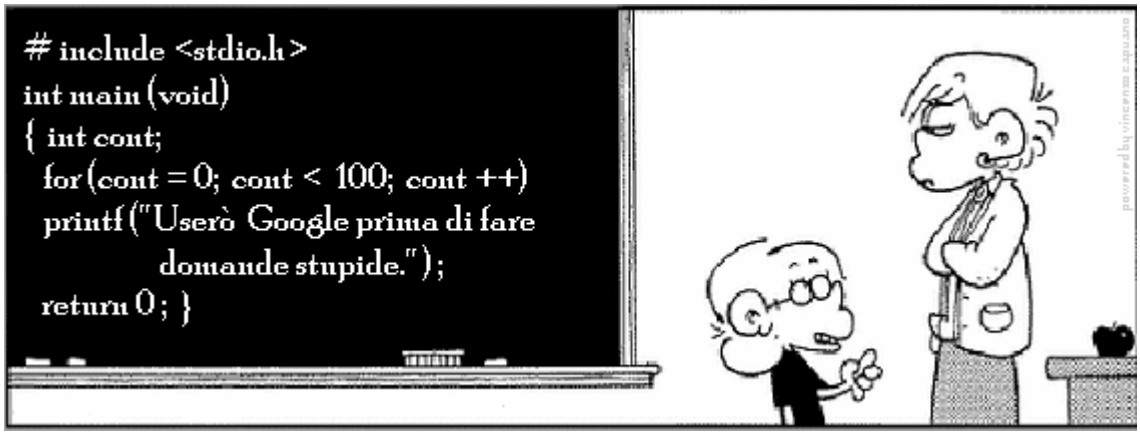


Firewall

Capuano Vincenzo
Guida Mario
Brasiello Domenico
Manco Giancarmine

www.sitoserio.it

Prefazione

Molti ritengono di non correre un reale rischio di intrusione nei propri computer, pensando che i propri dati, le proprie abitudini, i propri gusti **non siano di interesse per nessuno**.

Ma si sbagliano.

Molti ritengono che sia sufficiente **evitare di navigare in Internet** per essere al sicuro da virus o programmi dannosi.

Ma si sbagliano.

Molti non sanno neppure che esistono, oltre ai virus informatici, programmi che subdolamente **sottraggono informazioni**, prendono il controllo del computer e, inosservati, si propagano agli altri computer.

Oggi sia gli hackers che i crackers sono in grado di individuare un computer non protetto in mezzo a migliaia di altri sistemi grazie a specifici programmi di ricerca automatica.

Gli strumenti che offrono il massimo della sicurezza sono sostanzialmente due: i programmi **antivirus** aggiornati e i **firewall**.

Un **programma antivirus** protegge le macchine da **attacchi informatici**, impedisce la proliferazione degli stessi e di fatto rendono sicura la navigazione in Internet. A patto che siano **costantemente aggiornati**.

Un **firewall** può essere sia hardware che software ed offre una serie di strumenti per proteggere i dati e le informazioni che si conservano sul computer, oltre a **ridurre al minimo il rischio di intrusioni** da parte di hackers e crackers quando si è connessi ad Internet.

Vincenzo - Mario - Domenico - Giancarmine

La teoria è quando si sa tutto ma non funziona niente.
La pratica è quando funziona tutto ma non si sa il perché.
In ogni caso si finisce sempre a coniugare la teoria con la pratica:
non funziona niente e non si sa il perché.

Albert Einstein

Indice

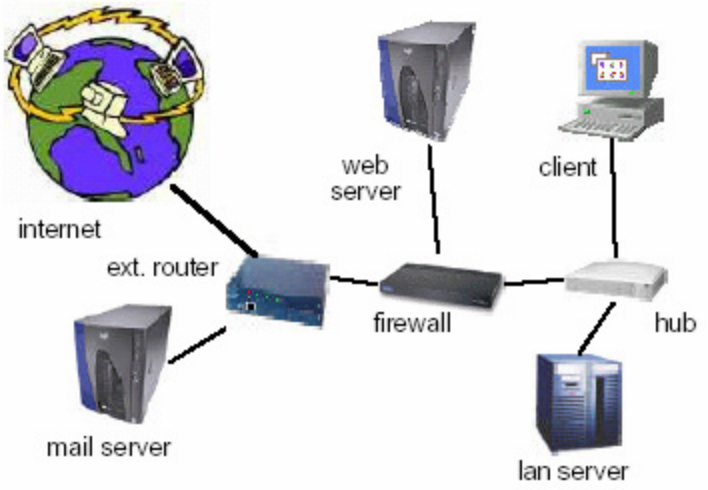
Introduzione

1.1 Che cosa è un firewall?

In un cantiere edile, vengono poste delle guardie all'uscita per verificare che non venga rubato materiale di costruzione; tutte le sere le guardie controllano gli operai. Tutte le sere uno di questi esce tutto sporco e con una carriola vuota, le guardie lo lasciano passare. Qual è il risultato? L'operaio ruba indisturbato le carriole! Di chi è la colpa? Quasi sicuramente di chi ha deciso dove devono essere posizionate le guardie o chi ha pensato le regole di controllo, oppure più semplicemente la colpa è delle guardie che non riescono a vigilare in modo accurato. Il problema può essere risolto egregiamente impartendo alle guardie degli ordini più accurati, oppure ingaggiando un corpo di guardie più efficiente.

L'esempio precedente può dare un'idea della sicurezza in una rete. A volte si ha solo l'illusione di essere immuni da attacchi esterni. Quando siamo connessi in Internet i nostri dati, il nostro computer, la nostra incolumità sono sempre in balia di attacchi: chiunque può entrare sul nostro computer se non si prendono i dovuti accorgimenti. La soluzione al problema è il firewall, ovvero un "guardiano" che vigila costantemente sulla nostra macchina.

Lo schema di lato rappresenta una rete aziendale "comune" con alcune caratteristiche riassunte nel rettangolo in basso a sinistra, si adatta abbastanza bene ad una serie di considerazioni relative a sicurezza, funzionamento e complessità.



Questo schema rappresenta una rete aziendale con le seguenti caratteristiche:
- collegamento ad Internet
- fornitura servizi Internet
- accesso del web server al lan server
- accesso del lan server al mail server
- rete interna connessa alle altre

N.B. Questa configurazione è una configurazione di esempio utilizzata per evidenziare delle casistiche.

Quando si parla di firewall può venire subito in mente quel meccanismo antincendio che presente in un edificio fa si che in caso di incendio il fuoco non passa spargersi da un punto all'altro della struttura in esame. In teoria, un firewall di Internet ha uno scopo simile: impedisce ai vari "pericoli" presenti su Internet di spargersi all'interno di una rete privata. In pratica, un firewall di Internet somiglia più ad un fossato di un castello medievale che ad un firewall di un edificio moderno. Per semplicità indicheremo il firewall di Internet con il termine firewall.

Un firewall è un componente attivo che seziona e collega due o più tronconi di rete. Usualmente la rete viene divisa in due sottoreti: una, detta esterna, comprende l'intera Internet mentre l'altra, detta interna, comprende una sezione più o meno grande di un insieme di computer locali.
Grazie alla sua posizione strategica, il firewall risulta il posto migliore ove imporre delle logiche di traffico per i pacchetti in transito e/o eseguire un monitoraggio di tali pacchetti.
La funzione principale del firewall è quella di proteggere i sistemi informatici presenti nella sezione interna dal "caos" presente nel lato esterno. Il firewall agisce sui pacchetti in transito da e per la zona interna potendo eseguire su di essi operazioni di:

- controllo
- modifica
- monitoraggio

Questo grazie alla sua capacità di "aprire" il pacchetto IP per leggere le informazioni presenti sul suo header.

Struttura di un pacchetto TCP/IP

Header IP (20 bytes)	1	2	3	4	5	6	7	8	9	10	
	Ver / TOS	DSF	Lungh. totale pacchetto		Identification		Flags	Offset	TTL	Proto (TCP)	
	11	12	13	14	15	16	17	18	19	20	
	Checksum		**Indirizzo IP sorgente (xxx.xxx.xxx.xxx)**				**Indirizzo IP destinazione (yyy.yyy.yyy.yyy)**				
Header TCP (20 bytes)	1	2	3	4	5	6	7	8	9	10	
	Porta src (aaaa)		**Porta dest (bbbb)**		Numero di sequenza del pacchetto				Ack Number		
	11	12	13	14	15	16	17	18	19	20	
	Ack Number		Offset / Flags		Windows size		Checksum		Urgent pointer		
Dati	Dati trasportati dal pacchetto										

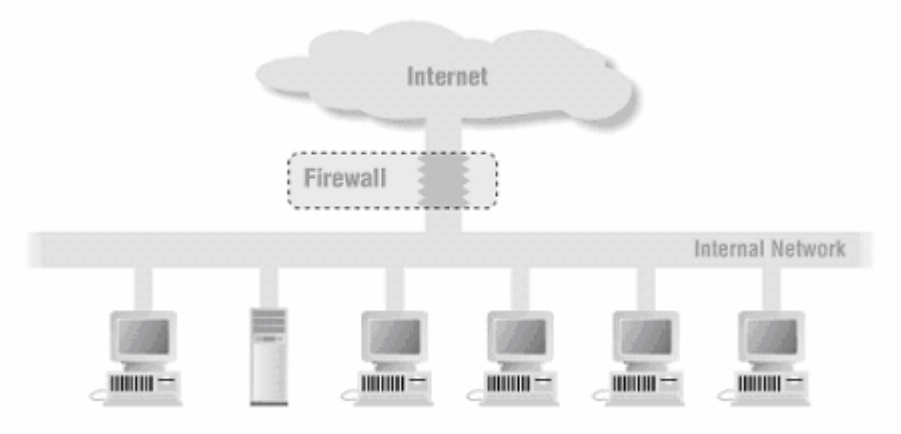
I firewall di questo tipo sono detti "packet-type". Esiste un'altra tipologia, quella dei firewall di tipo "application-type", che si differenzia dalla prima in quanto agisce sull'informazione contenuta nei dati che il pacchetto trasporta e non sull'header.
I firewall "packet-type" non possono, per esempio, individuare un virus perché non agiscono sul contenuto dei dati trasportati dal pacchetto IP.

Materialmente il firewall è un combinazione hardware e software. La parte hardware possiede due o più schede di rete su cui viene fatto girare un ambiente operativo che analizza e gestisce il traffico dei pacchetti in base ad una configurazione data dall'amministratore della rete.
L'ambiente operativo può essere, per cosi dire, chiuso e le sue funzionalità sono decise dal costruttore e si può agire solo modificando la configurazione interna ossia modificando le regole di selezione dei pacchetti o può essere aperto e permettere, oltre che di modificare la configurazione, anche di modificare e ampliare le sue capacità operative.

Questa filosofia, adottata dal sistema operativo Linux, permette al suo firewall di operare al pari di un qualsiasi altro dispositivo commerciale dal costo di decine di migliaia di euro includendo inoltre funzionalità che non si trovano su alcun altro prodotto e con la possibilità di eseguire, in relazione per esempio al passaggio di

pacchetti di dati prestabiliti, script personalizzati che possono inviare mail, accendere sirene acustiche, etc…

Un firewall molto spesso viene installato nel punto dove la rete è connessa a Internet, come mostrato in figura



in tal caso tutto il traffico che proviene da Internet dovrà passare obbligatoriamente attraverso il firewall.

Gli attacchi a questa rete possono provenire da:

- Client remoto legittimo
- Client remoto illegittimo
- Client locale legittimo
- Client locale illegittimo

Nel caso in cui poi siano stati compromessi dei sistemi, è possibile che gli attacchi provengano da qualunque punto, compresi server, router o firewall stesso.

Un firewall risponde a delle regole, anche complesse, "come le guardie del cantiere" ma non risponde comunque a tutti i possibili problemi di sicurezza.

Se si sta costruendo un firewall, la prima cosa di cui bisogna preoccuparsi è capire cosa proteggere. Quando ci si connette ad Internet, sono a rischio:

- i dati: le informazioni che si hanno sui propri computer
- le risorse: i computer
- la reputazione: un intruso può apparire su Internet con l'identità di un altro

La parte più difficile nella progettazione di un firewall non è installarlo, ma configurarlo, mantenerlo e mantenere sicura la rete nel tempo. Specialmente al variare delle richieste e dei servizi forniti, avere chiari i possibili attacchi a cui potenzialmente è soggetta la rete.

Un firewall abbandonato a se stesso "invecchia" al progredire delle tecnologie di attacco e in funzione dello sviluppo delle risorse della rete, delle vulnerabilità dei client e dei protocolli che utilizzano. Presto o tardi un firewall "sicuro", senza adeguata manutenzione diventerà "insicuro".

I punti di forza di un firewall sono:

1) corretto posizionamento nella rete
2) studio dei servizi che si vogliono fornire
3) impostazione delle regole pianificate
4) negazione di tutto il resto (prova del 9 degli studi fatti)
5) manutenzione e aggiornamento del firewall
6) manutenzione e aggiornamento della rete
7) verifica periodica dell'efficacia del sistema
8) consapevolezza dei limiti dei firewall

… i punti di debolezza invece:

1) configurazioni di default
2) mancanza di amministrazione
3) configurazioni troppo permissive verso host "insicuri"
4) installazione di applicazioni o estensioni insicure
5) stratificazione di situazioni non analizzate
6) sovrabbondanza di servizi pubblicamente accessibili
7) concentrazione di sicurezza in un unico punto e il dilagare di insicurezza circostante
8) suscitare un falso senso di sicurezza per il fatto di "esserci"

1.2 I limiti di un firewall

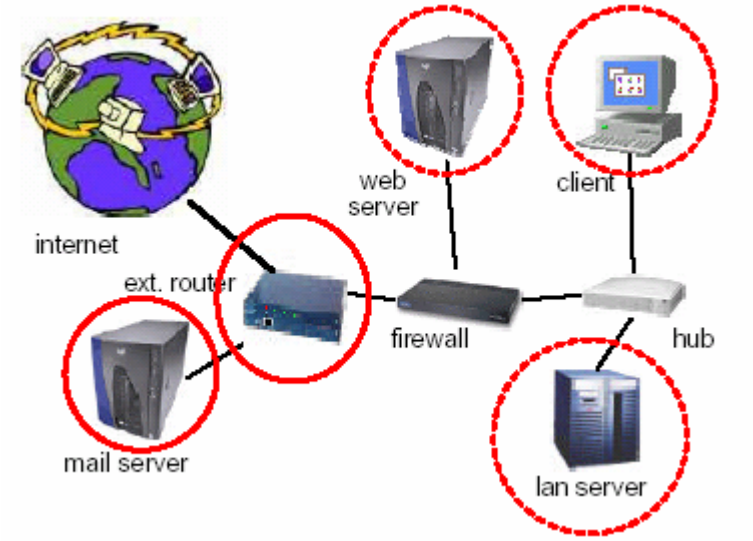
Vi sono due tipologie principali che generano problemi di sicurezza:

- **configurazioni non corrette:** si riscontrano problematiche che rientrano in questa categoria quando tramite una configurazione apposita del "sistema firewall" si sarebbero potuti evitare degli incidenti o quando si riscontrano impostazioni che forniscono dei punti di attacco superflui rispetto al funzionamento del sistema stesso. Questa tipologia è imputabile ad errori umani, errori di valutazione e conseguente impostazione.

- **problematiche tecniche:** si riscontrano problematiche che rientrano in questa categoria quando nonostante siano state impostate le configurazioni a regola d'arte sul firewall e non vi siano punti di attacco superflui, sia possibile forzare, sfruttando degli errori di programmazione o delle vulnerabilità del firewall stesso, le regole impostate ed accedere indebitamente ad un

componente che "teoricamente" si riteneva protetto. Questa tipologia è imputabile ad errori del firewall, errori di programmazione o limiti del sistema.

In generale la configurazione di un firewall non deve essere esclusivamente operativa ma deve essere sicura.



Sono evidenziati in rosso i componenti della rete che non sono protetti dal firewall rispetto ad attacchi provenienti da Internet (sono a monte del firewall).
A seconda delle tipologie di firewall e di traffico permesso nelle configurazioni è possibile che anche macchine difese dal firewall vengano attaccate con successo (sono le macchine evidenziate in rosso tratteggiato).

Esempi di configurazioni non corrette possono essere:

- permettere il traffico dal mail server (insicuro) al web server o alla rete interna (LAN server)
- permettere traffici non specificatamente legati ai servizi che si vogliono fornire verso le macchine della rete pubblica (nel nostro esempio, traffico ftp, smtp, verso il web server)
- permettere traffico verso eventuali servizi aperti sul firewall
- permettere traffico dal web server alla rete interna (qualora vi siano servizi dinamici di consultazione di database deve essere studiato un meccanismo SICURO di consultazione delle informazioni)
- permettere traffico da Internet verso la rete interna (qualora vi siano esigenze di telelavoro deve essere studiato un meccanismo SICURO di gestione)

E' possibile anche essere vittime di errori più difficili da individuare come:

- errori di configurazione su servizi pubblici (verso i quali sia permesso l'accesso pubblico attraverso il firewall)
- vulnerabilità nei servizi pubblici o loro add-on insicuri
- installare agenti (cgi, asp o simili) insicuri che permettano operazioni non volute
- predefinire delle regole ed applicarle indiscriminatamente al firewall

La struttura della rete è stata semplificata prendendo in considerazione i segmenti interessati da possibili errori di configurazione del firewall. Queste possono rientrare nelle seguenti tipologie:
- configurazioni di default
- configurazioni con specifiche negazioni e passaggio di quanto non specificatamente bloccato
- configurazioni non sufficientemente restrittive
- sistema ospite non sicuro
- relazioni di fiducia con host non protetti
- mancanza di comprensione delle problematiche di sicurezza inerenti una specifica configurazione
- errori di pianificazione della rete
- errori di valutazione dei servizi
- carenza di conoscenze avanzate su reti e protocolli

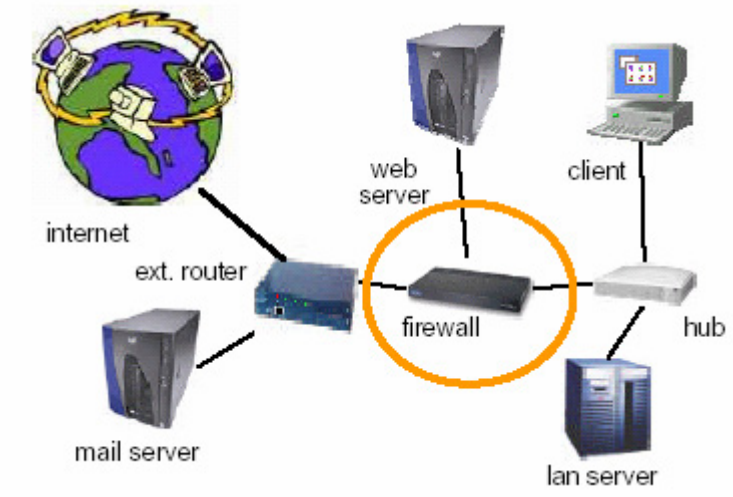
1.3 Problematiche tecniche

E' appurato che praticamente tutti i firewall abbiano avuto, essi stessi, problemi di sicurezza legati alla tecnologia utilizzata: errori di programmazione, errori di gestione delle eccezioni e quanti altri.

Una nota deve comunque essere fatta sulla gestione e sulla manutenzione del firewall:

- prevenire gli attacchi al sistema firewall verificandone l'efficacia periodicamente
- seguire e verificare gli aggiornamenti del sistema relativamente ai problemi di sicurezza riscontrati (con comunicazione dei produttori o pubblicazione su siti specializzati)
- analizzare i log (accessi) comprendendo gli eventi della rete e la loro reale gravità
- aggiornare e mantenere gli host della rete, client compresi (che possono essere vulnerabili anche se difesi dal firewall)
- aggiornare e mantenere i servizi della rete (che possono essere vulnerabili anche se difesi dal firewall) sono tutte attività necessarie ad una corretta amministrazione dei sistemi e ad una prevenzione degli incidenti.

E' doveroso ricordare che l'utilizzo di personale specializzato aiuta ad identificare e risolvere correttamente i problemi.



Le problematiche tecniche legate al firewall possono rientrare nelle seguenti tipologie:
- limiti di configurazione e mancanza di flessibilità nell'impostazione delle regole volute
- suscettibilità a specifici attacchi
- lentezza negli aggiornamenti
- limiti di analisi del traffico (non tutti i firewall arrivano allo stesso livello!)
- mancanza di protezione del traffico "legittimo"
- reazione automatica a situazioni della rete
- accentramento delle difese della rete in un unico strumento

1.4 I tipi di attacchi

Ci sono molti tipi di attacchi sui sistemi, e molti modi per catalogarli:

- intrusione
- Denial of Service (DoS - rifiuto di servizio)
- furto di informazioni

1.4.1 Intrusione

Gli attacchi più comuni sui sistemi sono le intrusioni; con le intrusioni, gli assalitori usano i computer delle loro vittime come se loro fossero gli utenti legittimi.

Gli assalitori hanno parecchi modi di trovare l'accesso, variando dal "social engineering attacks" (una volta ricavato il nome di un utente della rete privata, si contatta un amministratore del sistema e fingendo di essere quella persona

si cerca di ottenere la password di accesso; ottenuta la password si ha accesso alla rete), alla semplice "guesswork" (si effettuano una serie di prove in cui si cerca di individuare un nome utente e password validi).

I firewall aiutano a prevenire le intrusioni in vari modi. Idealmente, loro rendono impraticabile tutti i modi di ottenere l'accesso ad un sistema senza conoscere il nome dell'account e la password.

1.4.2 Denial of Service (DoS)

Un attacco Denial of Service impedisce all'utente vittima di utilizzare alcuni servizi presenti sul proprio computer.

Nel 1994, gli scrittori Josh Quittner e Michelle Slatalla furono vittime di una "bomba di posta elettronica". Apparentemente l'attacco fu una ritorsione per un articolo sulla comunità degli hacker, che loro avevano pubblicato nel periodico WIRED. Una serie interminabile di e-mail sommerse il loro servizio di posta elettronica tale che le altre comunicazioni non potessero essere inoltrate; in questo modo il loro collegamento ad Internet fu spento completamente.

Anche se dei casi di sabotaggio elettronico possono essere di tipo distruttivo, sempre più spesso seguono il modello visto nel caso di Quittner-Slatalla (flooding). Anche se questo tipo di attacco è il più semplice e il più diffuso, esistono anche tipi di attacchi che disabilitano, dirottano o sostituiscono i servizi.

1.4.3 Furto di informazioni

Alcuni tipi di attacchi permettono ad un assalitore di trovare i dati senza dovere usare direttamente i propri computer.

Il furto di informazioni non ha bisogno di essere attivo o particolarmente tecnico.

Le persone che vogliono scoprire informazioni su una certa persona potrebbero semplicemente chiamarla e chiedere (forse fingendosi qualcuno che ha diritto a sapere): questo è un furto di informazioni attivo. Se, invece, la persona interessata fornisce i propri dati, ad esempio il proprio numero di telefono, si parla di furto di informazioni passivo.

La maggior parte di persone che rubano informazioni tentano di avere l'accesso ai computer cercando nome utente e password. Fortunatamente per loro, e sfortunatamente per gli altri, quello è il genere di informazioni più semplice da ottenere quando si è in rete. Nome utente e password vengono

utilizzati all'inizio di molte interazioni in rete, e tali informazioni possono essere riusate nella stessa forma.

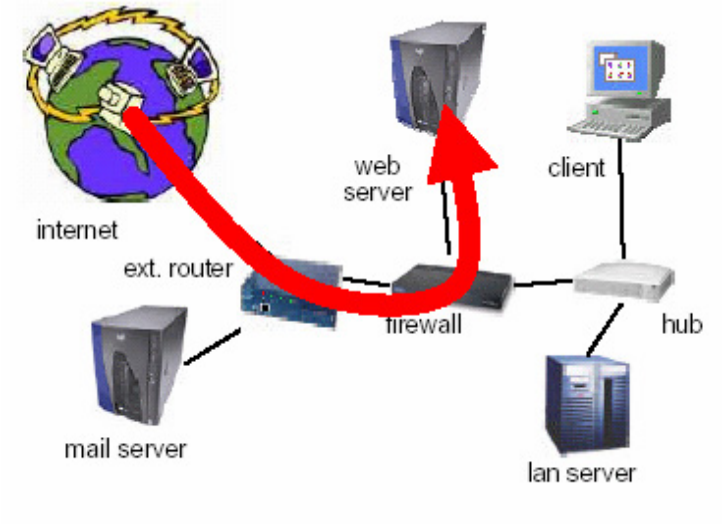
Un modo per catturare le informazioni riservate all'interno di una rete è quello di utilizzare uno "sniffer".

