

Università degli Studi di Padova

DIPARTIMENTO DI MATEMATICA

CORSO DI LAUREA IN INFORMATICA



**Analisi ed implementazione della sicurezza
perimetrale aziendale tramite l'utilizzo di
componenti del sistema operativo Linux**

Tesi di laurea

Relatore

Prof. Claudio Enrico Palazzi

Laureando

Campese Stefano

ANNO ACCADEMICO 2013-2014

I computer sono incredibilmente veloci, accurati e stupidi.
Gli uomini sono incredibilmente lenti, inaccurati e intelligenti.
L'insieme dei due costituisce una forza incalcolabile.

— Albert Einstein

Sommario

Il presente documento descrive il lavoro svolto durante il periodo di stage, della durata di circa trecentoventi ore, dal laureando Campese Stefano presso l'azienda *Sanmarco Informatica S.p.A.* Gli obiettivi del progetto erano molteplici. In primo luogo era richiesta la realizzazione di un *firewall* basato sulla manipolazione e il filtraggio dei pacchetti in transito nel computer. In secondo luogo era richiesta la creazione di filtri, regole di rete e di reindirizzamento NAT necessari per la creazione di un *firewall*.

“L’informatica non riguarda i computer più di quanto l’astronomia riguardi i telescopi.”

— Edsger Dijkstra

Ringraziamenti

Innanzitutto, vorrei esprimere la mia gratitudine al Prof. Claudio Enrico Palazzi, relatore della mia tesi, per l’aiuto e il sostegno fornitomi durante la stesura del lavoro.

Desidero ringraziare con affetto i miei genitori per il sostegno, il grande aiuto e per essermi stati vicini in ogni momento durante gli anni di studio.

Ho desiderio di ringraziare poi i miei colleghi ed amici per tutti i bellissimi anni passati insieme, i mille progetti e avventure vissute.

Ringrazio inoltre i colleghi che durante il periodo di Stage mi hanno aiutato a comprendere e capire il mondo delle Reti e della Sicurezza, incitandomi a portare avanti e a pubblicare il mio lavoro.

Padova, Febbraio 2014

Campese Stefano

Indice

1	Introduzione	1
1.1	Dati Aziendali	1
1.2	L'azienda	2
1.3	Visione aziendale	2
1.4	L'idea	2
1.5	Organizzazione del testo	3
2	Descrizione dello stage	5
2.1	Introduzione al progetto	5
2.2	Analisi preventiva dei rischi	5
2.3	Obiettivi	6
2.3.1	Obiettivi obbligatori	6
2.3.2	Obiettivi desiderabili	6
2.4	Pianificazione	6
2.4.1	Formazione	7
2.4.2	Progettazione e implementazione firewall	7
2.4.3	Verifica e Validazione	7
3	Tecnologie utilizzate	9
3.1	Introduzione	9
3.2	Modello TCP/IP	9
3.2.1	IP	9
3.2.2	TCP	11
3.3	Firewall	12
3.4	IPtables	15
3.4.1	Introduzione	15
3.4.2	Descrizione	15
3.5	Squid	16
3.5.1	Introduzione	16
3.5.2	Descrizione	16
3.6	L7-Filter	17
3.6.1	Introduzione	17
3.6.2	Descrizione	17
3.7	Kernel Linux	17
3.7.1	Introduzione	17
3.7.2	Descrizione	17
3.8	Software di Compilazione	18

3.9	Software di Monitoraggio	18
4	Progettazione e Scelte Tecnologiche	21
4.1	Studio dell'infrastruttura	21
4.1.1	Hardware preso in considerazione	22
4.1.2	Velocità e tipologia linea ADSL	23
4.1.3	Presenza di zone DMZ	25
4.1.4	VLAN	25
4.1.5	Connessioni VPN	25
4.1.6	Tipologia di cablatura	25
4.2	Compiti che deve svolgere il Firewall	26
4.3	Scelta di un firewall adatto	28
5	Implementazione Firewall	37
5.1	Introduzione	37
5.1.1	IPtables	37
5.1.2	Squid	39
5.2	Implementazione regole IPtables	39
5.2.1	Regole	41
5.3	Implementazione regole IPtables-Layer7	44
5.4	Implementazione Proxy tramite Squid	48
6	Verifica e validazione	53
6.1	Introduzione	53
6.2	Verifica Funzionamento IPtables	54
6.3	Verifica Funzionamento IPtables-Layer7	55
6.4	Verifica Funzionamento server proxy	55
6.5	Conclusioni Verifica	57
7	Conclusioni	59
7.1	Raggiungimento degli obiettivi	59
7.2	Conoscenze acquisite	59
7.3	Considerazioni finali	60
	Bibliografia	63

Elenco delle figure

1.1	Logo Sanmarco Informatica S.p.a.	1
3.1	Struttura Modelli OSI e TCP/IP	10
3.2	Struttura di un pacchetto IP	11
3.3	Logo Squid	16
3.4	Esempio di schermata di IFtop	19
3.5	Esempio di schermata di Htop	20
3.6	Esempio di schermata di IPtraf	20
4.1	Esempio di infrastruttura	22
4.2	Esempio di Soekris, modello 5501	24
4.3	Esempio scheda madre Soekris, modello 5501	24
4.4	Schema di collegamento VPN Client-Azienda	26
4.5	Schema di collegamento VPN <i>punto-punto</i> , ovvero vpn che permette di unire due o più sottoreti	26
4.6	Logo IPCop	30
4.7	Logo Voyage Linux	31
4.8	Logo Voyage Linux	32
4.9	Differenze dei file di configurazione del kernel in base alla CPU	33
4.10	Velocità di punta raggiunta da Soekris 4801 con linea asimmetrica	34
4.11	Velocità di punta raggiunta da Soekris 6501 con linea simmetrica	35
6.1	Regola NAT validata	54
6.2	Regole di input validate	55
6.3	Regole di forward non validate	55
6.4	Regole di blocco layer7 validate	56
6.5	Regole di blocco layer7 validate	56
6.6	Esempio di blocco siti internet da parte di Squid	57

Elenco delle tabelle

4.1	Tabella delle caratteristiche hardware dei modelli	23
4.2	Tabella rapporto connessioni VPN modello dispositivo	35
7.1	Consuntivo degli obiettivi raggiunti	59

Capitolo 1

Introduzione

Il seguente capitolo ha la funzione di elencare le informazioni generali dell'azienda presso la quale è stato svolto lo stage e di dare un'idea dello stage svolto.

1.1 Dati Aziendali

Di seguito sono riportate le informazioni sull'azienda nella quale è stato svolto il tirocinio, inoltre in figura 1.1 si può notare il logo aziendale.

Nome: Sanmarco Informatica S.p.A.

Partita IVA: 01712150240

Indirizzo: Via Vittorio Veneto, 153, Grisignano di Zocco Vicenza

Telefono: +39 0444 614464

Fax: +39 0444 419300

Email: direzionecommerciale@sanmarcoinformatica.it

Sito web: <http://www.sanmarcoinformatica.it>



figura 1.1: Logo Sanmarco Informatica S.p.a.

1.2 L'azienda

L'azienda *Sanmarco Informatica S.p.A.* è nata negli anni '80 come software house specializzata negli applicativi per aziende manifatturiere.

A partire dalla sua nascita ha lavorato a stretto contatto con le migliori aziende manifatturiere, fornendo loro, servizi e prodotti innovativi che le aiutano a crescere ed espandersi.

Lo sviluppo dei prodotti è basato sulla semplicità di utilizzo. Il software di punta sviluppato da Sanmarco Informatica è un gestionale chiamato *Galileo*.

Attorno a questo progetto sono nati altri prodotti e servizi che hanno permesso all'azienda di spaziare in più settori dell'informatica.

L'azienda, infatti, oltre allo sviluppo software si occupa della creazione, installazione e manutenzione delle infrastrutture necessarie come ad esempio Server, firewall, reti LAN e reti WAN.

1.3 Visione aziendale

Un approccio professionale, serio e metodico è ciò in cui crede l'azienda al fine di aumentare la qualità dei prodotti e dei servizi forniti.

L'obiettivo di Sanmarco Informatica è quello di rendere l'azienda il partner ideale per la consulenza, la fornitura di prodotti e servizi a supporto dei processi aziendali e professionali.

1.4 L'idea

Lo Stage è stato svolto in materia di Reti e Sicurezza.

L'idea alla base dello stage è la creazione e l'implementazione di un *firewall* mediante utilizzo di componenti del kernel del sistema operativo Linux.

La creazione del firewall consente l'intercettazione e la manipolazione dei pacchetti in transito nei computer della rete aziendale.

Nello specifico per consentire la realizzazione del progetto è stato fatto uso del programma IPtables, per permettere di definire le regole per i filtri di rete e il reindirizzamento NAT.

Lo scopo della creazione di tali regole è la sicurezza informatica aziendale dei clienti.

La sicurezza informatica, a livello aziendale, è ben più complessa e più difficile da realizzare, a causa di infrastrutture di rete spesso molto grandi e varie.

Ogni azienda nei propri server, computer e storage immagazzina dati molto impor-

tanti, come, ad esempio, la contabilità aziendale, specifiche di nuovi prodotti e dati logistici.

Questo fa capire come sia necessario avere una rete con alte misure di sicurezza in modo da bloccare possibili intrusioni dall'esterno, che potrebbero manomettere o rubare tali dati.

1.5 Organizzazione del testo

Il secondo capitolo approfondisce la descrizione dello stage descrivendone obiettivi rischi e pianificazioni

Il terzo capitolo approfondisce le tecnologie utilizzate quali IPtables, il protocollo TCP/IP, il software Squid.

Il quarto capitolo approfondisce e spiega le scelte tecnologiche da perseguire nella scelta e nella progettazione di un firewall.

Il quinto capitolo tratta l'implementazione e la creazione di un firewall di Livello 7 del modello OSI.

Il sesto capitolo approfondisce e illustra i test eseguiti dopo aver terminato la creazione del firewall, mostrando in oltre la differenza di prestazioni tra firewall diversi.

Nel settimo capitolo descrive il raggiungimento obiettivi, conoscenze acquisite e la valutazione personale al termine dello stage.

Capitolo 2

Descrizione dello stage

In questo capitolo verrà descritto il progetto dello stage, i rischi preventivati, gli obiettivi prefissati e la pianificazione

2.1 Introduzione al progetto

Il progetto consiste nella creazione di un firewall mediante alcuni moduli del kernel del sistema operativo Linux. I moduli utilizzati permetteranno l'intercettazione e la manipolazione dei pacchetti in transito tra i computer della rete. L'obiettivo è dunque, la realizzazione di firewall basati sul filtraggio dei pacchetti.

Nello specifico verrà fatto uso del programma IPtables, per permettere di definire le regole, per i filtri di rete e per il reindirizzamento NAT.

La realizzazione del firewall dovrà permettere la manipolazione dei pacchetti a livello 7 del modello OSI, ovvero a livello applicativo.

Il firewall dovrà riconoscere e manipolare i pacchetti generati da determinate applicazioni, permettendo di eliminare, modificare o accettare i pacchetti generati.

La realizzazione del modulo kernel adeguato alla manipolazione dei pacchetti a livello applicativo sfrutterà l'esistenza di un progetto Open Source, ormai abbandonato, denominato: L7-filter

2.2 Analisi preventiva dei rischi

Durante la fase di analisi iniziale sono stati individuati alcuni possibili rischi a cui si potrà andare incontro. Si è quindi proceduto a elaborare delle possibili soluzioni per far fronte a tali rischi.

1. Documentazione L7-filter insufficiente

Descrizione: La documentazione per la realizzazione del modulo del kernel che permette il riconoscimento del livello applicativo risulta essere decisamente insufficiente e non aggiornata.

Soluzione: Utilizzo di libri di testo come: *“Designing and Implementing Linux*

firewalls and QoS using NetFilter, iproute2, NAT, and L7-filter” il testo non risulta essere aggiornato ma le spiegazioni sono ottimali.

2. Possibili rotture Hardware

Descrizione: L’hardware del firewall potrebbe danneggiarsi a causa del carico di lavoro sulla CPU e delle continue letture/scritture effettuate sulla memoria CompactFlash del firewall .

Soluzione: Utilizzo di kernel compilati appositamente per la CPU e utilizzo di CompactFlash nuove.

3. Possibili comunicazione firewall

Descrizione: Il firewall potrebbe comunicare in modo errato all’interno della rete rendendo così i test falsi.

Soluzione: Cambio cablatura e aumento della banda disponibile per il firewall.

2.3 Obiettivi

Gli obiettivi prefissati sono molteplici, e si suddividono in due categorie; obiettivi obbligatori e obiettivi desiderabili.

2.3.1 Obiettivi obbligatori

Il raggiungimento di questi obiettivi, al termine dello stage, è obbligatorio. Gli obiettivi in questione sono:

- * apprendimento NAT
- * apprendimento IPtables
- * sviluppo di strumenti firewall
- * sviluppo di strumenti firewall basati su livello 7 della pila OSI
- * implementazione regole firewall per la sicurezza aziendale

2.3.2 Obiettivi desiderabili

Il raggiungimento di questi obiettivi, al termine dello stage, non risulta essere obbligatorio, ma comunque il raggiungimento, è consigliato. Gli obiettivi in questione sono:

- * configurazione di regole firewall NAT
- * configurazione server Proxy
- * Configurazione reti WAN e VPN

2.4 Pianificazione

La durata dell’attività di stage è fissata a 320 ore. Al fine di essere il più efficiente possibile l’intera durata dello stage, è stata pianificata in tre periodi.

2.4.1 Formazione

Questo periodo è suddiviso in tre sottoperiodi:

- * formazione di base Linux
- * formazione di base NAT
- * formazione approfondita su NAT e IPtables

La formazione è stata effettuata grazie al materiale fornito in dall'azienda, e tramite colloquio con i colleghi e con il tutor aziendale e tramite l'utilizzo dei manuali dei tools utilizzati.

La durata complessiva di questa fase dello stage è stata di 120 ore.

2.4.2 Progettazione e implementazione firewall

Questo periodo risulta essere stato il più lungo di tutto lo stage. La progettazione e l'implementazione richiedono tempistiche piuttosto lunghe. In questo periodo è stato pensato, progettato e realizzato il firewall. la durata totale di tale periodo è di 190 ore.

2.4.3 Verifica e Validazione

Il periodo di verifica ha una durata di 10 ore. Durante questo periodo viene testato il firewall con le sue funzionalità.

Se il firewall passa i test allora verrà validato.

Capitolo 3

Tecnologie utilizzate

In questo capitolo verranno descritte tutte le tecnologie utilizzate durante lo svolgimento dello stage.

3.1 Introduzione

In questo capitolo saranno descritte tutte le tecnologie di cui si è reso necessario l'utilizzo durante lo stage. Le tecnologie descritte sono sia hardware, come la Soekris, che software, come ad esempio IPtables.

3.2 Modello TCP/IP

Il modello TCP/IP, assieme al modello OSI è una delle architetture di rete più utilizzate al mondo. Queste due architetture di rete sono strutturate in livelli. Il modello OSI è strutturato in 7 livelli, mentre il modello TCP/IP ne ha 5.

La comunicazione tramite TCP/IP avviene mediante lo scambio di pacchetti informativi. Un esempio della suddivisione in livelli dei modelli OSI e TCP è rappresentata in figura [3.1](#)

Il protocollo TCP/IP è l'unione di due protocolli distinti:

- * TCP
- * IP

3.2.1 IP

Introduzione

Il protocollo IP fa parte del livello internet e della serie di protocolli TCP/IP. È uno dei protocolli più importanti di internet dato che permette l'elaborazione e il trasporto dei datagrammi IP, chiamati anche pacchetti, senza tuttavia assicurarne la consegna.

