata dall'utente non trasiterà più attraverso di esso e darà la possibilità ad un ipotetico intruso di utilizzarla per prendere il controllo della rete interna.

Un sistema di firewalling, inoltre, non è in grado di proteggere la rete interna da attacchi generati dalla rete stessa. Infatti il traffico rimarrà contenuto all'interno senza mia oltrepassarlo e pertanto escludendolo.

1.3.3 Tipologie di firewall

La funzione principale che viene svolta da un generico firewall è il filtraggio sui singoli pacchetti IP che transitano sulla rete, quindi al livello 3 della piramide ISO/OSI.

Il meccanismo di firewalling può però essere diviso in tre grandi categorie:

- Packet Filtering: lavorano sui singoli pacchetti, a livello di rete (Network Level, livello 2 della pila Internet). In tali sistemi i dati possono lasciare la rete interna solo se le regole lo permettono esplicitamente. Non appena un pacchetto giunge nel firewall, esso viene filtrato in base a diversi criteri quali, per esempio, l'indirizzo IP sorgente o di destinazione, le informazioni relative al numero di porta, il protocollo utilizzato (TCP, UDP, ICMP).

Questo genere di sistemi possono anche venire utilizzati come dei router. Infatti, una volta effettuato il filtraggio, è necessario decidere verso quale interfaccia di rete inoltrare il pacchetto.

- Stateful Filtering: lavorano a livello di connessione. Non verranno infatti considerati i singoli pacchetti ma l'intero flusso di dati che forma ogni singola connessione. Operano a livello tre della pila Internet).

- Application Layer: sono utilizzati principalmente per controllare o monitorare il traffico generato da una LAN verso l'interfaccia esterna e lavorano a livello 4 della pila Internet. Alcuni proxy sono forniti di una cache nella quale vengono memorizzati i dati una volta scaricati dalla rete. Tutto ciò permette un risparmio di banda dato che se un utente richiedesse dei dati che si trovano già nella cache del proxy, questi ultimi gli verrebbero forniti istantaneamente evitando una nuova connessione con l'esterno.

Capitolo 2

IPTABLES: struttura e funzionalità principali

IPTABLES è un pacchetto open source operante in ambiente GNU/Linux che utilizza gli strumenti messi a disposizione dai kernel della serie 2.4.

Pertanto, per utilizzare IPTABLES, sarà necessario configurare il proprio kernel affinchè supporti il firewalling in kernel-space e tutti i moduli necessari per implementarlo. Il programma IPTABLES rappresenta, infatti, solo un' applicazione in spazio utente.

Quando un pacchetto IP entra nel firewall, viene passato al corrispondente driver all'interno del kernel. Quindi il pacchetto attraverserà un serie di stati prima di essere inviato ad un'applicazione locale o inoltrato attraverso un'altra interfaccia di rete.

IPTABLES è strutturato internamente attraverso delle catene (*chains*). Un pacchetto, una volta immesso in una di queste, può essere bloccato o accettato, a seconda delle regole impostate dall'utente. Un altro concetto importante è quello di tabella (*table*). Non appena IPTABLES viene caricato all'interno del kernel, verranno create tre tabelle e diverse catene già preimpostate.

2.1 Le tabelle

Innanzi tutto analizziamo in dettaglio le tabelle che vengono create:

2.1.1 FILTER

Si tratta della tabella più importante e viene utilzzata per effettuare il filtraggio vero e proprio sui pacchetti che transitano attraverso le interfacce del firewall (*packet filtering*).

2.1.2 NAT

Questa tabella deve essere utilizzata per eseguire NAT (*Network Address Translation*, quindi per tradurre l'indirizzo sorgente o destinazione presente nell'header dei pacchetti IP). IPTABLES implementa due tipi diversi di NAT:

1. SNAT (*Source Network Address Translation*): viene utilizzato per modificare l'indirizzo IP sorgente del pacchetto. Per esempio può essere utilizzato per permettere ad una rete locale con indirizzi IP privati di accedere ad Internet. Ogni volta che viene richiesta una connessione verso l'esterno, il firewall modificherà l'indirizzo mittente inserendo il proprio IP al posto di quello privato della postazione che effettua la richiesta. Ovviamente verrà eseguita l'operazione inversa una volta ritornata la risposta dal server contattato. Il firewall sostituirà l'indirizzo privato originario nel campo di destinazione del pacchetto.

2. DNAT (*Source Network Address Translation*): viene utilizzato nei casi in cui si hanno a disposizione degli indirizzi IP pubblici e si vuole reindirizzare la connessione verso un'altra macchina che si trova per esempio all'interno di una DMZ (*Demilitarized Zone*). Ciò è di fondamentale importanza: infatti risulterebbe impossibile, altrimenti, connettersi dall'esterno verso un indirizzo della rete privata celata dietro il firewall.

2.1.3 MANGLE

Impostando oppurtune regole in questa tabella sarà possibile modificare internamente il pacchetto che vi transita. Risultano possibili infatti modifiche ai valori TOS (*Type of Service*) e TTL (*Time to Live*) dell'header. Si può inoltre imprimere una marcatura al pacchetto per destinarlo ad un trattamento successivo attraverso appositi programmi esterni. Ad esempio, in alcune situazioni, potrebbe essere conveniente stabilire una diversa politica di routing basata sulla marcatura dei pacchetti o implementare un particolare sistema di QOS (*Quality of Services*). Tutto ciò è possibile utilizzando appositi strumenti messi a disposizione, per esempio, dal pacchetto *iproute2*.

2.2 Le catene e le regole

La configurazione di un firewall avviene attraverso la dichiarazione di regole (*rules*). Ogni regola dovrà essere inserita in un contesto più ampio, ossia all'interno di una catena (*chain*).

Una regola può essere definita come un'indicazione alla quale il firewall dovrà attenersi per decidere se accettare o rifiutare i pacchetti appartenenti alle differenti connessioni. In ultima analisi, consiste nell'intraprendere una particolare azione nel caso in cui si verifichino alcune precondizioni. Per ogni pacchetto e per ogni regola, il kernel, verificherà che tutte le condizioni siano rispettate (*match*) e, in tal caso, inoltrerà il pacchetto verso la catena specificata (operazione di jump).

Le catene principali, ordinate a partire dall'arrivo del pacchetto fino alla sua successiva ripartenza attraverso un'altra interfaccia di rete, sono:

PREROUTING: ogni pacchetto che giunge al firewall proveniente da qualsiasi interfaccia, perviene in questa catena, a monte della decisione di routing. Viene quindi principalmente utilizzata per il DNAT. Infatti, la modifica dell'indirizzo di destinazione, influenzerà direttamente la decisione di routing da prendere per quanto riguarda il pacchetto.

INPUT: ogni pacchetto destinato direttamente al firewall entra in questa catena. In tal modo sarà possibile definire a quali servizi presenti sulla macchina permettere l'accesso dalle diverse interfacce.

OUTPUT: ogni pacchetto che viene generato direttamente dal firewall transita da questa catena.

FORWARD: ogni pacchetto che viene generato da un'interfaccia di rete destinato ad un'altra entra in questa catena. Sarà quindi possibile gestire, a seconda delle esigenze della propria rete, le diverse interconnessioni permettendo, ad esempio, alla LAN interna di accedere ad Internet ma impedendo l'accesso dall'esterno alla propria rete privata.

POSTROUTING: ogni pacchetto perviene in questa catena a valle della decisione di routing, appena prima di lasciare il firewall. Viene principalmente utilizzata per il SNAT. Infatti, la modifica dell'indirizzo sorgente, appena prima che il pacchetto lasci il firewall, permette la sostituzione dell'IP di rete privata, quindi non indirizzabile da Internet, con un altro indirizzo pubblico (solitamente quello del firewall stesso).

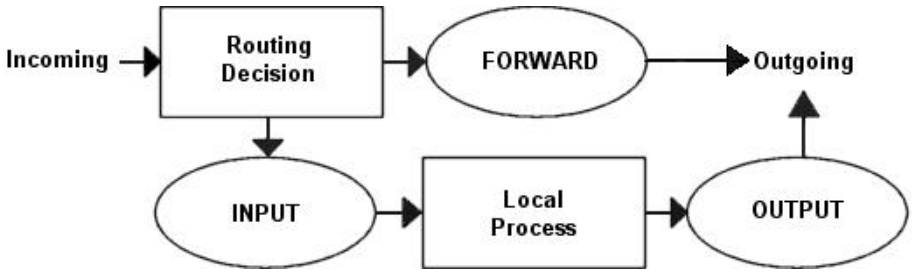

Figura 2.1: Le catene della tabella FILTER di IPTABLES

Ogni regola permette di ricercare dei riscontri su un'ampia gamma di parametri; tra i più comuni ed utilizzati si devono citare la verifica del protocollo, l'indirizzo di sorgente e di destinazione, le porte coinvolte nella connessione. Tutti questi elementi possono essere utilizzati per effettuare un corretto filtraggio e per prendere una decisione per quanto riguarda la sorte del pacchetto.

2.3 La macchina a stati

Un concetto molto importante e di estrema utilità che viene implementato da IPTABLES è quello di macchina a stati. Si tratta di una macchina virtuale che realizza un sistema di tracciamento delle connessioni (*connection tracking*). Sono previsti quattro stati:

NEW La macchina stabilisce che una connessione si trova in questo stato nel caso in cui un pacchetto fosse il primo relativo ad una particolare connessione. Per esempio, nel caso del protocollo TCP, la presenza del bit SYN indica escplicitamente l'inizializzazione di una nuova connessione.

ESTABLISHED La macchina entra in questo stato quando viene rilevato del traffico in entrambe le direzioni. Nel caso del protocollo TCP, una connessione viene definita ESTABLISHED non appena viene riscontrato un pacchetto di ritorno con i bit SYN e ACK attivi (termine del *three-way handshake*); nel caso di UDP, invece, quando viene ricevuto un pacchetto di risposta ad uno precedentemente inviato, mentre infine nel caso di ICMP si considera, ad esempio, il sopraggiungere di una risposta di tipo *echo-reply* (pong) in seguito ad una *echo-request* (ping).