1.5 Attraversare un firewall

Nell'analizzare un firewall è interessante mettersi nell'ottica del traffico e "studiare le reazioni del firewall".

Faremo tre esempi di traffico permesso per vedere le reazioni sugli host dietro il firewall:

Esempio 1:

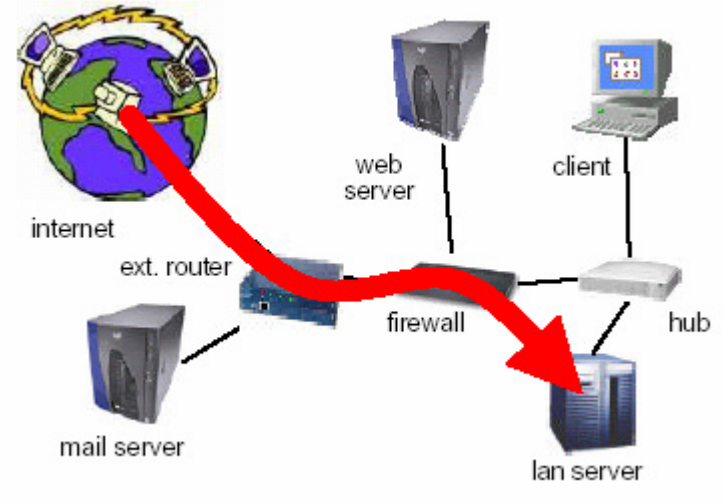


Da Internet un utente effettua una richiesta al web server:

" GET /cgi-bin/unsecure.cgi?
cat%20/etc/passwd HTTP/1.0"

sul firewall questa richiesta viene considerata legittima ed inoltrata al web server che trasmette in chiaro la lista degli utenti e relative password criptate.

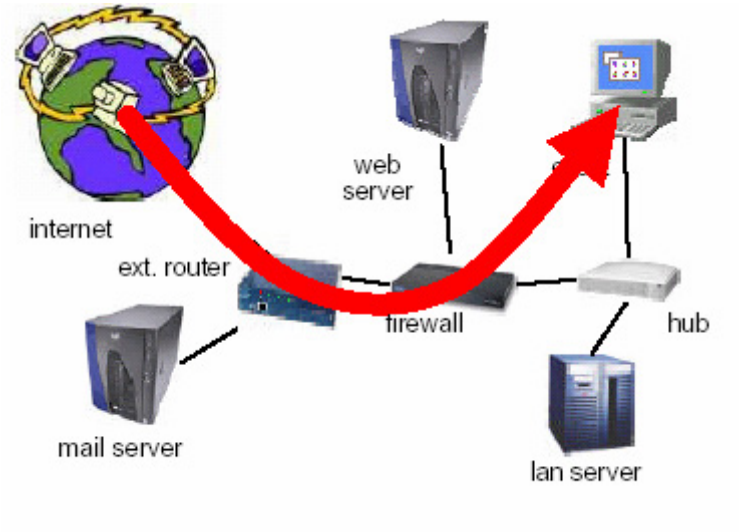
Esempio 2:



Da Internet e' accessibile un servizio (ftp, smtp, web, pop3, etc.) sul server.

Questo servizio e' vulnerabile ad un attacco che permette di eseguire comandi sul sistema. L'attaccante prende il controllo del lan server e può attaccare qualsiasi sistema sulla lan.

Esempio 3:



Da Internet viene inviata una mail con un allegato che, sfruttando un problema di sicurezza di un client di posta della rete interna o la buonafede dell'utente, si installa ed esegue una sequenza di comandi atti a prendere il controllo della stazione e a segnalare l'avvenuta intrusione all'attaccante.

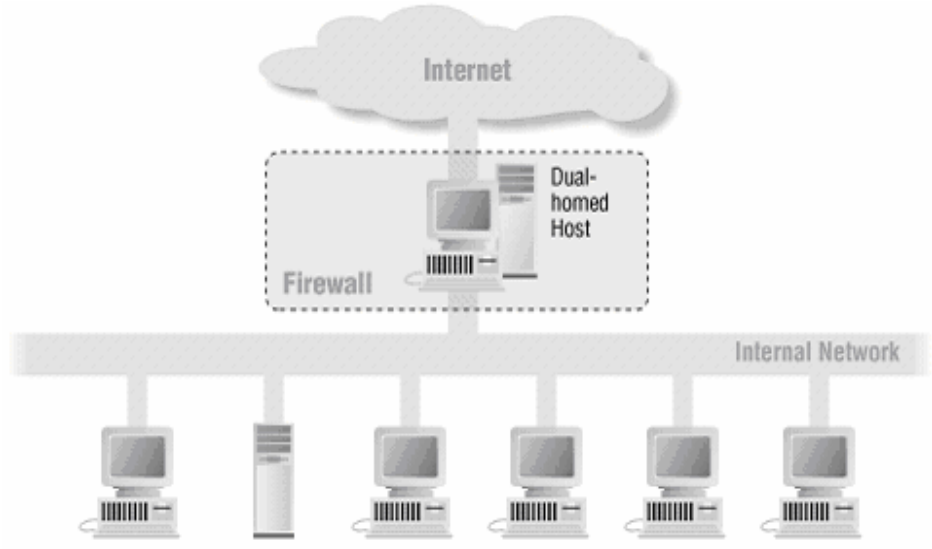
Architettura dei firewall

Nell'analizzare un firewall, di cui nel capitolo precedente abbiamo descritto ragioni e limiti, è utile comprendere quali sono i tipi di architetture su cui si fonda il firewall. In questo capitolo presenteremo tre tipi di architetture, spiegandone i pregi ed i difetti.

2.1 Dual Homed Host Architecture

La "dual-homed host architecture" prevede l'utilizzo di un computer, che ha almeno due interfacce di rete. Questo computer può avere un comportamento analogo ad un router tra le reti a cui sono legate le interfacce, infatti è capace di instradare i pacchetti IP tra le due reti di interesse. Bisogna notare, però, che la dual-homed host architecture prevede la disabilitazione della funzione di instradamento.

Come si può notare dalla figura, il punto forte di questo tipo di architettura è che i pacchetti IP da una rete (ad es. Internet) non possono passare direttamente ad un'altra rete (ad es. l'Internal Network). In tal caso i sistemi a valle del firewall possono comunicare con il dual-homed host computer, come i sistemi a monte, ma, i due, non possono comunicare direttamente tra loro. Il traffico IP tra i due sistemi è completamente bloccato.



I dual-homed host offrono un livello di controllo molto alto. Ad esempio, se si sta negando la possibilità ai pacchetti di andare direttamente dalle reti esterne a quelle interne, un problema di sicurezza può essere evidenziato nel momento in cui nella rete interna viene trovato una fonte esterna (cioè un pacchetto che proviene direttamente da Internet).

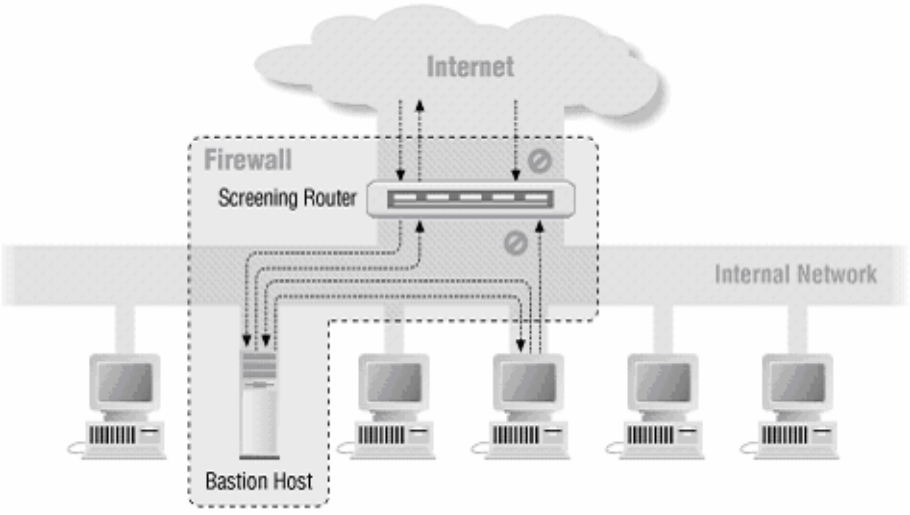
Un dual-homed host offre, quindi, dei servizi "proxandoli", cioè filtrandoli. Il proxying è molto meno problematico, ma non può essere disponibile per tutti i servizi.

L'architettura "screened subnet" che descriveremo nella prossima sezione offre delle scelte addizionali.

2.2 Screened Host Architecture

Come abbiamo visto la dual-homed host architecture offre servizi da un computer che è legato a reti multiple (ma ha instradamento spento). Al contrario, la "screened host architecture" offre servizi da un computer host (detto Bastion Host), che è legato alla sola rete interna, utilizzando semplicemente un router separato. In questa tipo di architettura, la sicurezza primaria è offerta dal "Packet Filtering".

Come si può notare dalla figura il bastion host risiede nella rete interna. Il packet filtering risiede nello screening router ed è settato in modo tale che il bastion host sia l'unico sistema della rete interna alla quale gli host della rete esterna (ad esempio Internet) possono collegarsi (ad es. per consegnare e-mail entranti). Qualsiasi sistema esterno che tenta di accedere ai sistemi interni o a servizi ad essi collegati dovrà effettuare una connessione al bastion host, il quale manterrà un livello di sicurezza alto.



Il packet filtering permette al bastion host di aprire i collegamenti ammissibili (quello che è "ammissibile" sarà determinato dalla particolare politica di sicurezza) alla rete esterna.

Configurando in modo appropriato il packet filtering si può:

- permettere agli altri host interni di aprire i collegamenti ad host su Internet per certi servizi
- respingere tutti i collegamenti da host interni (costringendo questi host ad usare proxy services attraverso il bastion host)

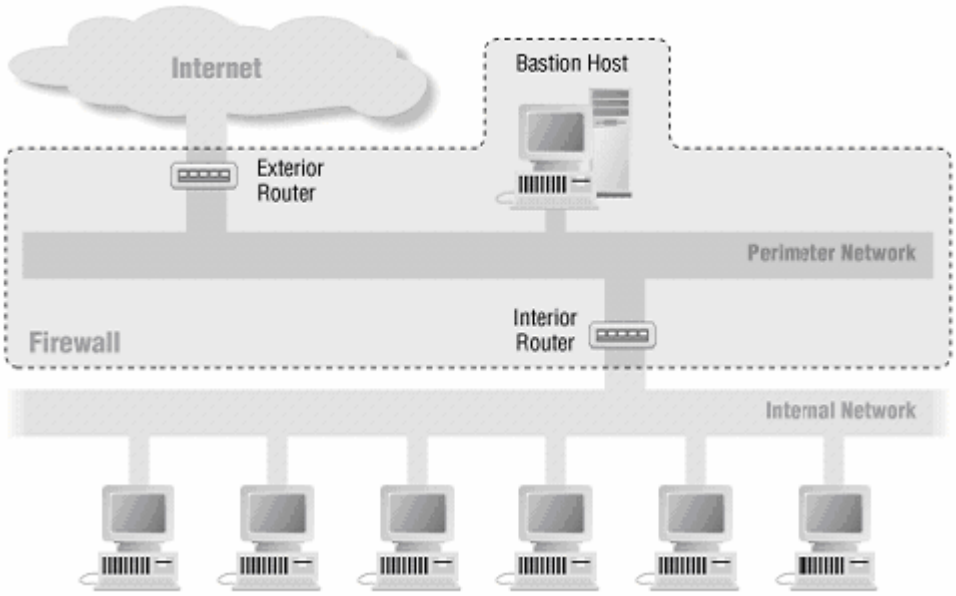
In generale questi due tipi di approcci possono essere mescolati o accoppiati per taluni servizi; alcuni possono essere permessi direttamente via packet filtering, mentre altri possono essere permessi indirettamente via proxy.

Come abbiamo visto la screened host architecture permette ai pacchetti di muoversi dalla rete esterna (Internet) alle reti interne. Ciò potrebbe far pensare che questo tipo di architettura sia più rischiosa della dual-homed host, che è disegnata in modo tale che nessun pacchetto esterno possa raggiungere direttamente la rete interna.

In generale è sempre più facile difendere un router, che offre un set molto limitato di servizi, che difendere un host. Nella maggior parte dei casi, la screened host architecture offre una *migliore sicurezza* ed una *migliore utilizzabilità* della dual-homed host architecture.

2.3 Screened Subnet Architecture

La "screened subnet architecture" aggiunge un strato addizionale alla sicurezza della screened host architecture, aggiungendo una rete di perimetro che isola ulteriormente la rete interna da quella esterna (ad es. Internet).



Per la loro natura i bastion host sono le macchine più vulnerabili in una rete. Nonostante gli innumerevoli sforzi per proteggerli, restano sempre il bersaglio preferito in un attacco.

Se, come in una screened host architecture, la rete interna è aperta agli attacchi, il bastion host resta sempre un obiettivo molto allettante: non ci sono altre difese tra esso e le altre macchine interne. Se qualcuno riesce ad entrare con successo nel bastion host, di in una screened host architecture, si può dire che è entrato completamente nella rete.

Isolando il bastion host in una rete di perimetro, si può ridurre l'impatto di un'intrusione sul bastion host. Da questa idea nasce la screened subnet architecture.

Il modo più semplice di realizzare una screened subnet è di utilizzare due screening router, ognuno connesso alla rete di perimetro. Il primo risiede tra la rete di perimetro

e la rete interna, mentre l'altro tra la rete di perimetro e la rete esterna (ad es. Internet).

In tal caso, anche se l'assalitore riesce ad entrare nel bastion host, per entrare nella rete interna, deve passare ulteriormente attraverso il router interno.

Illustriamo di seguito il funzionamento dei quattro elementi base di una screened subnet.

2.3.1 Rete di perimetro

La rete di perimetro è una rete supplementare che va ad interporsi tra la rete esterna e la rete interna, da proteggere. Se un assalitore irrompe con successo nelle porte esterne del firewall, la rete di perimetro offre un strato supplementare di protezione tra l'assalitore e gli host interni.

Per capire al meglio l'utilità della rete di perimetro consideriamo il seguente esempio.

È possibile per qualsiasi macchina di una data rete vedere il traffico per ogni macchina su quella rete. Questo è vero per la maggior parte di reti Ethernet, che è di gran lunga la rete locale più utilizzata.
Delle persone "curiose" possono riuscire a raccogliere password usate durante le varie sessioni Telnet, FTP, e sessioni di login. Se le password nel frattempo non sono state cambiate, queste persone possono avere l'accesso ai contenuti di file archiviati o di e-mail e così via.
Se si utilizza una rete di perimetro nel momento in cui una persona riesce ad irrompere in un bastion host sulla rete di perimetro, essa avrà solo la possibilità di curiosare sul traffico della rete di perimetro.
Poiché il traffico strettamente interno (ovvero, il traffico tra due host interni che sono presumibilmente sensibili o di proprietà riservati) non passa sul traffico della rete di perimetro, il traffico interno sarà al sicuro da occhi indiscreti, anche se il bastion host è compromesso.

2.3.2 Bastion host

Il bastion host è collegato alla rete di perimetro, e risulta essere il punto principale di contatto per i collegamenti entranti dalla rete esterna; per esempio:

- per servizi SMTP
- per i collegamenti a server FTP
- per servizi DNS.

I servizi dai client interni a server di Internet sono gestiti in questi modi:

- si configura il packet filtering sui router interni ed esterni, per permettere ai client interni di accedere direttamente ai server esterni
- si configura un server proxy sul bastion host (se il firewall usa software proxy) per permettere ai client interni di accedere indirettamente ai server esterni. Si può configurare anche un packet filtering che permetta ai client interni di parlare col server proxy sul bastion host e viceversa (le comunicazioni dirette tra client interni ed il mondo esterno sono proibite).

In ambo i casi, il packet filtering permette al bastion host di connettersi ad host su Internet ed accettare i collegamenti provenienti da esso.

2.3.3 Router interno

Il router interno (choke router) protegge la rete interna dalla rete esterna e dalla rete di perimetro filtrando i pacchetti. Permette i servizi a destinazione esterna dalla rete interna ad Internet, come Telnet in partenza, FTP, WAIS, GOPHER.
I servizi che il router interno permette tra il bastion host (sulla rete di perimetro stessa) e la rete interna non sono necessariamente gli stessi servizi che il router interno permette tra Internet e la rete interna. La ragione per la quale si limitano i servizi tra il bastion host e la rete interna è quella di ridurre il numero di macchine (ed il numero di servizi su quelle macchine) che possono essere attaccate dal bastion host.
E' utile limitare i servizi permessi tra il bastion host e la rete interna a quelli veramente importanti, come l'SMTP (così il bastion host può spedire le e-mail entranti), il DNS (così il bastion host può rispondere alle domande delle macchine interne, o chiedere a loro) e così via.
Si potrebbero fare delle limitazioni ulteriori permettendo tali servizi solo in alcuni casi; per esempio, sarebbe utile limitare l'SMTP solo ai collegamenti tra bastion host ed il server di posta interno.

2.3.4 Router esterno

In teoria, il router esterno (access router) protegge la rete di perimetro e la rete interna da Internet. In pratica, i router esterni tendono a permettere qualsiasi cosa a destinazione esterna dalla rete di perimetro, con un livello di packet filtering molto basso.
Le regole del packet filtering per proteggere le macchine interne dovrebbero essere essenzialmente le stesse per entrambi i router; se c'è un errore nelle regole, in modo da permettere un accesso ad un assalitore, l'errore probabilmente sarà presente su ambo i router.

Frequentemente, il router esterno è fornito da un gruppo esterno (per esempio, l'Internet Provider), e l'accesso ad esso può essere limitato.
Un gruppo esterno, che detiene un router, probabilmente sarà disposto a mettere su parecchie regole per il packet filtering, ma sicuramente non vorrà mantenere un set di regole complicato o variabile frequentemente.
Le uniche regole per il packet filtering che sono veramente importanti sul router esterno sono quelle che proteggono le macchine sulla rete di perimetro (ovvero, il bastion host e il router interno).
Uno dei compiti di sicurezza che il router esterno può compiere in modo utile (un compito che non può essere fatto in modo facilmente in altro punto) è quello di bloccare alcuni pacchetti entranti da Internet che ha contraffatto indirizzi di fonte. Questi pacchetti sembrano venire dall'interno, ma entrano da Internet.

Tipi di firewall

Il nostro cammino alla scoperta dei firewall prosegue e ci porta a conoscere i tipi di firewall. Sostanzialmente esistono due tipi di firewall: i "**packet filtering**" (firewall a filtraggio di pacchetto) che operano allo strato di rete, e i "**proxy system**" (gateway) che a seconda dei casi operano allo strato di applicazione o allo strato di trasporto.

3.1 Packet Filtering

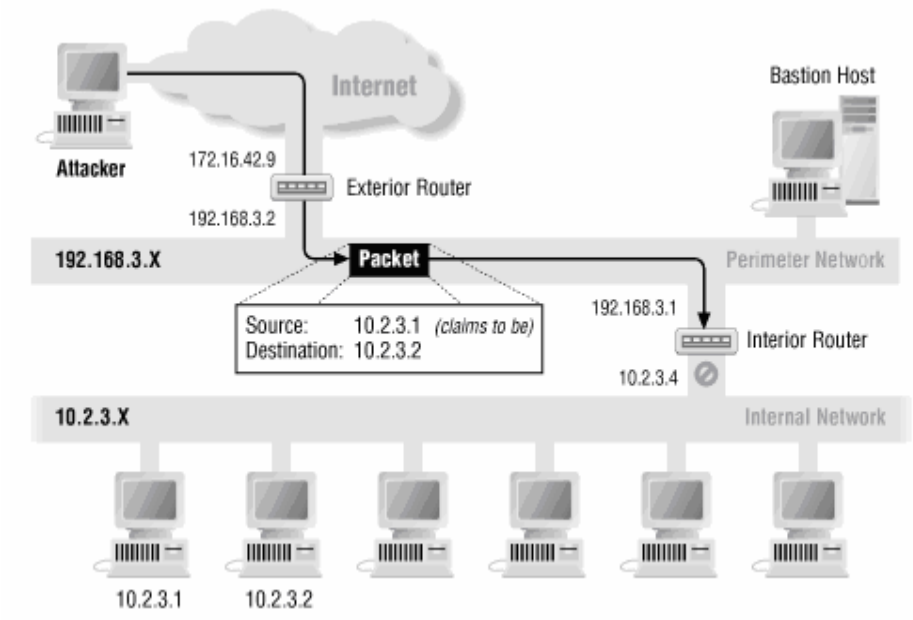
Il packet filtering è un meccanismo di sicurezza allo strato di rete che funziona controllando il flusso di dati in e da una rete ovvero, permette o respinge il trasferimento dei dati basandosi su:

- l'indirizzo di origine
- l'indirizzo di destinazione
- la sessione e i protocolli dei servizi che sono usati per trasferire i dati

Il vantaggio principale del packet filtering è che permette di fornire particolari protezioni per una rete intera.

Certe protezioni sono fornite solamente filtrando i router.

Per esempio, è utile rifiutare tutti i pacchetti che provengono dall'esterno ma che hanno come origine un indirizzo interno - ovvero, pacchetti che chiedono di venire da macchine interne, ma in realtà stanno entrando dall'esterno - perché tali pacchetti rappresentano degli attacchi (*address spoofing attack*). In questo caso, l'assalitore sta fingendo che stia venendo da una macchina interna.



Decisioni di questo tipo possono essere prese solo da un router di una rete di perimetro (*exterior router*), essendo l'unico capace di riconoscere questi tipi di pacchetti, guardando l'indirizzo di origine. Se il pacchetto viene dall'interno (dalla rete interna) o da fuori (dalla rete esterna).

Per prima cosa bisogna decidere quali servizi permettere e quali negare, poi bisogna tradurre le decisioni in regole sui pacchetti (nel router esterno).

I protocolli di solito sono bidirezionali, quindi nella progettazione delle regole bisogna tener conto che i pacchetti viaggiano in ambo i modi.

Bisogna effettuare un'attenta distinzione tra pacchetti in viaggio di ritorno ed a destinazione esterna e servizi in viaggio di ritorno ed a destinazione esterna; bisogna far entrare le "risposte" e non i servizi esterni, che vogliono entrare all'interno della rete.
Un servizio a destinazione esterna (ad es. il servizio di Telnet) comporta ambo i pacchetti, cioè a destinazione esterna (ad es. le battute di un tasto) e in viaggio di ritorno (ad es. le risposte visualizzate sullo schermo).

3.1.1 Vantaggi e svantaggi del packet filtering

Il packet filtering presenta i seguenti vantaggi:

- uno screening router protegge una intera rete;
- il packet filtering non richiede configurazioni di macchine client e non richiede nessuna conoscenza particolare o procedure per utenti;
- il packet filtering è disponibile in molti router.

Tuttavia i tools di filtraggio non sono perfetti. Ad esempio, possiamo avere le seguenti limitazioni**:**

- le regole del packet filtering sono difficili da configurare;
- una volta configurate, le regole sono difficili da testare;
- può avere dei bug (difetti); questi bug sono più frequenti nel packet filtering rispetto al proxying. Di solito, un proxy quando fallisce ferma i dati entranti (o siano che buoni o cattivi), mentre se fallisce un packet filtering i dati entrano comunque;
- alcuni protocolli non sono adatti al packet filtering.