Il protocollo IP tratta i pacchetti IP indipendentemente gli uni dagli altri definendo la loro rappresentazione, il loro routing e la loro spedizione. in figura [3.2](#) è

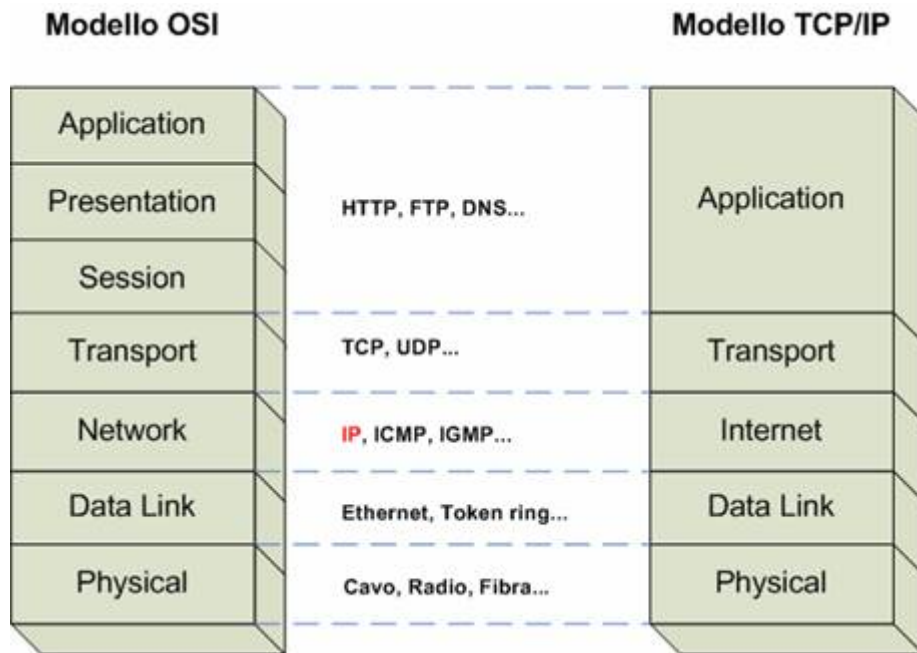


figura 3.1: Struttura Modelli OSI e TCP/IP

disponibile un esempio di pacchetto IP. Tale protocollo di interconnessione è classificato al livello di rete 3 del modello OSI e risulta essere nato per interconnettere reti eterogenee.

IP è un protocollo a pacchetto senza connessione e di tipo di *best effort* nel senso che fa il massimo di quello che può fare senza garantire alcuna forma di affidabilità della comunicazione in termini di controllo di errore, controllo di flusso e controllo di congestione a cui quindi dovranno supplire i protocolli di trasporto di livello superiore come ad esempio TCP.

La versione correntemente usata del protocollo IP è detta anche IPv4 per distinguerla dalla più recente IPv6, nata dall'esigenza di gestire meglio il crescente numero di computer connessi ad Internet.

Descrizione

Il principale compito di IP è l'Indirizzamento e l'instradamento dei pacchetti internet tra sottoreti eterogenee, che a livello locale utilizzano invece un indirizzamento proprio, tipicamente basato sull'indirizzo fisico o indirizzo MAC e protocolli di livello datalink del modello ISO-OSI.

Per far ciò è necessario assegnare un nuovo piano di indirizzamento a cui tutte le sottoreti devono sottostare per poter comunicare e interoperare tra loro: tale piano è rappresentato proprio dal Protocollo IP.

Questo comporta l'assegnazione a ciascun terminale che ne fa richiesta ,cioè si

32 bits			
Versione (4 bit)	Lunghezza dell'intestazione (4 bit)	Tipo di servizio (8 bit)	Lunghezza totale (16 bit)
Identificazione (16 bit)			<div>Flag (3 bit)</div> <div>Spostamento sezione (13 bit)</div>
Durata di vita (8 bit)		Protocollo (8 bit)	Somma di controllo intestazione (16 bit)
Indirizzo IP sorgente (32 bit)			
Indirizzo IP destinazione (32 bit)			
Dati			

figura 3.2: Struttura di un pacchetto IP

connette alla rete Internet, di un nuovo diverso indirizzo, univocamente associato all'indirizzo MAC locale, detto Indirizzo IP. L'assegnazione degli indirizzi avviene tramite i protocolli RARP, BOOTP o DHCP.

In sostanza dunque IP rappresenta la “colla” che unisce tra loro tutte le varie sottoreti, diverse tra loro, a livello di indirizzamento/instradamento, permettendone il dialogo o lo scambio di informazioni.

3.2.2 TCP

Introduzione

Il protocollo TCP definito anche come *Transmission Control Protocol* o *Transfer Control Protocol*, è un protocollo di rete con dati a pacchetto.

TCP appartiene alla suite di protocolli Internet, come ad esempio, TCP, UDP, IP, ICMP e ARP.

Questo protocollo è stato progettato appositamente per fornire un flusso di byte affidabile, rendendo quindi affidabile la comunicazione dati in rete tra un mittente e un destinatario.

Il protocollo TCP permette di gestire i dati provenienti dal livello inferiore ovvero dal protocollo IP.

TCP è presente solo sui terminali di rete, ovvero sui client, e non sui nodi interni di commutazione, ovvero switch e router. Questo protocollo viene implementato come software di rete all'interno del sistema operativo dei client.

Descrizione

Il TCP può essere classificato al livello trasporto (OSI level 4) del modello OSI, e di solito è usato in combinazione con il protocollo di livello rete (OSI level 3), ovvero IP.

La corrispondenza con il modello OSI non è perfetta, in quanto il TCP e l'IP nascono prima di questo modello.

Spesso la combinazione dei protocolli TCP e IP è indicata come TCP/IP e, viene erroneamente considerata un unico protocollo, motivo per cui si è scelto di spiegarli separatamente.

TPC, come già spiegato in precedenza, è nato con l'intento di superare i problemi della mancanza di affidabilità e controllo della comunicazione. Le comunicazioni di rete, come ad esempio le interconnessioni di reti locali in un'unica grande rete geografica, infatti avvengono mediante l'utilizzo del protocollo IP.

TCP, dunque è stato progettato e realizzato per utilizzare i servizi offerti dai protocolli di rete di livello inferiore, quali IP e protocolli di livello fisico e livello datalink, che definiscono in modo efficace il modo di trasferimento sul canale di comunicazione, ma che non offrono alcuna garanzia di affidabilità sulla consegna in termini di ritardo, perdita ed errore dei pacchetti informativi trasmessi.

Inoltre alcune funzionalità di TCP sono vitali per il buon funzionamento complessivo di una rete IP. Sotto questo punto di vista TCP può essere considerato come un elemento di rete che si occupa di garantire una qualità di servizio minima su una rete IP.

Il TCP nacque nel 1970 come frutto del lavoro di un gruppo di ricerca del dipartimento di difesa statunitense. I suoi punti di forza sono l'alta affidabilità e robustezza.

3.3 Firewall

Introduzione

In informatica, nell'ambito delle reti di computer, un firewall è un componente passivo di difesa perimetrale di una rete informatica, che può anche svolgere funzioni di collegamento tra due o più tronconi di rete, garantendo dunque una protezione in termini di sicurezza informatica della rete stessa.

Usualmente la rete viene divisa in due sottoreti: una, detta esterna, comprende l'intera Internet mentre l'altra interna, detta LAN (Local Area Network), comprende una sezione più o meno grande di un insieme di computer client locali.

In alcuni casi è possibile che si crei l'esigenza di creare una terza sottorete detta DMZ (o zona demilitarizzata) adatta a contenere quei sistemi che devono essere isolati dalla rete interna, ma che devono comunque essere protetti dal firewall ed essere raggiungibili dall'esterno (server pubblici) come per esempio dei server FTP o di posta.

Descrizione

La miglior definizione di firewall risulta la seguente:

Apparato di rete hardware o software di ingresso-uscita bidirezionale che, opportunamente configurato o settato e agendo in maniera centralizzata, filtra tutti i pacchetti entranti ed uscenti, da e verso una rete o un computer, secondo regole prestabilite che contribuiscono alla sicurezza della stessa.

Un firewall può essere realizzato con un semplice computer (con almeno due schede di rete, una per l'input l'altra per l'output, e software apposito), può essere una funzionalità logica (software) inclusa in un router oppure può essere implementato su un apparato hardware dedicato.

Esistono inoltre i cosiddetti "firewall personali", che sono programmi installati sui normali calcolatori client, che filtrano solamente i pacchetti che entrano ed escono da quel calcolatore, utilizzando in tal caso sola una scheda di rete.

La funzionalità principale in sostanza è quella di creare un filtro sulle connessioni entranti ed uscenti, innalzando il livello di sicurezza della rete, permettendo sia agli utenti interni che a quelli esterni di operare nel massimo della sicurezza.

Un firewall infatti agisce sui pacchetti in transito da e verso la rete interna permettendo di eseguire operazioni di:

- * controllo
- * modifica
- * monitoraggio

Queste operazioni avvengono mediante "l'apertura" del pacchetto IP e leggendo le informazioni presenti sul suo *header*, e in alcuni casi anche sul suo contenuto o *payload*. Solitamente questa tecnica di manipolazione dei pacchetti è definita come *stateful inspection*. Esiste anche una tecnica di manipolazione definita come *deep inspection* che oltre a leggere l'header dei pacchetti ne sfrutta altre proprietà.

Firewall custom e personalizzati

I firewall possono essere di molteplice fattura, possono essere hardware o software custom o personalizzati.

La differenza tra un firewall hardware e un firewall software è molto poca; solitamente, infatti un “firewall software” è un programma che esegue su un calcolatore che non è dedicato esclusivamente alla sicurezza, come per esempio un PC Desktop. Un firewall hardware invece, corrisponde ad un calcolatore che esegue un “firewall software” e che risulta dedicato interamente all’attività di sicurezza perimetrale della rete.

Ben diversa risulta la differenza tra “firewall custom” e “firewall personalizzato”. Un “firewall custom” è un firewall venduto da un produttore di tecnologie informatica, per esempio *Endian* (<http://www.endian.com/en/>) il quale ha determinate caratteristiche e non risulta modificabile. I punti di forza sono:

- * semplicità di utilizzo, l’interfaccia grafica è user friendly
- * facile installazione
- * supporto hardware e software completo
- * esistenza di community di supporto e piccole modifiche

i punti a sfavore invece risultano essere:

- * inesistente flessibilità di utilizzo, se cambiasse l’infrastruttura, probabilmente, sarebbe necessario cambiare firewall.
- * difficoltà di personalizzazione, lo strumento viene venduto con delle caratteristiche standard.
- * rilascio di aggiornamenti poco frequente, l’azienda produttrice preferisce sviluppare nuovi modelli che aggiornare quelli già esistenti, garantendosi un guadagno maggiore.
- * costo molto elevato, garantisce guadagni minimi.

Un “firewall personalizzato” invece ha caratteristiche ben diverse. I punti di forza sono:

- * maggior flessibilità e adattabilità all’infrastruttura.
- * sistema operativo open source, garantisce di modificare software senza problemi di licenze.
- * ampia scelta del sistema operativo utilizzabile.
- * hardware personalizzabile in base all’infrastruttura
- * aggiornamenti rilasciati ogni sei mesi.
- * abbattimento del costo di vendita e quindi un maggior guadagno

i punti a sfavore invece risultano essere:

- * supporto hardware separato dal supporto software
- * preparazione preventiva per l’installazione
- * conoscenze di sistemi operativi open source
- * minore usabilità, non è presente interfaccia grafica; si usa la linea di comando

3.4 IPtables

3.4.1 Introduzione

Per poter spiegare cos'è IPtables è necessario prima dare una definizione di NetFilter. NetFilter è un componente del kernel del sistema operativo Linux, che permette l'intercettazione e manipolazione dei pacchetti che attraversano il computer.

Tale modulo permette di realizzare alcune funzionalità di rete avanzate come la realizzazione di firewall basata sul filtraggio dei pacchetti, configurazioni anche complesse di NAT, creazione sistemi di traduzione automatica degli indirizzi IP, tra cui la condivisione di un'unica connessione Internet tra diversi computer di una rete locale, o ancora la manipolazione dei pacchetti in transito.

Per gestire NetFilter si usa il software IPtables, che permette di definire le regole per i filtri di rete.

3.4.2 Descrizione

Spesso con il termine IPtables ci si riferisce all'intera infrastruttura, incluso NetFilter.

IPtables, dunque, non è altro che un'applicazione che permette di configurare regole sul firewall di Linux mediante delle tabelle, le quali funzionano con delle catene di regole che sono chiamate *chains*.

IPtables permette di monitorare lo stato delle connessioni e in base a questo decidere se fare un reindirizzamento del pacchetto, modificarlo oppure fermarlo. NetFilter e IPtables sono delle componenti standard di tutte le moderne distribuzioni di Linux.

Entrambi questi moduli furono introdotti nel kernel Linux, a partire dalla versione 2.4.

Ogni kernel Linux ha i due moduli, NetFilter e IPtables che sono prestabiliti, tutta via nulla vieta di sostituirli e modificarli con versioni più meno aggiornate garantendo un'ottima flessibilità di utilizzo.

IPtables è progettato per poter essere facilmente esteso attraverso moduli che aggiungono funzionalità, come ad esempio:

- * predicati per identificare i pacchetti
- * operazioni da applicare ai pacchetti
- * supporto al protocollo NAT
- * analisi del traffico a livello applicativo

Ciascuna estensione può essere implementata come un modulo del kernel Linux, e fornisce specifiche ed obiettivi aggiuntivi.

3.5 Squid

3.5.1 Introduzione

Squid è un software open source con funzionalità di proxy e web cache, rilasciato sotto la GNU General Public License. Il software è reperibile al seguente indirizzo: <http://www.squid-cache.org/>.

In figura 3.3 è presente il logo di Squid.

Prima di proseguire è necessario precisare che le caratteristiche dettagliate di questo programma saranno elencate e spiegate in sezione 4.2; la cosa si è resa necessaria in quanto secondo l'autore aiuta a capire e comprendere il capitolo successivo.

3.5.2 Descrizione

Squid supporta la maggior parte dei protocolli di comunicazione tra i quali spiccano:

- * HTTP
- * HTTPS
- * FTP
- * TLS
- * SSL

Vale la pena segnalare che Squid, funziona sulla maggior parte dei sistemi operativi attualmente esistenti e utilizzati, questo lo rende un'ottima scelta poiché risulta avere una portabilità eccezionale. Ha una vasta varietà di usi, come ad esempio: rendere più veloce un server web usando una cache per richieste ripetute, fornendo sia un servizio di cache per il web che per i DNS e altri tipi di ricerche e il filtraggio del traffico all'interno della rete.



figura 3.3: Logo Squid

3.6 L7-Filter

3.6.1 Introduzione

L7-filter è un “classificatore” di pacchetti di comunicazione per Linux, più precisamente è un classificatore sfruttato dai moduli NetFilter e IPtables. A differenza di molti altri classificatori, non guarda i semplici valori dei pacchetti in transito, come ad esempio numeri di porta o gli indirizzi IP di destinazione e di origine.

L7-Filter, infatti, esegue la scansione sui dati del livello applicativo (livello 7 OSI) contenuti nei pacchetti di comunicazione, e mediante l'utilizzo di precise *espressioni regolari*, permette di riconoscere e determinare quali protocolli e quali applicazioni vengono utilizzati per la generazione dei pacchetti.

3.6.2 Descrizione

Il progetto L7-filter è nato nel 2003, il sito ufficiale (<http://l7-filter.sourceforge.net/>) contiene tutte le versioni del progetto.

Al momento dello stage l'ultima versione rilasciata risaliva al 13 Luglio 2009 con supporto garantito al kernel 2.6.30.

Tuttavia è stata rilasciata una nuova versione l'8 Ottobre 2013 che estende il supporto al kernel 2.6.35.

L7-Filter è stato sviluppato seguendo due diversi progetti:

- * creazione di moduli per il kernel tramite la modifica di NetFilter e IPtables e soprattutto tramite la ricompilazione del kernel
- * sviluppo di applicativo in grado di eseguire nell'*userspace* del kernel

A differenza del primo progetto il secondo è nato il primo Gennaio 2006 ed è stato sviluppato per più tempo poiché, a differenza del primo, non richiedeva la compilazione del kernel e la modifica dei sorgenti dei moduli sopracitati risultando quindi più semplice da sviluppare e più appetibile agli utilizzatori.