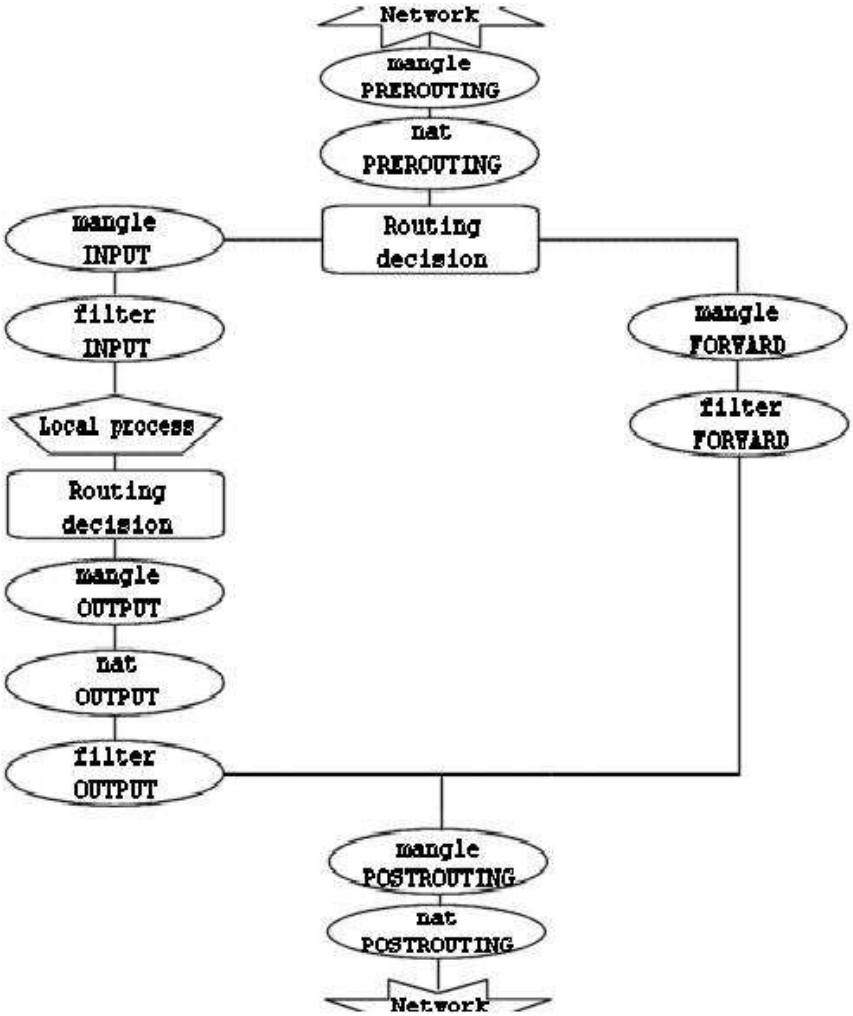

Figura 2.2: Schema di attraversamento delle catene di IPTABLES

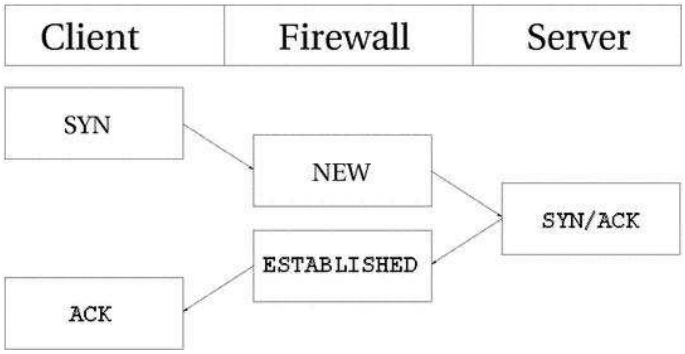

Figura 2.3: Inizializzazione di una connessione TCP vista dalla macchina a stati

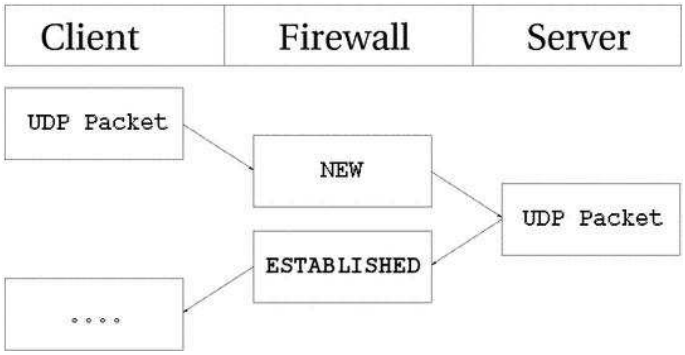

Figura 2.4: Una connessione UDP vista dalla macchina a stati

RELATED Una connessione viene considerata RELATED quando vi è una relazione con un'altra che si trova già nello stato ESTABLISHED. Ad esempio, nel caso del protocollo FTP, una connessione che viene inizializzata dalla porta 20 (ftp-data) è in relazione ad un'altra già attiva sulla porta 21 (ftp-command). Nel caso del protocollo ICMP, una connessione viene considerata RELATED se il pacchetto porta un messaggio di tipo *host-unreachable* o *network-unreachable* riportando quindi un errore di una connessione TCP o UDP già instaurata.

INVALID Se il pacchetto non può essere identificato diversamente non trovandosi in alcun altro stato viene marcato come INVALID.

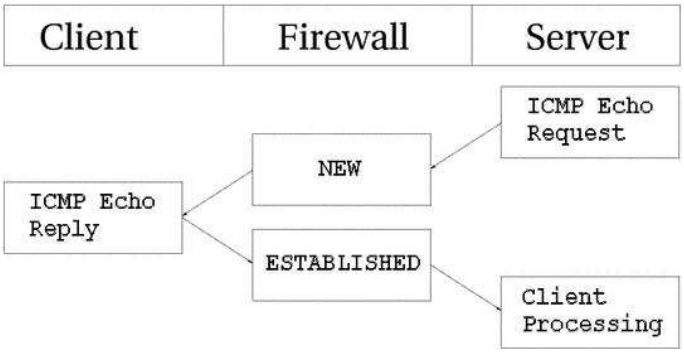

Figura 2.5: Comportamento della macchina a stati con un pacchetto ICMP di tipo echo-request (ping)

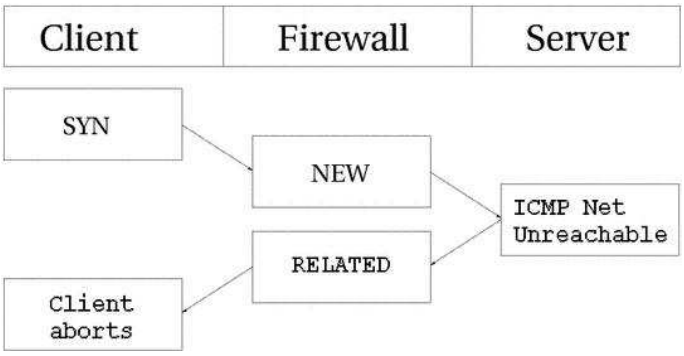

Figura 2.6: Comportamento della macchina a stati con un messaggio di errore veicolato da ICMP

2.4 Le catene speciali

Quando avviene il match tra una regola e un pacchetto, l'utente che configura IPTABLES è tenuto a specificare che sorte fare seguire al pacchetto stesso attraverso un'operazione di jump.

Un pacchetto può essere quindi inoltrato su un'altra catena, accettato o eliminato. Vengono preconfigurate all'avvio le seguenti catene:

ACCEPT Il pacchetto smette di attraversare la catena e le altre presenti nella tabella corrente e viene direttamente accettato dal sistema e inoltrato all'applicazione interessata ad esso.

DROP Il pacchetto viene rifiutato dal sistema.

LOG Alcune informazioni sul pacchetto vengono memorizzate nei log del firewall attraverso *syslog*. Il pacchetto continuerà quindi il suo percorso all'interno della catena.

DNAT Quando un pacchetto viene inoltrato nella catena DNAT, è possibile specificare a quale indirizzo IP inoltrare le connessioni (IPTABLES sostituirà quindi il campo destinazione del pacchetto con quello indicato dell'utente).

SNAT Quando un pacchetto viene inviato nella catena SNAT, è possibile specificare quale indirizzo IP utilizzare come sorgente del pacchetto. In questo modo sarà possile mascherare la rete interna privata servendosi di un indirizzo IP pubblico.

MASQUERADE Si tratta di un particolare tipo di SNAT nel quale l'indirizzo pubblico da impostare come mittente del pacchetto verra' individuato automaticamente dal sistema.

2.5 Le policy

Una volta che un pacchetto finisce di attraversare una catena senza che sia stata intrapresa nel frattempo alcuna azione, ritorna nella catena di partenza, al punto successivo in cui era stato dirottato. La sorte del pacchetto, una volta terminato l'attraverso delle catene INPUT, OUTPUT o FORWARD, è condizionato dalla policy della catena stessa impostata dall'utente. A questo punto infatti potrà solamente essere accettato oppure rifiutato.

2.6 Le funzionalità avanzate

IPTABLES mette a disposizione altri utili strumenti per un filtraggio avanzato. Per citare solo i pricipali:

- Attraverso il *limit match extension* è possibile imporre alcuni generici limiti alle operazioni di *jump*. Per esempio se ne può trarre vantaggio

limitando il logging di una particolare regola nel caso in cui quest'ultima risultasse essere troppo frequente e quindi renderebbe il file di log di troppo voluminoso e quindi di difficile interpretazione. In generale però il modulo può essere utilizzato per tutte le tipologie di regole. Sarà quindi anche possibile accettare solo un certo numero di connessioni all'interno di un determinato intervallo di tempo.

- È possibile creare dei filtri che si basano sull'indirizzo MAC (indirizzo fisico della scheda di rete).

- È possibile creare delle regole che tengano conto di connessioni a porte diverse, non necessariamento su di un intervallo numerico contiguo. Inoltre, il fatto di poter specificare degli intervalli di porte e di indirizzi IP differenti, anche nel caso di DNAT, permette di effettuare un semplice ma efficace *load balancing* tra più server. Ovviamente, per effettuare una ripartizione del carico più efficace, si rende necessario l'utilizzo di strumenti più avanzati di routing inclusi ad esempio nel pacchetto *iproute2*.

- È possibile creare delle regole sulla base dell'utente o del processo che ha generato il pacchetto. I dati relativi al proprietario vengono identificati attraverso UID, GID e PID.

- È possibile creare dei filtri che si basano sul valore del campo TOS (*Type of Service*) dell'header del pacchetto IP. Questo campo può permettere, tra le altre cose, di gestire il controllo del flusso dati. Infatti potrebbe capitare che un protocollo richieda alla rete che il pacchetto venga consegnato garantendo il minimo ritardo o il minimo costo oppure ancora la massima affidabilità. Raramente però queste modalità vengono gestite correttamente dai diversi router.

- È possibile rifiutare un pacchetto inoltrandolo sulla catena REJECT al posto della classica DROP. In questo modo sarà possibile specificare la modalità con cui si deve rispondere all'host che ha effettuato la richiesta. Un esempio classico consiste nell'inviare un pacchetto TCP di tipo *tcp-reset* (quindi con il bit RST attivo) qualora si voglia chiudere in modo pulito un tentativo di connessione (in caso contrario, l'host che ha tentato di effettuarla, non ricevendo alcuna risposta, rimarrà in attesa fino al sopraggiungere del timeout).

Il programma oggetto di questa ricerca non utilizzerà tutte le funzionalità messe a disposizione da IPTABLES, ma si serverà solo delle principali per

permettere ad un amministratore di configurare correttamente un firewall a protezione della propria rete. La struttura stessa del programma permetterà ad un utente esperto, una volta terminata la configurazione, di editare manualmente il file di configurazione prima dell'attivazione del firewall stesso. In tal modo sarà possibile aggiungere delle regole personalizzate o realizzare configurazioni più complesse che non vengono ancora contemplate dal programma.

Capitolo 3

Firewall-Config: struttura e utilizzo

Spesso la configurazione delle regole di IPTABLES risulta essere estremamente laboriosa. La soluzione che si è voluta fornire consiste nella progettazione di un software di configurazione che affronta il problema ponendosi su di un livello più alto.

All'utente che si servirà di tale programma non verrà richiesta una conoscenza approfondita riguardo la struttura dei comandi di IPTABLES, sebbene esso dovrà avere tutte le conoscenze tecniche di base adeguate ad affrontare il problema.

3.1 Scopo del programma

Firewall-Config si basa su di una semplice interfaccia che ha come scopo quello di essere di aiuto all'amministratore di rete nella configurazione del proprio firewall basato su IPTABLES.

L'interfaccia si basa su di un software opensource chiamato *dialog*, estremamente diffuso e già preinstallato in quasi tutte le distribuzioni GNU/Linux. Si tratta di un programma che permette la visualizzazione sulla console di una grande varietà di messaggi attraverso degli script avviati direttamente dalla shell.

Dialog implementa diverse finestre di dialogo e si basa sulle rinomate librerie *ncurses*. Queste librerie, distrubuite liberamente, mettono a disposizione dello sviluppatore una serie di API (*Application Programming Interface*) altamente flessibili ed efficenti che permettono di gestire la visualiz-

zazione di schermate su di un terminale a caratteri e quindi di creare delle UI (*User Interface*) in modalità testuale. È possibile, infatti, visualizzare finestre, pannelli, menu, form per l'inserimento dati e altro ancora. *Dialog* implementa molte delle funzionalità delle librerie.

3.2 Tipologie di rete supportate

Il primo passo per la configurazione di un firewall attraverso **Firewall-Config** spetta all'utente che deve progettare con attenzione ed essere a conoscenza della struttura della rete che vuole rendere sicura.

Il programma è in grado di supportare diverse tipologie di rete, tra le più comuni ed utilizzate.

3.2.1 Workstation

Struttura nella quale non vi è alcuna rete interna e il firewall viene attivato unicamente per proteggere la macchina su cui sta girando. Si può trattare indifferentemente di un computer connesso ad internet, anche attraverso una connessione dial-up, che svolge essenzialmente un ruolo di client, oppure anche una postazione con connessione permanente che mette a disposizione dei servizi.

3.2.2 Rete LAN

In questa struttura è presente una rete interna con indirizzi IP privati (LAN), quindi non indirizzabili direttamente dall'esterno, ed una connessione (solitamente permanente) ad Internet. In questo caso lo scopo di colui che gestisce tale struttura è quello di permettere agli utenti della rete interna di accedere ad Internet attraverso un IP pubblico (lo stesso dell'interfaccia esterna del firewall) e di evitare intrusioni dall'esterno.

Il primo obbiettivo può essere raggiunto utilizzando una particolare configurazione denominata SNAT (*Source Network Address Translation*). Ogni volta che un pacchetto generato da un terminale interno è destinato all'interfaccia esterna, il firewall sostituirà all'indirizzo IP sorgente privato nell'header del pacchetto, quello della propria interfaccia pubblica, in modo tale che gli eventuali pacchetti di risposta vengano inoltrati proprio a questo indirizzo. Ovviamente dovrà effettuare anche l'operazione inversa. Cioè, una volta che

un pacchetto di risposta giunge presso il firewall, dovrà inserire nuovamente l'indirizzo IP privato nel campo di destinazione del pacchetto, prima della decisione di routing. In tal modo, il pacchetto sarà inoltrato con successo all'interno della rete privata altrimenti non indirizzabile.

L'amministratore potrebbe anche decidere di permettere ai terminali presenti sulla rete interna di accedere solo a particolari siti oppure di vincolare il recupero delle informazioni da internet attraverso l'utilizzo di un server proxy.