3.1.2 Incapsulamento dei dati

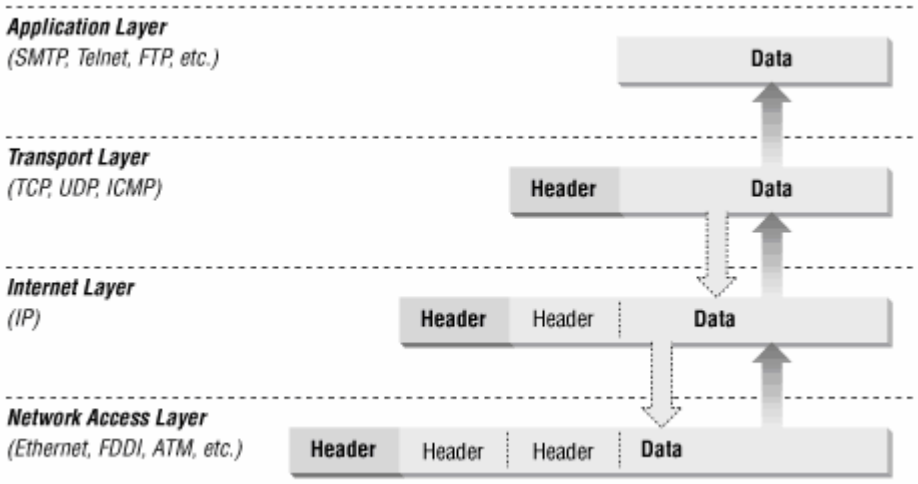
Per comprendere al meglio il packet filtering è importante capire i pacchetti e come sono maneggiati ad ogni strato della pila protocollare:

- strato di applicazione (ad es. FTP, Telnet, Http)
- strato di trasporto (TCP o UDP)
- strato di rete (IP)
- strato di collegamento (ad es. Ethernet, FDDI, ATM)

In ogni strato, un pacchetto è composto da due parti: l'intestazione (header) ed il corpo (body).

L'intestazione contiene delle informazioni sul protocollo attinente a quello strato, mentre il corpo contiene i dati per quello strato, che spesso consiste in un pacchetto proveniente dallo strato superiore. Ogni strato tratta le informazioni che ottiene dallo strato superiore, ed applica la sua propria intestazione a questi dati. Ad ogni strato, il pacchetto contiene tutte le informazioni passate dallo strato più alto; nulla è perso. Questo processo di preservare i dati, quando viene aggiunta una nuova intestazione, è noto come **incapsulamento**.

Allo strato di applicazione il pacchetto è costituito semplicemente dai dati, che devono essere trasferiti (ad es. parte di un file trasferito durante una sessione di FTP).



Nel momento in cui passiamo allo strato di trasporto, il TCP o l'UDP conservano i dati dello strato precedente, ma aggiungono una nuova intestazione. Allo strato successivo, l'IP considera il pacchetto intero come dati e aggiunge ad esso una nuova intestazione. E così via per gli altri strati.

All'altro lato del collegamento, questo processo è invertito.

Una volta che uno screening router ha esaminato un pacchetto può:

- far passare il pacchetto inviandolo alla sua destinazione
- bloccare il pacchetto

3.1.3 ICMP (Codici di Errore)

Se un pacchetto viene bloccato, il router può inviare un codice di errore, detto ICMP (*Internet Control Message Protocol*), il quale da indicazioni su cosa è successo. La ricezione da parte della macchina che sta spedendo i dati di un codice di errore di ICMP avvisa di non rispedire i pacchetti, poiché non verranno fatti passare.

Esistono due set di codici di ICMP:

- il "destination unreachable" - in particolare, l'*host unreachable* ed il *network unreachable*;
- il "destination administratively unreachable" - in particolare, l'*host administratively unreachable* ed il *network administratively unreachable*;

Il primo set di codici di errore di ICMP è riferito a problemi di rete abbastanza seri: l'host di destinazione non funziona o qualche cosa nell'unico percorso verso l'host non funziona (collegamento rotto). Questi codici di errore precedono il firewall e il packet filtering, quindi non dipendono da questi ultimi.

Il problema sorge nel momento in cui il router blocca un pacchetto a causa del packet filtering ed invia un messaggio del primo tipo all'host mittente; questo interpreta il messaggio di host irraggiungibile e taglia il collegamento, non permettendo l'invio di altri pacchetti che invece sarebbero potuti passare.

Questo problema viene risolto utilizzando il secondo set di codici, i quali offrono al router, che utilizza il packet filtering, la possibilità di avvisare gli host mittenti che i pacchetti inviati sono stati bloccati.

Molti sistemi non riconoscono ancora questi codici, ma ciò non provoca alcun problema. Infatti, si suppone che tali sistemi ignorino semplicemente l'errore di ICMP.

Si evidenziano vari problemi nel momento in cui si decide se il nostro sistema deve ritornare codici di errore di ICMP o meno.

Quale set di codici di errore ha senso per la nostra rete?

Ritornare i vecchi destination unreachable è tecnicamente incorretto (si ricorda che l'host può o non può essere raggiungibile, secondo la policy del packet filtering). Inoltre questi codici di errore possono causare azioni incorrette, come detto precedentemente (spegnendo tutti i collegamenti all'host o alla rete).

Ritornando i codici destination administratively unreachable si avverte che si sta utilizzando un sistema di filtraggio dei pacchetti. Questi codici possono provocare anche reazioni eccessive in realizzazioni di IP difettose.

C'è un'altra considerazione da fare. La generazione e il ritorno di codici di errore di ICMP comportano un piccolo sforzo da parte del router di filtraggio (un carico).
Un assalitore potrebbe inondare il router con richieste che il router stesso negherebbe, costringendolo a generare e a inviare pacchetti di errore di ICMP

con conseguente aumento di carico della CPU nel router (mentre è occupata a generare i pacchetti ICMP, non è capace di fare altre cose, come prendere decisioni per il filtraggio).

D'altra parte non ritornando codici di errore di ICMP si provocherà una piccola quantità di traffico di rete di eccesso, perchè il sistema che spedisce tenta e riprova a spedire il pacchetto che è bloccato dal router. Questo traffico non dovrebbe ammontare a molto, perché il numero di pacchetti bloccati dovrebbe essere una frazione minore del numero totale di pacchetti elaborati (inviati).
Se il router ritorna un codice di errore di ICMP per ogni pacchetto che viola la policy di filtraggio, si offre ad un assalitore la possibilità di sondare il sistema di filtraggio della nostra rete. Osservando quali pacchetti chiamano una risposta di errore di ICMP, gli assalitori possono scoprire quali tipi violano e non violano la policy di sicurezza.
Queste informazioni non dovrebbero essere offerte agli assalitori perché gli semplifica di molto il lavoro. L'assalitore sa che i pacchetti che non trovano l'errore di ICMP riescono ad andare verso la rete interna senza problemi, può concentrarsi, quindi, su di essi, cioè dove si è più vulnerabili, e tentare un attacco.

Il problema maggiore è che si perde meno tempo ad inviare un pacchetto, che a controllarlo (vedere se può passare o no, inviare il codice per il mancato passaggio).

La cosa più sicura potrebbe essere quella di non ritornare alcun codice di errore di ICMP all'esterno. Se il router offre abbastanza flessibilità, ha più senso configurarlo in modo da ritornare i codici di errore di ICMP solo ai sistemi interni (in questo modo si è avvisati immediatamente se qualcosa è andato a vuoto, piuttosto che attendere inutilmente), ma non a sistemi esterni (dove le informazioni potrebbero andare nelle mani degli assalitori). Se il router non offre questa flessibilità, si può avere lo stesso risultato impartendo al packet filtering delle regole appropriate, in modo da permettere il passaggio dei pacchetti di ICMP in viaggio di ritorno e di stare attenti a respingere i pacchetti di ICMP a destinazione esterna.

Ad esempio si può fare in modo che per ogni pacchetto il router esamini le regole in ordine (in modo sequenziale) finché non trova una regola compatibile con il pacchetto, o datagram, per eseguire l'azione specificata da quella regola.

Bisogna sempre specificare, nelle regole, host e reti con il proprio indirizzo IP e mai con hostname o nome di rete. Infatti, se si specificano le regole con hostname, il filtraggio potrebbe essere sovvertito se qualcuno corrompe accidentalmente o intenzionalmente la traduzione di nome-indirizzo (ad es. dando dati falsi al nostro sistema di servizio DNS).

Per capire al meglio quanto detto fino ad ora consideriamo un semplice esempio.

Diciamo che si vuole permettere qualsiasi traffico IP tra un host esterno fidato (host 172.16.51.50) ed un host della rete interna (la classe C pone le reti in 192.168.10.0).

Questa cosa si traduce nel considerare le regole riportate nella tabella seguente:

Regola	Direzione	Fonte Indirizzo	Destinazione Indirizzo	Set ACK	Azione
A	In viaggio di ritorno	*Host esterno di fiducia* **172.16.51.50**	*Interno* **192.168.10.0**	Alcuno	Licenza
B	A destinazione esterna	*Interno* **192.168.10.0**	*Host esterno di fiducia* **172.16.51.50**	Alcuno	Licenza
C	Altro	Altro	Altro	Alcuno	Nega

Osservando il set di regole assegnato risulta evidente che il "traffico" tra l'host 172.16.51.50 ed la rete 192.168.10.0 è consentito, mentre è negato quello fra altri due host qualsiasi.

3.1.4 Filtering by source port

E' possibile effettuare un filtraggio dei pacchetti controllando il porto della fonte, vedendo ad esempio se è il 23.

C'è un problema fondamentale con questo tipo di filtraggio. In questo caso si ha solo fiducia del porto di fonte e non della macchina di fonte, cioè il filtraggio è relativo al numero di porto e non alla macchina. Questa scelta non da molte garanzie, infatti un possibile ingresso proveniente da quel numero di porto può essere una macchina sicura, ma anche un assalitore.

Per risolvere questo problema bisognerebbe limitare l'accesso sul numero di porto specificato solo ai server fidati, cioè bisognerebbe filtrare sia in base al numero di porto sia in base all'indirizzo di fonte e destinazione.

3.1.5 Fattori limitanti e importanza dell'ordine delle regole

In molti firewall di Internet, il fattore che limita le prestazioni è la velocità del router nel collegamento ad Internet, non la velocità del sistema di filtraggio. La domanda corretta è: "È veloce abbastanza per le mie necessità?"

I collegamenti comunemente usati in Internet sono 56-Kb/s o 1.544-Mb/s (T1). Più il pacchetto è piccolo, più pacchetti verranno manipolati per ogni secondo, e più decisioni di filtraggio saranno prese per ogni secondo. Il più piccolo possibile pacchetto IP - un pacchetto "nudo" che contiene solo l'intestazione IP e nessun dato - è lungo 20 byte (160 pezzi). Così, una linea con velocità di 56 Kb/s può portare al massimo 350 pacchetti per secondo, ed una linea con velocità di 1.544 Mb/s (ad es. una linea di T-1) può portare al massimo 9,650 pacchetti per secondo, come mostrato nella tabella seguente:

Tipo di collegamento	Bit/Secondi (approssimato)	Packet/Secondi (Pacchetti di 20-byte)	Racket/Secondi (Pacchetti di 40-byte)
modem di v.32bis	14,400	90	45
56-Kb/s	56,000	350	175
T-1	1,544,000	9,650	4,825
Ethernet (pratico)	3,000,000	18,750	9,375
Ethernet (teorico)	10,000,000	62,500	31,250
T-3	45,000,000	281,250	140,625
FDDI	100,000,000	625,000	312,500

Un tipico pacchetto che attraversa un firewall è un pacchetto del tipo TCP/IP, perché la maggior parte dei servizi di Internet è basata sul TCP. La minima lunghezza possibile di un pacchetto TCP/IP, che contiene solo l'intestazione IP e l'intestazione TCP e nessun dato è 40 byte.

È probabile che la velocità di un firewall sia un problema se il firewall è collocato all'interno di una rete locale. In tal caso, il firewall avrà bisogno di girare in una LAN che ha una velocità teorica di 10 Mb/s e può essere molto alta.

Con più collegamenti, la massima velocità richiesta è quella della connessione più lenta.

Si dovrebbe permettere la semplice specifica delle regole.

E' preferibile specificare le regole per il packet filtering il più semplicemente possibile e di non aggiungere ulteriori complicazioni al nostro sistema.

E' utile specificare le regole basandosi sulle informazioni contenute nell'intestazione e le informazioni meta-packet. Le informazioni di intestazione includono:

- indirizzo IP sorgente ed indirizzo IP di destinazione

- scelte di IP
- protocollo, come TCP, UDP, o ICMP
- porto sorgente e di destinazione
- bit di ACK e informazioni per i pacchetti di TCP
- ed informazioni simili per alcuni altri protocolli che sta filtrando.

Le informazioni di meta-packet includono alcune informazioni sul pacchetto che il router conosce, ma che non sono presenti nelle intestazioni, ad esempio, da quale interfaccia di router il pacchetto è entrato o sta uscendo. La soluzione ideale è di basarsi su una combinazione delle due.

Per varie ragioni, molti tipi di filtraggio non ci lasciano guardare al porto di fonte di TCP/UDP nel prendere decisioni di filtraggio sul pacchetto, ma permettono di guardare solamente al porto di destinazione di TCP/UDP. Questo rende impossibile specificare certi generi di filtraggio.

Si dovrebbero applicare le regole nell'ordine specificato.

E' preferibile applicare le regole specificate nel packet filtering nell'ordine da noi prefissato. Sfortunatamente, alcuni prodotti, invece di applicare le regole nell'ordine specificato, tentano di ordinare di nuovo ed unire le regole (a modo loro) per rendere il tutto più efficiente. Questo provoca molti problemi:

- riordinando di nuovo le regole è più difficile dedurre la regola che sta esaminando, e l'azione che il router farà;
- se ci sono alcuni bug nella fusione o nel riordino del nuovo di set di regole, diviene impossibile dedurre quello che il sistema farà con un determinato set di filtri;
- riordinando le regole si può rompere un set di regole che funziona meglio rispetto al nuovo set di regole riordinato;

Consideriamo un esempio. Immaginiamo di far parte di una società per azioni e di lavorare su di un progetto speciale con un'università locale. La nostra rete è di classe B ed è la 172.16 (ad es. i nostri indirizzi IP possono andare da 172.16.0.0 a 172.16.255.255). L'università possiede la rete di classe A 10 (ad es. i loro indirizzi IP possono andare da 10.0.0.0 fino a 10.255.255.255).

Supponiamo di collegare direttamente la nostra rete all'università, utilizzando un router con il packet filtering, e che il nostro progetto con l'università utilizzi la subnet 172.16.6 (ad es. IP indirizzanti da 172.16.6.0 fino a 172.16.6.255). Lei vuole che ogni subnet dell'università entri nella sua subnet della società e viceversa. Nell'università, la subnet 10.1.99, non può accedere all'intera rete della società ma solamente alla subnet speciale 172.16.6.0/24.

Ecco le regole per soddisfare le nostre esigenze:

Regola	Fonte Indirizzo	Destinazione Indirizzo	Azione
A	10.0.0.0/8	172.16.6.0/24	Licenza
B	10.1.99.0/24	172.16.0.0/16	Nega
C	Alcuno	Alcuno	Nega

- la regola A da licenze all'università per giungere alla nostra subnet del progetto
- la regola B nega alla subnet ostile di accedere alla nostra rete
- la regola C respinge l'accesso da Internet alla nostra rete

Andiamo a vedere ora quello che succede a seconda di come vengono applicate le regole.

Le regole sono applicate nell'ordine ABC:
Vediamo cosa succede, per una serie di pacchetti, se le regole sono applicate nell'ordine ABC – lo stesso ordine specificato da noi.

Pacchetto	Fonte Indirizzo	Destinazione Indirizzo	Azione Desiderata	Azione Attuale (da Regola)
1	10.1.99.1	172.16.1.1	Nega	Nega (B)
2	10.1.99.1	172.16.6.1	Licenza	Licenza (A)
3	10.1.1.1	172.16.6.1	Licenza	Licenza (A)
4	10.1.1.1	172.16.1.1	Nega	Nega (C)
5	192.168.3.4	172.16.1.1	Nega	Nega (C)
6	192.168.3.4	172.16.6.1	Nega	Nega (C)

- il pacchetto 1 è inviato da una macchina dell'università appartenente alla subnet ostile ad una macchina sulla nostra rete (non sulla subnet del progetto); si vuole che ciò sia negato e questo è garantito dalla regola B
- il pacchetto 2 è inviato da una macchina dell'università appartenente alla subnet ostile ad una macchina sulla nostra subnet del progetto; si vuole che ciò sia permesso e questo è garantito dalla regola A
- il pacchetto 3 è inviato da una macchina casuale dell'università ad una macchina sulla nostra subnet del progetto; si vuole che ciò sia permesso e questo è garantito dalla regola A
- il pacchetto 4 è inviato da una macchina casuale dell'università alla nostra rete; si vuole che ciò sia negato e questo è garantito dalla regola C
- il pacchetto 5 è inviato da una macchina casuale da Internet alla nostra rete; si vuole che ciò sia negato e questo è garantito dalla regola C
- il pacchetto 6 è inviato da una macchina casuale da Internet ad una macchina sulla nostra subnet del progetto; si vuole che ciò sia negato e questo è garantito dalla regola C.

Così, se le regole sono applicate nell'ordine ABC, si ottiene ciò che volevamo.

Le regole sono applicate nell'ordine BAC:
Che cosa succede se il router riordina le regole a suo piacimento? Per dare una risposta alla nostra domanda supponiamo che il router riordini le regole nel modo BAC.

Regola	Fonte Indirizzo	Destinazione Indirizzo	Azione
B	10.1.99.0/24	172.16.0.0/16	Nega
A	10.0.0.0/8	172.16.6.0/24	Licenza
C	Alcuno	Alcuno	Nega

Consideriamo gli stessi sei pacchetti dell'esempio precedente ed andiamo a vedere le conseguenze dovute alle regole applicate nell'ordine BAC.

Pacchetto	Fonte Indirizzo	Destinazione Indirizzo	Azione Desiderata	Azione Attuale (da Regola)
1	10.1.99.1	172.16.1.1	Nega	Nega (B)
2	10.1.99.1	172.16.6.1	Licenza	Nega (B)
3	10.1.1.1	172.16.6.1	Licenza	Licenza (A)
4	10.1.1.1	172.16.1.1	Nega	Nega (C)
5	192.168.3.4	172.16.1.1	Nega	Nega (C)
6	192.168.3.4	172.16.6.1	Nega	Nega (C)

Se le regole sono applicate nell'ordine BAC, il pacchetto 2 che dovrebbe essere permesso è negato impropriamente dalla regola B.

Per evitare ciò è utile utilizzare un sistema di filtrazione che non modifichi l'ordine delle azioni da intraprendere.

Si dovrebbe essere capaci di accedere ai pacchetti accettati e bloccati.

Questi pacchetti sono interessanti, perché possono essere sondati da un malintenzionato che ha intenzione di sferrare un attacco. E' importante avere informazioni dal router, perché in tal caso si potrà avere un quadro completo di quello che sta facendo l'assalitore.

L'accesso dovrebbe essere flessibile; il packet filtering dovrebbe dare, ad esempio, la possibilità di accedere ad un archivio locale.

3.1.6 Dove si dovrebbe implementare il packet filtering?

Se si osservano le varie architetture di firewall delineate nel capitolo 2, si nota che c'è una varietà di luoghi dove è possibile implementare il packet filtering. Dove si dovrebbe fare? La risposta è semplice: dovunque si può.

Molte delle architetture (ad es. la screened host) coinvolgono un solo router. In quei casi, l'unico posto dove si potrebbe fare il packet filtering è proprio il router, così non ci sono molte decisioni da prendere.
Comunque, nelle altre architetture, come l'architettura che prevede due router o quelle che prevedono router multipli, è probabile che si faccia il packet filtering su qualcuno o tutti questi router.

La nostra raccomandazione è farlo dovunque si può. Per ogni router che è parte del proprio firewall, bisogna capire quello che si vuole filtrare, e preparare i filtri per permettere solo ad alcuni pacchetti di passare.
Questo può condurre alla duplicazione di alcuni filtri sui router multipli; in altre parole, si riesce a filtrare la stessa cosa in più posti. Ciò va bene, anche se è ridondante; bisogna tener conto, però, che una cosa del genere può salvare il gestore della rete nel giorno in cui ha un problema con uno dei suoi router.

3.1.7 Esempio

Per capire quanto detto finora consideriamo un semplice esempio.

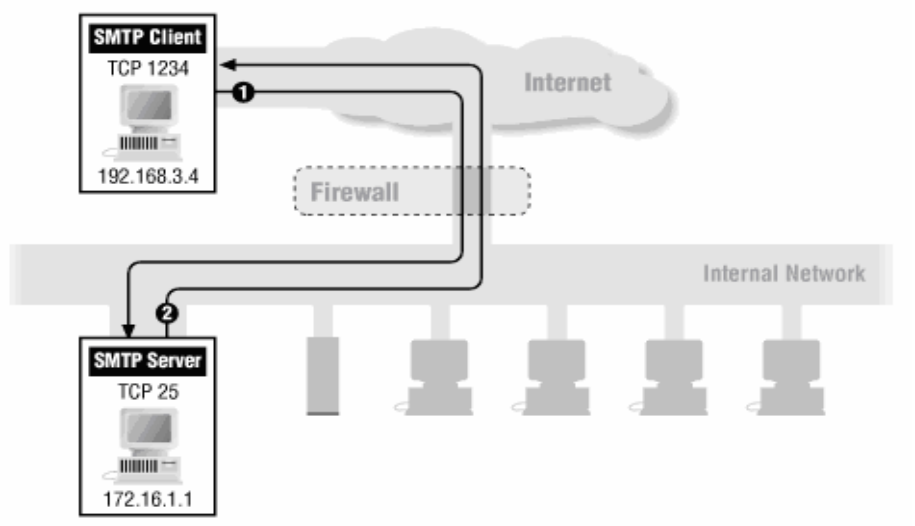
Supponiamo di aver impartito le seguenti regole:

Regola	Direzione	Fonte Indirizzo	Destinazione Indirizzo	Protocollo	Porto Dest.	Azione
A	In	Esterno	Interno	TCP	25	Licenza
B	Out	Interno	Esterno	TCP	>1023	Licenza
C	Out	Interno	Esterno	TCP	25	Licenza
D	In	Esterno	Interno	TCP	>1023	Licenza
E	O	Alcuno	Alcuno	Alcuno	Alcuno	Nega

- le regole A e B permettono i collegamenti SMTP in viaggio di ritorno (cioè le e-mail entranti).
- le regole C e D permettono i collegamenti SMTP a destinazione esterna (cioè le e-mail uscenti)
- la regola E è la regola di default che viene applicata se le altre regole non possono essere applicate

Per vedere quello che succede consideriamo dei pacchetti di prova. Diciamo che il nostro host ha l'indirizzo IP 172.16.1.1, e che qualcuno sta tentando di spedirgli posta dall'host remoto con indirizzo IP 192.168.3.4. Il client SMTP del mittente usa il porto 1234 per parlare col nostro server SMTP, il quale utilizza il porto 25 per colloquiare.