Attualmente tale progetto è stato rinominato in *ClearFoundation* il cui sito ufficiale risulta essere questo: <http://l7-filter.clearfoundation.com/>.

3.7 Kernel Linux

3.7.1 Introduzione

Linux è un kernel distribuito con licenza GNU General Public License; è stato creato nel 1991 dal grande Linus Torvalds. Integrato con il Sistema GNU, sviluppato da Richard Stallman, ha dato vita al sistema operativo GNU/Linux, chiamato solamente Linux.

3.7.2 Descrizione

Il kernel è il “cuore” di un sistema operativo e fornisce tutte le funzioni essenziali per il sistema, in particolare la gestione della memoria primaria, delle risorse hardware

del sistema e delle periferiche, assegnandole di volta in volta ai processi in esecuzione.

La controparte del kernel è la shell, ovvero l'interfaccia utente del sistema, la parte più esterna. I programmi chiedono le risorse al kernel attraverso delle chiamate di sistema o *system call* e non possono accedere direttamente all'hardware. Il kernel si occupa quindi di gestire il tempo processore, le comunicazioni e la memoria distribuendole ai processi in corso a seconda delle priorità (scheduling) realizzando così il multitasking.

L'architettura scelta da Torvalds per il kernel è *monolitica*. Attualmente Linux supporta gran parte dell'hardware disponibile per PC e supporta un numero non indifferente di architetture hardware tra le quali spiccano:

- * SPARC
- * PowerPC
- * ARM
- * x86
- * x64

Il codice sorgente di Linux è disponibile a tutti, è ampiamente personalizzabile, al punto da rendere possibile, in fase di compilazione, l'esclusione di codice non strettamente indispensabile. La flessibilità di questo kernel risulta essere il suo punto di forza.

3.8 Software di Compilazione

Per la compilazione dei moduli e del kernel personalizzato è necessario disporre degli strumenti corretti quali:

- * sistema operativo Linux, preferibilmente con architettura x86.
- * compilatore GNU C++ con versione ≥ 4.2
- * pacchetto *build-essential make* necessario alla preparazione dell'ambiente di configurazione del kernel del firewall
- * la libreria *libncurses5-dev* necessaria anch'essa alla preparazione dell'ambiente di configurazione del kernel del firewall

Tutti i pacchetti citati in precedenza sono open source e liberamente installabili in qualsiasi piattaforma Linux. Tuttavia per ottenere una maggior efficienza e compatibilità è necessario usare un sistema Linux di derivazione *Debian*.

3.9 Software di Monitoraggio

Per il monitoraggio delle reti e del firewall sono stati utilizzati fondamentalmente tre software:

IFtop: è un software di *monitoring* del sistema da shell. IFtop permette inoltre di monitorare le connessioni di rete, mostrando tutte le connessioni di rete attive in quel determinato istante.

Le connessioni vengono mostrate in ordine di consumo di banda. IFtop permette inoltre di mostrare l'utilizzo della CPU del firewall associato alle connessioni. In figura 3.4 è possibile vedere un esempio di come si presenta il software.



figura 3.4: Esempio di schermata di IFtop

Htop: è un *system monitoring* interattivo scritto per Linux. È stato progettato per sostituire il programma Unix top. Il programma mostra un elenco frequentemente aggiornato dei processi in esecuzione sul calcolatore; l'elenco normalmente è ordinato secondo l'utilizzo della CPU. Htop fornisce informazioni su quantità di CPU, RAM e SWAP utilizzate. In figura 3.5 è possibile vedere un esempio di come si presenta il software.

IPtraf: è un software basato su console che fornisce statistiche sulla rete. Permette di raccogliere informazioni sui dati generati dai protocolli TCP, IP, UDP, ICMP, ARP che stanno attraversando la rete in quel determinato istante. IPtraf inoltre permette di elaborare statistiche sulla velocità e sulla quantità di pacchetti trasmessi all'interno della rete, il software inoltre permette di monitorare il traffico in transito attraverso un'unica interfaccia di rete. In figura 3.6 è possibile vedere un esempio di come si presenta il software.

```

CPU[|||||||] 39.3% Tasks: 18 total, 1 running
Mem[|||||] 11/123MB Load average: 0.01 0.21 0.25
Swp[ ] 0/0MB Uptime: 00:12:00

  PID USER   PRI  NI  VIRT   RES   SHR  S  CPU% MEM%   TIME+  Command
 4112 root    20   0  2276  1140   924  R   8.9   0.9   0:05.79 htop
 4094 root    20   0  7936  2624  2168  S   0.0   2.1   0:01.29 sshd: root@pts/0
 4097 root    20   0  2844  1580  1188  S   0.0   1.2   0:00.22 -bash
 3899 ntp      20   0  4052  1236   952  S   0.0   1.0   0:01.01 /usr/sbin/ntpd -p /var/
 4103 root    20   0  7936  2628  2168  S   0.0   2.1   0:01.13 sshd: root@pts/1
 3930 root    -2   0  1688  1688  1420  S   0.0   1.3   0:00.16 /usr/sbin/watchdog
 3794 root    20   0  4196  1060   732  S   0.0   0.8   0:00.52 /usr/sbin/openvpn --wri
 3662 root    20   0  1876   680   568  S   0.0   0.5   0:02.47 /usr/sbin/syslogd --no-
 3711 dnsmasq 20   0  2196   588   460  S   0.0   0.5   0:00.16 /usr/sbin/dnsmasq -u dn
   1 root    20   0  1976   692   592  S   0.0   0.5   0:05.77 init [2]
 1619 root    16  -4  2088   800   484  S   0.0   0.6   0:04.77 udevd --daemon
 3533 daemon   20   0  1760   488   400  S   0.0   0.4   0:00.00 /sbin/portmap
 3818 root    20   0  5192  1012   656  S   0.0   0.8   0:00.01 /usr/sbin/sshd
 3892 root    20   0  1804   584   492  S   0.0   0.5   0:00.01 /usr/sbin/pppd
 3919 root    20   0  2032   788   640  S   0.0   0.6   0:00.02 /usr/sbin/cron
 3953 root    20   0  2488  1208   960  S   0.0   1.0   0:01.40 /bin/login --
 4067 root    20   0  2852  1584  1180  S   0.0   1.2   0:02.36 -bash
 4106 root    20   0  2844  1580  1188  S   0.0   1.2   0:00.14 -bash

F1Help F2Setup F3Search F4Invert F5Tree F6SortBy F7Nice F8Nice F9Kill F10Quit

```

figura 3.5: Esempio di schermata di Htop

```

IPTraf
TCP Connections (Source Host:Port) ----- Packets ----- Bytes ----- Flags ----- Iface -----
192.168.1.153:39705 > 6 695 --A- eth0
31.13.64.81:443 > 6 1823 -PA- eth0
192.168.1.152:54791 > 2 194 --A- eth0
192.168.1.153:17500 > 2 252 -PA- eth0
192.168.1.153:54848 > 3 246 --A- eth0
192.168.1.152:17500 > 2 252 -PA- eth0
108.160.162.50:80 > 2 283 --A- eth0
192.168.1.153:37013 > 1 1181 -PA- eth0
173.194.40.21:443 > 1 111 -PA- eth0
192.168.1.153:34012 > 1 52 --A- eth0
173.252.113.17:443 > 1 90 -PA- eth0
192.168.1.153:46401 > 1 52 --A- eth0
TCP: 7 entries ----- Active -----

UDP (510 bytes) from 192.168.1.153:17500 to 255.255.255.255:17500 on eth0
UDP (510 bytes) from 192.168.1.153:17500 to 192.168.1.255:17500 on eth0
UDP (68 bytes) from 192.168.1.153:5353 to 224.0.0.251:5353 on eth0
UDP (510 bytes) from 192.168.1.152:17500 to 255.255.255.255:17500 on eth0
UDP (510 bytes) from 192.168.1.152:17500 to 192.168.1.255:17500 on eth0
Elapsed Time: 0:00
Pkts captured (all interfaces): 38 | TCP flow rate: 0.00 kbits/s
Up/Dn/PgUp/PgDn-scroll M-more TCP info W-chg actv win S-sort TCP X-exit

```

figura 3.6: Esempio di schermata di IPtraf

Capitolo 4

Progettazione e Scelte Tecnologiche

In questo capitolo verranno elencate le modalità di studio dell'infrastruttura nella quale dovrà lavorare il firewall, i compiti che lui stesso dovrà svolgere e inoltre verrà eseguito un confronto tra firewall.

4.1 Studio dell'infrastruttura

Introduzione

Lo studio dell'infrastruttura è indispensabile per la realizzazione, implementazione e il funzionamento di un firewall. Gli aspetti da valutare sono molti e diversificati. Come prima cosa è necessario essere a conoscenza dell'infrastruttura nella quale il firewall andrà ad operare. Per infrastruttura di rete, si intende, tutto ciò che permette ai vari dispositivi all'interno della rete, di comunicare e scambiare dati tra loro.

Partendo da tale definizione capiamo come l'infrastruttura sia dunque *creata* dall'unione di:

- * modem
- * router
- * switch
- * access point
- * server
- * client
- * tutti i dispositivi che utilizzano la rete per comunicare quali ad esempio stampanti.

La quantità di dispositivi e la qualità della rete influenzano notevolmente la creazione dell'infrastruttura e la scelta del firewall. Un esempio di infrastruttura è evidenziato dalla figura [4.1](#).

Il sistema operativo che viene installato su tale firewall è, quasi sempre, una distribuzione Linux e molto spesso di derivazione *Debian*. La maggior parte di tali sistemi operativi sono sprovvisti di interfaccia grafica e sfruttano solo la linea di comando ed è stato sviluppato appositamente per eseguire su sistemi embedded con scarse risorse hardware.

I tre modelli presi in considerazione rispondono al nome di:

- * Soekris 4801
- * Soekris 5501
- * Soekris 6501

di seguito troviamo le caratteristiche hardware dei vari modelli.

Modelli	CPU	RAM	Numero Ethernet
Soekris 4801	AMD MGEODE GX1 266 Mhz	DDR 128 MByte	3 porte Ethernet 10/100 Mb/s, espandibile a 6
Soekris 5501	AMD MGEODE LX 666 Mhz	DDR 512 MByte	4 porte Ethernet 10/100/1000 Mb/s espandibile a 12
Soekris 6501	Intel CoreDuo 1200 Mhz	DDR2 1024 MByte	6 porte Ethernet 10/100/1000/10000 Mb/s espandibile a 12

tabella 4.1: Tabella delle caratteristiche hardware dei modelli

Dalla tabella è immediato notare come le caratteristiche hardware siano limitate, nonostante ciò tali calcolatori risultano essere perfetti per la creazione di firewall, poiché sono pensati e costruiti per lavorare ventiquattro ore al giorno senza la necessità di manutenzione.

Nelle figure 4.2 e 4.3 è possibile vedere come sono fatti questi dispositivi.

In figura 4.2 è possibile osservare una Soekris completa, mentre in figura 4.3 è riportata la sola scheda madre.

4.1.2 Velocità e tipologia linea ADSL

La velocità della linea ADSL, al contrario di quanto si possa pensare, influenza di molto la scelta del firewall, questo perché i firewall hanno CPU con basse frequenze di lavoro. Una maggiore velocità della linea implica un maggior mole di dati da filtrare e di conseguenza un maggior carico sulla CPU.

Il primo aspetto da osservare è la tipologia della linea che può essere asimmetrica o simmetrica:

- * **linea asimmetrica:** è una linea ADSL in cui il *download* e l'*upload* hanno una velocità diversa, solitamente la velocità di *download* risulta maggiore.
- * **linea simmetrica:** è una linea ADSL in cui il *download* e l'*upload* hanno una la stessa velocità.



figura 4.2: Esempio di Soekris, modello 5501

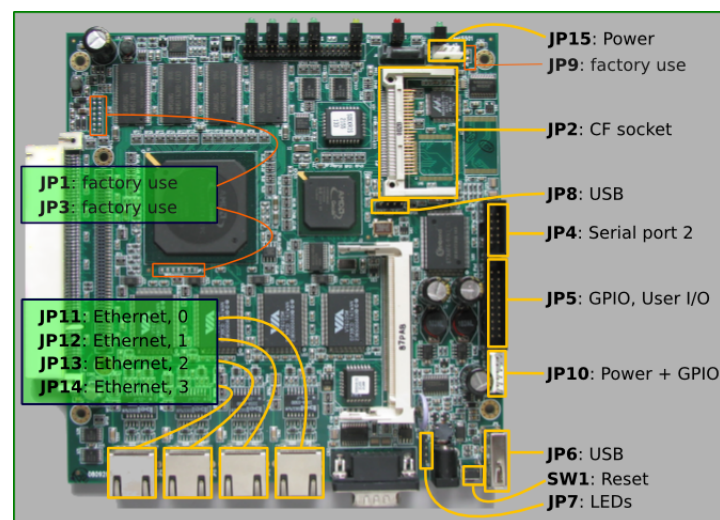


figura 4.3: Esempio scheda madre Soekris, modello 5501

Il secondo aspetto da considerare è la velocità effettiva della linea, anche in questo caso, come descritto in precedenza, maggiore è la velocità della linea, maggiore è la potenza di calcolo necessaria per sfruttarla a pieno.

4.1.3 Presenza di zone DMZ

La presenza di zone *DMZ* è un aspetto da non sottovalutare. Tutti i calcolatori presenti in questa zona devono risultare accessibili sia dall'interno della rete che dall'esterno generando una grande quantità di dati in rete.

Solitamente in DMZ sono presenti i Web server, i server FTP, e i mail server.

La presenza di Mail server implica la presenza di un software *antispam*, necessario ad aumentare la sicurezza aziendale. La presenza di un software antispam ha come diretta conseguenza l'aumento del carico di lavoro del firewall e quindi sarà richiesta una CPU più potente.

4.1.4 VLAN

La presenza di VLAN (Virtual LAN), ovvero la presenza di reti virtuali all'interno della rete aziendale, comporta un lavoro in più da parte del firewall.

Ogni rete VLAN si distingue dalle altre grazie ad un particolare *tag* presente all'interno dei pacchetti trasmessi. La distinzione dei pacchetti è a carico del firewall e quindi, se presenti, le VLAN aumentano il carico di lavoro del firewall.

4.1.5 Connessioni VPN

La presenza di VPN (Virtual Private Network), produce un aumento del carico di lavoro direttamente proporzionale con il numero di connessioni VPN attive nello stesso istante.

Maggiore è il numero di connessioni VPN che saranno attive nella rete maggiore sarà il carico di lavoro sulla CPU. Le connessioni VPN inoltre possono essere di due tipi:

- * client
- * punto-punto

Le connessioni *client*, mostrate in figura 4.4, sono le connessioni che permettono ad un utente di lavorare all'interno della rete aziendale senza essere fisicamente presente in azienda. Questa connessione infatti sfrutta la rete geografica per instaurare una connessione tra l'azienda e l'utente.

Per quanto riguarda la connessione *punto-punto*, mostrata in figura 4.5, è una particolare VPN che si occupa di collegare e connettere due reti distinte e situate in aree geografiche diverse tramite la connessione ad internet.

Questo tipo di connessione è più problematico da supportare in quanto richiede una CPU con discrete caratteristiche tecniche.