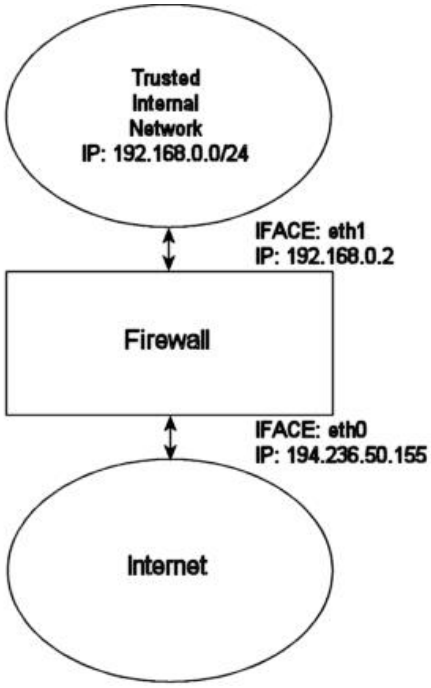

Figura 3.1: Un esempio di una rete privata LAN collegate ad un firewall

3.2.3 Rete DMZ

In questo caso la rete offre, attraverso delle macchine serventi, alcuni servizi che non si trovano più (o non più) solo sul firewall stesso. Principalmente esistono due motodologie per rendere disponibili dei servizi all'interno di una rete privata:

- Se la rete interna è configurata con IP privati e non indirizzabili dall'esterno, l'unica soluzione è quella di assegnare gli indirizzi pubblici

direttamente all'interfaccia esterna del firewall. Attraverso questa procedura chiamata IP aliasing (cioè conferire ad una stessa interfaccia di rete diversi indirizzi IP), il firewall sarà in grado di accettare non solo i pacchetti a esso destinati ma anche quelli indirizzati agli altri IP. A questo punto, attraverso un' operazione denominata DNAT (*Destination Network Address Translation*), il firewall modificherà l'indirizzo IP di destinazione di tutti i pacchetti indirizzati ad un particolare IP pubblico sostituendovi l'indirizzo privato corrispondente. Ovviamente dovrà svolgere l'operazione opposta non appena sopraggiungerà una risposta dalla rete interna destinata a quella pubblica. Tale sistema assicura una maggiore sicurezza dato che, nel caso in cui un'ipotetico instruso prendesse possesso di una macchina, si ritroverebbe comunque all'interno di una rete privata.

- È possibile, se si hanno a disposizione un elevato numero di indirizzi IP pubblici, create una sottorete interna (*subnetting*). Si può infatti dividere la propria rete in due macrosezioni; una di queste potrà essere utilizzata per creare una sottorete (*subnet*) da adibire a DMZ. In questo caso l'indirizzamento verso gli IP pubblici interni sarà svolto direttamente nell'ambito del routing.

3.2.4 Reti LAN e DMZ

In questa circostanza l'amministratore ha optato per una divisione funzionale delle macchine presenti all'interno della propria rete. Pertanto i terminali presenti nella LAN privata dovranno avere la possibilità di accedere ad Internet, mentre dovrà essere garantito l'accesso da parte degli utenti esterni ai servizi messi a disposizione dai server presenti nella DMZ. Tutte le considerazioni portate alla luce nei due casi precedenti valgono anche in questo scenario. Inoltre l'amministratore dovranno anche stabilire una politica di interconnessione tra le due interfacce interne. Dovrà infatti decidere se permettere o meno le connessioni da una verso l'altra e sotto quali condizioni.

3.2.5 Topologie complesse

Si tratta dello scenario più generico. Questo caso è analogo al precendente con l'eccezione del fatto che si dovrà stabilire quali servizi ogni singola rete DMZ metterà a disposizione verso l'esterno e la politica di interconnessione tra tutte le diverse interfacce.

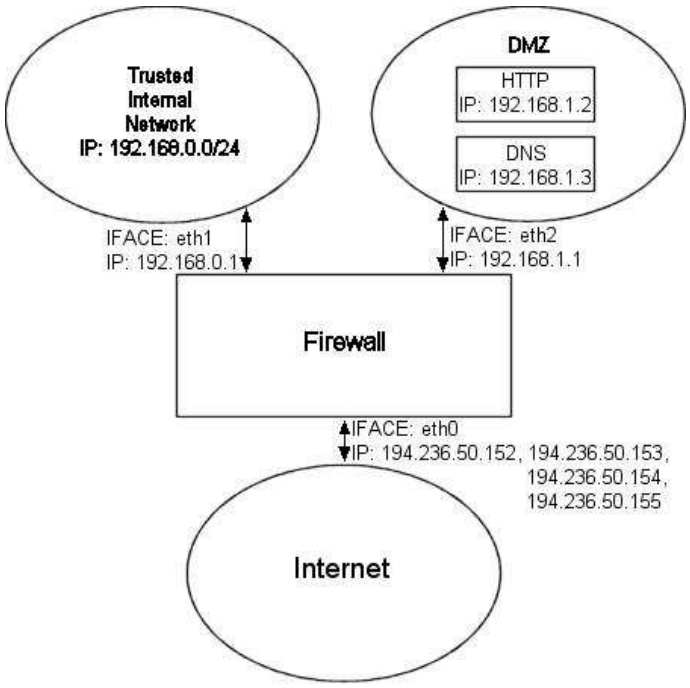

Figura 3.2: Un esempio di una rete privata LAN e una DMZ collegate ad un firewall

3.3 Struttura del software

Il software è stato progettato in modo tale da permettere all'utente di interagirvi attraverso diversi livelli. Il fine ultimo del programma risulta infatti quello di generare dinamicamente, a seconda delle preferenze dell'utente, uno script di shell in grado di impartire ad IPTABLES tutti i comandi necessari per implementare le regole del firewall.

Il programma permette una configurazione di tipo funzionale delle interfacce di rete e si occupa in modo del tutto trasparente della creazione delle regole adatte. Sarà infatti necessario dichiarare ogni interfaccia di rete in uso e la relativa funzione; **Firewall-Config** si comporterà in modo differente a seconda del ruolo. Le reti collegate al firewall vengono suddivise, a seconda della funzione che esse svolgono, nelle seguenti categorie:

- WAN (*Wide Area Network*): rappresenta il collegamento con l'esterno (solitamente Internet)
- LAN (*Local Area Network*): rappresenta la rete privata, all'interno della

quale non sono presenti indirizzi pubblici. La LAN è costituita da IP privati (RFC 1918).

- DMZ (*Demilitarized Zone*): rappresenta una rete interna collegata al firewall che mette a disposizione dei servizi verso l'esterno.

La configurazione del firewall viene infine salvata in un file di configurazione che utilizza un formato comprensibile, quindi facilmente modificabile a mano, che mantiene la stessa logica applicativa del software. In tal modo sarà possibile effettuare delle modifiche, anche sostanziali, senza dover necessariamente riavviare il programma. Risulta altresì semplice, in tal modo, rendere le impostazioni estremamente portabili, permettendo infatti lo spostamento del solo file di configurazione per avere l'intera configurazione su una macchina diversa, mantenendo la stessa logica di alto livello.

3.4 Analisi dei requisiti

Ovviamente, per eseguire tale programma si dovranno avere i privilegi di amministratore (root, UID 0) dato che i comandi che IPTABLES impartisce al kernel richiedono tali privilegi. **Firewall-Config** terminerà l'esecuzione non appena avviato nel caso in cui tale vincolo non venisse rispettato.

Per funzionare correttamente il software necessita di conoscere il percorso (*path*) di alcuni file utilizzati in diversi momenti della configurazione:

- *iptables*: viene utilizzato nel creare lo script di inizializzazione del firewall e serve per richiamare appunto il programma IPTABLES per generare le regole corrette per l'attivazione del firewall.

- *dialog*: viene utilizzato per la creazione dinamica delle interfacce utente (UI).

- *ifconfig*: programma utilizzato per gestire le diverse interfacce di rete. Viene utilizzato per creare, eliminare e visualizzare connessioni di rete.

- *grep*: programma ausiliario che viene utilizzato per recuperare dall'output di *ifconfig* le informazioni necessarie.

- *sh*: si tratta della shell da utilizzare per eseguire i comandi di IPTABLES nello script di inizializzazione del firewall.

- */etc/services*: si tratta del file di sistema che contiene la definizione di molti servizi noti. Specifica inoltre la porta e i protocolli utilizzati. Il programma si serve di questo file per aiutare l'utente a definire dei nuovi servizi.

Se il percorso di tali programmi non venisse indicato esplicitamente attraverso delle opzioni da linea di comando, durante la prima esecuzione, il programma provvederà autonomamente a ricercarli utilizzando il comando *which*. Terminata questa prima fase, tutti i percorsi precedentemente individuati verranno memorizzati all'interno del file di configurazione.

3.5 Opzioni attivabili da linea di comando

La linea di comando può essere utilizzata per impartire a **Firewall-Config** alcune opzioni. Le principali sono:

-c , –configuration-file: serve per specificare il percorso di un file di configurazione alternativo. Se non indicato diversamente il file viene ricercato o creato nella directory di default (*/etc/firewall-config.conf*).

-i , –init-file: serve per specificare il percorso dello script di inizializzazione. Se non indicato diversamente il file viene generato nella directory di default (*/usr/local/bin/firewall-init.rc*).

-d , –debug-file: viene utilizzato per specificare il percorso del file di debug. In tale file, se il debug fosse attivo, verrebbe inserita l'analisi dettagliata del file di configurazione.

–compile: impartendo questa opzione al programma (che può essere preceduta dall'opzione -c), verrà analizzato il file di configurazione e creato il file di inizializzazione del firewall, senza avviare l'interfaccia utente. Può essere utilizzata per la generazione dinamica del file di inizializzazione nel caso in cui la configurazione sia stata specificata manualmente o trasferita da un'altra macchina.

–exec: impartendo questa opzione viene direttamente eseguito lo script di inizializzazione e viene pertanto avviato il firewall senza però rileggere il file di configurazione.

–-iptables-path: il percorso completo del programma *iptables*

–-dialog-path: il percorso completo del programma *dialog*

–-sh-path: il percorso completo del programma *sh*

–-ifconfig-path: il percorso completo del programma *ifconfig*

–-grep-path: il percorso completo del programma *grep*

–-services-path: il percorso completo di */etc/services*

Le opzioni specificabili attraverso la linea di comando sono consultabili impartendo *–help* al programma.

3.6 Utilizzo del programma

Una volta avviato si verrà avvisati della prima esecuzione e **Firewall-Config** fornirà all'utente alcune indicazioni su come procedere con la configurazione.

La navigazione all'interno del programma avviene grazie all'ausilio di diversi menu. La struttura generale adottata in ognuno di questi presenta caratteristiche comuni quali la possibilità di di aggiungere, modificare, eliminare o visualizzare una determinata caratteristica o un insieme di opzioni (un'interfaccia, un servizio, una regola). Nelle pagine seguenti verranno analizzati i singoli menu e i servizi che essi realizzano.

3.6.1 *Device Management*

Il primo menu, *Devices Management*, non concerne direttamente con l'attivazione di un firewall ma può essere utilizzato per configurare le schede di rete presenti sulla macchina; sarà infatti possibile decidere, per ogni scheda, quale IP conferirle, quale subnet mask utilizzare e infine se assegnarle più indirizzi (IP Aliasing) permettendole di rispondere (attraverso il protocollo ARP) su diversi IP.

Dato che il primo passo della configurazione del firewall consisterà nel conferire una funzione ad ogni interfaccia, queste ultime dovranno essere attivate prima di tale momento. Il menu *Devices Management* vuole essere di aiuto a tal proposito.

Attraverso un successivo messaggio viene data all'utente la possibilità di attivare l'interfaccia immediatamente. In ogni caso, le configurazioni che avvengono attraverso questo menu, verranno implementate nello script di

avvio, e quindi le interfacce saranno attivate direttamente all'esecuzione di tale script.

3.6.2 Definizione dei servizi e delle reti

Prima di procedere alla configurazione delle regole del firewall, il programma permette all'utente di definire dei nome simbolici per i servizi e le reti che si vogliono utilizzare nella configurazione.

3.6.3 *Services Definitions*

Attraverso il menu *Services Definitions* risulta possibile assegnare un nome simbolico ad un servizio e metterlo in relazione ad una determinata porta e ad uno o più protocolli (TCP o UDP). Quando sarà necessario abilitare i servizi messi a disposizione dai server presenti nelle DMZ, verrà presentato l'elenco definito in questo menu. Tutte le definizioni saranno memorizzate nel file di configurazione del programma e quindi saranno disponibili anche ad una successiva esecuzione.

Il programma permette inoltre di importare direttamente dal file */etc/services* tutti i servizi necessari, evitandone in tal modo la definizione manuale.

3.6.4 *Networks Definitions*

Attraverso il menu *Networks Definitions* sarà possibile assegnare ad un indirizzo IP o ad un'intera rete un nome simbolico. Tale nome potrà in seguito essere utilizzato per la creazione delle regole del firewall. Il nome dovrà essere richiamato con il simbolo '$' preposto al nome stesso. Per esempio sarà possibile definire la propria rete interna attribuendole il nome simbolico $LAN e utilizzandola successivamente nelle regole di interconnessione.

Con questo metodo, risulta possibile modificare l'indirizzo IP o la rete a cui fa riferimento un nome simbolico senza dover necessariamente modificare le regole in cui è presente il nome stesso.