Pacchetto	Direzione	Fonte Indirizzo	Destinazione Indirizzo	Protocollo	Porto Dest.	Azione (da Regola)
1	In	192.168.3.4	172.16.1.1	TCP	25	Licenza (A)
2	Out	172.16.1.1	192.168.3.4	TCP	1234	Licenza (B)

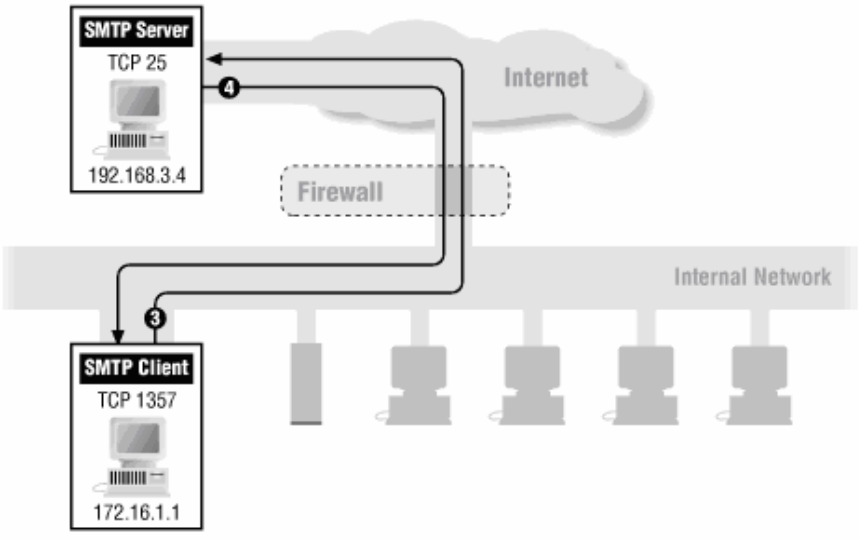


In questo caso, il packet filtering fa passare le e-mail entranti:

- la regola A permette ai pacchetti in partenza dal client SMTP di andare al suo SMTP server (rappresentato da pacchetto numero 1)
- la regola B permette alle risposte dal suo server SMTP di andare al client SMTP (rappresentato da pacchetto numero 2)

Cosa succede per un'e-mail in partenza da noi verso l'esterno? Diciamo che il nostro client SMTP usa il porto 1357 per parlare col server SMTP esterno:

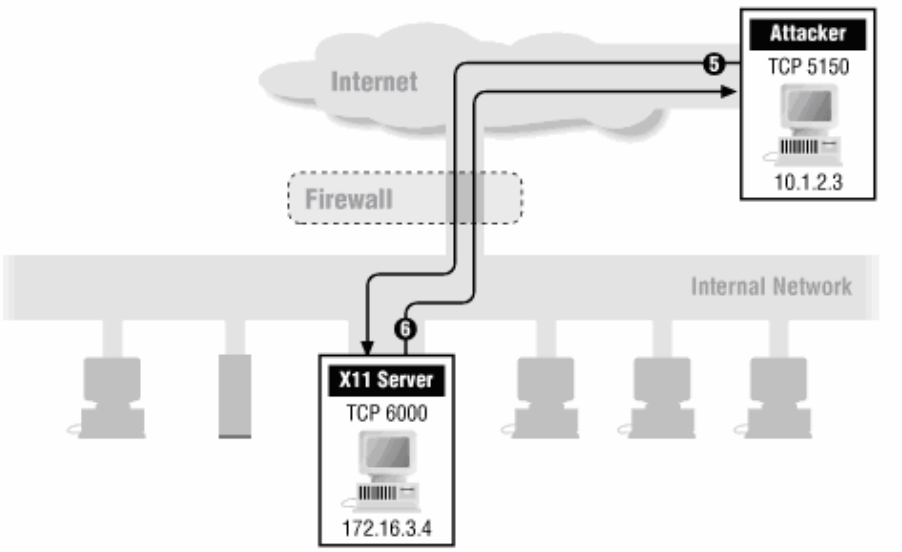
Pacchetto	Direzione	Fonte Indirizzo	Destinazione Indirizzo	Protocollo	Porto Dest.	Azione (da Regola)
3	Out	172.16.1.1	192.168.3.4	TCP	25	Licenza (C)
4	In	192.168.3.4	172.16.1.1	TCP	1357	Licenza (D)



- la regola C permette ai pacchetti in partenza dal nostro client SMTP di andare al server SMTP esterno (pacchetto numero 3)
- la regola D permette alle risposte partenti dal server SMTP esterno di andare al nostro client SMTP (pacchetto numero 4)

Mescoliamo, ora, le carte. Cosa accade se qualcuno dall'esterno (ad es. qualcuno sull'host 10.1.2.3) tenta di aprire un collegamento dal porto 5150 sul nostro server sul porto 6000 (ad es. presente all'host 172.16.3.4) per eseguire un attacco?

Pacchetto	Direzione	Fonte Indirizzo	Destinazione Indirizzo	Protocollo	Porto Dest.	Azione (da Regola)
5	In	10.1.2.3	172.16.3.4	TCP	6000	Licenza (D)
6	Out	172.16.3.4	10.1.2.3	TCP	5150	Licenza (B)



Le regole mostrate sopra permettono a questo collegamento di avere successo! Infatti, permettono a qualsiasi collegamento di entrare finché ambo gli ends del collegamento stanno usando dei porti superiori di 1023. Perché?

- le regole A e B permettono i collegamenti SMTP in viaggio di ritorno
- le regole C e D permettono i collegamenti SMTP a destinazione esterna
- ma B e D permettono tutti i collegamenti dove entrambi gli ends stanno usando dei porti superiori di 1023, e questo non è certamente quello che noi volevamo

Le stesse cinque regole di base, viste pocanzi, possono essere riformulate aggiungendo semplicemente il porto di fonte come criterio di controllo. Queste regole sono visibili nella tabella alla pagina seguente.

Regola	Direzione	Fonte Indirizzo	Destinazione Indirizzo	Protocollo	Porto Fonte	Porto Dest.	Azione
A	In	Esterno	Interno	TCP	>1023	25	Licenza
B	Out	Interno	Esterno	TCP	25	>1023	Licenza
C	Out	Interno	Esterno	TCP	>1023	25	Licenza
D	In	Esterno	Interno	TCP	25	>1023	Licenza
E	O	Alcuno	Alcuno	Alcuno	Alcuno	Alcuno	Nega

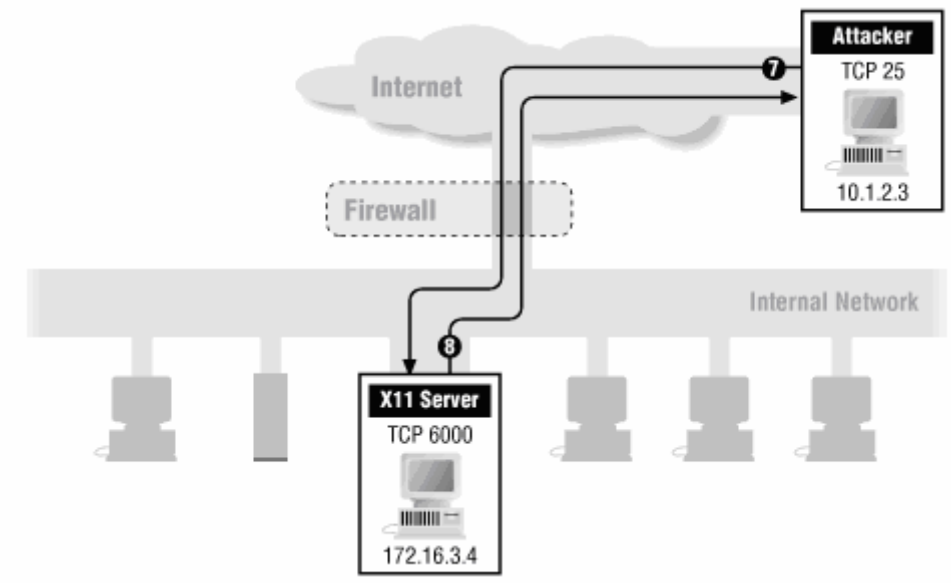
Ed ecco gli stessi pacchetti di prima, filtrati utilizzando le nuove regole:

Pacchetto	Direzione	Fonte Indirizzo	Destinazione Indirizzo	Protocollo	Porto Fonte	Porto Dest.	Azione (da Regola)
1	In	192.168.3.4	172.16.1.1	TCP	1234	25	Licenza (A)
2	Out	172.16.1.1	192.168.3.4	TCP	25	1234	Licenza (B)
3	Out	172.16.1.1	192.168.3.4	TCP	1357	25	Licenza (C)
4	In	192.168.3.4	172.16.1.1	TCP	25	1357	Licenza (D)
5	In	10.1.2.3	172.16.3.4	TCP	5150	6000	Nega (E)
6	Out	172.16.3.4	10.1.2.3	TCP	6000	5150	Nega (E)

Come si può vedere, quando il porto di fonte è considerato come un criterio, i pacchetti che creavano problema (numero 5 e 6, che rappresentano un attacco al server) finiscono per essere negati dalla regola predefinita.

Cosa succede se si ha a che fare con un assalitore più intelligente? Cosa succede se l'assalitore usa il porto 25 come porto di client sul suo end, e poi tenta di aprire un collegamento al server? Ecco i pacchetti che si vedrebbero:

Pacchetto	Direzione	Fonte Indirizzo	Destinazione Indirizzo	Protocollo	Porto Fonte	Porto Dest.	Azione (da Regola)
7	In	10.1.2.3	172.16.3.4	TCP	25	6000	Licenza (D)
8	Out	172.16.3.4	10.1.2.3	TCP	6000	25	Licenza (C)



Come si può vedere, i pacchetti sono permessi, e l'attacco può andare a buon fine con molta probabilità.

Quindi cosa si può fare? La soluzione è di considerare anche i bit di ACK come criterio di controllo. Consideriamo, quindi, di nuovo le stesse cinque regole di prima con i bit di ACK, aggiunti come criterio:

Regola	Direzione	Fonte Indirizzo	Destinazione Indirizzo	Protocollo	Porto Fonte	Porto Dest.	Set ACK	Azione
A	In	Esterno	Interno	TCP	>1023	25	Alcuno	Licenza
B	Out	Interno	Esterno	TCP	25	>1023	Sì	Licenza
C	Out	Interno	Esterno	TCP	>1023	25	Alcuno	Licenza
D	In	Esterno	Interno	TCP	25	>1023	Sì	Licenza
E	O	Alcuno	Alcuno	Alcuno	Alcuno	Alcuno	Alcuno	Nega

Ora, il pacchetto 7 fallirà:

Pacchetto	Direzione	Fonte Indirizzo	Dest. Indirizzo	Protocollo	Porto Fonte	Porto Dest.	Set ACK	Azione (da Regola)
7	In	10.1.2.3	172.16.3.4	TCP	25	6000	No	Nega (E)

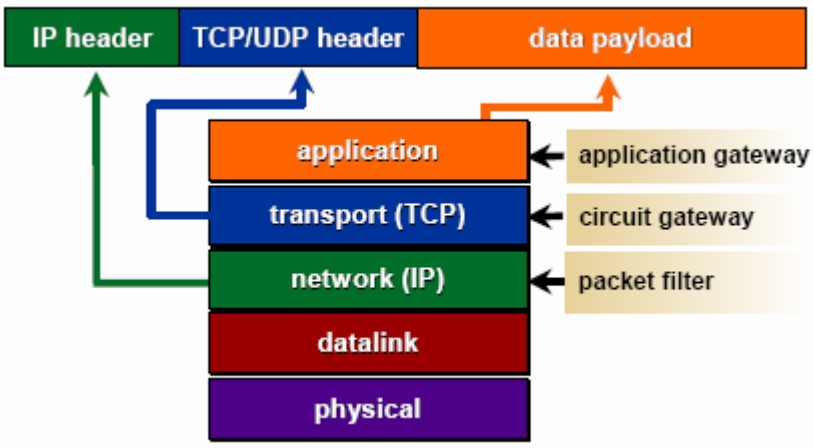
L'unica differenza è nelle regole B e D. Di queste due, la regola D è la più importante, perché controlla i collegamenti entranti. La regola B è applicata ai collegamenti in partenza .
La regola D ora ci dice di accettare i pacchetti entranti da cose che sono apparentemente server di SMTP (perché i pacchetti stanno venendo da porto 25), solo se i pacchetti hanno i bit di ACK; ovvero, solo se i pacchetti sono parte di un collegamento cominciato dall'interno.
Se qualcuno tenta di aprire un collegamento TCP da fuori, il primo pacchetto che lui spedisce non avrà i bit di ACK e quindi sarà bloccato.

3.2 Proxy System

A seconda del livello della pila protocollare in cui si decide di lavorare possiamo evidenziare due tipi di proxy system: gli Application-level Gateway che operano allo strato di applicazione e i Circuit-level Gateway che operano allo strato di trasporto.

Gli application-level gateway sono dispositivi programmati per analizzare il traffico di rete al livello applicativo. Essi conferiscono un ulteriore strumento per la sicurezza di rete. Ogni applicazione protetta dall'application-level-gateway viene identificata da questo attraverso un codice univoco, costringendo i vari collegamenti ad attraversare software controllato; per questo si garantisce un alto livello di sicurezza ma per ogni applicazione affidatagli è necessario un codice differente. Se questo dispositivo viene implementato per uno specifico servizio, un host (in un'applicazione Client-Server) che si vuol connettere a tale servizio si trova a

colloquiare con l'application-level gateway che effettua un'operazione di server nei confronti delle richieste del client e contemporaneamente opera come un client inoltrando le richieste (se la connessione è concessa) al server vero e proprio. Ciò comporta il fatto che il client si trova a colloquiare con un server diverso da quello "autentico" e quindi nasce l'esigenza, all'interno dell'application-level gateway, di avere degli applicativi il più possibile simili al server "autentico". Tali applicativi prendono il nome di "server proxy". Il server proxy valuta le richieste del client e decide quale far passare e quali trascurare. Se una richiesta è approvata, il server proxy parla col vero server.



I circuit-level gateway sono dispositivi programmati per analizzare il traffico di rete al livello di trasporto. E' infatti un proxy non "application-aware".
Essi creano un circuito tra client e server a livello trasporto, rompendo il modello client/server per la durata della connessione. Essi non prevedono la comprensione dei dati in transito.

I server proxy spesso sono usati al posto dei router per il controllo del traffico, per non permettere al traffico di passare direttamente tra le reti. Poiché i proxy devono capire quale protocollo sta per essere usato, possono anche implementare specifici protocolli di sicurezza (ad es. un proxy FTP potrebbe essere configurato per permettere l'FTP in entrata e bloccare l'FTP in uscita).

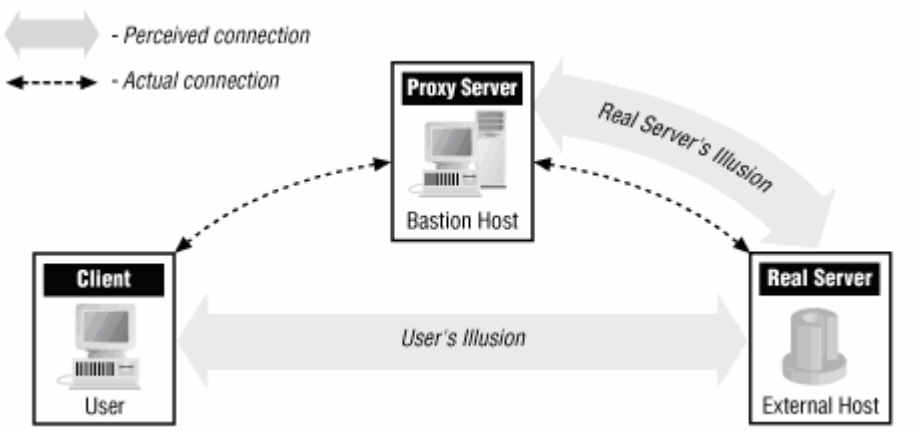
I server proxy sono applicazioni specifiche. Per supportare un nuovo protocollo attraverso un proxy, questo deve essere sviluppato per quel protocollo. Il proxying non richiede un hardware speciale, ma al contrario richiede del software speciale per la maggior parte dei servizi.

In una rete costituita da più host non è opportuno che ogni singolo utente possa accedere ad Internet direttamente dal suo host. Infatti se non si riesce a garantire alcuna protezione dal mondo esterno, si potrebbe utilizzare un unico host per l'accesso ad Internet per tutti gli utenti della rete. Tuttavia, questa non è la soluzione migliore in quanto sorgono alcuni problemi:

- gli utenti che vogliono accedere ai servizi di rete non possono farlo direttamente, ma devono accedere ad un dual-homed host, tramite il quale viene fornito il servizio richiesto. Inoltre la soluzione adottata risulta essere sconveniente perché costringe gli utenti ad effettuare trasferimenti multipli.
- il problema è peggiore quando siamo di fronte a sistemi operativi multipli; se ad esempio il sistema natio di un utente è un Macintosh, ed il dual-homed host

è un sistema di Unix, il sistema di Unix potrebbe essere completamente estraneo a quello dell'utente. L'utente sarà limitato ad usare gli strumenti disponibili sul dual-homed host, e questi strumenti possono essere completamente diversi da quelli utilizzati usualmente dall'utente.

L'utente ha solo l'illusione di un collegamento diretto col server di Internet a cui vuole realmente accedere, con un minimo di interazione diretta col dual-homed host. La figura illustra la differenza tra la realtà e l'illusione con i proxy system.



3.2.1 Vantaggi e svantaggi del proxying

Si evidenziano alcuni vantaggi nell'utilizzo dei servizi proxy:

- **permettono agli utenti di accedere direttamente ai servizi Internet:** con l'approccio dual-homed host senza proxy, l'utente deve accedere nell'host prima di usare alcuni servizi di Internet. Questo spesso reca disturbo. Con il proxy invece, gli utenti hanno l'impressione di interagire direttamente con servizi di Internet. Chiaramente, ciò non è vero in quanto l'accesso non è diretto, ma di solito i vari utenti non se ne accorgono in quanto il tutto è trasparente. Il proxy services permette agli utenti di accedere ai servizi Internet dai loro propri sistemi, senza che i pacchetti passino direttamente tra il sistema dell'utente e Internet. Il percorso è indiretto, e passa o attraverso un dual-homed host, o attraverso un bastion host e una combinazione di screening router.
- **regole più granulari e semplici rispetto al packet filtering.**

Il proxy server è soprattutto un dispositivo di sicurezza. Un suo utilizzo per aumentare l'accesso ad internet con limitati indirizzi IP causerà molti svantaggi. Un proxy server consentirà un maggior accesso dall'interno della rete protetta verso l'esterno, ma manterrà l'interno completamente inaccessibile dall'esterno. Ciò implica l'impossibilità di avere connessioni server, talk o archie oppure mail dirette verso i computer presenti all'interno.

Gli svantaggi che si evidenziano enormemente possono essere così riassunti:

- **indisponibilità per servizi nuovi:** anche se il proxy software è disponibile per i vecchi servizi, come ad esempio FTP e Telnet, per servizi più nuovi è più difficile trovarlo. C'è di solito un ritardo tra l'introduzione di un servizio e la disponibilità di proxying server.
- **possono richiedere server diversi per ogni servizio:** ci può essere l'esigenza di avere un server proxy per ogni protocollo. Il proxy server deve capire il protocollo per determinare cosa permettere e respingere, e per mascherarsi come un client al vero server e come il vero server al proxy client. Installare e configurare tutti questi server può essere molto faticoso.
- **richiedono modifiche a client, procedure, o ad entrambi:** i server proxy richiedono modifiche a client e/o procedure. Queste modifiche presentano degli inconvenienti; infatti, non si possono usare i tools con le istruzioni normali. A causa di queste modifiche, alcune applicazioni proxy non funzionano.
- **un esempio:** un report su cui state lavorando sul vostro computer è stato lasciato all'interno della rete protetta con firewall. Vi trovate a casa, e decidete di lavorarci ancora un po'. Ma questo non è possibile. Non potete raggiungere il vostro computer perché si trova al di là del firewall. Per prima cosa cercherete di connettervi al firewall, ma dal momento che tutti hanno un accesso proxy server, nessuno avrà impostato un account per voi.
- **un esempio:** supponete che vostra figlia frequenti il college e volete inviarle una e-mail per parlare con lei di alcuni affari personali. Di sicuro preferireste poter ricevere la posta direttamente sulla vostra macchina, ponendo piena fiducia nell'amministratore del sistema; tuttavia bisogna ricordare che si tratta sempre di posta privata e nulla vieta all'amministratore di dargli una sbirciatina.
- **l'incapacità di utilizzare i pacchetti UDP** rappresenta un grande svantaggio dei proxy server.
- **sono lenti:** a causa del sovraccarico maggiore, qualsiasi altro mezzo per ottenere questo accesso sarà più veloce.

3.2.2 Come lavora un proxy?

I dettagli di come lavora il proxying differiscono da servizio a servizio. Dei servizi offrono facilmente o automaticamente il proxying; per tali servizi, si setta il proxying facendo cambi di configurazione a server normali. Per la maggior parte di servizi, comunque il proxying richiede un software adatto sul server. Sul cliente, c'è bisogno di una delle seguenti possibilità:

- **Custom client software:** con questo approccio, il software deve sapere come contattare il server proxy invece del vero server quando

un utente fa una richiesta (per esempio, per FTP o Telnet), e come dire al server proxy che il vero server è connesso. E' adatto solamente per certe piattaforme.

- **Custom user procedures:** con questo approccio, l'utente usa un software client standard per comunicare al proxy server di attivare una connessione con il vero server, invece di comunicare direttamente con il vero server.

3.2.3 Application-level proxy VS Circuit-level proxy

Un *application-level proxy* è un proxy server che lavora a livello applicativo, cioè capisce ed interpreta i comandi nel protocollo applicativo. Al contrario un *circuit-level proxy* crea un circuito tra il client ed il server senza interpretare il protocollo applicativo. Un esempio di un application-level proxy è un'applicazione come *Sendmail*, che perfeziona un protocollo store-and-forward. La versione più estrema di un circuit-level proxy è uno degli ingressi dei moderni proxy ibridi, che al di fuori sembrano un proxy ma all'interno lavorano come un router di filtraggio.

Gli application-level proxy rompono completamente il modello client/server:

- i server sono più protetti rispetto al packet filtering
- possono autenticare i client
- mancanza di trasparenza per i client
- possono esporre il S.O. del firewall ad attacchi

I circuit-level proxy rompono, invece, il modello client/server per la sola durata della connessione:

- i server sono ancora più protetti:
  - isola da tutti gli attacchi che riguardano l'handshake TCP
  - isola da tutti gli attacchi che riguardano la frammentazione dei pacchetti IP
- può autenticare i client (ma allora richiede modifiche alle applicazioni)
- molte limitazioni proprie del packet filter rimangono

Il vantaggio di un circuit-level proxy è che offre un servizio per una vasta varietà di protocolli diversi, a differenza degli application-level che per ogni applicazione richiedono uno specifico proxy.
Bisogna, però, evidenziare una nota dolente del circuit-level proxy server e cioè che provvede in modo superficiale al controllo attraverso il proxy, in quanto è abbastanza complicato determinare se i comandi inviati attraverso di esso sono sicuri o provenienti da un attacco.

3.2.4 Utilizzare il proxying con i servizi di Internet

Per far si ché il proxying interferisca con comunicazioni tra un client e server, deve essere adattato separatamente ad ogni servizio. Alcune cose che sono facili diventano molto più difficili quando è coinvolto un proxy.

Il servizio ideale per il proxying è quello che fa un collegamento TCP in una direzione (da un client interno ad un server esterno) ed ha solo comandi sicuri.

Siccome il TCP è un protocollo a collegamento-diretto, basta preparare una sola volta il proxy, continuando ad utilizzare successivamente sempre lo stesso collegamento. L'UDP non ha nessun concetto sui collegamenti; ogni pacchetto è un'operazione separata che richiede una decisione separata dal proxy server. Il TCP è perciò più indicato in un proxying.

È facile per un proxy server intercettare il collegamento iniziale da un client ad un server. È più difficile intercettare un collegamento di ritorno. Ai fini della conversazione si deve essere consapevoli dell'esistenza del proxy server, o del fatto che il server ha bisogno di saper interpretare e cambiare il protocollo per essere certi che il collegamento di ritorno sia fatto correttamente.

Per alcuni servizi il proxying può essere tecnicamente facile, ma non da un punto di vista della sicurezza. Se un protocollo è pericoloso, utilizzare il proxying senza far nient'altro non lo renderà più sicuro.

Se è difficile la distinzione tra operazioni sicure ed insicure in un protocollo o impossibile usare il servizio per tutte le operazioni, il proxying non è la giusta soluzione al problema della sicurezza. In questo caso si può essere costretti ad utilizzare un host vittima (vedi cap. 2 – bastion host).

La maggior parte dei proxy server è disegnata per situazioni nelle quali il client è a valle del firewall ed il server è a monte. Il proxying per client esterni (a monte del firewall) viene utilizzato solo in alcune situazioni:

- si utilizzano procedure cambiate per offrire servizi in viaggio di ritorno per i propri utenti
- se si offre un servizio speciale ai client

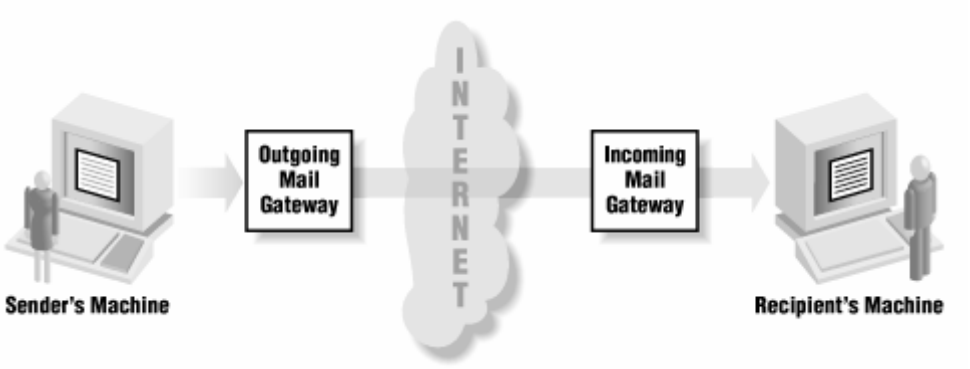
3.2.5 Proxying senza il server proxy

Alcuni servizi, definiti "store and forward", come SMTP, sostengono naturalmente il proxying. Questi servizi sono progettati in modo tale che i messaggi (le e-mail per SMTP) siano ricevuti da un server e poi immagazzinati, finché possono essere spediti ad un altro server. Per SMTP, i messaggi sono spediti verso la destinazione. Con tale schema, ogni server intermedio si comporta come un proxy per il mittente originario o per il server.