4.1.6 Tipologia di cablatura

La tipologia di cablatura utilizzata influenza molto la scelta del firewall. La rete infatti può essere basata su standard di trasmissione dati diversi quali:

- * 100 MB/s
- * 1000 MB/s



figura 4.4: Schema di collegamento VPN Client-Azienda

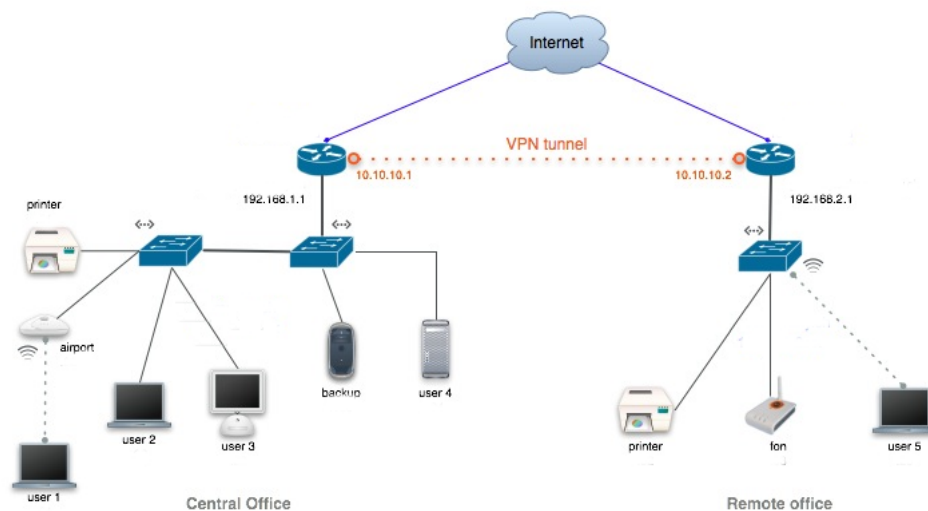


figura 4.5: Schema di collegamento VPN *punto-punto* , ovvero vpn che permette di unire due o più sottoreti

* 10000 MB/s

La diversa velocità della LAN incide sulla scelta diversa del firewall. Un firewall con schede Ethernet a 100 Mb/s non sfrutterà mai appieno la capacità di banda disponibile di una LAN 1000 MB/s o 10000 MB/s.

4.2 Compiti che deve svolgere il Firewall

I compiti che il firewall dovrà svolgere sono essenziali nella scelta del dispositivo. L'elenco di compiti :

- * filtraggio dei pacchetti, basato su regole IPtables
- * filtraggio dei pacchetti, basa su IPtables con L7-Filter attivo
- * implementazione di web proxy, basato su Squid
- * filtraggio su Mail-Server ovvero, azione antispam

Maggiore sarà il numero di compiti assegnati al firewall maggiore sarà la potenza di calcolo richiesta.

Filtraggio pacchetti con IPtables

Il filtraggio di pacchetti tramite IPtables è alla base della creazione di firewall. Una volta scelti i criteri corretti di filtraggio e analisi dei pacchetti, sarà il firewall che valuterà, sulla base dei suddetti criteri, ogni singolo pacchetto che transiterà all'interno della rete.

Tale operazione non risulta essere pesante dal punto di vista computazionale ed è ben supportata da CPU con caratteristiche inferiori rispetto alle CPU dei modelli di Soekris presi in esame.

Il filtraggio dei pacchetti avviene a livello 3 o 5 del modello OSI e quindi avviene a livello TCP/IP. Lavorando a livello 3 e 5 il firewall, per la distinzione e il filtraggio dei pacchetti il firewall ha la necessità di “leggere” solamente l'header di ogni singolo pacchetto e quindi è una scansione superficiale che non comporta eccessivi sforzi computazionali.

Filtraggio pacchetti con IPtables e L7-Filter

Il filtraggio di pacchetti tramite IPtables e L7-Filter funziona diversamente rispetto al filtraggio standard. Al contrario di quanto descritto nella sezione precedente, in questo caso si lavora a livello 3, 5 e 7 del modello OSI; quindi, oltre alle normali regole, ne esistono altre necessarie a filtrare i pacchetti a livello 7.

Il filtraggio dei pacchetti a livello 7 è un processo che richiede un maggior sforzo computazionale.

Al contrario di prima, ora non è più sufficiente la “lettura” degli header dei pacchetti, ma è necessaria anche una scansione più approfondita indispensabile per determinare il protocollo utilizzato per la trasmissione del pacchetto e l'applicazione che lo ha generato.

La lettura approfondita del pacchetto, inoltre non è sufficiente, ma si rende necessario il *matching* del protocollo tramite l'utilizzo di apposite espressioni regolari e quindi il tutto ha un peso computazionale non indifferente.

Implementazione di Web Proxy

l'implementazione di un Web Proxy tramite l'utilizzo di Squid potrebbe influenzare la scelta del firewall se rapportato al numero di utenti che saranno poi collegati al proxy stesso.

Un proxy, in questo caso Squid, infatti, è un programma che si interpone tra un client ed un server, inoltrando le richieste e le risposte dall'uno all'altro. Il client si collega al proxy invece che al server, e gli invia delle richieste. Il proxy a sua volta si collega al server e inoltra la richiesta del client, riceve la risposta da inoltrare al client.

Squid può essere usato per una o più delle seguenti ragioni:

- * **connettività:** permette ad una rete privata di accedere all'esterno, configurando un dispositivo in modo che faccia da proxy tra gli altri computer e Internet, in modo da mantenere un unico calcolatore connesso all'esterno, ma permettere a tutti di accedere.
- * **caching:** se si sfrutta tale capacità il firewall può immagazzinare per un certo tempo i risultati delle richieste di un utente più frequenti, e se un altro utente effettua le medesime richieste, può rispondere senza dover consultare il server originale diminuendo i tempi di attesa.
- * **monitoraggio:** può permettere al firewall di tener traccia di tutte le operazioni effettuate, come ad esempio, tutte le pagine web visitate, consentendo la realizzazione di statistiche ed osservazioni nell'utilizzo della rete.
- * **controllo:** permette l'applicazione di regole definite dall'amministratore di sistema per determinare quali richieste inoltrare e quali rifiutare, limitare l'ampiezza di banda utilizzata dai client, filtrare le pagine Web in transito.
- * **privacy:** può garantire un maggiore livello di privacy mascherando il vero indirizzo IP dei client in modo che il server non venga a conoscenza di chi ha effettuato la richiesta.

Alla luce di questo, risulta immediato capire, come un maggior numero di client connessi alla rete LAN comporti un maggior lavoro da parte del firewall. Con l'aumento dei client, infatti, c'è un aumento direttamente proporzionale delle richieste ricevute dal firewall e come conseguenza c'è un maggior carico sulla CPU.

Filtraggio antispam

I filtri antispam non sono software particolarmente complessi, la maggior parte si limita a verificare la sussistenza di una serie di criteri, per lo più legati alle parole e alla loro formattazione, riscontrabili nelle email.

Questo comporta la scansione di tutta la posta elettronica da parte del firewall o del server, nel caso il carico di lavoro sia sul firewall questo subirà un incremento nell'utilizzo della CPU.

Solitamente, tuttavia, il filtro antispam viene installato ed eseguito sul MailServer proprio per evitare sovraccarichi inutili al firewall.

4.3 Scelta di un firewall adatto

Come già spiegato in precedenza, un firewall è un componente passivo di difesa perimetrale che può anche svolgere funzioni di collegamento tra due o più tronconi di rete. Usualmente la rete viene divisa in due sottoreti: una, detta esterna, comprende l'intera Internet mentre l'altra interna, detta LAN, comprende una sezione più o meno grande di un insieme di computer locali. In alcuni casi è possibile che si crei l'esigenza di creare una terza sottorete detta DMZ (o zona demilitarizzata) atta a contenere quei sistemi che devono essere isolati dalla rete interna ma devono

comunque essere protetti dal firewall.

Alla luce di quanto detto in sezione 4.1 la scelta corretta del firewall deve essere valutata attentamente, prendendo in considerazione tutti gli aspetti già descritti. I punti su cui si basa la scelta del firewall, rispetto a quanto detto prima, quindi, sono due:

- * Sistema Operativo utilizzato
- * Hardware del dispositivo

Sistema Operativo

Il sistema operativo influenza molto le *performance*, e la capacità di svolgere determinate azioni da parte del firewall. Oltre al sistema operativo, fine a se stesso, è da prendere in considerazione il kernel utilizzato; un kernel compilato appositamente per il dispositivo offre performance di molto superiori rispetto ai kernel standard. Basti pensare che in alcuni casi un kernel ottimizzato per la CPU montata dal dispositivo può portare ad un incremento dell'85% delle prestazioni in termini di *throughput* ovvero in termini di velocità di linea.

La scelta del sistema operativo è ricaduta tra i seguenti:

- * IPCop
- * PfSense
- * Voyage Linux

elenchiamo ora le caratteristiche:

IPCop: è un firewall software basato su una distribuzione Linux che ha lo scopo di fornire un semplice e configurabile firewall utilizzando l'hardware di un comune PC.

L'ultima versione stabile di IPCop è la 2.0.6, rilasciata il 2012/10/28.

Tale sistema supporta appieno NetFilter standard con capacità di NAT, il supporto per DMZ via DNAT, il supporto DHCP il supporto NTP (Network Time Protocol) per sincronizzare la data e l'ora.

Inoltre IPCop offre la possibilità di attivare un proxy e un IDS (Intrusion detection system) che permette di rilevare intrusioni all'interno della rete.

Il sistema supporta quattro schede di rete e un numero illimitato di connessioni VPN, oltre ad offrire la possibilità di backup e restore della configurazione. È inoltre facilmente estendibile grazie a numerosi moduli presenti in Internet.

IPCop presenta inoltre una web GUI con interfaccia User friendly, che è raggiungibile digitando, nella barra degli indirizzi del browser, *l'indirizzo IP* del firewall con aggiunta la porta 8443; Esempio:

`http://192.168.1.54:8443`

Questo firewall inoltre offre la possibilità di amministrazione da remoto tramite l'utilizzo del protocollo SSH.

In figura 4.6 è raffigurato il logo di IPCop.



figura 4.6: Logo IPCop

PFSense: è un firewall software open source basato sulla distribuzione Linux *FreeBSD*. Ha lo scopo di fornire un firewall potente, sicuro e completamente configurabile utilizzando, come nel caso precedente, l'hardware di un comune PC.

Nel cuore del sistema il firewall PF (Packet Filter) è preso in prestito dal sistema *OpenBSD* da cui deriva. L'ultima versione stabile di PFSense è la 2.0.3, rilasciata il 2013/04/15.

Tale sistema supporta appieno caratteristiche quali Stateful Firewall, NAT, Load Balancing, VPN server, Generazione di grafici in tempo reale sull'andamento della rete e tramite l'aggiunta di moduli il supporto viene esteso anche alla creazione di proxy e al trattamento dei dati VOIP.

Una caratteristica di questo sistema operativo, che vale la pena spiegare, è la HA o High Availability.

Tale tecnologia permette al sistema di autoreplicarsi in un'altra macchina, in modo tale da avere sempre un alter ego pronto a sostituirsi all'originale in caso di malfunzionamento.

Come nel caso di IPCop anche questo sistema operativo presenta un'interfaccia grafica lato web, in questo caso la porta va configurata al momento dell'installazione, e un'interfaccia SSH per il collegamento da remoto. Questo firewall inoltre offre la possibilità di amministrazione da remoto tramite l'utilizzo del protocollo SSH. In figura 4.7 è raffigurato il logo di PFSense.

Voyage Linux: a differenza degli altri sistemi operativi che sono pensati come veri e propri firewall e quindi risultano poco adattabili ed elastici, questa distribuzione Linux è un sistema operativo a tutti gli effetti.



figura 4.7: Logo Voyage Linux

Voyage Linux, infatti è una distribuzione Linux di derivazione *Debian* che supporta tutti i pacchetti creati per *Debian*.

rispetto agli altri due sistemi operativi questa distribuzione si divide in tre sotto-distribuzioni:

- * Voyage Linux standard
- * Voyage MPD
- * Voyage MuBox

Le due versioni, Voyage MuBox e Voyage MPD nonostante siano simili all'edizione standard sono state sviluppate e concepite per un utilizzo multimediale, che quindi risulterebbero inadatte come firewall, motivo per cui dora in avanti si terrà in considerazione solamente la versione: *Voyage Linux* standard. L'ultima versione stabile di Voyage Linux è la 0.9.2, rilasciata il 2013/12/19.

A differenza degli altri due SO citati, Voyage Linux non presenta alcun tipo di interfaccia grafica e l'utilizzo avviene solo mediante linea di comando.

Anche in questo caso è presente un modulo che permette l'amministrazione e la manutenzione da remoto, quindi tramite SSH. Voyage è stato pensato appositamente per permettere di avere la capacità di elaborazione e la flessibilità di un PC all'interno di un firewall.

La cosa che contraddistingue maggiormente tale distribuzione, e che non ha limiti di capacità di operare, ovvero può eseguire qualsiasi tipo di ruolo e/o funzione se usato come firewall; questo perché, a differenza dei sistemi citati in precedenza, espandibili in parte solo grazie a delle apposite mod, Voyage essendo una distribuzione *Debian based* supporta qualsiasi tipo di software sviluppato per il controllo e la gestione della rete.

In figura 4.8 è raffigurato il logo di Voyage Linux.

Alla luce di quanto detto, risulta palese come la soluzione più equilibrata e personalizzabile sia l'ultimo sistema operativo citato, ovvero Voyage Linux.



figura 4.8: Logo Voyage Linux

Infatti, nonostante tutti e tre i sistemi siano molto funzionali e riescano ad eseguire su dispositivi con caratteristiche di CPU e RAM minimali, Linux Voyage risulta essere la soluzione migliore.

Come detto in precedenza, Voyage Linux, essendo una distribuzione di derivazione *Debian*, cioè uno dei branch Linux maggiormente sviluppati e supportati, permette una personalizzazione e uno sfruttamento dell'hardware massimo.

Inoltre, nonostante non abbia caratteristiche tecniche ben definite, tale sistema operativo garantisce la massima adattabilità e funzionalità in quanto permette l'installazione di tutti i pacchetti necessari alla creazione di un firewall di altissimo livello. Inoltre non disponendo di un interfaccia grafica, il sistema rimane snello e fluido anche durante la manutenzione, in quanto la potenza di calcolo necessaria al mantenimento della GUI non viene impiegata e quindi la CPU, e di conseguenza le prestazioni del firewall, non ne risentono negativamente. Un ultimo aspetto positivo è che Voyage Linux è stato ideato e progettato per lavorare in “*read only mode*”, ovvero il sistema operativo, nel suo normale funzionamento, che esclude la manutenzione, compie solo operazioni di lettura dal disco di sistema.

Tale meccanismo garantisce che difficilmente si verifichino errori nel file system a causa di scritture errate sul disco da parte del sistema e di conseguenza si ha un'affidabilità maggiore rispetto ai concorrenti che non dispongono di questo meccanismo.

Hardware dispositivo

I dispositivi presi in considerazione, come già detto, sono quelli riportati in tabella 4.1, come si nota le specifiche hardware sono decisamente minimali, questo perché il sistema che dovrà essere eseguito richiede, indipendentemente dalla versione, requisiti hardware minimi, che spesso si aggirano attorno a CPU 386 e 64MB di RAM.

Alla luce di questo è immediato capire come sulla scelta dell'hardware, il sistema operativo, al contrario di quanto si possa pensare, incide minimamente.

Più che il sistema operativo, infatti, la componente che maggiormente incide è il kernel del sistema. Un kernel generico infatti, può offrire prestazioni fino all'85% minori rispetto ad un kernel ottimizzato per la CPU e il dispositivo.

Normalmente i sistemi descritti in precedenza adottano kernel compilato per CPU 486 generiche che non offre grandi prestazioni, ma ricompilando il kernel per CPU

MDGEOD o Intel, ovvero per CPU x86 con scheduler personalizzato, le prestazioni subiscono incrementi vertiginosi.

La scelta dell'hardware verrà fatta, quindi, sulla base del sistema Voyage Linux e kernel ottimizzato.

In figura 4.9 è possibile notare le differenze tra i file di configurazione del kernel standard per CPU 486 generiche e il kernel ottimizzato.

Gli aspetti che incidono maggiormente, sono quelli largamente descritti nelle sezioni 4.1 e 4.2. I dispositivi presi in considerazione, tuttavia, sono in grado di svolgere appieno le funzioni di firewall implementando proxy, stateful firewall, connessioni VPN ed eventualmente antispam.