3.6.5 *Interfaces Configuration*

Il primo vero ed importante passo consiste nel fornire al programma lo schema della tipologia di rete che si vuole proteggere. Per fare ciò, sarà necessario

specificare le interfacce di rete effettivamente presenti sul firewall e le funzioni che esse dovranno svolgere all'interno della rete stessa. Tutto ciò viene realizzato attraverso il menu *Interfaces Configuration*.

Il programma richiede, per ogni interfaccia, un nome simbolico, la scheda di rete a cui essa fa riferimento e la funzione che svolgerà. Le scelte disponibili sono tre:

1. WAN: l'interfaccia svolge la funzione di collegamento tra il firewall e il mondo esterno (solitamente Internet ma potrebbe anche trattarsi del segmento principale della rete aziendale).

2. LAN: l'interfaccia viene utilizzata per collegare il firewall ad una rete privata all'interno della quale non sono presenti macchine serventi. L'accesso ad internet, in questo caso, può essere garantito attraverso l'operazione di NAT (*Network Address Translation*), gestita attraverso il menu *NAT Rules*.

3. DMZ: l'interfaccia viene utilizzata per collegare al firewall una rete che presenta delle macchine adibite a fornire dei servizi agli utenti esterni. In questo caso, una volta conferito un nome simbolico, sarà richiesta la tipologia di tale rete. Le scelte disponibili sono due:

   (a) Aliasing: se la rete è stata configurata con IP privati, l'interfaccia esterna del firewall dovrà essere attivata con gli indirizzi pubblici che si vogliono utilizzare per l'interno di tale rete come alias dell'IP principale dell'interfaccia.

   Verranno successivamente richiesti gli IP pubblici che dovranno essere mappati all'interno di tale rete. Infine, affinchè risulti possibile accedervi, sarà necessario rendere escplicito il mapping tra ogni singolo indirizzo pubblico e quelli privati presenti all'interno di questa DMZ attraverso la procedura di DNAT (*Source Network Address Translation*). Per fare ciò è disponibile il menu *Public IP Mapping*.

   (b) Subnetting: se la rete DMZ interna è stata configurata con IP pubblici o comunque risulta essere direttamente indirizzabile dall'esterno. In questo caso non verranno effettuate ulteriori richieste dato che ogni singolo IP presente all'interno di questa rete sarà direttamente accessibile dall'esterno.

È importante notare come l'assegnamento di un nome simbolico a ciascuna interfaccia di rete, la renda completamente indipendente dal dispositivo

fisico a cui esse è legata. In tal modo sarà sempre possibile modificare la scheda di rete di riferimento senza intaccare alcuna altra impostazione. Il nome simbolico deve quindi necessariamente essere univoco e non riutilizzabile (il programma provvederà ad avvisare nel caso riscontri questo genere di problema).

È prevista la configurazione di un'unica interfaccia di tipo WAN ma di un numero illimitato di reti LAN o DMZ. Ciò è dovuto al fatto che l'utilizzo di più reti WAN verso l'esterno avrebbe costretto una configurazione avanzata delle tabelle di routing basate sull'indirizzo sorgente oltre che ovviamente su quello di destinazione del pacchetto. Si tratta di impostazioni di non banale realizzazzione e che richiedono inoltre una particolare configurazione del kernel del sistema operativo.

3.6.6 *NAT Rules*

Questo menu può essere utilizzato per permettere ad interfacce LAN o DMZ (quest'ultima solo se configurata con IP privati al proprio interno), già definite attraverso il menu *Interfaces Configuration*, di accedere alla rete esterna. La creazione di un NAT porterà all'abilitazione del SNAT (*Source Network Address Translation*) e alla generazione di regole adatte a tale scopo nello script di inizializzazione. Una volta selezionata l'interfaccia, verrà richiesto quale indirizzo IP pubblico si vuole utilizzare per il NAT dell'interfaccia.

L'utente è tenuto ad indicare esplicitamente se desidera utilizzare l'opzione di *MASQUERADE*, quindi servirsi dell'IP pubblico del firewall ricavato dinamicamente da IPTABLES. In tal caso, se l'indirizzo cambiasse nel tempo (eventualità piuttosto comune se si considera una connessione di tipo dialup), la procedura di NAT continuerebbe a funzionare. L'opzione di *MASQUERADE* è decisamente comoda ma richiede un discreto utilizzo di risorse dato che l'indirizzo IP dovrà sempre essere mantenuto aggiornato.

Nel caso in cui si abbia a che fare solo con IP statici, è consigliabile rispondere negativamente alla domanda e specificare alla successiva richiesta l'indirizzo pubblico da utilizzare per effettuare il SNAT dell'interfaccia in oggetto. L'IP può essere l'indirizzo pubblico del firewall stesso o uno qualsiasi degli alias (se specificati) dell'interfaccia WAN.

3.6.7 *Public IP Mapping*

Attraverso questo menu sarà possibile permettere delle connessioni provenienti dall'interfaccia esterna (WAN) verso un IP della rete interna DMZ. Dato

che un indirizzo di rete privata non può essere indirizzato direttamente da Internet, è necessaria una particolare conversione che verrà effettuata per ogni singolo pacchetto che transita dal firewall. L'operazione, denominata DNAT (*Source Network Address Translation*), viene appunto utilizzata per fare in modo che le richieste di connessione indirizzate ad un particolare IP pubblico (configurato quindi come alias sull'interfaccia di rete WAN), vengano reindirizzate verso la rete privata oltre il firewall.

Attraverso tale procedimento il firewall, prima della decisione di routing, modificherà il campo di destinazione del pacchetto, permettendo a quest'ultimo di dirigersi nella direzione della macchina giusta. Ovviamente, nel senso contrario, verrà effettuata l'operazione inversa: verrà sostituito all'indirizzo sorgente privato quello pubblico utilizzato per mascherare la macchina all'interno della DMZ. Tutte queste operazioni avvengono in modo assolutamente trasparente per l'utente.

Il menu presenterà l'elenco degli indirizzi IP pubblici configurati attraverso *Interfaces Configuration* per quanto riguarda le interfacce definite come DMZ e alle quali è stato applicata la tecnica dell'IP aliasing. L'operazione di mapping consiste nell'indicare un indirizzo privato all'interno della stessa interfaccia DMZ a cui appartiene quello pubblico. Verrà in ogni caso presentato un valore di default ragionevole anche se, ovviamente, la scelta dovrà rispecchiare la struttura della rete.

3.6.8 *Enable Services*

Per mettere a disposizione del mondo esterno dei servizi, si dovrà necessariamente procedere alla loro abilitazione attraverso questo menu. Dato che la politica su cui è basato questo sistema consiste nel rifiutare tutte le connessioni qualora non diversamente specificato, sarà necessario indicare quali servizi rendere disponibili all'esterno per ogni singolo indirizzo che si possiede. Il menu mostrerà infatti tutti gli IP pubblici presenti nelle interfacce definite come DMZ.

Una volta selezionato un indirizzo, il programma permetterà di indicare quali servizi rendere disponibili. La lista che viene mostrata è la stessa definita attraverso *Services Configuration* e non è modificabile attraverso questo menu. Pertanto, se non è presente un servizio che si vuole attivare, sarà necessario tornare al menu *Services Definitions* e procedere all'aggiunta del nuovo servizio.

L'abilitazione dei servizi avviene attraverso un comodo menu di tipo

checklist: il tal modo sarà possibile aggiungere o eliminare anche più elementi alla volta.

3.6.9 Le Interconnessioni

Il menu successivo viene utilizzato per la configurazione delle diverse interconnessioni tra la varie interfacce. Per ogni interconnessione sarà necessario specificare una particolare politica:

- *Allow*: permette tutte le connessioni tra le due interfacce.
- *Deny*: blocca tutte le connessioni tra le due interfacce.

Risulterà inoltre possibile creare delle eccezzioni alle policy impostate. Infatti, nel caso di policy *allow*, sarà sempre possibile specificare di non accettare alcune tipologie di connessioni, mentre, nel caso di policy *deny*, si potranno permettere altri tipi di richieste. Queste eccezzioni vengono chiamate regole (*rules*) in **Firewall-Config**.

Non appena si aggiunge una nuova interfaccia tramite il menu *Interfaces Configuration*, il programma si occuperà di impostare automaticamente la policy di default per le interconnessioni con tutte le altre interfacce esistenti. La configurazione di default prevede:

- Connessioni da qualsiasi interfaccia verso una LAN o DMZ: *deny*. La reti interne sono considerata private e quindi nessuna connessione può avvenirvi se non esplicitamente indicato.
- Connessioni da LAN verso DMZ: *allow*. Vengono permesse le connessioni da una rete LAN verso una DMZ.
- Da LAN o DMZ verso WAN: *allow*. Di default è possibile effetturare qualsiasi connessione tra le reti interne e quella esterna. In tal modo si presuppone che gli utilizzati dei terminali interni siano in qualche modo fidati.
- Da tutte le interfacce verso il firewall: *deny*. Viene considerato lo scenario più probabile, cioè che il firewall non metta a disposizione alcun servizio.

Ogni volta che viene aggiunta una nuova interfaccia, anche manualmente modificando direttamente il file di configurazione, il programma provvederà ad impostare la politica di interconnessione più adatta.

Per procedere alla configurazione delle diverse interconnessioni, l'utente potrà interagire con **Firewall-Config** utilizzando il menu *Interconnections Configuration*.

3.6.10 *Interconnections Configuration*

Permette di impostare la politica e le regole di interconnessione tra tutte le interfacce (esterne e interne di qualsiasi tipo).

È importante mettere in evidenza che la modifica della politica di interconnessione porta all'eliminazione di tutte le regole precedentemente create dato che non avrebbe più senso mantenerle attive essendo cambiata la policy.

Questo menu può essere utilizzato per esempio per limitare la navigazione di alcuni utenti, imponendo loro delle restrizioni, per permettere connessioni via rete dalle diverse interfacce verso il firewall attivando alcuni servizi, oppure per abilitare la connessione tra un application server che si trova in una DMZ e un database server presente in una rete LAN che mette a disposizione il servizio tramite una particolare porta.

3.6.11 *Firewall Options*

Il menu *Firewall Options* propone all'utente diverse configurazioni aggiuntive applicabili al firewall per migliorarne la sicurezza e, in ultima analisi, il servizio offerto alla rete. Alcune di queste si realizzano attraverso delle modifiche al filesystem virtuale */proc*, altre aggiungono delle regole particolari allo script di inizializzazione. Vediamole in dettaglio:

- *refuse_rfc1918*: attivando questa opzione saranno bloccati tutti i pacchetti aventi indirizzi IP definiti nella RFC 1918 provenienti dall'interfaccia esterna. Essi sono infatti da considerarsi privati e quindi non possono, o meglio, non dovrebbero uscire al di fuori di una rete locale. Verranno bloccati i seguenti blocchi di IP, descritti in notazione CIDR:
  - 10.0.0.0/8
  - 172.16.0.0/12
  - 192.168.0.0/16
  - 127.0.0.0/8

  Nel caso in cui tali IP provenissero dall'interfaccia esterna, molto probabilmente si tratterebbe di un attacco di tipo *IP Spoofing*.

- *tcp_integrity*: attivando questa opzione verranno effettuti dei controlli ulteriori sui pacchetti che veicolano il protocollo TCP. In particolare verranno rifiutati:

  - Pacchetti con entrambi i bit SYN e ACK attivi in cui la macchina a stati indichi tale connessione come NEW.
  - Connessioni che si trovano nello stato NEW senza che sia stato ricevuto alcun pacchetto con il bit SYN attivo.

  In entrambi i casi verrano quindi rifiutate solo connessioni e pacchetti anomali.

- *refuse_invalid*: verranno bloccate tutte le connessioni che si trovano in uno stato INVALID. Ciò accade quando la macchina a stati non è in grado di stabilire la situazione di un particolare pacchetto.

- *refuse_netbios*: viene rifiutato tutto il traffico generato dal servizio Netbios di Windows, quindi tutti i pacchetti che hanno come porta sorgente e destinazione l'intervallo 135-139, proveniente dalla sola interfaccia esterna. Tali pacchetti, non solo non saranno accettati, ma non verranno neppure inclusi nel file di log. Questa opzione può essere utilizzata se non si è interessati a tale traffico e non si vogliono dei log troppo voluminosi dato che le richieste in broadcast di Netbios risultano essere molto frequenti.

- *refuse_netbios_all*: si tratta di un'opzione analoga alla precedente. In questo caso però verrà bloccato il traffico generato da Netbios proveniente da qualsiasi interfaccia di rete.

- *refuse_rip2*: vengono rifiutati i pacchetti UDP utilizzati da alcuni router per inviare in broadcast messaggi del protocollo RIP, utilizzando come porta sorgente e destinazione la 520. Anche in questo caso i pacchetti (inviati solitamente ogni 30 secondi) non verranno memorizzati all'interno del file di log.

- *refuse_dhcp*: viene rifiutato il traffico UDP indirizzato verso l'indirizzo di broadcast 255.255.255.255 utilizzato in alcuni casi dal servizio DHCP. I pacchetto bloccati non saranno memorizzati nel file di log.

- *refuse_multicasting*: se non si desidera utilizzare dei servizi multicasting, questa opzione permette di bloccare tutto il traffico destinato a 224.0.0.0/8 proveniente dall'interfaccia di rete esterna.