Se si esaminano i "riceventi", ad esempio gli headers delle e-mail (questi headers tracciano il percorso di un messaggio attraverso la rete dal mittente al destinatario), si scopre rapidamente che pochi messaggi viaggiano direttamente dalla macchina del mittente alla macchina del destinatario. Il messaggio passa almeno attraverso quattro macchine:

- la macchina del mittente
- l'ingresso di posta in partenza al luogo del mittente (o il provider di Internet del mittente)
- l'ingresso di posta in arrivo al luogo del destinatario
- la macchina del destinatario

Ognuno dei server intermedi (gli ingressi di posta) si comporta come un server proxy per il mittente, anche se il mittente non sta trattando direttamente con loro. La figura seguente illustra questa situazione.



3.3 SOCKS per proxy server

Il pacchetto SOCKS, originalmente scritto da David Koblas e Michelle Koblas, è un esempio del tipo di proxy server che richiede client classici. SOCKS è disponibile in modo free ed è divenuta il pacchetto standard di proxying su Internet. Il pacchetto si avvia a divenire uno standard ufficiale di Internet (vedi http://www.socks.nec.com**)** .

SOCKS è estremamente generico in modo tale da essere sostenuto da nuovi client. Questo è ciò che lo rende così popolare, ma ha lo svantaggio che non può offrire un accesso intelligente o controllo di accesso.

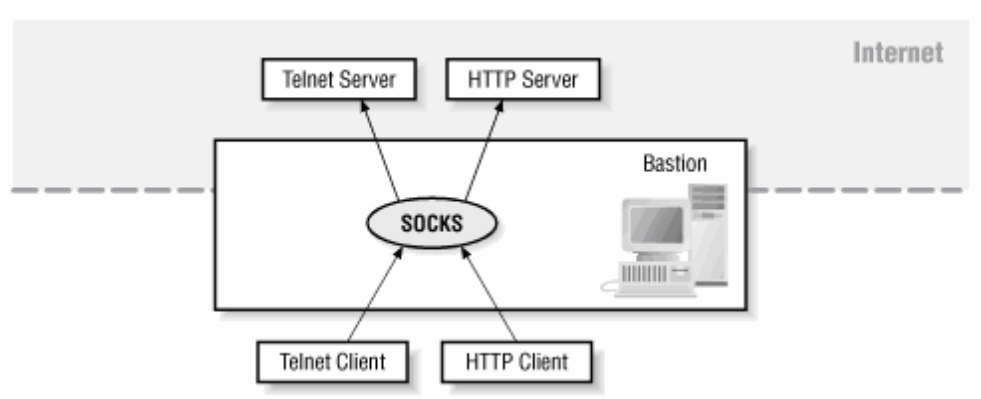
Un inconveniente di SOCKS è che funziona solamente per client basati sul TCP; non funziona per client basati sull'UDP. Se si sta utilizzando un cliente basato sull'UDP bisogna utilizzare un altro pacchetto, ovvero l'UDP Packet Relayer. Questo programma opera analogamente a SOCKS, solo che si utilizza per i client basati su UDP. Come SOCKS, l'UDP Packet Relayer è disponibile in modo free su Internet.

Il primo vantaggio di SOCKS è la sua popolarità. Poiché SOCKS è usato in modo esteso, le realizzazioni di server e i client SOCKS (ad es. versioni di vari programmi, come FTP e Telnet, sono stati gia convertiti per usare SOCKS) sono comunemente disponibili. Il pacchetto SOCKS include i seguenti componenti:

- il SOCKS server: questo server va installato su sistema Unix, anche se è stato portato su molte piattaforme diverse da Unix
- la libreria SOCKS di client per le macchine Unix
- versioni SOCKS-ified di programmi client standard UNIX, come FTP e Telnet

In aggiunta sono disponibili, come pacchetti separati, le librerie client per sistemi Macintosh e Windows.

La figura mostra l'uso di SOCKS per il proxying.



Molti programmi client di Internet supportano egregiamente SOCKS, integrandolo come opzione in fase di compilazione o di esecuzione.

Come si converte un programma client per l'utilizzo di SOCKS?

C'è bisogno di cambiare il programma in modo tale che possa parlare con il SOCKS server, piuttosto che direttamente col mondo reale. Questo si fa ricompilando il programma con la libreria SOCKS.

Convertire un programma client per l'utilizzo di SOCKS è abbastanza facile. Il pacchetto SOCKS fa certe assunzioni riguardo il funzionamento del programma client, e la maggior parte dei programmi client già seguono queste assunzioni. Per convertire un programma client, bisogna sostituire tutte le funzioni standard di rete con le corrispettive funzioni SOCKS. Ecco le funzioni:

Funzione di rete standard	**Versione SOCKS**
connect()	Rconnect()
getsockname()	Rgetsockname()
bind()	Rbind()
accept()	Raccept()
listen()	Rlisten()
select()	Rselect()

Di solito ciò si può fare semplicemente aggiungendo "CFLAGS = " linea del Makefile del programma:

- Dconnect=Rconnect
- Dgetsockname=Rgetsockname
- Dbind=Rbind
- Daccept=Raccept
- Dlisten=Rlisten
- Dselect=Rselect

Poi si ricompila e si collega il programma con la libreria SOCKS del client.

La macchina client ha bisogno non solo di avere i client SOCK modificati, ma anche qualche cosa per dirgli quale server SOCKS contattare per avere determinati servizi (su macchine Unix, il file */etc/socks.conf*).

3.3.1 Configurare il Proxy Server

Il programma SOCKS ha bisogno di due file di configurazione separati. Il primo per specificare gli accessi autorizzati, il secondo per instradare le richieste al proxy server appropriato. Il file di accesso dovrebbe essere localizzato sul server. Il file di instradamento dovrebbe trovarsi su ogni macchina Unix.

3.3.1.1 Il file di accesso

Con socks 4.2c Beta, il file di accesso è denominato "sockd.conf". Dovrebbe contenere 2 righe, una riga permit e una riga deny. Ogni riga conterrà tre campi:

- L'identificatore (permit/deny)
- L'indirizzo IP
- Il modificatore di indirizzo

L'identificatore è di tipo permit oppure deny. È consigliabile avere sia la riga permit, sia la riga deny.

L'indirizzo IP è un indirizzo a 4 byte come nella tipica notazione IP. Ossia, 192.168.1.0.

Anche il modificatore di indirizzo è un tipico indirizzo IP rappresentato da un numero di 4 byte e funziona come una netmask. Supponiamo che questo numero sia a 32 bit (serie di 1 e di 0). Se il bit è un 1, il bit corrispondente dell'indirizzo che si sta controllando deve essere uguale al bit corrispondente presente nel campo dell'indirizzo IP.

Ad esempio, se la riga è:

```
permit 192.168.1.23 255.255.255.255
```

saranno ammessi solo gli indirizzi IP i cui bit corrispondono a 192.168.1.23, ossia, solo 192.168.1.23. La riga:

```
permit 192.168.1.0 255.255.255.0
```

ammetterà ogni numero all'interno del gruppo da 192.168.1.0 a 192.168.1.255, ossia l'intero dominio di Classe C. Non si dovrebbe invece avere la riga:

```
permit 192.168.1.0 0.0.0.0
```

dal momento che ammetterà qualsiasi indirizzo, indistintamente.

Pertanto, prima di tutto si abilitino tutti gli indirizzi che si vogliono abilitare e si neghino i restanti. Per ammettere tutti coloro che sono presenti nel dominio 192.168.1.xxx, le righe:

```
permit 192.168.1.0 255.255.255.0
deny 0.0.0.0 0.0.0.0
```

funzioneranno correttamente. E’ da notare il primo "0.0.0.0" della riga deny. Con un modificatore 0.0.0.0, il campo dell'indirizzo IP non è rilevante. Tutti zero è la norma perché è semplice digitarli.

È consentita più di una voce per ciascun tipo.

3.3.1.2 Il file di instradamento

Il file di instradamento in SOCKS è chiamato "socks.conf", appare molto simile a quello del file di accesso, pertanto potrebbe essere facile confonderli.

Il file di instradamento informa il client SOCKS quando utilizzare socks e quando non farlo. Naturalmente non è necessario SOCKS per parlare con se stessi. Esistono tre voci:

- deny
- direct
- sockd

Deny specifica a SOCKS quando respingere una richiesta. Questa voce possiede gli stessi tre campi già descritti per sockd.conf: identificatore, indirizzo e modificatore. Generalmente, dal momento che questo viene gestito anche da sockd.conf, il file di accesso, il campo del modificatore

è impostato a 0.0.0.0. Se si vuole precludere se stessi dal chiamare qualsiasi posto, può essere fatto in questo punto.

La voce direct specifica gli indirizzi per i quali non deve essere utilizzato socks. Si tratta degli indirizzi che possono essere raggiunti senza il proxy server. Ancora una volta, abbiamo i tre campi: identificatore, indirizzo e modificatore. Nel nostro esempio corrisponderebbe a:

```
direct 192.168.1.0 255.255.255.0
```

che permette di andare direttamente ovunque nella nostra rete protetta.

La voce sockd infine specifica al computer quale host ha in esecuzione il demone server socks. La sintassi è:

```
sockd @=<serverlist> <IP address> <modifier>
```

Si noti la voce @=. Questa permette di impostare gli indirizzi IP di una lista di proxy server. Nel nostro esempio, viene utilizzato un unico proxy server. Tuttavia è possibile averne molti per consentire un carico maggiore e per avere a disposizione una ridondanza in caso di errore.

I campi di indirizzo IP e di modificatore funzionano esattamente come negli altri esempi. Attraverso questi è possibile specificare dove devono andare gli indirizzi.

Le insidie della Rete

Capuano - Guida - Brasiello - Manco

Il pericolo proveniente dalla Rete non deve essere minimamente sottovalutato. Gli hacker malintenzionati sono ovunque e bisogna essere sempre preparati a respingere un attacco e a predisporre le contromisure ad una possibile intrusione. In questo caso la migliore mossa è conoscere le loro armi e questo capitolo ci aiuterà a conoscerle.

4.1 Il furto di informazioni

4.1.1 Mapping

Nel "mondo reale" un attacco è spesso preceduto dalla raccolta di informazioni, questo è vero anche nel mondo informatico. Prima di attaccare una rete, gli attaccanti vorrebbero sapere gli indirizzi IP delle macchine connesse alla rete, i sistemi operativi che usano, ed i servizi che offrono. Con queste informazioni, i loro attacchi possono essere più precisi ed avere meno probabilità di essere scoperti. Il processo di raccolta di queste informazioni è noto come *mapping*. Un programma come ping può essere usato per determinare gli indirizzi IP delle macchine sulla rete osservando semplicemente quali indirizzi IP rispondono ad un messaggio ping. La *scansione delle porte* (*port scanning*) è una tecnica che consiste nel contattare sequenzialmente (o con una richiesta di connessione TCP, o con un semplice datagram UDP) i numeri di porta di una macchina ed osservare cosa succede in risposta. Queste risposte a loro volta possono essere utilizzate per determinare i servizi offerti (per esempio, http o FTP) dalla macchina. Nmap è una utilità open source ampiamente usata per "esplorazione di rete e verifica di sicurezza" che effettua la scansione delle porte. Molti firewall, come quelli venduti da Check point, individuano attività di mapping e scansione delle porte, come altre "attività maliziose", e riportano queste attività al gestore della rete.

4.1.2 Packet sniffing

Un *packet sniffing* è un programma che gira su un dispositivo attaccato alla rete che riceve passivamente tutti i frame dello strato di collegamento che passano attraverso l'adattatore di rete del dispositivo. In un ambiente broadcast, come una LAN Ethernet, questo significa che il packet sniffing riceve tutti i frame che vengono trasmessi da o a tutti i terminali sulla LAN. Qualunque terminale con una scheda Ethernet può facilmente essere utilizzato come packet sniffer, dato che l'adattatore Ethernet deve essere solo impostato a *modo promiscuo* per ricevere tutti i frame Ethernet in transito. Questi frame, a loro volta, possono essere passati a programmi applicativi che estraggono i dati di strato di applicazione. Per esempio, le password in applicazioni varie. Avendo ottenuto le password per gli account di utente, gli attaccanti possono utilizzare in seguito questi account per lanciare attacchi di "negazione di servizio" (*denial*

of service), come discusso in seguito. Il packet sniffing è un'arma a doppio taglio: può essere prezioso per un amministratore di rete per il monitoraggio e la gestione della rete ma può anche essere usato da un hacker senza scrupoli. Il software per il packet sniffing è disponibile gratuitamente in vari siti Web, e come prodotto commerciale. La chiave per rilevare il packet sniffing è individuare interfacce di rete che stanno funzionando in modalità promiscua. All'interno di un'azienda, i gestori di rete possono installare in tutti i computer dell'azienda software che avverte il gestore quando un'interfaccia viene configurata in modalità promiscua. Si possono anche mettere in atto da remoto vari trucchi per evidenziare interfacce promiscue. La chiave per non temere il packet sniffing è di cifrare tutti i dati (in particolare le password) che attraversano un link della rete.

4.2 Effettuare un attacco

E' possibile falsificare l'indirizzo IP sorgente della connessione in modo da far credere di essere un altro host per poter superare certe difese o per portare a termine certe tipologie di attacchi. In questo caso si parla di IP Spoofing. Poiché i protocolli TCP, UDP, ICMP (e altri) sono incapsulati in pacchetti IP tutti questi protocolli sono di riflesso teoricamente spoofabili. A seconda della tipologia di attacco che l'attaccante vuole portare a termine e della posizione sulla rete in cui egli si trova ci sono differenti tecniche basate sull'IP Spoofing.
Le categorie di attacchi che esamineremo in questo paragrafo sono:

1. IP Spoofing non cieco
2. IP Spoofing cieco
3. DoS

Il concetto di Spoofing non cieco è molto semplice: l'attaccante cerca di farsi passare per un host che fa parte della sottorete in cui egli stesso si trova, quindi i pacchetti indirizzati all'host (di cui si è rubato l'IP) sono visibili anche a lui e perciò è a conoscenza del *Sequence number* e dell'*Acknowledgement number* aggiornati.
Per quanto riguarda lo Spoofing cieco, l'attaccante sta cercando di farsi passare per un host qualsiasi (che non è nella sua sottorete) e quindi in qualche maniera dovrà cercare di "indovinare" il *Sequence number* corretto per continuare la connessione.
Infine gli attacchi di tipo DoS (Denial of Service) sono attacchi diretti a bloccare un determinato host impedendogli di svolgere le sue normali attività e quindi negando agli utenti di poter accedere ai servizi a cui l'host era preposto.

4.2.1 IP Spoofing non cieco

Solitamente i vari host di una sottorete sono collegati da apparecchi (HUB) che, per far arrivare un pacchetto ad un determinato host, lo trasmettono in

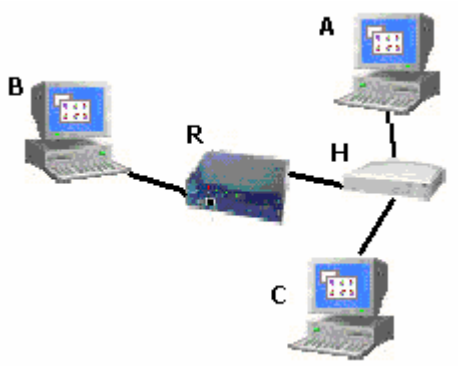
broadcast a tutti i computer della sottorete. Quando il pacchetto arriva ad un determinato host, quest'ultimo esamina l'indirizzo di destinazione contenuto nel pacchetto. Se questo indirizzo coincide con quello dell'host stesso, il pacchetto viene processato, altrimenti viene scartato in quanto era destinato ad un altro host. Tuttavia esiste una modalità particolare in cui è possibile impostare la scheda di rete: la "modalità promiscua". Quando la scheda di rete si trova in questa modalità, permette di processare tutti i pacchetti che arrivano.
Come già accennato sopra, nel caso dello Spoofing non cieco l'attaccante sta cercando di farsi passare per un host che fa parte della sua sottorete; quindi, impostando la scheda di rete in modo promiscuo, egli riesce a leggere tutti i pacchetti inviati, anche quelli che non sono destinati ad esso stesso, in particolare lui farà attenzione ai pacchetti indirizzati all'host che intende impersonare e può così scoprire il Sequence number e l'Acknowledgement number della connessione in corso e cercare di inserirvisi. Alcuni tipi di attacchi che possono essere messi a segno con questa tecnica sono la chiusura di una connessione esistente e l'Hijacking.
C'è da notare che ultimamente al posto degli HUB sono sempre più usati degli apparecchi chiamati switch, che invece di trasmettere i pacchetti in broadcast li trasmettono solo al destinatario corretto. Questo permette di eliminare certe tipologie di attacchi quali Sniffing e IP Spoofing non cieco, aumentando allo stesso tempo le prestazioni della rete.

4.2.1.1 Chiusura di una connessione esistente

Avendo a disposizione Sequence number e Acknowledgement number di una connessione in corso l'attaccante può spedire in un momento preciso un pacchetto creato accuratamente con l'intento di far cadere la connessione. Per raggiungere questo obiettivo può usare uno dei due flag RST o FIN compresi nell'header dei pacchetti TCP.

Chiusura di una connessione esistente usando il flag RST

Il flag RST, tra le altre cose, viene utilizzato per reinizializzare una connessione che è diventata instabile per qualche motivo. Un pacchetto TCP avente lo scopo di resettare una connessione ha solamente il Sequence number valido, mentre l'Acknowledgement number è disabilitato. Per procedere al reset di una connessione esistente l'attaccante procede secondo alcuni passi. Immaginiamo che esista una situazione del tipo in figura dove:



A e C = Host della stessa sottorete
B = Host di una rete diversa da A e C
H = HUB
R = Router che delimita la sottorete

Supponiamo che tra gli host A e B esista una connessione e che l'attaccante si trovi nella postazione C. Per cercare di resettare la connessione esistente, come prima cosa l'attaccante aspetterà di ricevere un pacchetto proveniente dalla connessione A-B. Supponendo che riceva un pacchetto proveniente da B verso A, egli prima calcolerà il Sequence number a partire dall'Acknowledgement number del pacchetto ricevuto, poi costruirà e spedirà un pacchetto con le seguenti impostazioni:

Campi del pacchetto IP:

- IP sorgente uguale a quello di A (IP Spoofato)
- IP di destinazione uguale a quello di B

Campi del pacchetto TCP:

- Porta sorgente uguale alla porta usata da A
- Porta di destinazione uguale alla porta usata da B
- Sequence number contenente il valore appena calcolato
- Flag RST impostato

Il risultato sarà il reset della connessione.

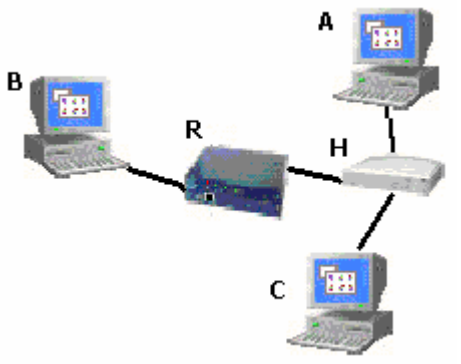
Ci sono però dei problemi tecnici: questo metodo, infatti, funziona solo se il pacchetto dell'attaccante arriva prima della reale risposta dell'host A. L'host B riceverà due pacchetti con lo stesso Sequence number (uno mandato dall'attaccante, e l'altro mandato dall'host A), quindi prenderà per buono il primo arrivato e scarterà il secondo credendolo un duplicato.

Chiusura di una connessione esistente usando il flag FIN

Il flag FIN sta ad indicare che “non ci sono più dati da parte del mittente”. Se vengono ricevuti altri pacchetti, dopo che un host ha spedito il flag FIN e questo è stato accettato, questi vengono ignorati. Per procedere alla chiusura di una connessione esistente usando il flag FIN l'attaccante procede in modo simile al caso precedente, con alcune differenze.

La caratteristica di questa tecnica è che l'attaccante ha la possibilità di controllare se la connessione è stata chiusa in quanto l'host che riceverà il pacchetto di FIN risponderà con un pacchetto di Acknowledgement.
A differenza del caso precedente, il pacchetto che verrà costruito deve avere sia il Sequence number che l'Acknowledgement number validi.

Riprendiamo la situazione precedente:



A e C = Host della stessa sottorete
B = Host di una rete diversa da A e C
H = HUB
R = Router che delimita la sottorete

Supponiamo che tra gli host A e B esista una connessione e che l'attaccante si trovi nella postazione C. Per cercare di chiudere la connessione esistente, come prima cosa l'attaccante aspetterà di ricevere un pacchetto proveniente dalla connessione A-B. Supponendo che riceva un pacchetto proveniente da B verso A, egli prima calcolerà il Sequence number e l'Acknowledgement number in base al pacchetto ricevuto, poi costruirà e spedirà un pacchetto con le seguenti impostazioni:

Campi del pacchetto IP:

- IP sorgente uguale a quello di A (IP Spoofato)
- IP di destinazione uguale a quello di B

Campi del Pacchetto TCP:

- Porta sorgente uguale alla porta usata da A
- Porta di destinazione uguale alla porta usata da B
- Sequence number contenente il valore appena calcolato
- Acknowledgement number contenente il valore appena calcolato
- Flag FIN impostato

A questo punto resterà in ascolto in attesa di un pacchetto di Acknowledgement da parte di B. Se lo riceve é sicuro che da questo momento B risponderà a tutti i pacchetti ricevuti da A con un pacchetto di reset credendo siano bugs facendo, tra l'altro, cadere anche l'altro senso di connessione.

4.2.1.2 Hijacking

L'hijacking è una tecnica molto raffinata che permette di intromettersi in una connessione esistente e prenderne il controllo. Molte volte l'hijacking viene snobbato in quanto trovandosi già nella sottorete dell'host che si vuole impersonare basterebbe usare uno sniffer per catturare username e password, per poi collegarsi "legalmente". Tuttavia è possibile trovarsi in situazioni in

cui, ad esempio, vengono usate password "usa e getta" e quindi, anche se l'attaccante riuscisse a sniffarne qualcuna, quando andrà ad usarle queste saranno già scadute.

Stato di desincronizzazione

Prima di proseguire con la spiegazione sul funzionamento dell'hijacking è necessario chiarire cosa si intende per stato di desincronizzazione di una connessione. Per semplicità da adesso in poi indicheremo con:

- SVR_SEQ il Sequence number del prossimo byte che il server spedirà
- SVR_ACK il prossimo byte che il server si aspetta di ricevere
- SRV_WND la grandezza della finestra di ricezione del server
- CLT_SEQ il Sequence number del prossimo byte che il client spedirà
- CLT_ACK il prossimo byte che il client si aspetta di ricevere
- CLT_WND la grandezza della finestra di ricezione del client

In una situazione di "calma" durante una connessione, cioè un momento in cui non vengono spediti dati da entrambe le parti, le seguenti equazioni sono vere:

SVR_SEQ = CLT_ACK e CLT_SEQ = SRV_ACK

Invece mentre sono trasferiti dei dati sono vere queste altre espressioni:

CLT_ACK ⇐ SVR_SEQ ⇐ CLT_ACK + CLT_WND
SRV_ACK ⇐ CLT_SEQ ⇐ SRV_ACK + SRV_WND

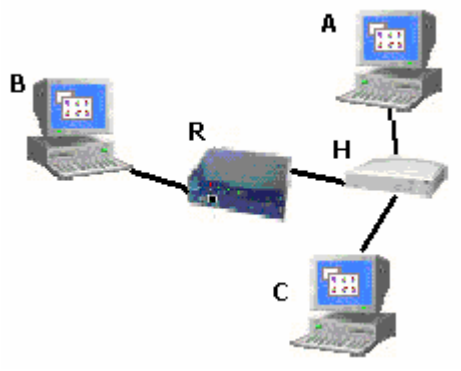
da cui si può capire che un pacchetto è accettabile se il suo Sequence number appartiene all'intervallo [SRV_ACK, SRV_ACK + SRV_WND] per il server e [CLT_ACK, CLT_ACK + CLT_WND] per il client. Se il Sequence number supera o precede questo intervallo il pacchetto viene scartato e viene spedito un pacchetto contenente nell'Acknowledgement number il Sequence number del prossimo byte atteso.
Il termine desincronizzazione si riferisce ad una situazione in cui durante una connessione in un periodo di "calma" sono vere le seguenti equazioni SVR_SEQ != CLT_ACK e CLT_SEQ != SRV_ACK (dove != sta a significare diverso).