<pre> CONFIG_NO_BOOTMEM=y # CONFIG_MEMTEST is not set # CONFIG_M386 is not set # CONFIG_M486 is not set # CONFIG_M586 is not set # CONFIG_M586TSC is not set # CONFIG_M586MMX is not set # CONFIG_M686 is not set # CONFIG_MPENTIUMII is not set # CONFIG_MPENTIUMIII is not set # CONFIG_MPENTIUMM is not set # CONFIG_MPENTIUM4 is not set # CONFIG_MK6 is not set # CONFIG_MK7 is not set # CONFIG_MK8 is not set # CONFIG_MCRUSOE is not set # CONFIG_MEFFICEON is not set # CONFIG_MWINCHIP3D is not set # CONFIG_MGEODEGX1=y # CONFIG_MGEODE_LX is not set # CONFIG_MCYRIXIII is not set # CONFIG_MVIAC3_2 is not set # CONFIG_MVIAC7 is not set # CONFIG_MCORE2 is not set # CONFIG_MATOM is not set # CONFIG_X86_GENERIC is not set </pre>	<pre> CONFIG_NO_BOOTMEM=y # CONFIG_MEMTEST is not set # CONFIG_M386 is not set # CONFIG_M486 is not set # CONFIG_M586 is not set # CONFIG_M586TSC is not set # CONFIG_M586MMX is not set # CONFIG_M686 is not set # CONFIG_MPENTIUMII is not set # CONFIG_MPENTIUMIII is not set # CONFIG_MPENTIUMM is not set # CONFIG_MPENTIUM4 is not set # CONFIG_MK6 is not set # CONFIG_MK7 is not set # CONFIG_MK8 is not set # CONFIG_MCRUSOE is not set # CONFIG_MEFFICEON is not set # CONFIG_MWINCHIP3D is not set # CONFIG_MGEODEGX1 is not set # CONFIG_MGEODE_LX=y # CONFIG_MCYRIXIII is not set # CONFIG_MVIAC3_2 is not set # CONFIG_MVIAC7 is not set # CONFIG_MCORE2 is not set # CONFIG_MATOM is not set # CONFIG_X86_GENERIC is not set </pre>	<pre> CONFIG_NO_BOOTMEM=y # CONFIG_MEMTEST is not set # CONFIG_M386 is not set # CONFIG_M486 is not set # CONFIG_M586 is not set # CONFIG_M586TSC is not set # CONFIG_M586MMX is not set # CONFIG_M686 is not set # CONFIG_MPENTIUMII is not set # CONFIG_MPENTIUMIII is not set # CONFIG_MPENTIUMM is not set # CONFIG_MPENTIUM4 is not set # CONFIG_MK6 is not set # CONFIG_MK7 is not set # CONFIG_MK8 is not set # CONFIG_MCRUSOE is not set # CONFIG_MEFFICEON is not set # CONFIG_MWINCHIP3D is not set # CONFIG_MGEODEGX1 is not set # CONFIG_MGEODE_LX is not set # CONFIG_MCYRIXIII is not set # CONFIG_MVIAC3_2 is not set # CONFIG_MVIAC7 is not set # CONFIG_MCORE2=y # CONFIG_MATOM is not set # CONFIG_X86_GENERIC=y </pre>
CPU MGEODE GX1	CPU MGEODE LX	CPU INTEL CORE 2
	<pre> # CONFIG_PARAVIRT_GUEST is not set # CONFIG_MEMTEST is not set # CONFIG_M386 is not set # CONFIG_M486=y # CONFIG_M586 is not set # CONFIG_M586TSC is not set # CONFIG_M586MMX is not set # CONFIG_M686 is not set # CONFIG_MPENTIUMII is not set # CONFIG_MPENTIUMIII is not set # CONFIG_MPENTIUMM is not set # CONFIG_MPENTIUM4 is not set # CONFIG_MK6 is not set # CONFIG_MK7 is not set # CONFIG_MK8 is not set # CONFIG_MCRUSOE is not set # CONFIG_MEFFICEON is not set # CONFIG_MWINCHIP3D is not set # CONFIG_MGEODEGX1 is not set # CONFIG_MGEODE_LX is not set # CONFIG_MCYRIXIII is not set # CONFIG_MVIAC3_2 is not set # CONFIG_MVIAC7 is not set # CONFIG_MPSC is not set # CONFIG_MCORE2 is not set </pre>	CPU 486 Standard

figura 4.9: Differenze dei file di configurazione del kernel in base alla CPU

La scelta dell'hardware quindi, dipende principalmente dall'infrastruttura associata al firewall, descriviamo ora gli aspetti che determinano la scelta:

velocità e tipologia ADSL: come riportato in sottosezione 4.1.2 la tipologia, ma soprattutto la velocità influenzano la scelta.

Considerando ora tre fasce di velocità in uscita, le tre fasce sono standard, vediamo quale sia la scelta migliore. Le tre fasce in uscita sono:

- * **15 Mbit/s:** se la velocità è compresa tra 0 e 15 Mbit/s e l'ADSL è sia simmetrica che asimmetrica è sufficiente la Soekris 4801.
La Soekris 4801 supporta anche linee a 20 Mbit/s con ADSL asimmetrica, come mostrato in figura 4.10.



figura 4.10: Velocità di punta raggiunta da Soekris 4801 con linea asimmetrica

- * **da 15 Mbit/s a 50 Mbit/s:** se la velocità è compresa tra 15 e 50 Mbit/s e l'ADSL è sia simmetrica che asimmetrica è consigliata la Soekris 5501.
Come per il modello più piccolo la Soekris 5501 supporta anche linee a 60 Mbit/s con ADSL asimmetrica.
- * **da 50 Mbit/s a 100 Mbit/s:** se la velocità è maggiore di 50 Mbit/s e l'ADSL è sia simmetrica che asimmetrica è caldamente consigliata la Soekris 6501.

A differenza delle altre, il limite superiore massimo di portata è stato fissato a 100 Mbit/s, in realtà la CPU permetterebbe di salire ancora e quindi è stato stimato che il limite massimo sia attorno ai 120 Mbit/s per linea sincrona e 150 Mbit/s circa, per linea asimmetrica.

La difficoltà nel testare tale cosa è che in Italia attualmente, non sono ancora disponibili linee consumer con tale velocità. In figura 4.10 è possibile osservare la velocità raggiunta con una linea a 100 Mbit/s.

Le velocità di punta riportate sono state testate con kernel ottimizzati per le CPU

quantità VPN: come riportato nella sottosezione 4.1.5 il numero di connessioni VPN provoca un massiccio aumento del carico di lavoro da parte del firewall, questo perché ogni utente che si collega tramite questa tecnologia “apre un tunnel” tra la sua rete e la rete protetta dal firewall.

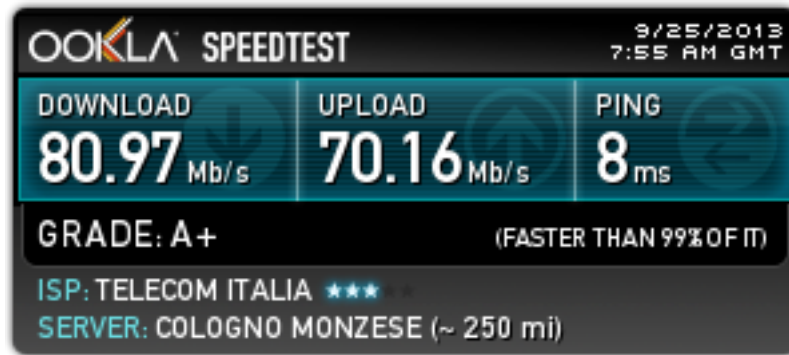


figura 4.11: Velocità di punta raggiunta da Soekris 6501 con linea simmetrica

L'oneroso compito di mantenere aperto il tunnel giace sulle spalle del firewall, che quindi, deve gestire una nuova sottorete, mascherandola e rendendola disponibile alla rete interna. Quindi, maggiore sarà la quantità di client connessi tramite VPN, maggiore sarà lo sforzo del firewall.

Una rete VPN inoltre può essere utilizzata come base per unire due sottoreti distinte e quindi in questo caso il carico di lavoro risulta essere ancora maggiore; riportiamo in tabella ora le reti VPN consigliate rispetto all'hardware utilizzato.

Modelli	Numero VPN consigliato	VPN tra sottoreti
Soekris 4801	5	Si, ma altamente sconsigliato, la CPU subisce sovraccarichi con possibile blocco del servizio
Soekris 5501	15	Si, ma la CPU subisce sovraccarichi con possibili disservizi
Soekris 6501	25	Si, soluzione migliore

tabella 4.2: Tabella rapporto connessioni VPN modello dispositivo

Cablatura: in questo caso è facile capire quale scelta effettuare in quanto la scelta è strettamente legata all'hardware presente.

Se si avrà una LAN con velocità 10/100 Mbit/s la scelta sarà una Soekris 4801, con velocità 10/100/1000 Mbit/s la scelta ricadrà su Soekris 5501, mentre per LAN con velocità fino a 10000 Mbit/s sarà necessaria una Soekris 6501 con degli adattatori particolari, poiché solitamente una LAN a tale velocità, è basta su tecnologia a fibra ottica.

Al termine di questa discussione si capisce come per una piccola media impresa la soluzione ottimale sia una Soekris 4801, che dispone di un hardware sufficiente a mantenere i servizi di firewall fondamentali con linee ADSL fino a 15 Mbit/s.

Si noti che la scelta dell'hardware in base alla velocità non è obbligato, ma consigliato, in quanto permette di sfruttare appieno la banda disponibile.

Una Soekris 4801 installata con una linea ADSL sincrona a 50 Mbit/s, funzionerà comunque, la la banda massima che questa sarà in grado di garantire è 15 Mbit/s e quindi la banda verrebbe sfruttata al 30% della propria portata.

Capitolo 5

Implementazione Firewall

Questo capitolo si preme di spiegare il modo di funzionare di IPtables e Squid mostrando inoltre il modo di creare le regole secondo le proprie necessità

5.1 Introduzione

5.1.1 IPtables

Introduzione

Prima di poter iniziare a creare delle regole firewall con IPtables è necessario spiegarne le modalità di funzionamento.

I dati che transitano in una rete sono divisi in pacchetti di dimensioni prefissate, con NetFilter ed IPtables è possibile controllare il contenuto di ogni singolo pacchetto, e definire le azioni da compiere in base alle caratteristiche di quelli ricevuti. Ad esempio, si può definire una regola che impedisce la ricezione di pacchetti provenienti da un particolare indirizzo o che utilizzano una determinata porta per effettuare la connessione.

Il sistema NetFilter è basato su regole raggruppate in catene chiamate, *chain*, che a loro volta sono raggruppate in tabelle chiamate *tables*. Ogni tabella definisce un tipo diverso di operazioni che è possibile effettuare sui pacchetti; ogni catena definisce come vengono trattati i pacchetti nelle diverse fasi della loro elaborazione.

Le catene sono una forma di lista di controllo degli accessi, chiamata solitamente con l'acronimo *ACL*.

Ogni regola è costituita da due parti: la specifica delle caratteristiche che un pacchetto deve avere affinché la regola stessa venga applicata (*match*) e un obiettivo o *target*, che indica cosa fare quando il pacchetto rispetta le caratteristiche indicate.

A ciascuna catena è anche associata una politica di default, che definisce come vengono trattati i pacchetti che non corrispondono ad alcuna regola. Le caratteristiche più di frequente utilizzate per costruire delle regole sono l'indirizzo di partenza

o di destinazione del pacchetto e il numero di porta associato alla connessione.

Tabelle

Esistono quattro tabelle prestabilite, ognuna delle quali contiene delle catene predefinite. Esiste anche la possibilità di creare altre tabelle.

Inizialmente, tutte le catene sono vuote e hanno una politica che permette a tutti i pacchetti di passare senza essere bloccati o alterati in alcun modo, esse vanno poi modificate a seconda delle proprie esigenze.

Le tabelle predefinite sono le seguenti:

tabella Filter: è responsabile del filtraggio dei pacchetti, permette cioè di bloccarli o di farli passare. Ogni pacchetto passa attraverso la tabella filtro.

La tabella contiene le seguenti catene predefinite:

- * INPUT: tutti i pacchetti destinati al sistema passano attraverso questa
- * OUTPUT: tutti i pacchetti creati dal sistema passano attraverso questa
- * FORWARD: tutti i pacchetti che hanno come destinazione finale un altro sistema e che non sono stati generati dal sistema stesso.

tabella NAT: questa tabella è responsabile dell'impostazione delle regole per la modifica degli indirizzi e porte dei pacchetti. Il primo pacchetto di una connessione passa attraverso questa tabella, e il risultato del passaggio del primo pacchetto determina come tutti gli altri pacchetti della stessa connessione verranno modificati.

La tabella contiene le seguenti catene predefinite:

- * PREROUTING: passano attraverso questa catena i pacchetti in entrata, il passaggio avviene prima che la tabella di routing venga consultata per effettuare l'instradamento.
- * POSTROUTING: passano attraverso questa catena i pacchetti in uscita dopo la consultazione della tabella di routing .
- * OUTPUT: permette un DNAT limitato sui pacchetti generati localmente.

tabella Mangle: questa tabella è responsabile delle modifiche alle opzioni dei pacchetti, come ad esempio quella che determina la qualità del servizio. Tutti i pacchetti passano attraverso questa tabella.

i pacchetti che passano per questa tabella possono essere modificati aggiungendo, ed esempio, un particolare tag per identificare i pacchetti. Questa tabella contiene le seguenti catene:

- * PREROUTING: esamina tutti i pacchetti che in qualche modo entrano nel sistema. Questo processo avviene prima che il routing decida se il

pacchetto debba essere inoltrato o se sia destinato al sistema. Viene utilizzata per manipolare l'header del pacchetto.

- * INPUT: tutti i pacchetti destinati al sistema passano per questa catena.
- * FORWARD: tutti i pacchetti che vengono instradati dal sistema ma di cui il sistema non è né sorgente iniziale né destinazione finale, passano per questa catena.
- * OUTPUT: tutti i pacchetti generati dal sistema passano per questa catena.
- * POSTROUTING: tutti i pacchetti che lasciano il sistema, sia quelli in OUTPUT sia quelli in FORWARD, passano poi per questa catena.

Regole

Le regole di IPtables sono divise in due parti, la specifica e l'obiettivo.

La specifica definisce l'insieme di caratteristiche dei pacchetti che devono essere gestite dalla regola stessa, e si ottiene combinando ad esempio l'interfaccia di invio/ricezione del pacchetto, l'indirizzo IP sorgente o di destinazione, il protocollo utilizzato e le porte.

L'obiettivo di una regola è invece l'azione da compiere se un pacchetto rispetta la regola, e può essere uno degli obiettivi predefiniti come ACCEPT, che accetta il pacchetto, DROP, che scarta il pacchetto, o RETURN che fa ritornare il pacchetto all'interno di un'altra catena.

Se un pacchetto rispecchia le specifiche definite da una regola questo verrà trattato secondo quanto definito e il contatore della regola e della catena sarà incrementato, al fine di testimoniare il passaggio del pacchetto.

5.1.2 Squid

Il funzionamento di Squid risulta molto più semplice, le regole di *Allow* o *Deny*, rispettivamente regole di accesso e blocco vengono scritte all'interno del file di configurazione di Squid tramite la generazione di ACL. Ogni ACL costituisce una regola che può essere di accesso o blocco.

5.2 Implementazione regole IPtables

L'implementazione di regole firewall durante tutto l'arco del tirocinio è stato molto vario.

La quantità di regole e la loro funzionalità dipendevano dalle necessità dei clienti. Le regole riportate in seguito quindi, sono state scelte per fornire un esempio di firewall, sfruttando le regole più utilizzate durante il periodo di stage.

Intestazione del file firewall.sh

Nella prima parte del file contenente le regole, devono essere impostate le interfacce e in loro relativi indirizzi IP .

In questo caso vediamo che sono state definite ed impostate due interfacce, ovvero

ETH1 ed ETH0 che corrispondono, rispettivamente alle due schede di rete omonime. Le prime due variabili, come riportato nel listato 5.1 corrispondono all'indirizzo IP del firewall e al *default gateway* della rete.

La variabile IP_ETH1 associa l'indirizzo IP alla scheda Ethernet ETH1, mentre l'ultima variabile si occupa di definire lo spazio degli indirizzi che sarà disponibile in LAN.