- *refuse_icmp*: con questa opzione verranno rifiutati tutti i pacchetti del protocollo ICMP.
  Verrà in ogni caso lasciato passare il tipo *destination-unreachable*, comunemente utilizzato per indicare il mancato raggiungimento dell'host di destinazione.

- *accept_ping*: questa opzione permette di rifiutare tutti i pacchetti ICMP ad eccezione dei ping (*echo-request* e *echo-reply*, tipo 0 e 8 rispettivamente). Questa regola verrà applicata per tutti e soli i pacchetti provenienti dall'interfaccia esterna.

- *accept_traceroute*: vengono accettate le operazioni di traceroute effettuate sia utilizzando il protocollo ICMP (quindi utilizzando i tipi *ttl-zero-during-reassembly* e *ttl-zero-during-transit*) sia l'UDP (con porta di destinazione compresa nell'intervallo 33434-33523).

- *use_unclean*: viene utilizzato il modulo *unclean* di IPTABLES che si dovrebbe occupare di bloccare le connessioni non regolari. Il modulo viene considerato sperimentale dal manuale di IPTABLES .

- *limit_ping*: limita il numero dei ping ad un particolare valore per intervallo di tempo. Questa opzione può essere utilizzata per evitare attacchi DoS di tipo *Ping Flooding*.

- *refuse_impossible_flags*: vengono rifiutati tutti i pacchetti TCP che hanno una combinazione di flag impossibile o comunque non utilizzata. Normalmente in tal modo viene bloccata l'azione di molti programmi di tipo port-scanning.

- *use_rst_for_ident*: la comunicazione con un programma che ricerca il servizio IDENT (porta 113) viene chiusa in modo pulito attraverso un pacchetto TCP con il bit RST attivo.

- *accept_source_route*: questa opzione, sconsigliata, permette di accettare dei pacchetti con all'interno l'opzione di *source route*. Nonostante possa essere utilizzato per testare la rete, il source route potrebbe causare enormi problemi di sicurezza dato che renderebbe possibile, per un'ipotetico malintenzionato, specificare per ogni singolo pacchetto il percorso da seguire in modo da eludere i controlli del firewall.
  L'opzione viene gestita attraverso il file
  /proc/sys/net/ipv4/conf/all/accept_source_route.

- *refuse_internal_spoofing*: vengono rifiutati tutti i pacchetti provenienti da interfacce interne aventi come mittente un indirizzo IP non appartenente alle reti stesse. In questo modo si vuole evitare lo spoofing dei pacchetti generati internamente.
- *limit_syn*: limita la frequenza di arrivo di pacchetti TCP con il bit SYN attivo (che rappresenta l'inizializzazione di una nuova connessione TCP, il primo passo del *three-way handshake*). In tal modo sarà possibile evitare attacchi DoS di tipo *Syn Flooding*.
- *good_nat*: viene impedito che indirizzi di rete locale escano al di fuori del firewall, verso l'interfaccia esterna.
- *log_martians*: effettua il log dei pacchetti che presentato un indirizzo IP impossibile. Viene gestito attraverso il file */proc/sys/net/ipv4/conf/all/log_martians*.
- *accept_redirects*: attivando questa opzione, verranno accettati i messaggi ICMP di tipo redirect.
  Viene utilizzato il file */proc/sys/net/ipv4/conf/all/accept_redirects*.
- *rp_filter*: viene effettuata una validazione dell'indirizzo mittente come specificato nella RFC 1812.

3.6.12 *Blacklist*

Attraverso questo menu sarà possibile definire una vera e propria blacklist elencando una serie di indirizzi IP o reti dai quali non si vuole accettare alcuna connessione.

3.6.13 *Show Actual Configuration*

Vengono mostrate, attraverso un'unica schermata, tutte le impostazioni della configurazione corrente. Può risultare utile per ottenere una visione di insieme della propria rete e dei settaggi che sono stati effettuati.

3.6.14 *Save Configuration*

La configurazione fino a questo punto specificata attraverso i diversi menu viene salvata nel file di configurazione. Si ha inoltre la generazione dello script di inizializzazione del firewall.

3.6.15 *Apply Saved Configuration*

Attraverso questo menu sarà possibile attivare tutte le impostazioni precedentemente salvate.

Capitolo 4

Firewall-Config: Implementazione

Il programma **Firewall-Config** è stato scritto utilizzando il Perl come linguaggio di programmazione. Si è scelto tale linguaggio innanzi tutto per ottenere una maggiore semplicità nell'elaborazione delle stringhe, per esempio nel gestire l'output di *ifconfig* (per recuperare le informazioni relative alle interfacce configurate sulla macchina) o nella realizzazione di un parser in grado di caricare nelle strutture dati predisposte la configurazione.

Il codice del programma può essere diviso, da un punto di vista funzionale, in diverse sezioni, ognuna predisposta a svolgere un particolare compito:

- Parser del file di configurazione
- Creazione e gestione dell'interfaccia utente per la configurazione
- Salvataggio del file di configurazione
- Generazione dello script di inizializzazione del firewall

È importante notare come il programma non è in grado di recuperare le impostazioni della configurazione direttamente dallo script di inizializzazione una volta generato. Questa scelta progettuale è dovuta al fatto che le informazioni presenti nello script di shell non sono strutturate e quindi non posseggono la logica di alto livello presente invece nel file di configurazione vero e proprio.

Analizziamo ora brevemente come il flusso dati attraverso tipicamente la struttura del programma:

1. L'utente avvia **Firewall-Config**. Nel caso in cui non sia stato specificato alcun file di configurazione attraverso la linea di comando e non venga trovato nella posizione di default, il programma avvisa che si tratta della prima esecuzione e si occupa di verificare la versione di *dialog* in uso e ricercare all'interno del filesystem i programmi di supporto necessari.

2. Se viene individuato un file di configurazione, viene attivato il parser e vengono caricati all'interno delle strutture dati predisposte tutte le impostazioni presenti.

3. A questo punto viene attivata l'interfaccia utente che rappresenta la parte più voluminosa del codice del programma.

4. Quando l'utente vuole salvare la configurazione, il programma si occuperà di creare il file di configurazione e contemporaneamente generare lo script di inizializzazione del firewall.

5. A questo punto lo script risulta essere completamente indipendente dal programma che l'ha generato. Potrà quindi essere avviato direttamente o spostato su una diversa macchina.

4.1 Strutture dati

Un ruolo di rilievo per la corretta memorizzazione di tutte le informazioni immesse dall'utente, è rappresentato dalle strutture di memorizzazione dei dati. Il Perl mette a disposizione, oltre alle comuni variabili scalari e agli array, anche un'altra struttura dati: l'*hash*. Esso permette di richiamare il valore di una variabile attraverso una chiave alfanumerica.

Un'altra importante struttura dati che viene utilizzata dal programma per memorizzare le caratteristiche di ogni rete, viene messa a disposizione dal modulo *Net::Netmask*. La creazione di un'istanza di un'oggetto di tipo *Net::Netmask* avviene semplicemente richiamando il metodo costruttore passando come prametri un'indirizzo IP e una subnet mask. Il modulo mette a disposizione un numero consistente di utili operazioni sull'oggetto. Per esempio sarà possibile verificare se un particolare indirizzo IP si trovi all'interno di una determinata rete, enumerarne tutti gli IP che vi appartengano, ottenerne la notazione CIDR.

Gli hash principali utilizzati per la memorizzazione dei dati sono:

- `%Programs`: vengono memorizzati tutti i percorsi dei programmi di supporto.

  ```
  $Programs{iptables}
  $Programs{dialog}
  $Programs{ifconfig}
  ...
  ```

- `%Create_Interfaces`: si tratta di un hash multidimensionale utilizzato per la memorizzazione di tutte le informazioni relative alle interfacce che si vogliono attivare all'avvio, cioè quelle definite attraverso il menu *Devices Management*. L'hash è caratterizzato da una chiave primaria (l'interfaccia che si vuole attivare) e da tutti i dati relativi alla configurazione quali l'indirizzo IP, la subnet mask e gli eventuali alias che si desiderano attivare.

  ```
  $Create_Interfaces{eth0}{ip}
  $Create_Interfaces{eth0}{netmask}
  ...
  ```

- `%Interfaces`: contiene tutte le informazioni riguardanti le interfacce che vengono configurate attraverso il menu *Interfaces Configuration*. Anche in questo caso si tratta di un hash di hash e come chiave primaria viene utilizzato il nome simbolico conferito all'interfaccia. Viene memorizzato l'indirizzo IP, la rete a cui esso appartiene sotto forma di oggetto *Net::Netmask*, il device della scheda di rete, la funzione che svolge l'interfaccia (WAN, LAN o DMZ), eventualmente i servizi che essa mette a disposizione (memorizzati sotto forma di array) se si tratta di una rete DMZ, la mappatura tra indirizzo IP pubblico esterno e privato nel caso si voglia utilizzare DNAT (le impostazioni saranno memorizzate a loro volta sotto forma di hash).

  ```
  $Interfaces{lan}{ip}
  $Interfaces{lan}{type}
  $Interfaces{lan}{device}
  ...
  ```

- `%Interconnection`: vengono memorizzate in un unico hash tutte le informazioni relative all'interconnessione tra ogni interfaccia, indipendentamente dalla funzione che essa svolge all'interno della rete. L'hash ha due chiavi. La prima rappresenta il nome simbolico dell'interfaccia

sorgente, il secondo quello di destinazione. Vengono inoltre utilizzati solo due valori atti a memorizzare la politica di interconnessione e le eventuali regole definite dall'utente (memorizzate sotto forma di array).

```
$Interconnection{lan}{wan}{policy}
$Interconnection{lan}{wan}{rules}
```

Come interfaccia di destinazione dell'interconnesione può essere utilizzata anche la parola chiave FIREWALL che rappresenterà quindi il collegamento tra una particolare rete e il firewall stesso.

- `%Names`: in questo hash viene memorizzata la mappatura tra i nomi simbolici e gli indirizzi IP o le reti definiti attraverso il menu *Networks Definitions*.
- `%Services`: in questo hash viene memorizzata la mappatura tra i nomi dei servizi, la porta e i protocolli utilizzati definita attraverso il menu *Services Definitions*.
- `%Blacklist`: in questo hash vengono memorizzate le reti dalle quali non si desidera ricevere connessioni e che quindi formeranno la blacklist del firewall.
- `%Options`: vengono memorizzate le opzioni attivabili attraverso il menu *Firewall Options*.

Un'altra struttura dati che svolge un ruolo importante è l'hash `%Text`. In esso vengono memorizzate tutte le stringhe di testo che saranno presentate all'utente attraverso l'interfaccia. Questa scelta permette una facile modifica di tutti i testi evitando una ricerca complessa all'interno del codice. In tal modo potrebbe risultare comoda anche un'eventuale traduzione del programma in un'altra lingua.

A titolo di esempio si è voluto riportare un frammento di codice che vuole essere esemplificativo dell'uso di molte delle strutture dati presentate fino a questo momento. La funzione *set_default_policies()* qui presentata viene utilizzata dal programma per stabilire la corretta politica di interconnessione tra le diverse interfacce:

```
#############################################################
# set_default_policies($review,%iface)
# Set the default policies for a particular interface.
# If $review exists, set the policy only if it's not
```

```
# already been defined.
# LAN -> LAN deny
# LAN -> DMZ/WAN allow
# DMZ -> WAN allow
# DMZ -> DMZ|LAN deny
# WAN -> * deny

sub set_default_policies {
        my ($review,%iface) = @_;
        my ($name,$type) = ($iface{name},$iface{type});
        for (keys(%Interfaces)) {
                my $iface_type = make_default($Interfaces{$_},'type');
                # avoid to create name->name interconnection
                next if $_ eq $name;

                # Do not create WAN->LAN interconnections.
                next if $type eq 'wan' and $iface_type eq 'lan';

                # Review current policies: set them if are not defined
                next if $review eq 'review' and defined
                $Interconnection{$name}{$_}{policy};

                if ($type eq 'lan') {
                        # Allow all from the LAN to the outside world,
                        # not to other LAN
                        $Interconnection{$name}{$_}{policy} =
                        $iface_type eq 'lan' ? 'deny' : 'allow';

                        # Deny all incoming connections
                        $Interconnection{$_}{$name}{policy} = 'deny';

                } elsif ($type eq 'dmz') {
                        # Allow connections to the WAN but not to LAN
                        # or other DMZ interfaces
                        $Interconnection{$name}{$_}{policy} =
                        $iface_type eq 'wan' ? 'allow' : 'deny';

                        # Allow connections from LAN but not from WAN
                        # or other DMZ interfaces
                        $Interconnection{$_}{$name}{policy} =
                        $iface_type eq 'lan' ? 'allow' : 'deny';
```

```
                } elsif ($type eq 'wan') {
                        # Deny connections between a WAN and a DMZ
                        # interface
                        $Interconnection{$name}{$_}{policy} = 'deny';
                }

                # Clear all previously defined rules
                $Interconnection{$_}{$name}{rule} = undef;
        }

        # Deny all the connections to the firewall
        $Interconnection{$name}{INPUT}{policy} = 'deny';
}
```

4.2 Il file di configurazione

Il file di configurazione è stato progettato in modo tale da presentare una sintassi semplice e intuitiva. Tutto ciò permette all'utente eventuali modifiche manuali senza necessariamente utilizzare l'interfaccia di configurazione. Il file è diviso in diverse sezioni; ognuna di esse contribuisce alla creazione di una determinata struttura dati. Ogni sezione è racchiusa tra delle parentesi graffe ed è caratterizzata dal seguente formato:

```
nome_sezione [nome_componente] {
        attributo1 valore1
        attributo2 valore2
        attributo3 valore3
        ...
}
```

Ogni riga vuota o preceduta dal simbolo '#' viene ignorata. Il numero degli spazi o dei caratteri di tabulazione utilizzati per separare i diversi elementi non interferisce con la corretta interpretazione del file di configurazione. Procediamo con l'analisi delle diverse sezioni.