Se una connessione si trovasse in una situazione di questo tipo e dei dati venissero spediti da una delle parti potrebbero presentarsi due casi distinti:

- se CLT_SEQ < SVR_ACK + SVR_WND e CLT_SEQ > SVR_ACK il pacchetto è accettabile e i dati vengono memorizzati per un uso futuro, ma non vengono processati
- se CLT_SEQ > SVR_ACK + SVR_WND o CLT_SEQ < SVR_ACK il pacchetto non è accettabile e viene scartato

In pratica se una connessione è in questo stato i due host non possono scambiarsi dati.

L'attacco
Riprendiamo la situazione precedente:



A e C = Host della stessa sottorete
B = Host di una rete diversa da A e C
H = HUB
R = Router che delimita la sottorete

Supponiamo che esista una connessione Telnet da A verso B e che l'attaccante si trovi nella postazione C. Cosa succederebbe se quest'ultimo in un periodo di "calma" della connessione tra A e B mandasse un pacchetto spoofato a B in modo da far credere che provenga da A? Semplice, B aggiornerebbe l'Acknowledgement number di A (SVR_ACK) in base al pacchetto ricevuto desincronizzandosi dal Sequence number reale di A (CLT_SEQ). A questo punto i pacchetti spediti da A verranno scartati in quanto per B hanno un Sequence number errato. Vediamo un esempio per capire meglio:

1. A spedisce un pacchetto a B contenente 10 Byte di dati con Sequence number=100 e Acknowledgement number=500
2. B aggiorna Sequence number e Acknowledgement number:
   Sequence number=500
   Acknowledgement number =100+10
3. B spedisce un pacchetto ad A contenente 15 Byte di dati con Sequence number=500 e Acknowledgement number=110
4. Il pacchetto arriva ad A che riaggiorna ACK e SEQ:
   Sequence number=110
   Acknowledgement number=500+15

A questo punto si intromette l'attaccante con un pacchetto Spoofato, usando il Sequence number e l'Acknowledgement number corretti.

5. C spedisce un pacchetto spoofato contenente 20 Byte di dati con Sequence number=110 e Acknowledgement number=515
6. B riaggiorna l'ACK e il SEQ:
   Sequence number=515
   Acknowledgement number=110+20

A questo punto B è desincronizzato rispetto ad A in quanto il prossimo byte che B si aspetta da A è il 130, mentre il Sequence number di A è a 110. Quindi i pacchetti che A spedirà a B da questo momento in poi verranno scartati.
L'attaccante però sa cosa si aspetta B e perciò può mandare dei pacchetti creati appositamente per essere accettati. Ricordiamo che nel nostro esempio la connessione in corso era una sessione Telnet, quindi adesso l'attaccante può mandare comandi di shell a B come se fosse A.
C'è da notare che nell'esempio appena descritto non c'è una desincronizzazione da entrambe le parti, infatti abbiamo che CLT_SEQ != SRV_ACK ma SVR_SEQ = CLT_ACK. Quindi l'host A accetterà tutti i pacchetti spediti da B come risposta ai comandi dell'attaccante, e quindi vedrà tutto quello che questi sta facendo. Per evitare ciò l'attaccante deve creare una situazione di desincronizzazione anche nell'altro senso di trasmissione spedendo un pacchetto spoofato ad A come se provenisse da B. Ricordiamo che essendo l'attaccante nella stessa sottorete di A é in grado di vedere l'output dei propri comandi sniffando i pacchetti di risposta spediti da B.

Ci sono varie tecniche per ottenere la desincronizzazione di una connessione. Quella vista nell'esempio è quella usata solitamente e consiste appunto nello spedire un pacchetto spoofato contenente dei dati sia al server che al client. Più è grande il numero di byte spediti in questi pacchetti e più è grande la desincronizzazione che si andrà a creare tra i due host.
Esiste comunque un secondo e più raffinato metodo di desincronizzazione, che consiste nell'intromettersi nel protocollo three-way handshake usato per creare una connessione. Il funzionamento si può sintetizzare in 4 punti:

1. l'attaccante aspetta di ricevere il pacchetto SYN/ACK, proveniente dal server e diretto verso il client (secondo passo del three-way handshake), della connessione da desincronizzare
2. appena lo ha identificato spedisce un pacchetto di RST (Spoofato) verso il server e immediatamente dopo uno di SYN (sempre Spoofato) con gli stessi parametri (porta TCP e indirizzo IP) usati per la connessione da desincronizzare, ma con un differente Sequence number

3. il server chiuderà la prima connessione grazie al pacchetto RST, e ne aprirà una uguale ma con un Sequence number diverso, spedendo il pacchetto SYN/ACK
4. l'attaccante non appena identifica quest'ultimo, spedisce il pacchetto ACK necessario a completare l'instaurazione della connessione

A questo punto la connessione è aperta, ma è in uno stato di desincronizzazione in quanto per il client il Sequence number corretto è quello che era presente nel pacchetto SYN/ACK intercettato dall'attaccante al punto 1, mentre per il server quello corretto è quello introdotto dall'attaccante nel punto 2.

Il problema dell'ACK Storm

Se il Sequence number di un pacchetto supera o precede l'intervallo di accettazione il pacchetto viene scartato e viene spedito un pacchetto contenente il Sequence number del prossimo byte atteso. Se ci si trova in uno stato di desincronizzazione con tutta probabilità i pacchetti che il client e server si scambieranno avranno errori di questo tipo, e quindi verranno generati questi pacchetti di errore che però a loro volta non sono validi. Questo genera un loop di pacchetti di Acknowledgement chiamato appunto ACK Storm. Se la desincronizzazione è completa, cioè da entrambe i sensi di comunicazione, viene creato un ACK Storm per ogni pacchetto spedito da entrambe gli host. Se la desincronizzazione è parziale (solitamente viene desincronizzata solo la direzione del client verso il server) verrà creato un ACK Storm solo dai pacchetti che provengono da quella direzione. Quando vengono spediti dei dati in un canale desincronizzato c'è da notare che oltre ad essere provocato un ACK Storm, il pacchetto non viene confermato e quindi dopo un pò il mittente lo ritrasmetterà credendo che sia andato perso, creando un altro ACK Storm. Sarà compito dell'attaccante spedire dei pacchetti di conferma per impedire questi ulteriori ACK Storm. Questi loop si fermeranno solo quando uno dei pacchetti dell'ACK Storm andrà perso.

4.2.2 IP Spoofing cieco

Quando si parla di IP Spoofing solitamente ci si riferisce all'IP Spoofing cieco. Con questa tecnica l'attaccante cerca di farsi passare per un host qualunque di internet, non facente parte della sottorete in cui si trova. In questo caso si parla di Spoofing cieco in quanto l'attaccante non avrà nessuna possibilità di vedere i pacchetti mandati in risposta ai pacchetti (spoofati) che ha spedito. Questi ultimi infatti saranno indirizzati all'host che egli sta impersonando, impedendogli quindi di venire a conoscenza del Acknowledgement number e Sequence number corretti per continuare la connessione. Come si può dedurre

le possibilità dello Spoofing cieco sono infinitamente inferiori a quelle che si poteva avere con lo Spoofing non cieco, ma comunque qualcosa si può fare.
Molte volte alcuni server danno un accesso privilegiato a degli host fidati attraverso servizi di rlogin senza password o servizi simili. In un caso di questo tipo l'unico controllo è fatto sul numero IP sorgente della connessione, quindi se un attaccante riuscisse ad aprire una connessione spoofando il proprio IP potrebbe lanciare dei comandi al server facendogli credere di essere l'host fidato. Se siete stati attenti fino a questo punto avrete già intuito che questa non è un'operazione semplice. Andiamo a rivedere brevemente il contenuto dei 3 pacchetti TCP necessari per instaurare una connessione:

Pacchetto 1:
- Direzione: Client al Server
- Flag Attivi: SYN
- Sequence number: X (numero generato dal client)

Pacchetto 2:
- Direzione: Server al Client
- Flag Attivi: SYN, ACK
- Sequence number: Y (numero generato dal server)
- Acknowledgement number: X + 1

Pacchetto 3:
- Direzione: Client al Server
- Flag Attivi: ACK
- Sequence number: X + 1
- Acknowledgement number: Y + 1

Se un attaccante tentasse di instaurare una connessione con il proprio IP Spoofato, come detto prima, non sarebbe in grado di vedere il pacchetto di risposta del server, ovvero il secondo dei 3 pacchetti. Questo è un problema, in quanto non sapendo il Sequence number del pacchetto 2 non saprà qual è l'Acknowledgement number corretto per il pacchetto 3 e quindi l'instaurazione della connessione non potrà andare a buon fine. L'unica soluzione a questo problema è cercare di predire il Sequence number corretto. Sebbene questa tipologia di attacco fosse stata introdotta a livello teorico molto tempo prima, la prima registrazione ufficiale risale alla vigilia di Natale del 1994 e fu portata a termine dall'hacker Kevin Mitnick.

4.2.2.1 Piccolo dettaglio tecnico

Supponiamo che un attaccante voglia mettere in pratica quanto appena descritto, cioè tentare di aprire una connessione con un server spoofando il proprio IP in maniera da farsi passare per un host fidato. Un problema che si presenta all'attaccante ancora prima della predizione del Sequence number del secondo pacchetto è che l'host che egli cerca di impersonare, non appena riceverà il secondo pacchetto, si renderà conto che non sta cercando di aprire una connessione: quindi avviserà il server mandandogli un pacchetto di reset rendendo nullo il lavoro dell'attaccante. Per impedire che ciò accada, quest'ultimo o aspetta che l'host da impersonare sia spento, o può cercare di "buttarlo giù" con un attacco del tipo Denail of Service. L'attacco classico è il SYN flood, anche se ultimamente molti host incorporano una protezione per questo tipo di attacco. Se l'host dovesse avere questo tipo di protezione, l'attaccante potrà provare con un altro dei numerosi DoS esistenti.

4.2.2.2 Generazione del Sequence Number

E adesso veniamo al problema vero e proprio, come fa l'attaccante a predire un numero a 32 bit? Bisogna tenere conto che il TCP è stato creato per la gestione di connessioni e non per risolvere problemi di sicurezza, quindi non dobbiamo stupirci se molti sistemi operativi relativamente vecchi risultano vulnerabili a questa tipologia di attacco. Per capire com'è possibile predire il Sequence number andiamo a vedere come i server scelgono questo numero. Fondamentalmente esistono tre modi in cui il Sequence number può essere generato, due dei quali sono vulnerabili all'IP Spoofing. Vediamoli in dettaglio.

Generazione in base alla regola dei 64k
Questo sistema è molto semplice, ma nonostante ciò è molto usato. Le regole usate per generare il Sequence number sono le seguenti:

- incrementa ogni secondo il contatore del Sequence number di una costante, solitamente 128000
- se è stata aperta una connessione incrementa il contatore del Sequence number di un'altra costante, solitamente 64000

Generazione in base al Tempo
Anche questa è una tecnica semplice ed abbastanza usata. Per generare il Sequence number, dopo un'inizializzazione casuale al boot, il contatore del Sequence number viene incrementato ogni periodo di tempo prefissato. Un esempio può essere l'incremento di 1 ogni microsecondo.

Generazione Random
Con questa tecnica il Sequence number viene generato quasi casualmente, ed è pressoché impossibile predirlo. Un esempio di questa tecnica si può trovare nei nuovi kernel di Linux.

4.2.2.3 Predizione del Sequence Number

Come fa quindi l'attaccante a predire il Sequence number per portare a termine il suo attacco? Prima di tutto deve scoprire quale dei tre algoritmi per la generazione del Sequence number è in uso sul server che vuole colpire. Per fare ciò manda qualche pacchetto SYN non spoofato per vedere ed analizzare le risposte del server. Questi pacchetti di risposta gli permettono di capire con una certa facilità a quale dei tre algoritmi si trova di fronte. Per vedere se il server sta usando l'algoritmo che usa la regola dei 64k gli è sufficiente calcolare la differenza tra 2 pacchetti ricevuti e vedere se il numero ottenuto è divisibile per 64000. Per capire se il server sta usando l'algoritmo che usa la regola in base al tempo dovrà fare dei campionamenti su una serie di pacchetti. Innanzitutto spedirà una serie di pacchetti SYN e salverà il tempo in cui gli arriva il pacchetto SYN/ACK di risposta e il relativo Sequence number. Quindi andrà a calcolare la differenza tra i vari tempi registrati ottenendo l'intervallo di tempo trascorso tra due generazioni del Sequence number. Infine, facendo la divisione tra il risultato ottenuto e la differenza tra i Sequence number corrispondenti ai tempi presi in considerazione otterrà l'incremento per ogni unità di tempo. Se i risultati di tutti i campionamenti danno un valore simile si può pensare che il server usi questo algoritmo per generare i Sequence number. Chiaramente se questi due test non vanno a buon fine probabilmente il server usa un algoritmo di generazione random del Sequence number e per l'attaccante non sarà possibile continuare l'attacco. Se così non è il passo successivo sarà cercare di indovinare il prossimo Sequence number che verrà generato.
Nel caso della generazione in base alla regola dei 64k predire il Sequence Number è abbastanza semplice. Infatti all'attaccante sarà sufficiente spedire un pacchetto SYN non spoofato al server, leggere il Sequence Number del pacchetto giunto in risposta ed aggiungerci 64000. Il risultato con tutta probabilità sarà il prossimo Sequence Number per meno di un secondo, tempo dopo cui verrà incremento di 128000. Come si può intuire 1 secondo nel mondo dei computer è un tempo molto grande, quindi questo algoritmo è particolarmente vulnerabile ad un attacco di IP-Spoofing. Nel caso della generazione in base al tempo, invece, le cose per l'attaccante si complicano un po'. Il Sequence Number in questo caso verrà calcolato in base ai risultati dei campionamenti fatti in precedenza.

4.2.2.4 L'attacco

Adesso che sappiamo come l'attaccante fa a predire il Sequence number possiamo passare ad analizzare i passi successivi di un attacco di IP Spoofing cieco. Per prima cosa l'attaccante si assicurerà che l'host che intende impersonare sia spento, e nel caso non lo fosse, prenderà qualche iniziativa.
Dopo aver capito di fronte a che tipo di algoritmo per la generazione del Sequence number si trova procederà secondo i seguenti passi:

1. spedirà un pacchetto SYN non spoofato
2. in base al pacchetto SYN/ACK di risposta e all'algoritmo usato calcolerà il prossimo probabile Sequence number.
3. spedirà un pacchetto SYN spoofato con l'indirizzo sorgente dell'host fidato
4. spedirà il pacchetto ACK spoofato con l'indirizzo sorgente dell'host fidato con il Sequence number appena calcolato + 1.

In certi casi, specialmente in quelli in cui il Sequence number viene calcolato in base al tempo, indovinare il numero corretto è abbastanza difficile. Certe implementazioni del protocollo TCP in alcuni sistemi facilitano il compito all'attaccante. In questi sistemi quando viene ricevuto un Sequence number troppo alto rispetto a quello corretto viene generato un pacchetto di reset. Se, invece, il Sequence number è più basso di quello corretto non viene fatto assolutamente niente. Questo fatto viene sfruttato spedendo una serie di pacchetti ACK con Sequence number crescenti partendo da un valore leggermente inferiore a quello predetto per arrivare ad uno un pò più alto. A questo punto l'attaccante non ha la sicurezza matematica di essere riuscito ad aprire la connessione, ma ha comunque una probabilità abbastanza elevata. Poiché stava cercando di aprire una connessione con il servizio rlogin non gli sarà chiesta nessuna password per entrare. Adesso con tutta probabilità, quindi, avrà accesso ad una shell. Il passo successivo solitamente sarà quello di inviare il comando *echo "+ +" > .rhosts* che ha l'effetto di consentire l'accesso al sistema senza password tramite servizi del tipo di rlogin a tutti i numeri IP. Comunque esistono modi diversi di procedere, tutto dipende della fantasia e dagli obiettivi dell'attaccante.
L'ultimo particolare da tenere in considerazione è che se il server spedisce dei dati, l'attaccante non saprà esattamente quando lo fa e quanti byte spedisce, quindi non potrà spedire i pacchetti di Acknowledgement per confermare la ricezione dei dati. Questo effettivamente non è un problema per lui, infatti ciò causerà solamente la ritrasmissione dei dati dopo un certo tempo limite.

4.2.3 Denial of Service (DoS)

Un attacco di tipo DoS ha come scopo finale escludere un determinato host dalla rete rendendolo irraggiungibile. Fondamentalmente esistono 4 categorie di DoS:

1. attacchi per l'esaurimento di banda, che si basano sull'inondare la rete dell'host bersaglio in maniera da consumare tutta la larghezza di banda a lui disponibile. Questo tipo di attacco si può attuare in 2 modi distinti. Il primo caso si ha quando l'attaccante ha a disposizione una connessione più veloce della vittima e quindi riesce a saturarne la banda direttamente. Il secondo metodo si ha quando l'attaccate pur non avendo una connessione veloce riesce a saturare la banda della vittima grazie all'utilizzo di altri host che hanno lo scopo di amplificare l'attacco
2. attacchi per l'esaurimento delle risorse, che hanno come scopo colpire il sistema piuttosto che la rete in cui si trova. In genere questo si traduce con il consumo di risorse come cicli di CPU, della memoria, ecc.
3. attacchi contro difetti di programmazione, che vanno a colpire bug software o hardware. Solitamente si verificano quando vengono spediti dei dati non previsti all'elemento vulnerabile
4. attacchi DoS generici, che non rientrano nei casi precedenti

Molti attacchi di tipo DoS per il loro funzionamento fanno uso dell'IP Spoofing rendendo quasi impossibile capirne la provenienza. Andiamo adesso ad analizzare due esempi abbastanza popolari di DoS che fanno uso dell'IP Spoofing.

4.2.3.1 Smurf

Lo Smurf è un attacco DoS che ha lo scopo di esaurire l'intera banda dell'host vittima, sfruttando intere sottoreti che fungono da amplificatori. Solitamente ad ogni sottorete è associato un indirizzo IP di broadcast che ha per lo più funzioni diagnostiche. Se un pacchetto è indirizzato ad un indirizzo di broadcast significa che deve essere elaborato da tutti gli host della sottorete. Questo può essere utile in certi casi, ad esempio per verificare quanti host sono attivi in una sottorete é sufficiente fare un ping all'indirizzo di broadcast, anziché farne uno ad ogni indirizzo. Lo smurf combina questo meccanismo di risposta multipla, con l'IP Spoofing per creare un attacco pressoché impossibile da fermare. Il funzionamento dello smurf è molto semplice: l'attaccante spedisce una serie di pacchetti ICMP "ECHO REQUEST" Spoofati con l'indirizzo dell'host vittima a degli indirizzi di broadcast. Il risultato è che tutti gli host della sottorete elaborano il pacchetto e generano come risposta un pacchetto ICMP "ECHO REPLY" indirizzato all'host vittima. Per capire meglio prendiamo in esame questo esempio:

A = Host Attaccante
V = Host Vittima
S = Sottorete contenente 100 Host
B = Indirizzo di broadcast della sottorete S

1. A spedisce un pacchetto ICMP "ECHO REQUEST" spoofato con l'indirizzo di V all'indirizzo B di broadcast della sottorete
2. tutti i 100 host della sottorete rispondono con un pacchetto ICMP "ECHO REPLY" a V credendo che sia lui ad aver spedito il pacchetto precedente

Come si può capire con questa tecnica anche un host che ha a disposizione una banda limitata, conoscendo l'indirizzo di broadcast di qualche rete che ha un fattore di amplificazione elevato, può mettere in ginocchio connessioni molto veloci. Supponiamo infatti che l'attaccante abbia a disposizione una comune connessione a 56 kbps e che riesca a inviare con continuità ad una velocità di 5 k/s pacchetti ICMP "ECHO REQUEST" a degli indirizzi di broadcast che abbiano un fattore di amplificazione di 100. Facendo due semplici conti ci si accorge che l'attaccante riesce a generare un traffico di 500 k/s e cioè 4 Mbit al secondo.

4.2.3.2 SYN Flood

Il SYN Flood è un attacco DoS che ha come scopo ultimo l'esaurire le risorse del sistema vittima. Fino a qualche anno fa il SYN Flood era molto temuto, ma adesso molti sistemi incorporano delle protezioni che permettono loro di essere immuni a questo tipo di attacco. Come abbiamo visto più volte in questo articolo, quando un client vuole instaurare una connessione, spedisce al server un pacchetto SYN. Quando il server riceve questo pacchetto alloca una certa quantità di risorse per una futura connessione e spedisce il pacchetto SYN/ACK aspettandosi il pacchetto ACK necessario per completare la connessione. Se il terzo e ultimo pacchetto non arriva entro un certo tempo (da qualche decina di secondi a qualche minuto a seconda del sistema) il server libera le risorse allocate in precedenza. Il problema sfruttato da SYN Flood è che i server, anche se molto potenti, esauriscono velocemente le risorse usate per la realizzazione dei collegamenti.
Quindi se in un piccolo lasso di tempo (inferiore al tempo necessario al server per liberare le risorse) vengono ricevute molte richieste parziali di connessioni il server esaurisce le risorse. Ad esempio a sistemi, che sono in grado di gestire migliaia di connessioni contemporaneamente, possono bastare poche decine di richieste parziali di connessioni pendenti per esaurire questo tipo di risorsa. L'idea dell'attacco è spedire una serie di pacchetti SYN in un breve lasso di tempo in modo da consumare tutte le risorse del server. C'è solo un particolare da tenere in considerazione, e cioè che i pacchetti che

l'attaccante spedisce devono avere l'indirizzo sorgente Spoofato con un indirizzo di un host inesistente o spento. Questo si spiega con il fatto che se l'host esiste, non appena riceverà il pacchetto SYN/ACK spedito dal server, risponderà con un pacchetto di reset ed il server libererà le risorse rendendo vani gli sforzi dell'attaccante. Ad un attaccante basterà spedire qualche pacchetto SYN ogni 10 secondi per tenere il server in una situazione di sovraccarico delle risorse. In una situazione di questo tipo infatti il server non riuscirà mai a liberare le risorse in quanto non appena sarà passato il tempo necessario per deallocare un po' di risorse arriverà subito un pacchetto SYN che le riallocherà. Come si può capire questo attacco è molto pericoloso in quanto si può attuare anche con una connessione lenta, inoltre spoofando gli indirizzi IP sorgenti diventa quasi impossibile risalire all'attaccante.

4.3 Internet Services

Di seguito descriveremo alcuni dei principali servizi di Internet, quali Telnet ed FTP, evidenziando alcuni dei loro punti deboli.

4.3.1 Telnet

Telnet fornisce un metodo di comunicazione standard e bidirezionale, attraverso il quale due host possono comunicare.

Il protocollo Telnet, descritto nelle **RFC 854** e **855**, permette a due host di comunicare tra loro. Grazie a questo protocollo è possibile impartire comandi al server remoto come se si fosse localamente collegati a quest'ultimo. Una sessione telnet, utilizza il protocollo **TCP** per la comunicazione e necessita di un **Telnet Server** in ascolto sulla porta **23**.