Listing 5.1: Definizione delle variabili e degli indirizzi IP

```
IP_ETH0=11.11.11.10
GW_ETH0=11.11.11.9
IP_ETH1=192.168.2.254/24
NET_ETH1=192.168.2.0/24
```

Le istruzioni riportate nel listato 5.2 si occupano dell'associazione delle variabili, appena definite alle interfacce di rete, il tutto secondo quanto detto prima. oltre ad associare le interfacce di rete queste istruzioni permettono di attivare le schede di rete necessarie alla creazione del firewall.

Listing 5.2: Attivazione delle schede di rete e assegnazione degli indirizzi IP

```
ip l s lo up
ip a a 127.0.0.1/8 brd + dev lo

if [ $IP_ETH0 ]; then
    ip l s eth0 up
    ip a a $IP_ETH0 brd + dev eth0
    ip r a default via $GW_ETH0 dev eth0
fi
if [ $IP_ETH1 ]; then
    ip l s eth1 up
    ip a a $IP_ETH1 brd + dev eth1
fi
if [ $IP_ETH2 ]; then
    ip l s eth2 up
```

Successivamente nel file contenente le regole devono essere presenti i comandi del caricamento, in memoria, dei moduli del kernel necessari al corretto riconoscimento dei pacchetti che transitano attraverso il firewall.

Un esempio di moduli è riportato del listato 5.3. Assieme al comando di caricamento dei moduli è una buona regola aggiungere i comandi di azzeramento dei contatori di IPtables, al fine di avere la garanzia di non trovare falsi positivi durante le fasi di test.

Listing 5.3: Comandi per il caricamento dei moduli del kernel e comandi di pulizia delle tabelle di IPtables.

```
modprobe -a ip_nat_ftp
modprobe -a ip_conntrack_ftp
modprobe -a ip_conntrack_pptp
```

```
modprobe -a ip_conntrack
modprobe -a ip_conntrack_netlink
modprobe -a ipt_layer7
modprobe -a xt_layer7

iptables -F
iptables -t nat -F
iptables -X
```

5.2.1 Regole

Regole di NAT

Il termine *NAT*, come già spiegato in precedenza significa letteralmente *Network Address Translation*, ovvero “traduzione di indirizzi di rete”.

La sua funzione la si capisce con esempio: supponiamo, di avere tre client da proiettare in internet, attraverso il firewall configurato come NAT con due schede di rete di cui, una si affaccia su internet e una sulla nostra LAN.

supponiamo inoltre che l’indirizzo IP della scheda di rete, del nostro firewall, che si affaccia in internet, sia 10.100.0.1 e che l’indirizzo IP, della scheda di rete, che si affaccia sulla nostra LAN sia 192.168.0.1 e che, di conseguenza, per compatibilità di classi di indirizzi, gli indirizzi IP dei nostri tre client siano 192.168.0.2, 192.168.0.3 e 192.168.0.4, ovvero indirizzi privati e quindi poco adatti ad una connessione ad internet.

La funzione delle regole NAT è quindi quella di mascherare gli indirizzi IP dei client connessi al firewall, facendo risultare le connessioni verso internet come se venissero istanziate dall’indirizzo pubblico 10.100.0.1 definito in precedenza.

La regola riportata in 5.4 infatti, si occupa di mascherare tutti le richieste effettuate dagli indirizzi IP provenienti dai client connessi al firewall, in modo tale da nascondere l’indirizzo privato che questi assumono nella LAN.

Listing 5.4: Regole di NAT

```
# NAT
iptables -t nat -A POSTROUTING -o eth0 -s $NET_ETH1 -j
MASQUERADE
```

Regole di filtraggio pacchetti Windows

Le regole riportate in seguito si rendono necessarie al fine di diminuire le possibilità di attacchi verso i Client che hanno installato le versioni di Windows NT,2000,XP,2003. Tali versioni del sistema operativo, infatti, sfruttano le porte 445, 137 e 139 per la condivisione di dati tramite il protocollo *Server Message Block*, il cui acronimo è SMB. Tale protocollo, infatti è deputato alla condivisione di file, stampanti, porte seriali e a particolari comunicazioni di rete; per il suo funzionamento si basa sui

protocolli TCP e UDP. Le regole riportate in 5.5 quindi, sono indispensabili per negare l'accesso a dati aziendali da parte di persone non autorizzate.

Le regole sono definite in modo tale da eliminare qualsiasi pacchetto che tenta di aprire la connessione verso tali porte.

Listing 5.5: Regole di filtraggio pacchetti Windows

```
# Drop SMB Windows
iptables -A FORWARD -m state -i eth1 -s $NET_ETH1 -o eth0 -p
    udp --dport 445 --state NEW -j DROP
iptables -A FORWARD -m state -i eth1 -s $NET_ETH1 -o eth0 -p
    tcp --dport 445 --state NEW -j DROP

iptables -A FORWARD -m state -i eth1 -s $NET_ETH1 -o eth0 -p
    udp --dport 137:139 --state NEW -j DROP
iptables -A FORWARD -m state -i eth1 -s $NET_ETH1 -o eth0 -p
    tcp --dport 137:139 --state NEW -j DROP
```

Regole di filtraggio pacchetti generati dai Client

Le regole riportate in seguito vengono utilizzate per filtrare il traffico generato dai client, e che attraversa il firewall. In questo caso, le regole sono tutte in “ACCEPT” quindi, i pacchetti generati dalle interfacce *ppp+*, *tun+* e *tap+* vengono accettati.

Prima di definire quali accettare o eliminare, è necessario accettare tutte le nuove connessioni derivanti dalla LAN che tentano di uscire in internet; di questo si occupa la prima regola.

Accettare tutte le connessioni è necessario definire da quali interfacce accettare il traffico in uscita e il traffico in entrata.

Le regole riportate in seguito riferiscono le interfacce di utilizzo standard all'interno di una rete aziendale. L'interfaccia *ppp+* è necessaria per il corretto funzionamento del protocollo Point-to-Point, le interfacce *tun+* e *tap+*, invece, sono nello specifico, dei driver virtuali che rappresentano, dunque delle periferiche virtuali che sfruttano le schede di rete fisiche.

Tali interfacce sono utilizzate per la creazione di particolari connessioni quali, per esempio, VPN ed SSH.

Tali connessioni sono molto utilizzate in ambito aziendale perché possono consentire di lavorare da remoto (connessione SSH) o di instaurare connessioni punto-punto tra azienda e dipendente (connessione VPN).

Le regole riportate nel listato 5.6 dunque, permettono il corretto funzionamento di tali connessioni.

Listing 5.6: Regole di filtraggio pacchetti generati dai Client

```
# Client
iptables -A FORWARD -m state -i eth1 -s $NET_ETH1 -o eth0 --
    state NEW -j ACCEPT
```

```

iptables -A FORWARD -i ppp+ -j ACCEPT
iptables -A FORWARD -o ppp+ -j ACCEPT

iptables -A FORWARD -i tun+ -j ACCEPT
iptables -A FORWARD -o tun+ -j ACCEPT

iptables -A FORWARD -i tap+ -j ACCEPT
iptables -A FORWARD -o tap+ -j ACCEPT

iptables -A FORWARD -i eth0 -j LOG

```

Regole di Input/Output standard

Queste regole si occupano di gestire correttamente il traffico di rete che riceve in Input e in Output il firewall. La prima regola, `iptables -P INPUT DROP` si occupa di bloccare tutto il traffico in ingresso.

Il motivo di tale regola è semplice; bloccare tutto il traffico in ingresso permette, successivamente, di accettare solamente le connessioni e il traffico desiderato bloccando invece quello non necessario o comunque quello considerato dannoso.

Come si può notare dal listato 5.7, queste regole, sono suddivise in sotto gruppi:

Listing 5.7: Regole di Input/output del firewall

```

#Regole di Input/output
iptables -P INPUT DROP

#Gruppo 1
iptables -A INPUT -m state --state INVALID -j DROP
iptables -A INPUT -f -j DROP
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j
ACCEPT

#Gruppo 2
iptables -A INPUT -i eth0 -p tcp --dport 22 -s assistenza.
miazienza.com -m state --state NEW -j ACCEPT
iptables -A INPUT -i eth0 -p udp --dport 5000:5002 -m state --
state NEW -j ACCEPT

#Gruppo 3
iptables -A INPUT -i eth1 -j ACCEPT
iptables -A INPUT -i ppp+ -j ACCEPT
iptables -A INPUT -i tun+ -j ACCEPT
iptables -A INPUT -i tap+ -j ACCEPT
iptables -A INPUT -i eth0 -j LOG

```

Gruppo 1: Le regole di questo gruppo si occupano eliminare i pacchetti derivanti da connessioni di cui non si conosce l'origine, i pacchetti frammentati e inoltre, la regola:

```
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j
ACCEPT
```

si occupa di mantenere tracciabili tutte le connessioni che risultano ancora aperte.

Gruppo 2: questo insieme di regole ha la funzione di consentire il traffico generato dai database dei server aziendali, in particolare, la regola deputata a tale funzione è:

```
iptables -A INPUT -i eth0 -p udp --dport 5000:5002 -m
state --state NEW -j ACCEPT
```

e inoltre di permettere la manutenzione del firewall da remoto, in questo caso la regola è:

```
iptables -A INPUT -i eth0 -p tcp --dport 22 -s assisten-
za.miazienda.com -m state --state NEW -j ACCEPT
```

Nel caso della seconda regola il nome dell'indirizzo sorgente, andrà sostituito con il nome del cliente, come da imposto dallo standard aziendale.

Gruppo 3: quest'ultimo gruppo di regole permette di accettare tutto il traffico in ingresso verso le interfacce *ppp+*, *tun+* e *tap+*; l'ultima regola server per la generazione di un log di sistema, nel quale vengono mostrati i pacchetti scansionati che corrispondono alle specifiche delle regole.

5.3 Implementazione regole IPtables-Layer7

Come spiegato in precedenza, le regole IPtables a livello 7 sono regole particolari, che permettono il riconoscimento dei pacchetti a livello applicativo, ovvero permettono il filtraggio dei pacchetti basandosi sull'applicazione o sul protocollo che li ha generati. Questo tipo di regole si basa su particolari protocolli definiti precedentemente che permettono il riconoscimento dell'applicazione che genera traffico di rete. I protocolli su cui si basa, sono essenzialmente delle espressioni regolari.

Nel caso all'interno del firewall siano presenti tali regole, IPtables analizza i pacchetti in profondità in modo poter analizzare il protocollo applicativo che li ha generati, basandosi, sulle espressioni regolari citate in precedenza. Nel caso di un riscontro positivo "protocollo applicativo - espressione regolare" allora IPtables eseguirà le azioni descritte nella regola.

Questo genere di regole viene utilizzato solitamente per l'eliminazione di pacchetti non desiderati derivati da particolari applicazioni. Come si può notare dagli esempi che seguiranno, le regole sfruttano la tabella *mangle*, spiegata nella sotto sezione 5.1.1. I pacchetti che sono generati dal protocollo specificato, infatti, sono modificati con un *tag* che permetterà poi l'eliminazione ad opera di NetFilter. Le regole riportate nelle sezioni sottostanti sono utilizzate per bloccare il funzionamento dei protocolli elencati, per esempio la regola:


```
iptables -t mangle -A POSTROUTING -m layer7 --l7proto edonkey  
-j DROP
```

impedirà l'utilizzo di software P2P, ad esempio *Emule*, che sfruttano tale protocollo.

Regole per il P2P

Queste regole si occupano di eliminare tutti i pacchetti generati dai protocolli usati da software di *peer to peer*. I protocolli vengono specificati tramite l'inserimento del nome, come ad esempio *edonkey*.

Tali regole, riportate nel listato 5.8, vengono implementate, su richiesta del cliente, per impedire l'utilizzo di software di P2P che altrimenti, se utilizzati, potrebbero saturare la banda della rete o aprire connessioni non sicure.

Listing 5.8: Regole di filtraggio P2P

```
iptables -t mangle -A POSTROUTING -m layer7 --l7proto edonkey  
-j DROP  
  
iptables -t mangle -A POSTROUTING -m layer7 --l7proto gnutella  
-j DROP  
  
iptables -t mangle -A POSTROUTING -m layer7 --l7proto ares -j  
DROP  
  
iptables -t mangle -A POSTROUTING -m layer7 --l7proto 100bao -  
j DROP  
  
iptables -t mangle -A POSTROUTING -m layer7 --l7proto goboogy  
-j DROP  
  
iptables -t mangle -A POSTROUTING -m layer7 --l7proto hotline  
-j DROP  
  
iptables -t mangle -A POSTROUTING -m layer7 --l7proto kugoo -j  
DROP  
  
iptables -t mangle -A POSTROUTING -m layer7 --l7proto imesh -j  
DROP  
  
iptables -t mangle -A POSTROUTING -m layer7 --l7proto napster  
-j DROP  
  
iptables -t mangle -A POSTROUTING -m layer7 --l7proto mute -j  
DROP  
  
iptables -t mangle -A POSTROUTING -m layer7 --l7proto openft -  
j DROP
```

```
iptables -t mangle -A POSTROUTING -m layer7 --l7proto poco -j DROP

iptables -t mangle -A POSTROUTING -m layer7 --l7proto pplive -j DROP

iptables -t mangle -A POSTROUTING -m layer7 --l7proto soribada -j DROP

iptables -t mangle -A POSTROUTING -m layer7 --l7proto soulseek -j DROP

iptables -t mangle -A POSTROUTING -m layer7 --l7proto bittorrent -j DROP

iptables -t mangle -A POSTROUTING -m layer7 --l7proto thecircle -j DROP

iptables -t mangle -A POSTROUTING -m layer7 --l7proto fasttrack -j DROP

iptables -t mangle -A POSTROUTING -m layer7 --l7proto directconnect -j DROP

iptables -t mangle -A POSTROUTING -m layer7 --l7proto applejuice -j DROP
```

Regole per siti di file sharing

Queste regole si occupano di eliminare tutti i pacchetti generati dai protocolli di siti di file sharing, come ad esempio *mediafire.com*. I protocolli vengono specificati tramite l'inserimento del nome, come ad esempio *mediafire*. Anche in questo caso tali regole, riportate nel listato 5.9, vengono implementate, su richiesta del cliente, per impedire il download di file personali che potrebbero saturare la banda.

Listing 5.9: Regole di filtraggio file sharing

```
iptables -t mangle -A POSTROUTING -m layer7 --l7proto depositfiles -j DROP

iptables -t mangle -A POSTROUTING -m layer7 --l7proto rapidshare -j DROP

iptables -t mangle -A POSTROUTING -m layer7 --l7proto hotfile -j DROP

iptables -t mangle -A POSTROUTING -m layer7 --l7proto mediafire -j DROP
```

Regole per file

Queste regole si occupano di eliminare tutti i pacchetti generati dai protocolli che si occupano della trasmissione dei file. Tali regole, riportate nel listato 5.10, vengono applicate per impedire il download di file che potrebbero essere dannosi per i computer aziendali.

Listing 5.10: Regole di filtraggio dei file

```
iptables -t mangle -A POSTROUTING -m layer7 --l7proto exe -j DROP
iptables -t mangle -A POSTROUTING -m layer7 --l7proto flash -j DROP
iptables -t mangle -A POSTROUTING -m layer7 --l7proto mp3 -j DROP
iptables -t mangle -A POSTROUTING -m layer7 --l7proto ogg -j DROP
iptables -t mangle -A POSTROUTING -m layer7 --l7proto perl -j DROP
iptables -t mangle -A POSTROUTING -m layer7 --l7proto rar -j DROP
iptables -t mangle -A POSTROUTING -m layer7 --l7proto zip -j DROP
iptables -t mangle -A POSTROUTING -m layer7 --l7proto postscript -j DROP
```

Regole per messaggistica istantanea

Queste regole si occupano di eliminare tutti i pacchetti generati da protocolli di software di messaggistica istantanea quali, per esempio, MSN, Skype o Gtalk. Tali regole, riportate nel listato 5.11, sono state applicate solo in alcuni casi, poiché software di messaggistica istantanea permettono di avere una comunicazione efficace tra i dipendenti delle aziende.