4.2.1 *programs*

La sezione *programs* definisce i percorsi dei programmi di supporto utilizzati da **Firewall-Config**. Se si desidera procedere alla creazione manuale del

file di configurazione sarà necessario specificare in questa sezione i percorsi di tutti i file necessari (sh, iptables, ifconfig, grep, dialog, services) dato che non verranno ricercati automaticamente.

```
programs {
        programma1 <percorso1>
        programma2 <percorso2>
        ...
}
```

4.2.2 *create_interface*

Ognuna di queste sezioni viene utilizzata per attivare un particolare dispositivo. Il programma le leggerà e inserirà in testa allo script di inizializzazione del firewall i comandi di *ifconfig* necessari all'attivazione delle interfacce qui specificate.

```
create\_interface <device> {
        ip <indirizzo ip>
        netmask <network mask>
        [ip_alias <ip1,ip2,ip3,...>]
}
```

4.2.3 *interface*

In questa sezione sarà possibile configurare tutte le interfacce presenti sulla macchina secondo il seguente formato:

```
interface <nome> {
        device <device>
        ip <indirizzo ip>
        type <lan|wan|dmz>
        [network <network in notazione CIDR>]
        [ip_alias <ip1,ip2,ip3>]
        [nat <ip|MASQUERADE>]
        [dmz <aliasing|subnetting>]
        [public_ip <ip_pubblico1->ip_privato1>]
        [public_ip <ip_pubblico2->ip_privato2>]
        ...
        [service <ip_pubblico1:servizio1>]
```

```
        [service <ip_pubblico2:servizio2>]
        ...
}
```

I valori opzionali dipendono dal ruolo dell'interfaccia all'interno della rete. Infatti, per una LAN risulterà possibile specificare la tipologia di NAT, mentre per una DMZ, si potrà stabilire quali servizi offrire al mondo esterno attraverso l'opzione *service*.

4.2.4 *interconnection*

Le impostazioni di interconnessione tra le singole interfacce, definite attraverso le sezioni *interface*, devono essere esplicitate utilizzando il seguente formato:

```
interconnection <nome1>-><nome2> {
        policy <allow|deny>
        [<allow|deny> <regola>]
}
```

Dove <regola> è una stringa del tipo:

```
<network_sorgente>|*:<porta_sorgente>|*->
<network_destinazione>|*:<porta_destinazione>|*(<tcp|udp>)
```

Si devono quindi esplicitamente indicare l'indirizzo IP o la rete di sorgente e destinazione con le rispettive porte (sarà comunque sempre possibile utilizzare il carattere '*' per indircare tutte le reti o tutte le porte) insieme ai protocolli coinvolti. Per ogni interfaccia sarà necessario inoltre definire l'interconnessione tra questa e il firewall stesso. Ciò avviene definendo il collegamento verso 'FIREWALL'. Per esempio:

```
interconnection LAN->FIREWALL {
        ...
}
```

4.2.5 *definitions*

Questa sezione è utilizzata per memorizzare le definizioni di indirizzi IP o di intere reti:

```
definitions {
        \$nome_simbolico <ip|network>
        ...
}
```

Il nome conferito alla rete che si vuole definire deve essere preposto dal simbolo '$', adeguatamente protetto da un backslash.

4.2.6 *services*

Attraverso la sezione *services* sarà possibile definire i servizi che verranno messi a disposizione dal programma:

```
services {
        <nome_servizio1> <porta1> (<tcp|udp>)
        <nome_servizio2> <porta2> (<tcp|udp>)
        ...
}
```

4.2.7 *blacklist*

La sezione *blacklist* può essere utilizzata per definire la lista nera del firewall:

```
blacklist {
        <network|ip>
        ...
}
```

4.2.8 *options*

Tutte le opzioni supplementari da impartire al firewall si trovano in questa sezione:

```
options {
        nome_opzione1 <yes|no>
        nome_opzione2 <yes|no>
        nome_opzione3 <yes|no>
        ...
}
```

4.3 Lo script di inizializzazione

Una volta completata la configurazione attraverso l'interfaccia utente, il programma, oltre a salvare tutte le impostazioni nel file di configurazione, genererà automaticamente uno shell script di inizializzazione che rappresenta il vero meccanismo di avvio del firewall.

La generazione di tale file viene effettuata in base alle impostazioni indicate dall'utente. Il file può essere logicamente diviso nelle seguenti sezioni:

1. Attraverso il comando *ifconfig* vengono attivate le interfacce di rete che l'utente ha deciso di inizializzare all'avvio attraverso il menu *Devices Management* o, analogamente, nelle sezioni *create_interface* del file di configurazione. Se esempressamente indicato, verranno inoltre attivati i diversi alias (IP aliasing).

2. Vengono caricati nel kernel i moduli di IPTABLES necessari al corretto funzionamento del firewall. Se tali moduli non fossero disponibili o già inclusi nel kernel stesso, non verrà mostrato all'utente alcun messaggio di errore. I moduli che vengono caricati sono i seguenti:

   - *ip_tables*
   - *iptable_filter*
   - *iptable_nat*
   - *ipt_state*
   - *ipt_LOG*
   - *ipt_limit*
   - *ipt_REJECT*
   - *ipt_multiport*
   - *ip_conntrack*
   - *ip_conntrack_ftp*
   - *ip_nat_ftp*
   - *ipt_unclean*

3. Vengono configurati attraverso il filesystem virtuale */proc* i parametri di rete riguardanti l'ipv4. Alcune di queste impostazioni vengono esplicitamente indicate dall'utente attraverso il menu *Firewall Options*.

4. Vengono eliminate tutte le catene esistenti e vengono impostate le policy di default per quelle principali:

- INPUT: DROP
- OUTPUT: ACCEPT
- FORWARD: DROP

5. Vengono create due catene che saranno utilizzate rispettivamente nel caso in cui un pacchetto venisse accettato o rifiutato. Nel primo caso verranno effettuti alcuni controlli nell'ambito del protocollo TCP, nel secondo verrà tenuta traccia dei pacchetti nel file di log (attraverso la chain LOG) prima di essere definitivamente rifiutati.

6. Viene generata la blacklist eventualmente definita dall'utente.

7. Vengono accettati in ingresso, sia per quanto riguarda la catena INPUT, sia per FORWARD, tutte le connessioni già stabilite o in relazione ad altre attraverso la macchina a stati e le condizioni di ESTABLISHED e RELATED.

8. Vengono imprementati tutti i controlli definiti dall'utente attraverso il menu *Firewall Options*.

9. Viene creata la catena *check_tcp* nella quale si effettueranno tutti i controlli di integrità legati al protocollo TCP.

10. I pacchetti di tipo ICMP verranno reindirizzati in una catena dedicata nella quale verrà applicata la politica definita dall'amministratore per quanto riguarda questo protocollo.

11. Viene creata una catena per ogni singola interconnessione nella quale i pacchetti saranno accettati solo se espressamente indicato dalla policy o dalle regole impostate dall'utente.

12. Vegono abilitate le impostazioni di SNAT e DNAT qualora specificate dalla configurazione.

Lo script sarà generato in modo tale da essere simile, come struttura, ad un qualsiasi altro programma di inizializzazione.
Difatti dovrà essere lanciato indicando l'azione da eseguire:

```
Usage: firewall-init.rc {start|stop|status|restart}
```

- *start*: avvia il firewall attivando tutte le regole e le catene precedentemente illustrate.

- *stop*: elimina tutte le regole precedentemente create e ripristina la politica originaria accettando quindi ogni connessione.
- *restart*: riavvia il firewall.
- *status*: mostra, attraverso il comando *IPTABLES -L*, tutte le regole in funzione.

Conclusioni

Firewall-Config è stato progettato per consentire all'utente di modellare in modo astratto sistemi di reti di complessità arbitraria, realizzando in seguito in modo automatico la configurazione di un sistema di firewalling basato su IPTABLES.

L'elemento di novità introdotto dal software consiste nel proporre una soluzione alternativa all'amministratore di rete per la configurazione del proprio firewall, rendendola semplice ed intuitiva, e presentando all'utente dei menu attraverso la console.

Contrariamente ad altri software esistenti, **Firewall-Config** utilizza esclusivamente la shell di sistema, senza costringere l'utente all'installazione di interfacce grafiche o web-based. Inoltre, il meccanismo di astrazione utilizzato guida l'utente nella progettazione del firewall, senza costringerlo a preoccuparsi di dettagli implementativi.

Si possono facilmente immaginare possibili estensioni future dell'architettura di **Firewall-Config**; ad esempio, si potrebbe introdurre la possibilit di definire e gestire connessioni VPN (Virtual Private Network), utilizzando la stessa struttura intuitiva e ad alto livello con cui sono state gestite le configurazioni di IPTABLES.

In conclusione, **Firewall-Config** è un sistema estendibile e user-friendly per progettare e implementare soluzioni di firewalling su piattaforma GNU/Linux.

Appendice A: esempio di file di configurazione

Nel seguente file di configurazione di esempio viene simulata una delle situazioni più comuni. È stata infatti definita un'interfaccia WAN e una DMZ. All'interno della rete DMZ, configurata tra l'altro con IP privati, sono presenti due server, un webserver e un server di posta. Nella DMZ si trovano inoltre dei terminali ai quali deve essere garantita la possibilità di accedere ad Internet:

```
programs {
        dialog /usr/bin/dialog
        ifconfig /sbin/ifconfig
        grep /bin/grep
        iptables /sbin/iptables
        services /etc/services
}

create_interface eth0 {
        ip 192.168.2.100
        netmask 255.255.255.0
}

create_interface eth1 {
        ip 131.175.124.136
        netmask 255.255.255.128
        ip_alias 131.175.124.137
        ip_alias 131.175.124.138
}

interface lan {
        device eth0
```

```
        type dmz
        ip 192.168.2.100
        dmz aliasing
        nat 131.175.124.136
        network 192.168.2.0/24
        public_ip 131.175.124.137->192.168.2.200
        public_ip 131.175.124.138->192.168.2.6
        service 131.175.124.137:smtp
        service 131.175.124.137:imap
        service 131.175.124.137:pop3
        service 131.175.124.138:http
        service 131.175.124.138:https
}

interface wan {
        device eth1
        type wan
        ip 131.175.124.136
}

interconnection lan->FIREWALL {
        policy deny
}

interconnection lan->wan {
        policy allow
}

interconnection wan->FIREWALL {
        policy deny
}

interconnection wan->lan {
        policy deny
}

services {
        ssh 22 (tcp,udp)
        http 80 (tcp,udp)
        ftp 21 (tcp,udp)
        nicname 43 (tcp,udp)
```

```
        https 443 (tcp,udp)
        smtp 25 (tcp,udp)
        imap 143 (tcp,udp)
        pop3 110 (tcp,udp)
        telnet 23 (tcp,udp)
        domain 53 (tcp,udp)
        snmp 161 (tcp,udp)
}

definitions {
}

blacklist {
}

options {
        limit_ping no
        good_nat yes
        accept_traceroute yes
        refuse_rip2 no
        tcp_integrity yes
        use_rst_for_ident yes
        refuse_icmp no
        log_martians yes
        accept_ping yes
        refuse_rfc1918 yes
        rp_filter yes
        refuse_impossible_flags yes
        refuse_internal_spoofing yes
        use_unclean no
        refuse_netbios no
        refuse_dhcp yes
        refuse_netbios_all no
        accept_source_route no
        refuse_multicasting yes
        accept_redirects no
        limit_syn no
        refuse_invalid no
}
```