Le funzioni fornite a due host che comunicano tramite una sessione telnet sono:

- **Network Virtual Terminal (NVT)**;
- **Negoziazione delle opzioni**;
- **Viste simmetriche di terminali e processi**;

Network Virtual Terminal
NVT, come indica il nome, è un ambiente virtuale creato ai due estremi della connessione, dove operano rispettivamente il client ed il server Telnet. Le funzioni disponibili per la connessione sono negoziate tra i due host partendo da un insieme minimo, il quale deve essere necessariamente disponibile. In questo modo è possibile offrire un ambiente identico per i due host che instaurano la connessione in base alle caratteristiche comuni disponibili per

ciascuno e preventivamente negoziate. Grazie a NVT, **non** è quindi necessario **utilizzare software specifici** sia dal lato client che da quello server.

Negoziazione delle opzioni
In quanto non tutti i client ed i server Telnet supportano le medesime funzioni, è stato implementato un meccanismo di negoziazione delle opzioni grazie al quale, una sessione può avvenire utilizzando le caratteristiche comuni ai due host. Le opzioni di base disponibili possono poi essere successivamente ampliate. La negoziazione, che avviene solitamente all'inizio di una connessione, utilizza quattro tipi di richieste:

- **DO <codice opzione>**: Richiede all'NVT ricevente di **implementare** l'opzione richiesta;
- **DON'T <codice opzione>**: Richiede all'NVT ricevente di **cessare** l'utilizzo dell'opzione richiesta;
- **WILL <codice opzione>**: Propone all'NVT ricevente di **implementare** l'opzione indicata;
- **WON'T <codice opzione>**: Propone all'NVT ricevente di **cessare** l'utilizzo dell'opzione indicata;

Viste simmetriche di terminali e processi
La capacità di negoziare le opzioni, essendo simmetrica può dare origine ad un ciclo infinito di richieste qualora uno dei due host interpretasse erroneamente le richieste. Telnet prevede delle funzioni per evitare questo tipo di problema:

- La richiesta di definizione di un'opzione può essere inviata ad un NVT solo se viene richiesto un cambiamento dello stato delle opzioni;
- Se un NVT riceve una richiesta di definizione di un'opzione già definita non deve inviare nessun messaggio di accettazione al fine di evitare la possibile nascita di un loop senza fine;
- E' possibile definire una nuova opzione anche quando il flusso dati della sessione è in corso. L'opzione sarà attiva dalla sua definizione in avanti;

Funzioni di controllo standard
Essendo tra gli obbiettivi del protocollo quello di definire uno metodo **standard** per la comunicazione, indipendentemente dal sistema su cui è in funzione il terminale, sono stati definite alcune funzioni di controllo standard:

- **Interrupt Process (IP)**: sospende un processo in corso su un server Telnet;
- **Break (BRK)**: indica che è stato premuto il tasto break;
- **Abort Output (AO)**: porta a termine il processo in esecuzione senza visualizzare l'output sul terminale utente;

- **Are You There (AYT)**: permette di sapere se l'host remoto è ancora online;
- **Erase Character (EC)**: permette di correggere i caratteri di una stringa di testo prima di inviarla al server;
- **Erase Line (EL)**: permette di cancellare una riga completa dall'input dei comandi;
- **Synchronize (SYNCH)**: permette di riprendere il controllo della sessione in caso di problemi;

Caratteri di Escare

Ogni comando inviato da un client ad un server Telnet è composto da **due** parti: il codice di **Escape** detto anche **IAC** (*Interpret As Command*) ed il codice del **comando**. I codici di Escape hanno lo scopo di distinguere i comandi dai dati durante la sessione Telnet tra due host.

Alcuni dei principali comandi Telnet sono:

close : chiude la sessione corrente;
display : visualizza i parametri operativi della sessione;
open <*host*> <*porta*> : indica a quale server e su quale porta deve avvenire il collegamento;
quit : esce dal client telnet;
send <*codice*> : invia dei caratteri speciali al server;
set <*parametro*> : imposta i parametri operativi;
unset <*parametro*> : elimina l'impostazione di un parametro operativo;
status : visualizza lo stato della connessione;
? : visualizza l'help in linea;

Per capire al meglio come funziona Telnet, consideriamo il seguente esempio nel quale si effettua l'accesso ad una shell di un host in rete.

```
homer@Apollo13:~$ telnet
telnet> open 192.168.0.1
```
Avvio del client e connessione all'host 192.168.0.1
```
Trying 192.168.0.1...
Welcome to Joker Linux Slackware Box!!!
Joker login: homer
Password: du367fh34r
```
L'host risponde quindi passa alla fase di autenticazione...
```
Linux 2.4.18.
Last login: Wed Jul 23 09:35:32 +0200 2003 on pts/0 from
192.168.0.100.
No mail.
homer@Joker:~$
```
Connessione avvenuta con successo. A questo punto è possibile impartire comandi come da locale. (Esiste anche la possibilità di implementare un telnet server con accesso anonimo)

Essendo Telnet un protocollo flessibile, può essere utilizzato per diversi compiti come per esempio: effettuare sessioni **POP3**, **IMAP**, **SMTP**, oppure collegarsi alla console di router.

Telnet è sicuro?
Gli hacker possono interferire con ogni tipo di comunicazione di rete. Ad esempio un hacker può intercettare una sessione Telnet nel seguente modo:

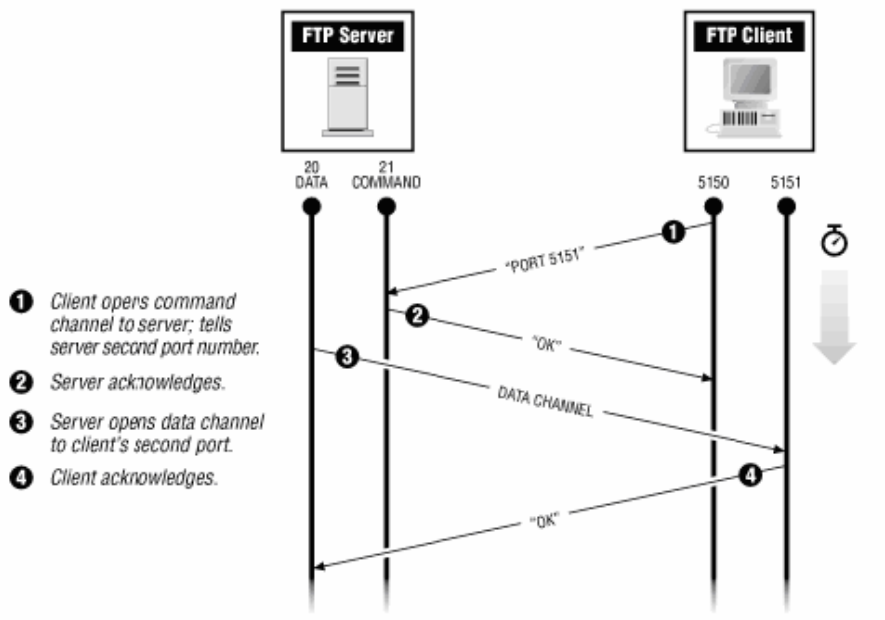
1. Prima di iniziare l'attacco l'hacker osserva passivamente le trasmissioni della sessione senza interferire in alcun modo.
2. Nel momento opportuno l'hacker invia una grande quantità di dati nulli al server contenenti la sequenza estesa di byte “IAC NOP IAC NOP” (NOP = no operation), il demone Telnet del server interpreta questi dati come valore nullo e li rimuove dal canale. Tuttavia la ricezione da parte del server di questi messaggi interrompe la sessione Telnet in corso.
3. Il risultato di questa sessione è una sessione Telnet desincronizzata.
4. Per costringere il client a uno stato di desincronizzazione, l'hacker adotta la stessa tecnica usata per il server.
5. Per controllare la sessione Telnet l'hacker cerca di intercettare il numero di sequenza dei pacchetti in modo da poter continuare la connessione.

Gli hacker possono usare questo metodo solo se la sessione può trasportare dati nulli. Anche se la sessione accetta dati nulli, un hacker avrà difficoltà pratiche (in quanto dovrà mantenere una certa sincronizzazione tra il client valido e il server) a determinare il momento esatto in cui inviare i dati nulli. Se i dati vengono inviati nel momento errato, l'attacco interromperà la sessione Telnet o creerà un 'interferenza senza dare all'hacker la possibilità del controllo della sessione.

4.3.2 Protocollo per il trasferimento di file (FTP)

FTP è un protocollo dello strato di applicazione utilizzato per trasferire file da una macchina ad un altra.
L’FTP anonimo è di gran lunga il più usato in Internet. I server FTP anonimi sono il meccanismo standard per distribuire programmi, informazioni, e altri file che i siti desiderano rendere disponibili in Internet. Se un sito offre un server FTP anonimo, chiunque in Internet può iniziare un collegamento di FTP al sito con un nome di login “anonimo”, e accedere a qualunque file che gli amministratori del server hanno scelto di rendere pubblico in un'area limitata.

L’FTP usa due collegamenti TCP separati: uno per portare comandi e risultati tra il client ed il server (*command channel*), e l'altro per portare alcuni file e l’elenco delle directory trasferiti (*data channel*). Sul server il command channel usa il porto 21, ed il data channel usa il porto 20. Il client usa porti maggiori di 1023 per entrambi i collegamenti.



All'inizio di una sessione FTP il client alloca due porti TCP, con un numero di porto maggiore di 1023 **nota**. Il primo è usato per aprire il collegamento di command channel al server, al tempo stesso riferisce al server il numero del secondo porto che lo stesso client usa per il data channel. A questo punto, è il server ad aprire i data channel. Questo tipo di collegamento è denominato "normale" (*normal-mode*).

La figura precedente mostra un tipo di collegamento FTP normale.

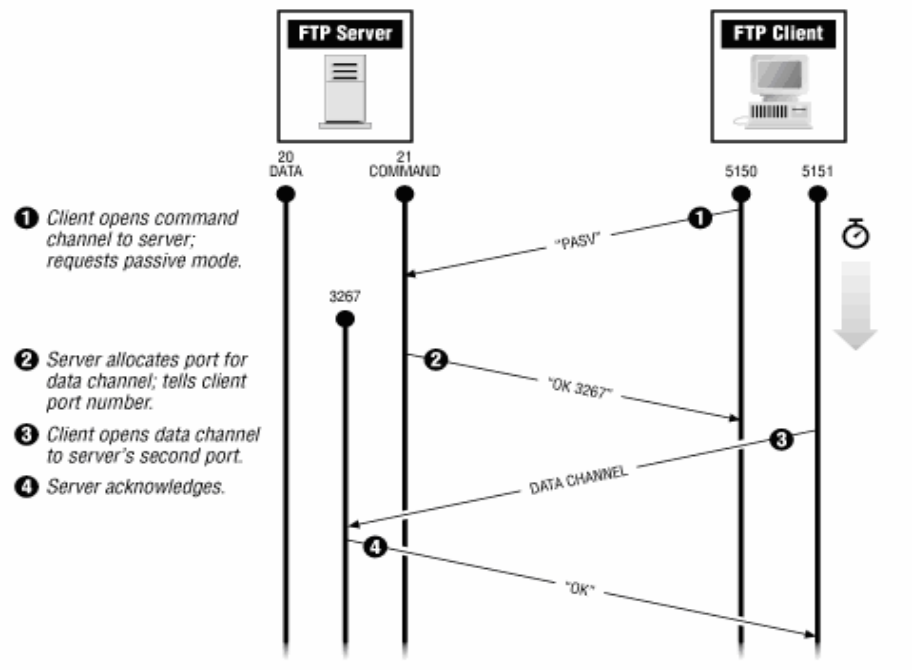
Molti server FTP e molti client FTP supportano un modo alternativo che permette ai client di aprire sia la connessione di "controllo", sia quella per i dati. Questa modalità, denominata "passiva" (*passive mode* o *PASV mode*), è sicuramente più sicura di quella normale in quanto nella passiva è sempre il client FTP ad attivare una connessione, qualunque essa sia, quella di controllo o quella per i dati, e quindi si ha il notevole vantaggio che il client FTP non può ricevere nessuna connessione.

Per una connessione FTP passiva, il client alloca due porti TCP. Usa il primo porto per contattare il server FTP, come nella modalità normale. Invece di pubblicare il comando di porto per dire al server il secondo porto del client, il client pubblica il comando di PASV, per indicare l'uso di una modalità passiva. Questo comporta, per il server, l'allocazione di un secondo porto per il canale dati (per ragioni architettoniche, i server usano porti casuali con numero maggiore di 1023 e non il porto 20 come nel modo normale). Il client, una volta ricevuto il numero del porto del canale dati sul lato server, apre il collegamento di dati dal suo porto verso il server.

---

nota Perché si utilizza un porto più grande di 1023?
Solitamente i porti da 0 a 1023 sono adibiti ad uso locale; questi porti sono normalmente utilizzati dai server e non dai client. Questa convenzione fu introdotta da Unix ed è stata seguita anche dagli altri sistemi operativi, come ad es. Macintosh ed MS-DOS.

La figura seguente mostra un tipo di collegamento FTP passivo.



Se il nostro client FTP (o uno dei server di FTP con cui comunichiamo) non supporta la modalità passiva, e noi vogliamo permettere l'FTP con il packet filtering (piuttosto che via proxy), si dovrà inserire una eccezione nelle regole per permettere al server di aprire il data channel al client. Agendo in questo modo si è vulnerabili ad assalitori che lanciano un collegamento dal proprio porto 20 su un porto client con un numero maggiore di 1023. Perciò, si dovrebbe restringere questa eccezione quanto possibile.
Alcune realizzazioni di packet filtering esaminano i comandi spediti sull'FTP ed osservano i comandi di porto che il client invia al server; in questo modo il server è informato su quale porto il client è in ascolto per aprire il data channel con il server stesso.

Strategie di sicurezza

Siamo arrivati finalmente alla meta del nostro lungo cammino, che ci ha portato a conoscere i firewall. In questo capitolo esamineremo alcune strategie di sicurezza.
Non prenderemo però in considerazione i firewall "application-type" in quanto sono:

- poco performanti soprattutto su reti ad alto traffico;
- usualmente sono implementati tramite software proprietario chiuso;
- sono implementati in hardware;
- necessitano di un software lato client;
- sono costosi.

5.1 La difesa contro il caos di Internet

Anche il firewall Linux, come i più blasonati firewall commerciali, possiede ovviamente la capacità di filtrare i pacchetti in transito e limitare quindi l'accesso dall'esterno ai soli servizi pubblici interni eliminando la possibilità di accesso alle risorse private presenti sulla rete aziendale.
Tale capacità viene messa in pratica tramite l'applicazione di regole applicate all'intero blocco di informazioni presenti nell'header del pacchetto. Il firewall può decidere se accettare o rigettare il pacchetto in base, per esempio, all'indirizzo e/o alla porta del sorgente o del destinatario, in base al tipo di pacchetto (TCP,UDP,ICMP…) e così via.
Queste regole possono essere applicate in diversi momenti del processo di trasferimento del pacchetto dalla rete esterna alla rete interna.
Iptables prevede l'analisi e l'applicazione di regole sui pacchetti in processi che vengono chiamati di PREROUTING, INPUT, FORWARD, OUTPUT e POSTROUTING.

Per capire quando e come agiscano questi processi seguiamo il percorso di un pacchetto IP dalla scheda di rete esterna a quella interna senza la presenza del software di firewall iptables. Dopo essere entrato dalla scheda di rete esterna, il pacchetto viene aperto e ne viene analizzato l'header alla ricerca dell'indirizzo di destinazione. Questo indirizzo viene confrontato con la tabella di routing della macchina e quindi instradato verso una porta locale o verso la scheda di rete appropriata se l'indirizzo di destinazione è differente da quello associato al firewall.

Prima di proseguire, visto che abbiamo tirato in ballo la tabella di routing, vediamo che funzione svolge nel processo di trasporto del pacchetto.

La tabella di routing, viene utilizzata per decidere dove instradare il pacchetto IP in base all'indirizzo di destinazione presente nell'header. Essa contiene un'associazione tra blocchi di indirizzi Internet e risorse con cui tali indirizzi possono essere raggiunti. Le risorse possono essere interfacce di rete locali o indirizzi IP di computer detti "gateway".

Di seguito vi è un esempio di una banale tabella di routing di una machina con una sola interfaccia ethernet. Linux nomina le interfacce di rete ethernet con il suffisso eth, numerandole poi in base alla posizione all'interno degli slot ISA/PCI.

Kernel IP routing table

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
127.0.0.0	*	255.0.0.0	U	0	0	0	lo
192.168.0.0	*	255.255.255.0	U	0	0	0	eth0
0.0.0.0	192.168.0.1	0.0.0.0	UG	0	0	0	eth0

In evidenza:

- l'indirizzo 127.0.0.1, il cosi detto indirizzo di loopback, mappato sull'interfaccia (virtuale) lo;
- gli indirizzi della rete locale da 192.168.0.0 a 192.168.0.255 mappati sulla scheda di rete eth0;
- la rotta di default, l'indirizzo 0.0.0.0/0.0.0.0, l'ultima risorsa nel caso che le altre rotte non vengano applicate al pacchetto e che, di solito, corrisponde all'indirizzo del router/gateway verso INTERNET.

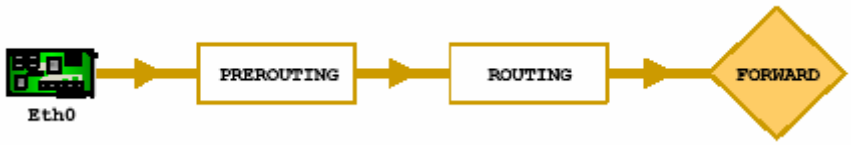
Ritorniamo all'analisi del tragitto percorso dal nostro pacchetto IP e vediamo cosa accade in presenza del firewall iptables.

Il pacchetto entra dall'interfaccia esterna e viene sottoposto, prima del processo di routing, all'applicazione delle direttive presenti nella lista PREROUTING.

Usualmente, in tale processo vengono inserite regole che tendono a evidenziare il pacchetto per distinguerlo dagli altri pacchetti ed eseguire su di esso adeguate operazioni nelle successive fasi del processo di trasporto.
In tale fase vengono anche applicate le regole per la gestione del destination NAT (DNAT) che vedremo nei prossimi paragrafi.

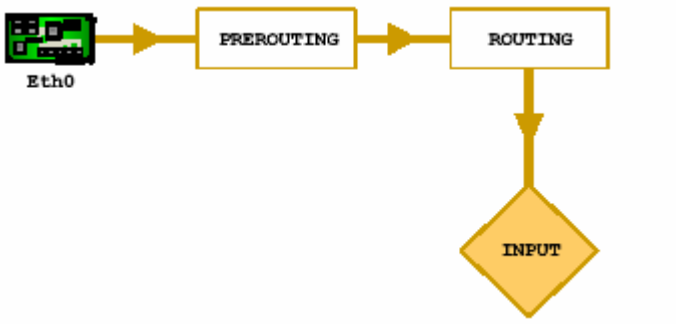
Il pacchetto subisce il processo usuale di routing in base alla tabella presente nella macchina locale.

Se il pacchetto, in base alla tabella di routing, è destinato alla interfaccia di rete interna vengono applicate le regole descritte nella lista di FORWARD.



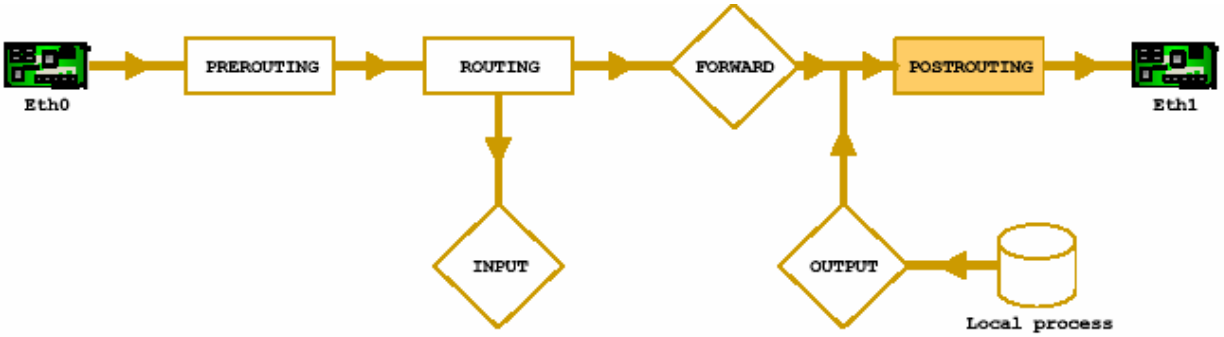
Usualmente sono questi i filtri più importanti in quanto definiscono cosa può passare dall'esterno verso l'interno e cosa no.

Se il pacchetto è destinato, in base alla tabella di routing, alla macchina locale vengono applicate le regole descritte nella lista di INPUT. Questi filtri proteggono il firewall stesso da accessi indesiderati.



Se il pacchetto ha come sorgente la macchina locale, ossia è stato generato da un processo della macchina locale vengono applicate, al pacchetto, le regole di OUTPUT.

Sia nel caso di forward che di output, prima di uscire dalla scheda di rete interna, il pacchetto subisce l'applicazione delle direttive di POSTROUTING. In tale fase vengono di solito applicate le regole per il source NAT (SNAT) che vedremo nei prossimi paragrafi.



In ognuno di questi step ogni direttiva si chiede sostanzialmente: "se l'header del pacchetto verifica certe condizioni, che cosa devo fare del pacchetto" ?

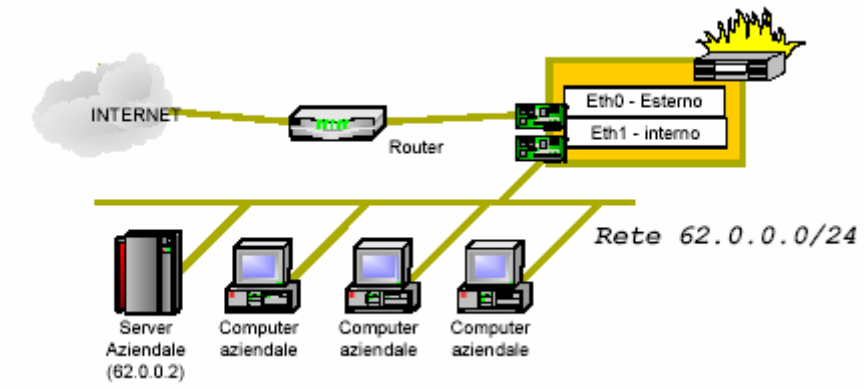
La risposta a questa domanda può essere o di accettare il pacchetto che continua nel suo percorso all'interno delle altre direttive e degli altri step o rigettare il pacchetto che viene definitivamente buttato via.
In ogni step, se il pacchetto non verifica nessuna delle condizioni impostate, può essere definita una regola di default da applicare al pacchetto che verrà quindi accettato o rigettato.

Usualmente si ritengono sicuri ed accettabili i pacchetti provenienti dall'interno e destinati verso l'esterno e quindi, in particolare, i pacchetti che hanno come sorgente il firewall saranno permessi e quindi l'impostazione ovvia di default del processo di OUTPUT sarà quella di ACCEPT.
Quello che si vuole invece evitare, a meno di eccezioni, e che dall'esterno si possa impunemente accedere verso l'interno. Ecco perché è usuale impostare a DROP la configurazione di default dei processi di FORWARD e di INPUT**.**
Sarebbe un errore lasciare ad ACCEPT queste regole e tentare di chiudere tutti i servizi interni. Il firewall serve, in particolare, per chiudere tutto, anche quello che neanche si sa di avere aperto. Negando tutto e accettando solo ciò che si considera come corretto si è sicuri che se dall'esterno di accede ad una data risorsa interna è perché siamo stati noi a richiederlo.
Il primo banale script di configurazione del nostro firewall sarà quindi (il flag -P imposta per l'appunto la politica di default per il processo):

```
-P INPUT DROP
-P FORWARD DROP
-P OUTPUT ACCEPT
```

5.2 Strategie di difesa

Impariamo a difenderci considerando, come semplice esempio, una piccola rete connessa ad Internet composta da un piccolo gruppo di computer client con installato il sistema operativo Windows® 2000 Professional e da un server aziendale con indirizzo 62.0.0.2 su cui vi è installato il sistema operativo Windows® 2000 Server con il server web Microsoft Internet Information Server.