Listing 5.11: Regole di filtraggio software di messaggistica istantanea

```
iptables -t mangle -A POSTROUTING -m layer7 --l7proto msnmessenger -j DROP
iptables -t mangle -A POSTROUTING -m layer7 --l7proto msn-filetransfer -j DROP
iptables -t mangle -A POSTROUTING -m layer7 --l7proto gtalk -j DROP
iptables -t mangle -A POSTROUTING -m layer7 --l7proto skype -j DROP
```

```
iptables -t mangle -A POSTROUTING -m layer7 --l7proto irc -j  
DROP
```

Regole per social network

Queste regole si occupano di eliminare tutti i pacchetti generati da protocolli di siti sociali quali *Twitter* e *Facebook*, impedendone quindi l'utilizzo.

Le regole sono riportate nel listato [5.12](#),

Listing 5.12: Regole di filtraggio siti di social network

```
iptables -t mangle -A POSTROUTING -m layer7 --l7proto facebook  
-j DROP  
  
iptables -t mangle -A POSTROUTING -m layer7 --l7proto twitter  
-j DROP
```

Note di implementazione

Le regole firewall devono essere ordinate per importanza.

La regola `iptables -P INPUT DROP`, che blocca tutto il traffico in entrata, per esempio, deve essere posta prima di regole quali:

```
iptables -A INPUT -i eth0 -p udp --dport 5000:5002 -m state  
--state NEW -j ACCEPT o iptables -A INPUT -m state --state  
ESTABLISHED,RELATED -j ACCEPT
```

altrimenti queste verranno annullate.

Le regole citate precedentemente sono un esempio di regole applicate maggiormente al firewall, L'elenco completo delle regole può essere visionato nella documentazione aziendale all'interno delle *schede clienti*.

5.4 Implementazione Proxy tramite Squid

Introduzione

Squid, come citato in sezione [5.4](#), è un proxy server utile per miglioramento la gestione delle connessioni di reti LAN ad Internet.

La funzione principale di Squid è quella di ottimizzare, ovvero di velocizzare, i collegamenti ad Internet di una rete. Il proxy infatti si farà carico di memorizzare tutte le richieste web effettuate dai client connessi in modo da restituire le stesse pagine con più efficienza e rapidità. Questo processo è definito come *web caching*. Squid offre il servizio di *web caching* per protocolli come HTTP (Hyper Text Transport Protocol), FTP (File Transfer Protocol) SSL (Secure Sockets Layer) e DNS (Domain Name Server).

L'utilizzo di tale software non si ferma al solo *web caching*, infatti, può essere utilizzato anche per selezionare i contenuti vietati e quelli permessi alla visione, può essere utile per controllare l'attività della rete interna e per vietare l'utilizzo di alcuni programmi.

Il file di configurazione di Squid risulta essere di oltre 4000 righe, motivo per cui, all'interno del documento saranno riportate solamente le impostazioni standard e quelle maggiormente richieste.

A questo punto procediamo con la spiegazione di una configurazione standard di Squid. Prima di procedere con la spiegazione del file di configurazione, è necessario specificare che la versione utilizzata è la 3.3.0.

Configurazione

Per una corretta configurazione è necessario modificare il file *squid.conf*, che solitamente, si trova in questa posizione:

`/etc/squid/squid.conf`

Le regole, o per meglio dire, le *direttive* per comodità sono state suddivise in tre blocchi:

- * direttive generali
- * direttive ACL
- * direttive di ACL di accesso al proxy

Direttive generali

Per direttive generali, si intendono tutte quelle istruzioni che si rendono necessarie ad una corretta configurazione generale di Squid.

queste direttive hanno diverse funzioni:

http_port: questa direttiva permette di definire l'indirizzo IP e la porta di ascolto del proxy, l'indirizzo IP e la porta saranno poi necessari per collegare i vari client della rete.

cache_mem: questa direttiva permette di settare la dimensione massima della *cache*, poiché quest'ultima, in generale, risiede in RAM è consigliato avere un valore non troppo eccessivo.

maximum_object_size: definisce la dimensione massima che può avere un oggetto salvato nella cache.

minimum_object_size: definisce la dimensione minima che deve avere un oggetto per non essere salvato in cache.

maximum_object_size_in_memory: specifica la dimensione massima degli oggetti conservabili nel Memory Storage, ovvero nell'area di SWAP.

cache_replacement_policy lru: questa direttiva definisce la politica e l'algoritmo di sostituzione file all'interno della cache.

La politica *LRU*, definita nella direttiva, è l'algoritmo più utilizzato in quanto rimpiazza gli oggetti che non sono più richiesti da tempo, creando spazio per nuovi oggetti.

memory_replacement_policy lru: questa direttiva risulta essere uguale alla precedente, l'unica differenza è che si occupa di ripulire la memoria di SWAP della cache.

forwarded_for: questa direttiva, se attivata, permette il mascheramento dell'indirizzo IP che sta navigando in rete. Questa regola, ammette solo due valori che sono "on" "off" che rispettivamente attivano e disattivano la direttiva.

high_response_time_warning: Questa direttiva rappresenta il numero di millisecondi massimo entro cui dare una risposta ad un client che ha effettuato una richiesta. Se la risposta non viene data entro il tempo definito in questa direttiva allora è da considerarsi scaduta.

Le direttive riportate nel listato 5.13, al fine di rendere più comprensibile la loro funzione, sono state riportate con i valori di configurazione standard applicati in azienda.

Listing 5.13: Direttive generali di squid

```
http_port 192.168.0.101:3128

cache_mem 8 MB

maximum_object_size: 4096 KB

minimum_object_size: 0 KB

maximum_object_size_in_memory: 8192 KB

cache_replacement_policy lru:

memory_replacement_policy lru:

forwarded_for on

high_response_time_warning 6000
```

Direttive ACL

Le direttive ACL, riportate nel listato 5.14, di Squid sono le direttive necessarie alla creazione delle catene di controllo degli accessi dei client che sono connessi al proxy. Questo tipo di direttive permette di definire dei file al cui interno saranno presenti particolari espressioni regolari deputate al riconoscimento di siti internet a cui si vuole negare l'accesso a determinati utenti .

All'interno del file `social.acl`, per esempio compariranno espressioni regolari in grado di identificare la navigazione in siti internet quali, ad esempio, *Facebook* o *Twitter*.

Questa tipologia di direttive deve essere composta in due parti, la prima si occupa di definire la ACL vera e propria definendone il nome e il file contenente le espressioni regolari, la seconda invece si occupa di permettere (allow) o negare l'accesso ai siti che vengono riconosciuti.

Generalmente questa tipologia di direttive viene utilizzata per negare l'accesso a siti poco affidabili quali siti di *gaming*, di *filesharing*, e *social network*

Listing 5.14: Direttive ACL

```
acl social url_regex "/etc/squid/squidblock/social.acl"
acl badlanguage url_regex "/etc/squid/squidblock/badlanguage.
acl"

acl entertainment url_regex "/etc/squid/squidblock/
entertainment.acl"

acl games url_regex "/etc/squid/squidblock/games.acl"
acl mp3 url_regex "/etc/squid/squidblock/mp3.acl"
acl pirates url_regex "/etc/squid/squidblock/pirates.acl"
acl redlight url_regex "/etc/squid/squidblock/redlight.acl"

http_access deny social
http_access deny badlanguage
http_access deny entertainment
http_access deny games
http_access deny mp3
http_access deny pirates
http_access deny redlight
```

direttive di ACL di accesso al proxy

Questo particolare insieme di direttive ACL permettono di garantire o negare l'accesso al proxy, quindi a Squid, solamente ad alcuni utenti della LAN. Supponiamo quindi di voler permettere l'accesso a Squid a tre categorie di elementi diversi come:

- * tutti i client della LAN
- * tutti i client il cui indirizzo IP fa parte di un determinato range
- * tutti i client i cui utenti sono in possesso delle credenziali

Le direttive ACL necessarie a permettere l'accesso al proxy di tutta la LAN sono riportate nel listato [5.15](#).

Listing 5.15: ACL di accesso per l'intera LAN

```
acl lan src 0.0.0.0/0.0.0.0
http_access allow lan
```

Come si nota, la definizione della ACL vera e propria si basa sugli indirizzi IP sorgenti. La presenza del range di IP 0.0.0.0/0.0.0.0 vicino al tag *src* significa esattamente che l'accesso al proxy è garantito a tutti gli IP sorgenti, e di conseguenza a tutta la LAN.

Le direttive ACL necessarie a permettere l'accesso al proxy ad un determinato range di IP sono riportate nel listato [5.16](#).

Listing 5.16: ACL di accesso per range di indirizzi IP

```
acl someIP src 192.168.0.5/192.168.0.25
http_access allow someIP
```

Come si nota, anche in questo caso la definizione della ACL vera e propria si basa sugli indirizzi IP sorgenti. La presenza del range di IP, che in questo caso è diverso da 0.0.0.0/0.0.0.0, vicino al tag *src* significa esattamente che l'accesso al proxy è garantito a tutti gli IP sorgenti che sono inclusi in quel range.

Per quanto riguarda le direttive di accesso tramite l'utilizzo di credenziali, la costruzione dell'intera ACL risulta diversa. La prima istruzione, infatti risulta essere:

```
auth_param basic program /usr/lib/squid/ncsa_auth /etc/squid-
/squidpasswd
```

questa istruzione determina il tipo di algoritmo, che risulta essere *basic*, ovvero l'algoritmo base per l'utilizzo di credenziali.

Oltre a determinare l'algoritmo tale istruzione si occupa di trovare il file con le corrispondenze password-utente che dovrà essere utilizzato e scansionato nella fase di autenticazione.

All'interno della definizione di ACL vera e propria si può notare che è stato aggiunto il tag *REQUIRED*, questo tag si occupa impedire l'accesso a meno di non essersi autenticati.

Le direttive descritte sono riportate nel listato [5.17](#)

Listing 5.17: ACL di accesso con credenziali

```
auth_param basic program /usr/lib/squid/ncsa_auth /etc/squid/
squidpasswd
acl protected proxy_auth REQUIRED
http_access allow protected
```


Capitolo 6

Verifica e validazione

Questo capitolo si preme di spiegare il modo eseguire i test per validare un firewall personalizzato

6.1 Introduzione

I test di verifica e validazione sono stati eseguiti al termine di ogni configurazione di firewall. L'attività di verifica sfrutta l'analisi dinamica che ha lo scopo di verificare il corretto funzionamento del sistema attraverso l'esecuzione dei software verificandone la correttezza rispetto alle regole applicate. L'insieme di tutti i test è presente nella documentazione aziendale.

In questo capitolo quindi sarà spiegato il modo di verificare il funzionamento corretto del firewall portando due esempi per ogni test. La verifica del funzionamento delle regole IPtables si basa sull'osservazione delle tabelle e delle catene in cui risiedono le regole.

Per quanto riguarda il proxy Squid, il funzionamento delle direttive lo si può osservare nel Log del software stesso oppure, nel caso di particolari ACL deputate al blocco di siti internet, la verifica avviene tramite browser.

Per quanto riguarda le regole IPtables, è necessario fare una premessa: Le verifica permette di capire se una regola esegue o meno la propria funzione, tuttavia non permette comprendere quali siano gli errori commessi. Detto questo, è necessario specificare, inoltre, che gli errori possibili sono solamente dovuti ad errate immissioni di indirizzi IP, porte di comunicazione e interfacce.

IPtables, infatti, non permette l'inserimento di regole sintatticamente errate.

Prima di procedere con le verifiche delle varie regole, è necessario verificare il corretto caricamento di moduli del kernel necessari. Per verificare tale situazione è sufficiente lanciare il comando:

```
/sbin/lsmod
```

Questo comando infatti mostra tutti i moduli del kernel caricati in memoria. Il mancato caricamento dei moduli necessari, comporta un malfunzionamento del sistema che può causare il blocco dell'intera rete.

6.2 Verifica Funzionamento IPtables

Verificare il corretto funzionamento delle regole IPtables è molto semplice. IPtables, infatti, mette a disposizione una serie di comandi che permettono di visualizzare le tabelle con le catene di regole associate. Tali comandi permettono inoltre di vedere il numero di pacchetti che ogni singola regola, all'interno della catena, controlla e gestisce.

I contatori delle regole sono essenziali in fase di verifica, infatti permettono di capire se effettivamente una regola è corretta o meno.

Supponendo, ad esempio di avere una regola così definita:

```
iptables -t nat -A POSTROUTING -o eth0 -s 192.168.0.1/24 -j
MASQUERADE
```

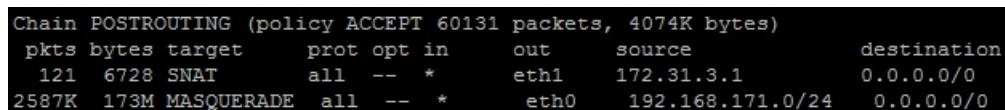
Nel caso in cui l'interfaccia `eth0` sia errata, o nel caso in cui la sotto rete `192.168.0.1/24` sia sbagliata, il contatore associato rimarrebbe fermo a 0 e la sotto rete non sarebbe in grado di avere un accesso ad Internet a causa di un'errata configurazione NAT.

Contrariamente, nel caso in cui la regola sia corretta osserveremo il contatore che viene incrementato.

Nonostante questo tipo di verifica faccia capire se la regola è errata o meno, come spiegato in sezione 6.1, non è possibile risalire con certezza all'errore commesso.

Supponendo quindi che la regola di NAT, citata in precedenza sia errata, non è permesso sapere quale sia l'errore. Nel caso in cui una regola risulti verificata è corretta allora è da considerarsi valida.

Come è possibile notare in figura 6.2 una regola di NAT corretta ha i contatori *pkts* e *bytes*, ovvero i contatori dei pacchetti e dei bytes trasmessi, che risultano essere diversi da zero.



Chain POSTROUTING (policy ACCEPT 60131 packets, 4074K bytes)									
pkts	bytes	target	prot	opt	in	out	source	destination	
121	6728	SNAT	all	--	*	eth1	172.31.3.1	0.0.0.0/0	
2587K	173M	MASQUERADE	all	--	*	eth0	192.168.171.0/24	0.0.0.0/0	

figura 6.1: Regola NAT validata

Per poter vedere tutti i contatori delle regole e quindi verificarne il funzionamento è sufficiente lanciare, da shell, i comandi:

iptables -nvL: questo comando mostra tutte le regole, e i relativi contatori, della tabella di Input Output del firewall.

iptables -t mangle -nvL: questo comando mostra tutte le regole, e i relativi contatori, della tabella di *mangle* del firewall.

iptables -t nat -nvL: questo comando mostra tutte le regole, e i relativi contatori, della tabella *nat* del firewall.