Appendice B: esempio di script di inizializzazione

Di seuito viene riportato lo script di inizializzazione generato dalla configurazione presentata in precedenza:

```
#!/bin/sh

#### start(): brings up the firewall

start()  {

### Networks Definition



### Devices to bring up:

/sbin/ifconfig eth0 down
/sbin/ifconfig eth0 192.168.2.100 netmask 255.255.255.0 up


/sbin/ifconfig eth1 down
/sbin/ifconfig eth1 131.175.124.136 netmask
255.255.255.128 up
/sbin/ifconfig eth1:0 131.175.124.137 netmask
255.255.255.128 up
/sbin/ifconfig eth1:1 131.175.124.138 netmask
255.255.255.128 up


### Load some modules (do not show error messages, the modules
```

```
### may be already loaded into the kernel):

modprobe ip_tables  2>/dev/null
modprobe iptable_filter  2>/dev/null
modprobe iptable_nat  2>/dev/null
modprobe ipt_state  2>/dev/null
modprobe ipt_LOG  2>/dev/null
modprobe ipt_limit  2>/dev/null
modprobe ipt_REJECT  2>/dev/null
modprobe ipt_multiport  2>/dev/null
modprobe ip_conntrack  2>/dev/null
modprobe ip_conntrack_ftp  2>/dev/null
modprobe ip_nat_ftp  2>/dev/null
modprobe ipt_unclean  2>/dev/null


### General ipv4 configuration:

echo "1" > /proc/sys/net/ipv4/ip_forward
echo "1" > /proc/sys/net/ipv4/conf/all/rp_filter
echo "0" > /proc/sys/net/ipv4/conf/all/accept_redirects
echo "1" > /proc/sys/net/ipv4/conf/all/log_martians
echo "0" > /proc/sys/net/ipv4/conf/all/accept_source_route


### iptables initialization: flush all existing chains and set
###  default policies to INPUT, OUTPUT and FORWARD:

/sbin/iptables -F
/sbin/iptables -F -t nat

/sbin/iptables -X

/sbin/iptables -P INPUT DROP
/sbin/iptables -P FORWARD DROP
/sbin/iptables -P OUTPUT ACCEPT


### Create the 'allow' chain. When a TCP connection is allowed
### arrives here. Accept only syn, established or related TCP
### connections:
```

```
/sbin/iptables -N allow
/sbin/iptables -A allow -p ! tcp -j ACCEPT
/sbin/iptables -A allow -p tcp --syn -j ACCEPT
/sbin/iptables -A allow -p tcp -m state --state ESTABLISHED,
RELATED -j ACCEPT
/sbin/iptables -A allow -p tcp -j DROP


### Create the 'logdrop' chain. Log a packet before
### dropping it:

/sbin/iptables -N logdrop
/sbin/iptables -A logdrop -p tcp -m limit --limit 1/s
--limit-burst 10 -j LOG --log-prefix "DROPPED tcp: "
/sbin/iptables -A logdrop -p udp -m limit --limit 1/s
--limit-burst 10 -j LOG --log-prefix "DROPPED udp: "
/sbin/iptables -A logdrop -p icmp -m limit --limit 1/s
--limit-burst 10 -j LOG --log-prefix "DROPPED icmp: "
/sbin/iptables -A logdrop -j DROP


### Accept all incoming connections already established or
### related:

/sbin/iptables -A INPUT -m state --state ESTABLISHED,
RELATED -j ACCEPT
/sbin/iptables -A FORWARD -m state --state ESTABLISHED,
RELATED -j ACCEPT


### Reject DHCP traffic from the WAN interface:

/sbin/iptables -A INPUT -i eth1 -p udp -d 255.255.255.255
--dport 67:68 -j DROP
/sbin/iptables -A FORWARD -i eth1 -p udp -d 255.255.255.255
--dport 67:68 -j DROP


### Create the 'check_tcp_flags' chain. Check TCP packets for
### strange flags combinations:
```

```
/sbin/iptables -N log_tcp_flags
/sbin/iptables -A log_tcp_flags -m limit --limit 1/s
-j LOG --log-prefix "BADFLAGS: "
/sbin/iptables -A log_tcp_flags -j DROP

/sbin/iptables -N check_tcp_flags
/sbin/iptables -A check_tcp_flags -p tcp --tcp-flags
ALL FIN,URG,PSH -j log_tcp_flags
/sbin/iptables -A check_tcp_flags -p tcp --tcp-flags
SYN,RST SYN,RST -j log_tcp_flags
/sbin/iptables -A check_tcp_flags -p tcp --tcp-flags
SYN,FIN SYN,FIN -j log_tcp_flags
/sbin/iptables -A check_tcp_flags -p tcp --tcp-flags
ALL SYN,RST,ACK,FIN,URG -j log_tcp_flags
/sbin/iptables -A check_tcp_flags -p tcp --tcp-flags
ALL ALL -j log_tcp_flags
/sbin/iptables -A check_tcp_flags -p tcp --tcp-flags
ALL NONE -j log_tcp_flags


### Create the 'check_tcp' chain:

/sbin/iptables -N check_tcp
/sbin/iptables -A check_tcp -p tcp -j check_tcp_flags
/sbin/iptables -A check_tcp -p tcp --tcp-flags SYN,ACK SYN,ACK
-m state --state NEW -j REJECT --reject-with tcp-reset
/sbin/iptables -A check_tcp -p tcp ! --syn -m state
--state NEW -j logdrop

/sbin/iptables -A INPUT -p tcp -j check_tcp

/sbin/iptables -A FORWARD -p tcp -j check_tcp


### Create the 'icmp_packets' chain:

/sbin/iptables -N icmp_packets
/sbin/iptables -A icmp_packets -i eth1 -p icmp --icmp-type
echo-request -j ACCEPT
/sbin/iptables -A icmp_packets -i eth1 -p icmp --icmp-type
```

```
ttl-zero-during-reassembly -j ACCEPT
/sbin/iptables -A icmp_packets -i eth1 -p icmp --icmp-type
ttl-zero-during-transit -j ACCEPT
/sbin/iptables -A INPUT -i eth1 -p udp --dport
33434:33523 -j ACCEPT
/sbin/iptables -A FORWARD -i eth1 -p udp --dport
33434:33523 -j ACCEPT
/sbin/iptables -A icmp_packets -p icmp --icmp-type
destination-unreachable -j ACCEPT
/sbin/iptables -A icmp_packets ! -i eth1 -p icmp -j ACCEPT
/sbin/iptables -A icmp_packets -i eth1 -p icmp --icmp-type
echo-reply -j ACCEPT
/sbin/iptables -A icmp_packets -i eth1 -p icmp --icmp-type
echo-request -j ACCEPT
/sbin/iptables -A icmp_packets -p icmp -j logdrop

/sbin/iptables -A INPUT -p icmp -j icmp_packets

/sbin/iptables -A FORWARD -p icmp -j icmp_packets


### Reject RFC 1918 IP addresses coming from the WAN interface

/sbin/iptables -A INPUT -i eth1 -s 10.0.0.0/8 -j logdrop
/sbin/iptables -A INPUT -i eth1 -s 172.16.0.0/12 -j logdrop
/sbin/iptables -A INPUT -i eth1 -s 192.168.0.0/16 -j logdrop
/sbin/iptables -A INPUT -i eth1 -s 127.0.0.0/8 -j logdrop
/sbin/iptables -A FORWARD -i eth1 -s 10.0.0.0/8 -j logdrop
/sbin/iptables -A FORWARD -i eth1 -s 172.16.0.0/12 -j logdrop
/sbin/iptables -A FORWARD -i eth1 -s 192.168.0.0/16 -j logdrop
/sbin/iptables -A FORWARD -i eth1 -s 127.0.0.0/8 -j logdrop


### Reject RFC 1918 IP addresses coming to the WAN interface

/sbin/iptables -A OUTPUT -o eth1 -s 10.0.0.0/8 -j logdrop
/sbin/iptables -A OUTPUT -o eth1 -s 172.16.0.0/12 -j logdrop
/sbin/iptables -A OUTPUT -o eth1 -s 192.168.0.0/16 -j logdrop
/sbin/iptables -A OUTPUT -o eth1 -s 127.0.0.0/8 -j logdrop
/sbin/iptables -A FORWARD -o eth1 -s 10.0.0.0/8 -j logdrop
/sbin/iptables -A FORWARD -o eth1 -s 172.16.0.0/12 -j logdrop
```

```
/sbin/iptables -A FORWARD -o eth1 -s 192.168.0.0/16 -j logdrop
/sbin/iptables -A FORWARD -o eth1 -s 127.0.0.0/8 -j logdrop


### Reject connections from inside networks which do not come
from an internal IP:

/sbin/iptables -A INPUT -i eth0 ! -s 192.168.2.0/24 -j logdrop
/sbin/iptables -A FORWARD -i eth0 ! -s 192.168.2.0/24
-j logdrop


### Reject multicasting traffic:

/sbin/iptables -A INPUT -i eth1 -d 224.0.0.0/8 -j logdrop
/sbin/iptables -A FORWARD -i eth1 -d 224.0.0.0/8 -j logdrop


### Send a TCP reply with the RST flag set with IDENT:

/sbin/iptables -A INPUT -p tcp --dport 113
-j REJECT --reject-with tcp-reset
/sbin/iptables -A FORWARD -p tcp --dport 113
-j REJECT --reject-with tcp-reset


### Accept loopback traffic:

/sbin/iptables -A INPUT -i lo -j ACCEPT


### Interconnecions between all the interfaces:

## Interconnections between 'lan' and the firewall box.
## Policy: deny

/sbin/iptables -A INPUT -i eth0 -j logdrop


## Interconnections between 'lan' and 'wan'. Create the chain
## 'lan-wan'. Policy: allow
```

```
/sbin/iptables -N lan-wan
/sbin/iptables -A lan-wan  -j ACCEPT

# Chain created. Add to the FORWARD chain a rule to redirect
## here all the traffic between the two interfaces:

/sbin/iptables -A FORWARD -i eth0 -o eth1 -j lan-wan


## Interconnections between 'wan' and the firewall
## box. Policy: deny

/sbin/iptables -A INPUT -i eth1 -j logdrop


## Interconnections between 'wan' and 'lan'.
## Create the chain 'wan-lan'. Policy: deny

/sbin/iptables -N wan-lan

## Enable services for public IP addresses

# Enable service 'smtp' for the public ip
# 131.175.124.137 (mapped to 192.168.2.200):

/sbin/iptables -A wan-lan -p tcp
-d 192.168.2.200 --dport 25 -j allow
/sbin/iptables -A wan-lan -p udp
-d 192.168.2.200 --dport 25 -j allow

# Enable service 'imap' for the public
# ip 131.175.124.137 (mapped to 192.168.2.200):

/sbin/iptables -A wan-lan -p tcp
-d 192.168.2.200 --dport 143 -j allow
/sbin/iptables -A wan-lan -p udp
-d 192.168.2.200 --dport 143 -j allow

# Enable service 'pop3' for the public ip
# 131.175.124.137 (mapped to 192.168.2.200):
```

```
/sbin/iptables -A wan-lan -p tcp
-d 192.168.2.200 --dport 110 -j allow
/sbin/iptables -A wan-lan -p udp
-d 192.168.2.200 --dport 110 -j allow

# Enable service 'http' for the public ip
# 131.175.124.138 (mapped to 192.168.2.6):

/sbin/iptables -A wan-lan -p tcp
-d 192.168.2.6 --dport 80 -j allow
/sbin/iptables -A wan-lan -p udp
-d 192.168.2.6 --dport 80 -j allow

# Enable service 'https' for the public ip
# 131.175.124.138 (mapped to 192.168.2.6):

/sbin/iptables -A wan-lan -p tcp
-d 192.168.2.6 --dport 443 -j allow
/sbin/iptables -A wan-lan -p udp
-d 192.168.2.6 --dport 443 -j allow
/sbin/iptables -A wan-lan  -j logdrop

# Chain created. Add to the FORWARD chain a
# rule to redirect here all the traffic
# between the two interfaces:

/sbin/iptables -A FORWARD -i eth1 -o eth0 -j wan-lan


### NAT rules: allow natted networks to browse
## the internet (SNAT):

## SNAT for 'lan' using '131.175.124.136'

/sbin/iptables -t nat -A POSTROUTING -s 192.168.2.0/24
-o eth1 -j SNAT --to-source 131.175.124.136


### Public IP Mapping: redirect all the connections for a
## public IP to a private one inside a DMZ zone with IP
```

```
## aliasing configured (DNAT):

## DNAT for 'lan' to '192.168.2.200'

/sbin/iptables -t nat -A PREROUTING -i eth1
-d 131.175.124.137 -j DNAT --to-destination 192.168.2.200

## DNAT for 'lan' to '192.168.2.6'

/sbin/iptables -t nat -A PREROUTING -i eth1
-d 131.175.124.138 -j DNAT --to-destination 192.168.2.6

echo "Firewall started"
}


#### stop(): stop the firewall and remove all chains
#### previously created

stop() {
    /sbin/iptables -F
    /sbin/iptables -X
    /sbin/iptables -t nat -F
    /sbin/iptables -P INPUT ACCEPT
    /sbin/iptables -P OUTPUT ACCEPT
    /sbin/iptables -P FORWARD ACCEPT
    echo "Firewall stopped"
}


#### status(): Show iptables rules

status() {
    echo "
+ Table: filter
"
    /sbin/iptables -L
    echo "
+ Table: nat
"
    /sbin/iptables -t nat -L
```

```
}


#### restart(): stop and restart the firewall

restart() {
    stop
    start
}


#### Main Menu:

case "$1" in
    start) start ;;
    stop) stop ;;
    status) status ;;
    restart) stop; start ;;
    *) echo "Usage: $0 {start|stop|status|restart}" ;;
esac
```