La figura 6.2 risulta essere un altro esempio di regole corrette, in questo caso è necessario specificare che dalle immagini sono stati rimossi gli indirizzi IP pubblici dei clienti come da direttive aziendali. e 6.3

Chain INPUT (policy DROP 106K packets, 10M bytes)									
pkts	bytes	target	prot	opt	in	out	source	destination	
4800	1099K	DROP	all	--	*	*	0.0.0.0/0	0.0.0.0/0	state INVALID
0	0	DROP	all	--	*	*	0.0.0.0/0	0.0.0.0/0	
0	0	ACCEPT	all	--	lo	*	0.0.0.0/0	0.0.0.0/0	
459K	44M	ACCEPT	all	--	*	*	0.0.0.0/0	0.0.0.0/0	state RELATED,ESTABLISHED
0	0	ACCEPT	tcp	--	eth0	*	21	0.0.0.0/0	tcp dpt:22 state NEW
49	2164	ACCEPT	tcp	--	eth0	*	0.0.0.0/0	0.0.0.0/0	tcp dpt:1723 state NEW
0	0	ACCEPT	47	--	eth0	*	0.0.0.0/0	0.0.0.0/0	
868K	105M	ACCEPT	all	--	eth1	*	0.0.0.0/0	0.0.0.0/0	
3	156	ACCEPT	all	--	tap+	*	0.0.0.0/0	0.0.0.0/0	

figura 6.2: Regole di input validate

In figura 6.3, è possibile notare che le prime due regole hanno i contatori fermi a zero, in questo caso le regole risultano errate e quindi è necessario modificarle.

e 6.3

Chain FORWARD (policy DROP 1113K packets, 57M bytes)									
pkts	bytes	target	prot	opt	in	out	source	destination	
0	0	LOG	all	--	*	*	0.0.0.0/0	0.0.0.0/0	LOG flags 0 level 4 prefix 'Fragmented: '
0	0	DROP	all	--	*	*	0.0.0.0/0	0.0.0.0/0	
42323	1753K	LOG	all	--	*	*	0.0.0.0/0	0.0.0.0/0	state INVALID LOG flags 0 level 4 prefix 'Invalid: '
42323	1753K	DROP	all	--	*	*	0.0.0.0/0	0.0.0.0/0	state INVALID
56M	34G	ACCEPT	all	--	*	*	0.0.0.0/0	0.0.0.0/0	state RELATED,ESTABLISHED
0	0	DROP	udp	--	eth1	eth0	192.168.171.0/24	0.0.0.0/0	state NEW udp dpt:445
2393	134K	DROP	tcp	--	eth1	eth0	192.168.171.0/24	0.0.0.0/0	state NEW tcp dpt:445
644	50738	DROP	udp	--	eth1	eth0	192.168.171.0/24	0.0.0.0/0	state NEW udp dpts:137:139
632	35256	DROP	tcp	--	eth1	eth0	192.168.171.0/24	0.0.0.0/0	state NEW tcp dpts:137:139
1732K	127M	ACCEPT	udp	--	eth1	eth0	192.168.171.0/24	0.0.0.0/0	state NEW udp dpt:53

figura 6.3: Regole di forward non validate

6.3 Verifica Funzionamento IPtables-Layer7

La procedura di verifica e validazione di queste regole procede esattamente allo stesso modo di quanto descritto in sezione 6.2. Tuttavia, si è preferito descrivere la loro verifica separatamente a causa di una diversità fondamentale.

Come riportato in sezione 5.3 questa tipologia di regole permette di bloccare le comunicazioni di rete generate da diversi protocolli applicativi. La diretta conseguenza di quanto appena detto è immediata, infatti, per testare la validità e sufficiente eseguire il software di cui si vuole inibire il protocollo, e verificarne il mancato funzionamento. Supponendo infatti, di aver bloccato il protocollo P2P tramite una regola, allora, nel caso in cui la regola sia corretta, qualsiasi software che sfrutta il protocollo P2P non sarà abilitato ad uscire nella rete e di conseguenza non funzionerà.

Tuttavia la tipologia di verifica migliore risulta essere uguale a quella descritta in sezione 6.2 poiché permette di verificare tutte le regole contemporaneamente. In figura 6.4 e in 6.5 è possibile notare due esempi di regole funzionanti.

6.4 Verifica Funzionamento server proxy

Come descritto in precedenza, le direttive di Squid possono essere verificate in due modi. Per quanto riguarda le direttive generali, ovvero le direttive minime necessarie al suo funzionamento si possono verificare direttamente durante la fase di avvio del servizio o all'interno dei log del proxy stesso.

```
Chain PREROUTING (policy ACCEPT 411 packets, 83009 bytes)
pkts bytes target      prot opt in     out     source      destination
7640 1259K      all  --  *      *      0.0.0.0/0   0.0.0.0/0   LAYER7 l7proto unset
5552 3330K      all  --  *      *      0.0.0.0/0   0.0.0.0/0   LAYER7 l7proto unknown

Chain INPUT (policy ACCEPT 235 packets, 59536 bytes)
pkts bytes target      prot opt in     out     source      destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target      prot opt in     out     source      destination

Chain OUTPUT (policy ACCEPT 168 packets, 20896 bytes)
pkts bytes target      prot opt in     out     source      destination
4      248 DROP      all  --  *      *      0.0.0.0/0   0.0.0.0/0   LAYER7 l7proto facebook
```

figura 6.4: Regole di blocco layer7 validate

```
Chain PREROUTING (policy ACCEPT 1396 packets, 185K bytes)
pkts bytes target      prot opt in     out     source      destination
9240 1515K      all  --  *      *      0.0.0.0/0   0.0.0.0/0   LAYER7 l7proto unset
6301 3439K      all  --  *      *      0.0.0.0/0   0.0.0.0/0   LAYER7 l7proto unknown

Chain INPUT (policy ACCEPT 1149 packets, 140K bytes)
pkts bytes target      prot opt in     out     source      destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target      prot opt in     out     source      destination

Chain OUTPUT (policy ACCEPT 1759 packets, 161K bytes)
pkts bytes target      prot opt in     out     source      destination
4      248 DROP      all  --  *      *      0.0.0.0/0   0.0.0.0/0   LAYER7 l7proto facebook
19     1634 DROP      all  --  *      *      0.0.0.0/0   0.0.0.0/0   LAYER7 l7proto bittorrent
```

figura 6.5: Regole di blocco layer7 validate

Ogni qual volta viene avviato il servizio, infatti le direttive generali, come ad esempio quelle riportate in sezione 5.4, devono essere caricate nel sistema e nel caso una di queste non sia corretta il servizio non si avvia e di conseguenza le direttive generali non sono considerabili valide. Quanto scritto in precedenza, vale soprattutto per le direttive generali minime, come ad esempio, la direttiva che definisce l'indirizzo IP e la porta di ascolto del proxy.

Per quanto riguarda le direttive che definiscono le liste di controllo degli accessi, i test di verifica risultano meno veloci, ma non meno immediati. Supponiamo, infatti che sia necessario validare e verificare la direttiva definita nel listato 6.1 che si occupa di negare l'accesso ai siti di social network come ad esempio *Facebook*. Per verificarne il funzionamento è sufficiente provare a navigare in uno dei siti definiti all'interno del file `social.acl`.

È bene specificare, a scanso di equivoci, che questa tipologia di test risulta applicabile solamente se il client, con il quale sto navigando, risulta essere connesso al proxy.

Se la pagina internet che si presenta è esattamente quella riportata in figura 6.6 allora la regola è da considerarsi corretta e di conseguenza valida.

Listing 6.1: Esempio di ACL che blocca i siti di social network

```
acl social url_regex "/etc/squid/squidblock/social.acl"
```

```
http_access deny social
```

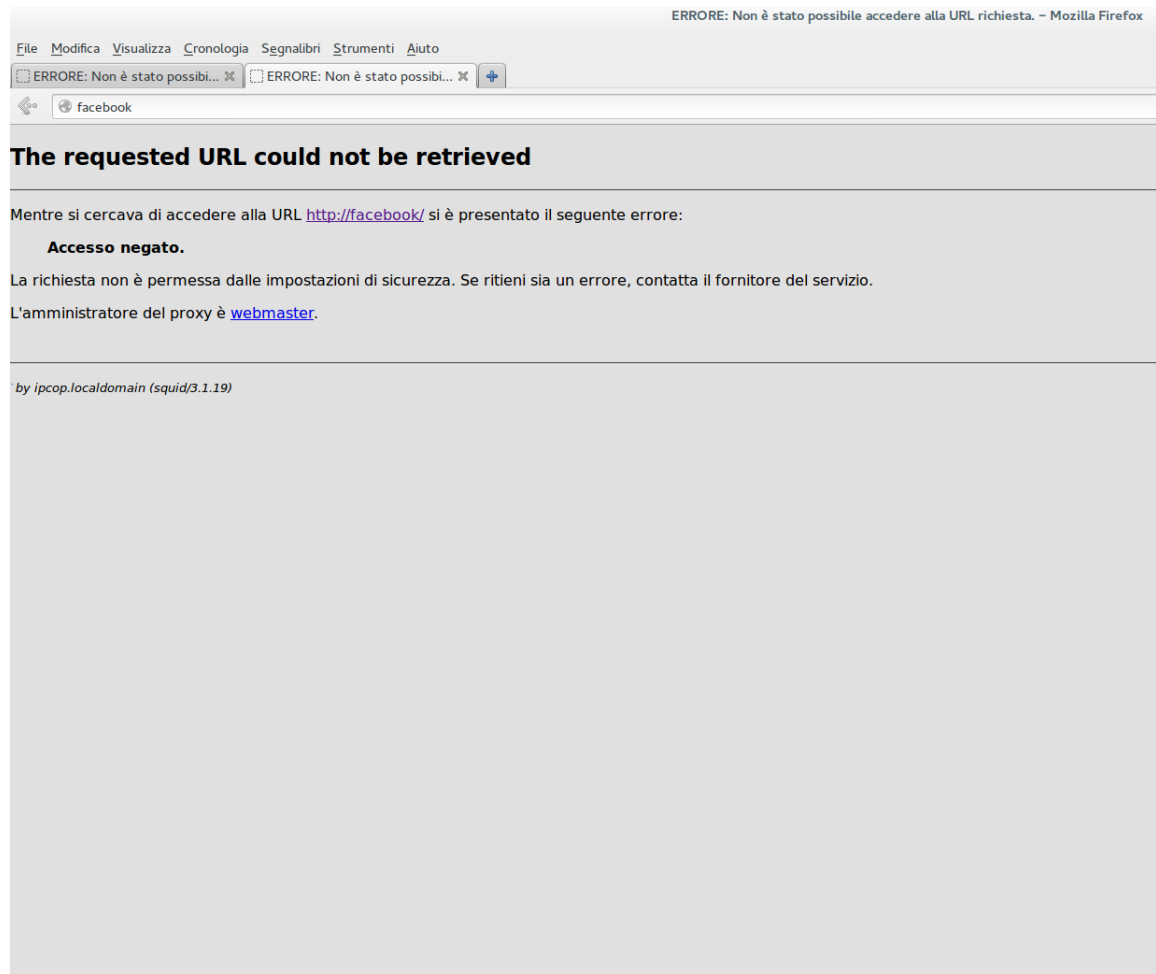


figura 6.6: Esempio di blocco siti internet da parte di Squid

6.5 Conclusioni Verifica

Quanto descritto nelle sezioni precedenti, è stato il modo di verificare e di validare le regole IPtables e le direttive Squid utilizzato durante il periodo di stage.

Nella verifica di reti consistenti sono stato seguito dal tutor interno al fine di non omettere la verifica di alcune regole, di valutarne la correttezza e di correggere correttamente gli errori.

Capitolo 7

Conclusioni

7.1 Raggiungimento degli obiettivi

gli obiettivi, come citato in sezione 2.3, erano suddivisi in obbligatori e desiderabili. Gli obiettivi obbligatori sono stati raggiunti tutti appieno. Per quanto riguarda i requisiti desiderabili ne sono stati raggiunti due su tre. La motivazione risiede nel fatto, che durante il periodo di stage, non sono state implementate reti di tipo WAN né reti VPN. L'apprendimento di tale tecnologia, infatti, non era presente negli obiettivi obbligatori, inoltre per apprendere tale tecnologia il tempo di durata dello stage doveva essere maggiore. In tabella 7.1 è possibile vedere un tracciamento del raggiungimento degli obiettivi.

Obiettivo	Tipo Obiettivo	Raggiungimento
Apprendimento NAT	obbligatorio	Si
Apprendimento IPtables	obbligatorio	Si
sviluppo di strumenti firewall	obbligatorio	Si
sviluppo di strumenti firewall layer7	obbligatorio	Si
configurazione di regole firewall per la sicurezza	obbligatorio	Si
configurazione di regole firewall NAT	desiderabile	Si
configurazione server Proxy	desiderabile	Si
Configurazione reti WAN e VPN	desiderabile	No

tabella 7.1: Consuntivo degli obiettivi raggiunti

7.2 Conoscenze acquisite

Le conoscenze acquisite sono state molteplici. Nello svolgimento delle attività accademiche questo aspetto dell'informatica non è tenuto molto in considerazione,

motivo per cui le conoscenze che si acquisiscono sono minime anche se sufficienti. Durante l'attività di stage sono state acquisite conoscenze profonde nell'utilizzo del kernel Linux, di strumenti di monitoraggio e soprattutto nella creazione di reti ed infrastrutture. Nello specifico le conoscenze, di genere informatico, acquisite sono state le seguenti:

- * conoscenza profonda del funzionamento di un kernel Linux
- * conoscenza profonda dei problemi di sicurezza legata ai sistemi operativi
- * conoscenza del modo di implementare reti ed infrastrutture
- * conoscenze nelle creazioni di firewall
- * conoscenze nell'implementazione di server proxy
- * conoscenze del funzionamento delle comunicazioni tra server
- * conoscenze profonde dei protocolli di comunicazione

A livello personale inoltre è stato acquisito il modo corretto di comunicazione con i clienti aziendali e soprattutto il modo di lavorare bene in una delle migliori aziende informatiche. Tengo a precisare che la motivazione fondamentale per la quale ho svolto lo stage in questo ambito era proprio quella di acquisire nuove conoscenze che prima mancavano al fine avere una buona base per il futuro che mi aspetta.

7.3 Considerazioni finali

All'inizio dello stage le conoscenze in mio possesso non sarebbero state sufficienti per portare a termine le creazioni dei vari firewall, ma grazie allo studio delle nuove tecnologie fatto in azienda e grazie all'affiancamento con i colleghi, sono riuscito ad acquisire le conoscenze minime per poter iniziare consapevolmente le progettazioni e le creazioni dei prodotti. L'affiancamento di una persona durante tutta la durata del progetto, a causa della grande mole di lavoro, è stata di grande aiuto in quanto conosceva già le tecnologie e gli strumenti che da utilizzare per creare e progettare firewall.

Questo mi ha permesso di apprendere più velocemente i concetti essenziali da applicare.

Le maggiori difficoltà avute, sono state nella fase iniziale della progettazione del firewall e soprattutto nella creazione delle regole, in quanto all'inizio non si conoscevano bene i meccanismi delle reti aziendali. La parte più complessa da progettare e realizzare è stato la realizzazione del firewall a livello 7.

Come detto nei capitoli precedenti per la creazione di tale firewall a livello 7 è stata sfruttato un progetto pre-esistente, tuttavia si sono incontrati enormi difficoltà nell'adattare il progetto ai firewall, in quanto si è reso necessario modificare e ricompilare il kernel Linux. In alcuni casi i firewall realizzati, inoltre avevano delle personalizzazioni molto complesse come ad esempio il *traffic shaping* che serviva a bilanciare il traffico in uscita su due o più linee.

L'esperienza di stage presso un'azienda credo sia molto importante per uno studente prossimo ad entrare nel mondo del lavoro, in quanto permette di conoscere le dinamiche di lavoro adottate in azienda, le quali differiscono da quelle adottate nei progetti proposti durante il corso di studi.

Questo tipo d'esperienza inoltre aiuta a comprendere non solo il modo di lavorare in azienda, ma anche il modo di rapportarsi con i clienti dell'azienda stessa aiutando anche ad ampliare i propri orizzonti lavorativi.

Oltre al bagaglio culturale e alle conoscenze acquisite credo sia stato molto importante l'ottimo rapporto con il tutor aziendale e con il team di sistemisti nel quale sono stato inserito, che mi ha permesso di completare gli obiettivi fissati grazie alla possibilità di scambiare opinioni e conoscenze con loro e di vedere dispositivi già realizzati e installati al lavoro.

Bibliografia

Riferimenti bibliografici

Gheorghe, Lucian. *Design and implementig Linux firewall and QoS*. PACKT publishing, 2007.

Tanenbau, Andrew S. *Computer Network*. Pearson Education Italia, 2008.

Siti Web consultati

IPCop-firewall. URL: <http://ipcop.org/docs.php>.

L7-Filter - Application Layer Packet Classifier. URL: <http://l7-filter.sourceforge.net/>.

Linux Voyage - x86 Embedded Linux. URL: <http://linux.voyage.hk/>.

Netfilter - Netfilter and IPtables project. URL: <http://www.netfilter.org/>.

PFSense - firewall. URL: <http://www.pfsense.org/>.

Squid-Optimising Web Delivery. URL: <http://www.squid-cache.org/Doc/config/>.