Appendice C: alcune schermate

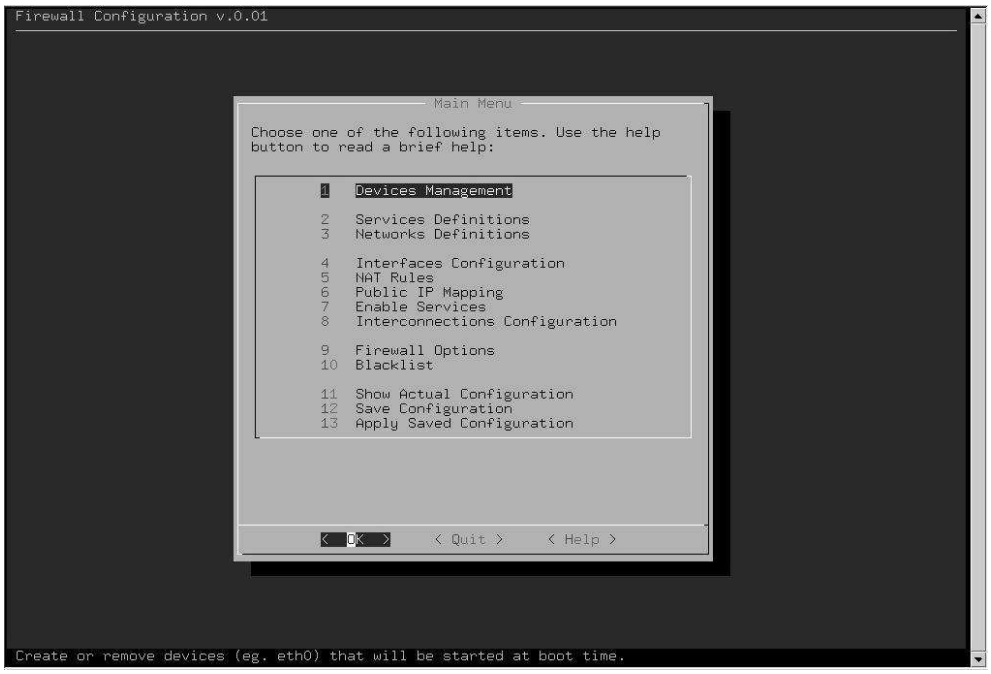

Figura 4.1: Il menu principale

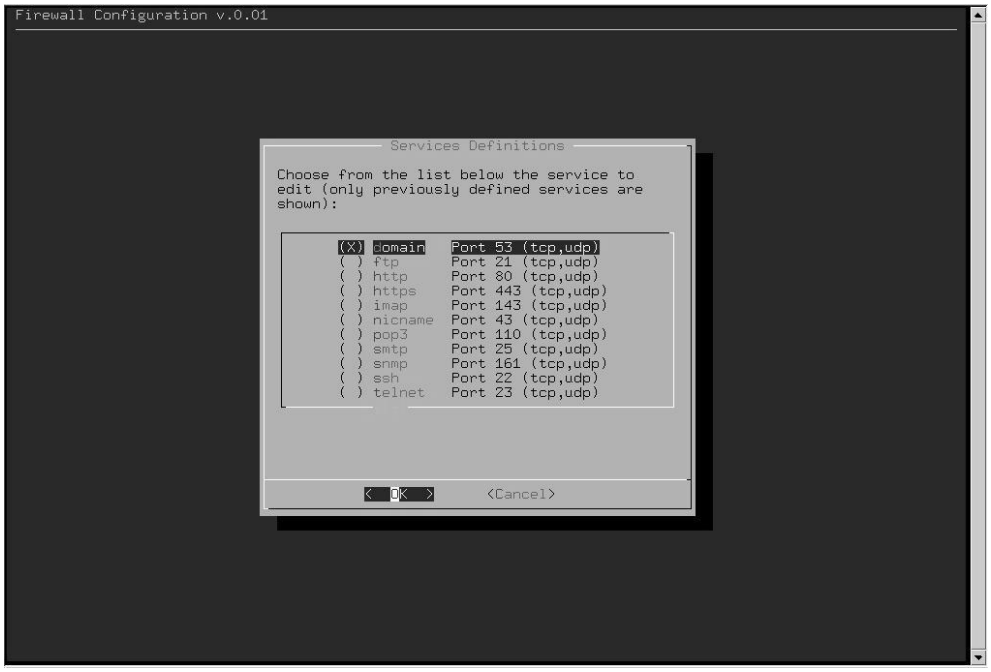

Figura 4.2: Menu per la gestione e la definizione dei servizi

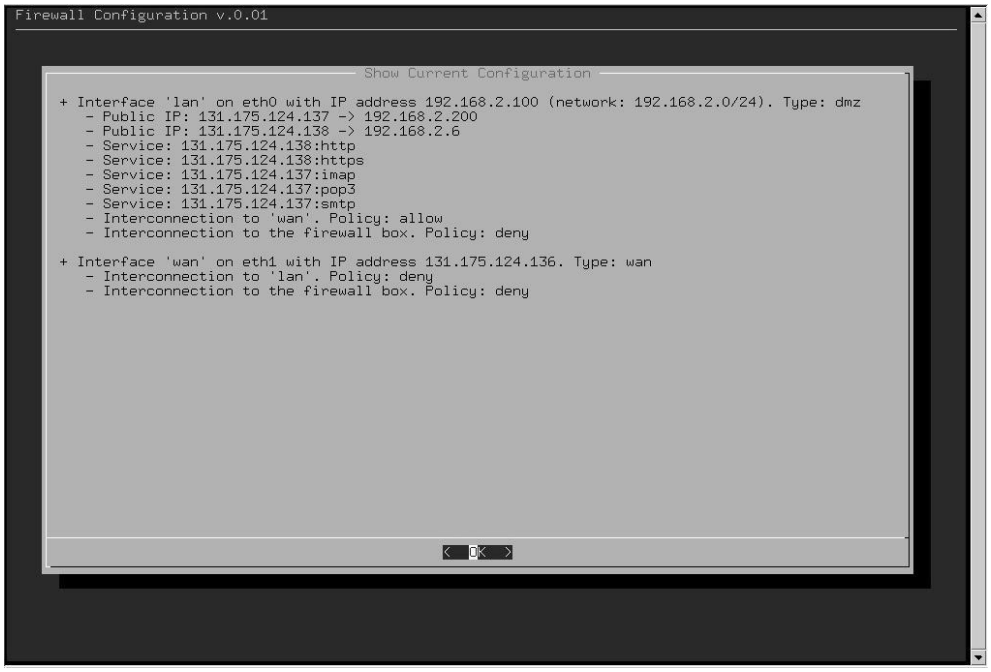

Figura 4.3: Sintesi della configurazione attuale

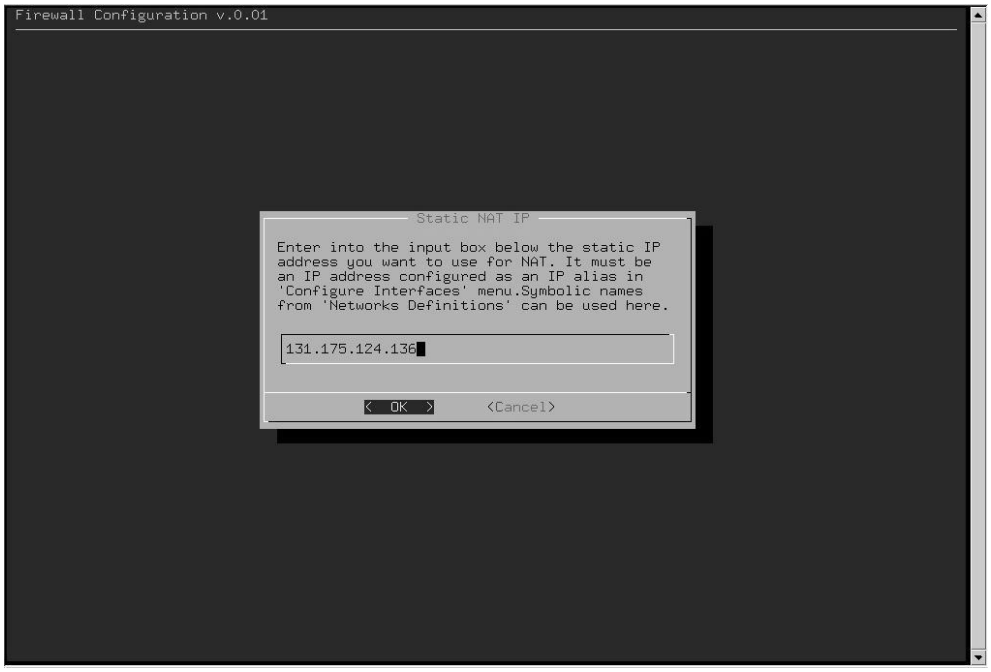

Figura 4.4: Configurazione del NAT

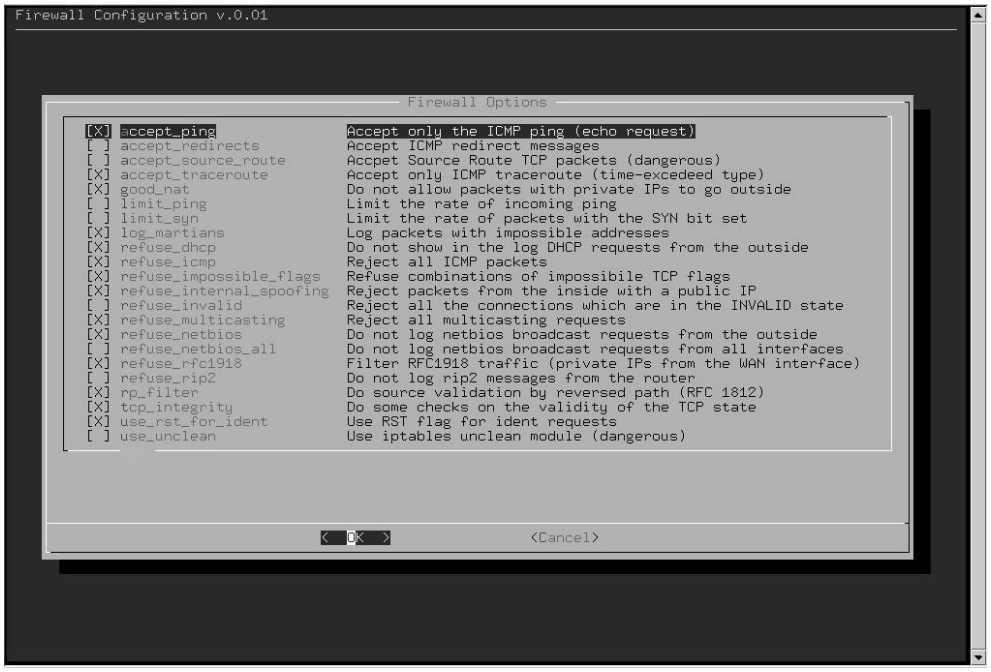

Figura 4.5: Opzioni del firewall

Bibliografia

[1] *Practical Unix Security* - Garfinkel Simon, Spafford Gene

[2] *Hacker! Linux* - Hatch, Lee, Kurtz

[3] *Internetworking with TCP/IP. Principles, Protocols and Architectures Volume 1* - Douglas E. Comer

[4] Networking Concepts HOWTO
http://www.netfilter.org/unreliable-guides/it/networking-concepts-HOWTO.html

[5] Linux Advanced Routing & Traffic Control
http://lartc.org/

[6] Packet Filtering HOWTO
http://www.netfilter.org/unreliable-guides/it/packet-filtering-HOWTO.html

[7] NAT HOWTO
http://www.netfilter.org/unreliable-guides/it/NAT-HOWTO.html

[8] Netfilter Hacking-HOWTO
http://www.netfilter.org/unreliable-guides/netfilter-hacking-HOWTO/index.html

[9] RFC 790: Assigned Numbers
http://www.faqs.org/rfcs/rfc790.html

[10] Networking Overview HOWTO
http://www.ibiblio.org/mdw/HOWTO/Networking-Overview-HOWTO.html

[11] IPTABLES HOWTO
http://www.linuxguruz.org/IPTABLES/howto/IPTABLES-HOWTO.html

[12] IP Masquerade HOWTO
http://en.tldp.org/HOWTO/IP-Masquerade-HOWTO/index.html

[13] Multicast HOWTO
http://www.tldp.org/HOWTO/Multicast-HOWTO.html

[14] IPTABLES Tutorial
http://www.linuxsecurity.com/resource_files/firewalls/IPTABLES-Tutorial/IPTABLES-tutorial.html

[15] Security Quickstart Redhat HOWTO
http://www.linuxsecurity.com/docs/LDP/Security-Quickstart-Redhat-HOWTO/index.html

[16] IP Subnetworking
http://www.tldp.org/HOWTO/mini/IP-Subnetworking.html

[17] Proxy ARP Subnet Mini HOWTO
http://www.tldp.org/HOWTO/mini/Proxy-ARP-Subnet/

[18] IP Alias Mini HOWTO
http://www.tldp.org/HOWTO/mini/IP-Alias/

[19] Network Administration Guide
http://www.tldp.org/LDP/nag/nag.html