Visto l'esiguo numero di macchine presenti sulla rete è stato possibile assegnare ad ogni macchina un indirizzo IP assegnato dal provider.
Il nostro scopo, come amministratori della rete, è quello di proteggerla da eventuali accessi non autorizzati provenienti dall'esterno permettendo l'accesso al solo server web.
A tale scopo separiamo la nostra rete da Internet inserendo un firewall Linux subito a valle del router e tentiamo di creare una configurazione tale che ci permetta di raggiungere lo scopo che ci siamo sopra preposti.

La cosa più evidente da fare è quella di bloccare l'accesso al nostro server permettendo l'accesso dall'esterno alla sua porta web.
Potremmo essere tentati di attivare una configurazione del tipo

```
-A FORWARD –p tcp –d 60.0.0.2 --dport ! web –j DROP
```
nota

ed effettivamente il nostro server sarebbe sicuro in quanto tutti i pacchetti ad esso destinati non diretti alla porta web verrebbero rigettati dalla regola sopra.

Quello che l'amministratore della rete ha dimenticato è che vi sono servizi aperti anche su tutti gli altri computer e che cosi facendo abbiamo lasciato ad un hacker di passaggio la possibilità di utilizzare tali porte per un eventuale attacco alla nostra rete.
Questo è dovuto al fatto di non aver specificato le politiche di default per i processi di INPUT, OUTPUT e FORWARD che, se non specificati esplicitamente, sono impostati ad ACCEPT. I pacchetti destinati a tutti gli altri pc della nostra rete vengono quindi implicitamente accettati e dunque tutti i servizi attivati sui vari pc interni sono accessibili dall'esterno ed eventualmente attaccabili.
Non si creda infatti che i computer Windows® 2000 Professional siano sicuri grazie al fatto di non essere dei server.
A parte le porte del protocollo NETBIOS che il sistema operativo attiva per la gestione della rete Microsoft vi possono essere altre porte aperte all'insaputa dell'amministratore di rete e, a volte, anche dello stesso utente. Per esempio, la persona che fa sviluppo ASP per il server aziendale potrebbe aver installato, sul proprio computer, un server IIS per velocizzare le fasi di sviluppo. Visto che il server IIS, a meno di non richiederlo esplicitamente, installa anche un server FTP e un server SMTP ecco che, all'insaputa dell'amministratore, c'è in rete un altro server web, un server ftp e un server di posta aperto al relay che può essere usato, per esempio, per azioni di spamming.
Si comprende quindi che il chiudere il solo server aziendale non evita alla rete di essere sottoposta ad attacchi esterni.
La filosofia corretta è, come detto in precedenza, di impostare a DROP i processi di FORWARD e di INPUT e di impostare la sola regola di accesso al server web come

```
-A FORWARD –p tcp –d 60.0.0.2 --dport web –j ACCEPT
```

In questo modo tutte le nostre macchine sono protette.

Il problema è che ora sono troppo protette in quanto gli utenti della nostra rete non possono più navigare e anche il server web è impossibilitato a rispondere alle, ora legittime, richieste dall'esterno. I pacchetti provenienti dalla rete interna e destinati all'esterno vengono infatti rigettati dalla politica di default del processo di FORWARD.

---

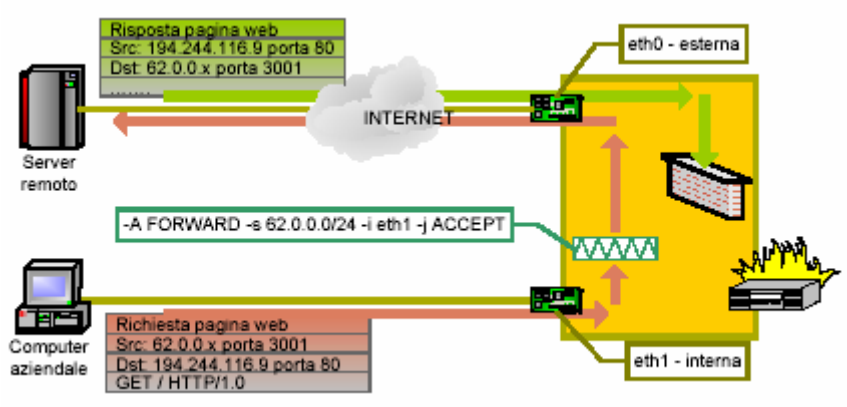
nota La regola esprime che nel processo di forward del pacchetto (-A FORWARD) i pacchetti di tipo tcp (-p tcp) [byte 10 dell'header IP] destinati al nostro server (-d 60.0.0.2) [byte 17-20 dell'header IP] e che tentano di collegarsi sul nostro server ad una porta diversa dalla porta 80 (--dport ! web) [byte 3-4 dell'header TCP] vengano rigettati (-j DROP).

Per risolvere questo, considerando sicure tutte le connessioni che hanno come sorgente la nostra rete, dobbiamo intervenire sulla configurazione del firewall ed accettare tali pacchetti. Una corretta impostazione potrebbe essere

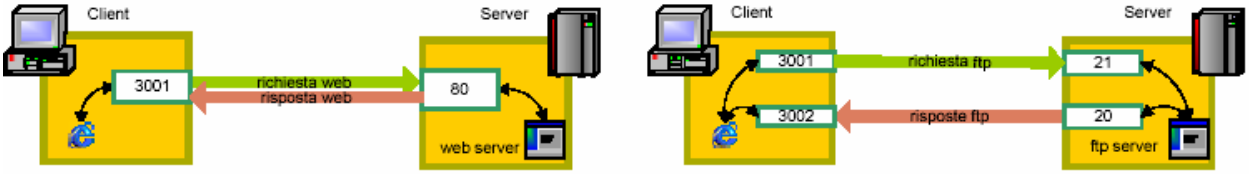
```
-A FORWARD -s 62.0.0.0/24 -i eth1 -j ACCEPT
```

dove l'indicazione dell'interfaccia di provenienza del pacchetto è stata esplicitata per evitare tecniche di spoofing dall'esterno.

Ora le cose cominciano ad andare un po' meglio. Siamo protetti ed il server web risponde alle richieste dall'esterno. I nostri utenti interni però non riescono ancora a navigare. I loro pacchetti di richiesta di servizi esterni filtrano correttamente verso l'esterno grazie alla regola appena inserita ma le risposte a tali richieste non riescono a rientrare in quanto vanno a cadere nella regola di DROP del processo di FORWARD.



Quello che dobbiamo garantire è che pacchetti provenienti da macchine localizzate in "territorio nemico" possano giungere a noi solo in risposta ad una nostra richiesta e non in generale. Il problema risiede però nel fatto che l'analisi del solo pacchetto in ingresso non permette di evincere se esso sia relativo o meno ad una nostra richiesta.
A causa di ciò, quasi tutti i firewall commerciali e tutti i firewall Linux ad eccezione di iptables non riescono a risolvere questo dilemma e si devono accontentare di "intuire" questa cosa.
Il metodo usuale con cui si tenta di individuare i pacchetti di risposta è basato sul modo con cui gli applicativi comunicano tra di loro.
Vi sono due modi con cui un client che si collega ad un server remoto attiva una comunicazione bidirezionale. Nel primo caso il client, che parla da una porta locale si collega sulla porta standard remota del servizi e su questo canale, aperto dal client viaggiano i pacchetti in ambedue le direzioni. Nel secondo caso il client comunica al server di aspettare una risposta dal server su una determinata porta locale. Il server si collega a questa nuova porta locale e i dati viaggiano in un senso sulla prima connessione e nell'altro sull'altra connessione.



In ambedue i casi comunque le porte locali sono comprese tra la porta 1024 e la 65535. Tale intervallo di porte prende il nome di porte non privilegiate dal fatto che esse possono essere aperte da software che si mettono in ascolto su di esse anche se il processo su cui girano non è controllato dall'utente amministrativo (root).

Al contrario le porte dalla 1 alla 1023 sono dette porte privilegiate e solo i processi che girano sotto l'utente di root possono mettersi in ascolto su tali porte.
Ciò che usualmente fanno i firewall per permettere ai server remoti di rispondere alle richieste dei client locali è di permettere il passaggio di tutti i pacchetti destinati a macchine interne sulle porte non privilegiate.
Tale soluzione risolve il problema ma non è esente da problemi. Un programma "maligno" potrebbe aprire una di queste porte per permettere l'accesso ad un hacker (la cosi detta backdoor) che, grazie a questa nuova regola di firewall, potrebbe dall'esterno entrare nella macchina. È si vero che il processo backdoor non gira sotto l'utente di root ma intanto un hacker è entrato e uno bravo potrebbe poi risalire i livelli del sistema e giungere al livello di root diventando quindi il padrone della macchina.
Inoltre molti dei sistemi operativi della famiglia Windows® non hanno il concetto di utenti e quindi delle backdoor installate su porte non privilegiate permetterebbero immediatamente all'hacker di prendere il completo controllo del sistema.
Ricordiamo inoltre che alcuni server si mettono in ascolto su porte non privilegiate. Un esempio tipico è il server proxy SQUID che si installa di solito sulla porta 3128 o 8080. Tale server sarebbe raggiungibile dall'esterno anche se il servizio proxy dovesse essere un servizio privato ad esclusivo uso interno.
Se è quindi vero che l'apertura delle porte non privilegiate è una soluzione al problema è anche vero che essa risulta una apertura indiscriminata e quindi potrebbe permettere accessi non desiderati dall'esterno.

Come risolve invece egregiamente il problema delle risposte a richieste locali il firewall iptables?

Dato che abbiamo già detto che è impossibile risalire alla correlazione tra richiesta e risposta analizzando il solo pacchetto in ingresso, iptables risolve la cosa tenendo traccia di tutte le richieste interne e verificando che i pacchetti di ritorno siano effettivamente associati ad una di queste richieste. Se il pacchetto verifica tale condizione viene inviato al client altrimenti il pacchetto viene rigettato.
Questa funzione è possibile grazie al modulo ip_conntrack. Tramite questo modulo viene aggiunta un'ulteriore analisi al pacchetto in transito che prescinde dall'analisi dell'header TCP/IP e che confronta il pacchetto con tutti gli altri pacchetti transitati in precedenza nel firewall. Quest'analisi ritorna un cosi detto "stato" del pacchetto relativo agli altri pacchetti. Lo stato di un pacchetto inteso all'interno dell'analisi del modulo ip_conntrack può assumere i seguenti valori:

- **NEW** → il pacchetto non è correlato a nessun altro pacchetto transitato in precedenza ed è quindi teso alla creazione di una nuova connessione
- **ESTABLISHED** → il pacchetto appartiene ad una connessione già esistente ossia è un pacchetto di risposta relativo ad richiesta di dati sulla

connessione esistente (caso richiesta pagina web)

- **RELATED** → un pacchetto che è correlato ma non appartiene ad una connessione esistente (caso risposta ftp)
- **INVALID** → non è stato possibile ricavare lo stato del pacchetto.

Avendo aggiunto questa nuova informazione all'analisi del pacchetto è ora possibile scrivere una regola che permetta l'ingresso nella nostra rete di pacchetti provenienti dalla "zona nemica" solo se correlati a richieste di host interni. La regola, da aggiungere alla configurazione del firewall risulta essere:

```
-A FORWARD –d 62.0.0.0/24 –m state --state ESTABLISHED,RELATED –j ACCEPT
```

In questo modo siamo riusciti a proteggere egregiamente la nostra rete.

Rileggiamo e commentiamo a parole cosa fa la nostra configurazione attuale:

```
-P INPUT DROP
-P FORWARD DROP
-P OUTPUT ACCEPT
-A FORWARD –s 62.0.0.0/24 –i eth1 –j ACCEPT
-A FORWARD –d 62.0.0.0/24 –m state --state ESTABLISHED,RELATED –j ACCEPT
-A FORWARD –p tcp –d 60.0.0.2 --dport web –j ACCEPT
```

ossia tutti i pacchetti in transito sul nostro firewall vengono rigettati ad eccezione dei pacchetti in partenza dalla nostra rete interna e dei pacchetti provenienti dall'esterno che o sono correlati a richieste provenienti dall'interno o sono pacchetti destinati al nostro server web.

Abbiamo quindi raggiunto lo scopo prepostoci.

5.3 Source Nat e Masquerade: navigare senza indirizzi Internet

Uno degli aspetti sottovalutati nei giorni in cui Internet si chiamava ancora Arpanet è stato quello relativo alle dimensioni dello spazio degli indirizzi IP e della loro assegnazione.
Nella visione di chi ha creato la Rete si riteneva che, per come erano stati definiti gli indirizzi IP, il loro numero fosse talmente grande da essere praticamente inesauribile.
L'analisi era talmente ottimistica che vennero assegnati, alle prime istituzioni che ne fecero richiesta, blocchi di indirizzi ampiamente sovradimensionati rispetto alle loro esigenze del tempo e anche future.
Nessuno avrebbe potuto neanche lontanamente immaginare l'enorme espansione delle Rete e l'enorme numero di computer che avrebbero richiesto di farne parte.
Oggi gli indirizzi IP liberi sono una risorsa molto importante e l'iter necessario per la loro assegnazione è complesso e prevede la richiesta di una descrizione analitica e particolareggiata del loro utilizzo da parte del richiedente.
È per questo motivo che i provider assegnano ai clienti finali un numero minimo di

indirizzi IP e fanno molto pesare, in termini di costi, l'assegnazione di indirizzi IP aggiuntivi.
Non è quindi rara la situazione di un cliente che si vede assegnare dal provider un numero di indirizzi IP minore del numero di macchine presenti sulla sua rete e che quindi si trova nella necessità di limitare la navigazione a pochi computer.
Un problema analogo lo ha chi si collega ad Internet tramite un modem e un abbonamento ad un Internet Service Provider (ISP). Il provider, in questo caso, assegna, per la durata del collegamento, un indirizzo IP al computer che diviene a tutti gli effetti una macchina della Internet.
Ma se questo computer è collegato anche ad una rete locale a cui accedono altri computer l'utente potrebbe voler condividere la connessione ottenuta con gli altri in modo che tutti possano accedere ad Internet usando quell'unico accesso. Ovviamente solo il computer con il modem può navigare in quanto è l'unico ad avere un indirizzo IP reale.
La risposta a queste richieste è il cosi detto masquerade che è un sottoinsieme di un metodo più generale di modifica del pacchetto IP chiamato source NAT o SNAT.

Che cos'è l'operazione di SNAT e come può aiutare a risolvere il problema del ridotto numero di indirizzi IP assegnati dal provider?

Tecnicamente l'SNAT modifica, nell'header dei pacchetti, l'indirizzo IP del sorgente facendo credere al destinatario del pacchetto che esso provenga da un altro indirizzo IP. Questo permette anche a chi non è fisicamente in Internet di navigare per la Rete.

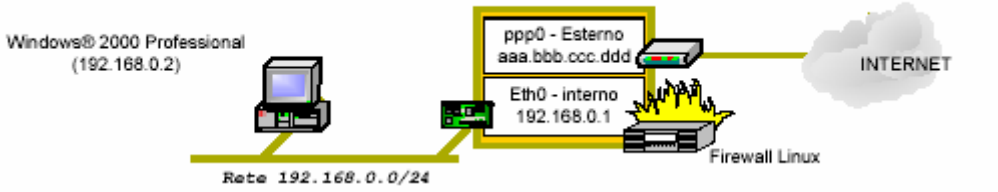
Spieghiamo con un esempio questo, a prima vista, controsenso.

Consideriamo due computer, uno con Windows® 2000 Professional e uno con Linux, collegati alla stessa rete locale.
Ai due computer sono assegnati due indirizzi IP intranet **nota** del blocco 192.168.0.0/24 e, in particolare, la macchina Linux ha un indirizzo IP assegnato alla sua scheda di rete eth0 192.168.0.1 mentre la macchina Windows® ha l'indirizzo 192.168.0.2 .
Il protocollo TCP/IP è indipendente dal sistema operativo e quindi, tramite esso, le due macchine "si vedono" sulla rete locale.
Sulla macchina Linux vi è anche un modem e vi è configurato un abbonamento tramite un ISP. Durante il periodo di collegamento con l'ISP, alla macchina viene assegnato un indirizzo Internet aaa.bbb.ccc.ddd . La macchina Linux ha quindi, per la durata del collegamento, due indirizzi IP assegnati, uno interno, con cui vede la macchina Windows®, e uno esterno, sulla connessione modem, tramite il quale risulta visibile e vede là Internet.

Ci chiediamo come si può ottenere di far navigare anche la macchina Windows® tramite il collegamento attivato dalla macchina Linux.

---

nota Gli indirizzi IP devono essere compresi tra 0.0.0.0 e 255.255.255.255 (in esadecimale 0.0.0.0 e ff.ff.ff.ff). Alcuni indirizzi IP sono però stati riservati alla costituzione di reti private e non sono utilizzabili su Internet. Gli indirizzi da 10.0.0.0 a 10.255.255.255, quelli tra 172.16.0.0 e 172.31.255.255.255 e quelli tra 192.168.0.0 e 192.168.255.255 sono chiamati indirizzi intranet e non può esistere alcuna macchina di Internet ad avere un indirizzo appartenente ad uno di questi intervalli.



Il primo step consiste nell'informare il modulo TCP/IP della macchina Windows® che esiste, sulla rete locale, una macchina che sa come inviare pacchetti ad Internet, ossia consiste nel configurare il gateway di default della macchina Windows® impostandolo con l'indirizzo IP della macchina Linux.

Ma attenzione, la macchina Linux ha, ora che è connessa, due indirizzi IP. Quale dei due va impostato sulla macchina Windows®?

Ovviamente l'indirizzo locale 192.168.0.1 in quanto l'altro è già un indirizzo Internet e la macchina Windows® 2000 non sa come raggiungere la Rete.
Con questa configurazione una richiesta della macchina Windows® destinata a Internet giunge sulla macchina Linux che instrada il pacchetto sulla sua rotta di default che, da quando è attiva la connessione modem, risulta settata proprio su tale linea.
Sembra che già tutto funzioni ma in realtà non funziona niente. L'unica macchina che il provider ha autorizzato a navigare è la macchina Linux. Quando le apparecchiature dell'ISP vedono arrivare dei pacchetti dalla macchina Windows® con indirizzo di sorgente 192.168.0.2 rigettano tali pacchetti in quanto, dal loro punto di vista, sono pacchetti anomali perché lungo tale connessione telefonica gli unici pacchetti che dovrebbero arrivare devono avere indirizzo di sorgente aaa.bbb.ccc.ddd.

Cosa fa allora il SNAT se attivato sulla macchina Linux?

Quando arriva il pacchetto dalla macchina Windows® l'iptables si segna l'header che dovrebbe assumere il pacchetto di risposta ad esso associato e sostituisce l'indirizzo di sorgente 192.168.0.2 del pacchetto in transito con il suo indirizzo Internet aaa.bbb.ccc.ddd instradandolo lungo la linea modem.
Ora le apparecchiature del provider non hanno più nessun motivo per rigettare il pacchetto in quanto, dal loro punto di vista, esso proviene dal soggetto legittimamente autorizzato a navigare lungo quella connessione.
La cosa non finisce però qui in quanto il server remoto invia la risposta alla macchina Linux che si vede arrivare una risposta per una richiesta che non ha fatto. Qui interviene di nuovo il SNAT che si accorge, controllando l'header del pacchetto con quello salvato sopra, che questo flusso di dati remoti va dirottato alla macchina Windows®.
Sostituisce allora all'indirizzo del destinatario, attualmente settato sull'header del pacchetto di risposta, l'indirizzo IP della macchina Windows® e glielo invia. La macchina Windows® si vede arrivare il pacchetto di risposta che si aspettava ed è felice.
La macchina Windows® naviga quindi per "interposta persona" in quanto è la macchina Linux che la fa navigare mascherando di volta in volta l'indirizzo sorgente della macchina locale.

Vediamo ora tecnicamente come si configurano le cose sulla macchina Linux con iptables.

La configurazione dell'SNAT è molto semplice e consiste nell'attivazione della tabella di NAT nel processo di POSTROUTING tramite l'impostazione:

```
-t nat –A POSTROUTING –o ppp0 –j MASQUERADE
```

deve ppp0 è l'interfaccia modem di uscita verso Internet.
Come si vede non appare alcun indirizzo IP nella configurazione in quanto il sistema prevede automaticamente a ricavare l'indirizzo dinamico assegnato di volta in volta dal provider e di utilizzarlo per lo SNAT.
Se la situazione è quella di un indirizzo statico assegnato come nel caso di un collegamento xDSL con pochi indirizzi IP è meglio utilizzare questa configurazione

```
-t nat –A POSTROUTING –o eth0 –j SNAT --to xxx.xxx.xxx.xxx
```

dove eth0 è l'interfaccia di rete lato Internet e xxx.xxx.xxx.xxx è l'indirizzo Internet di questa interfaccia.

Un'ultima domanda: perché questa regola va applicata nel processo di POSTROUTING?

La spiegazione è semplice; per far si che, almeno fino all'ultimo momento, prima di uscire dall'interfaccia verso Internet il pacchetto sia ancora identificato come proveniente dalla macchina interna e su cui poter applicare regole di filtro o di routing. Se questo mascheramento avvenisse prima, il pacchetto sembrerebbe del tutto indistinguibile da un pacchetto generato localmente dal box Linux e non potrebbero quindi essere applicarti eventuali filtri personalizzati.

5.4 Destination Nat: il server che c'è ma non c'è

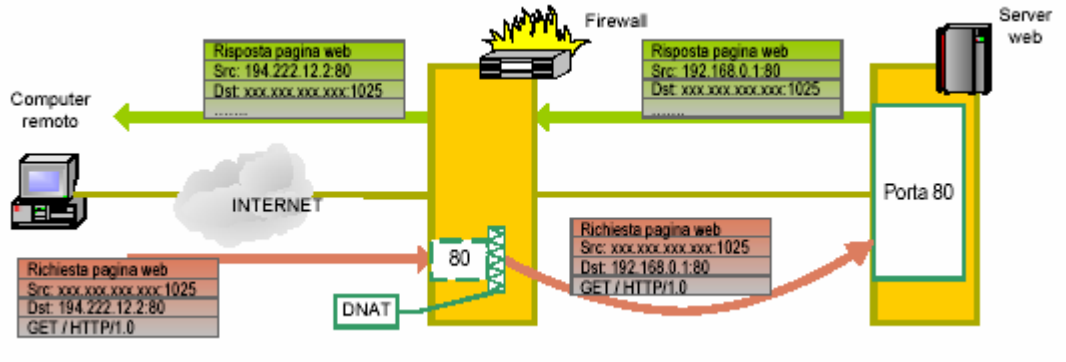
Analogamente al source NAT anche il destination network address traslation agisce modificando l'header dei pacchetti in transito ed in particolare l'indirizzo e la porta di destinazione.
Al contrario dell'SNAT, ma per lo stesso motivo, il DNAT viene applicato nel processo di PREROUTING cosi che tutti i restanti processi di iptables e di routing agiscono sul pacchetto già modificato.
L'operazione di DNAT può essere utilizzata per operazioni di "port forwarding" o di "trasparent proxy". Tralasciando questo secondo caso che tratteremo nel prossimo paragrafo, occupiamoci ora dell'operazione di "port forwarding", di capire cos'è e di come viene eseguita utilizzando il destination NAT.
Il "port forwarding" è quell'operazione grazie alla quale una porta presente su di un server riesce a "puntare" ad un servizio presente su di un'altra porta molto spesso attiva su di un altro server. Tutte le richieste di connessione sulla prima porta vengono inoltrate al servizio reale attivo sulla seconda porta.
Un utilizzo del "port forwarding" si ha nella situazione in cui uno o più servizi presenti in una rete con indirizzi intranet debbono essere esportati e visibili anche

sulla rete Internet.
Portiamo quindi ad esempio una azienda che abbia acquistato una linea xDSL da un fornitore di connettività Internet che le ha assegnato quattro indirizzi IP nell'intervallo da 194.244.12.0 a 194.244.12.3 .
Dei quattro indirizzi, lo zero è l' indirizzo della rete, l'uno è l'indirizzo del router che collega l'azienda al provider e il tre è l'indirizzo di broadcast. L'unico indirizzo libero risulta quindi l'194.244.12.2 e, visto che l'azienda in realtà possiede molti computer che desiderano navigare sulla Rete, ha utilizzato l'SNAT attivandolo su di una macchina Linux. L'indirizzo libero è stato assegnato all'interfaccia di rete esterna mentre a quella interna è stata data un indirizzo intranet della classe 192.168.0.0/24 .
L'azienda decide quindi di trasferire il proprio sito, attualmente in housing presso un provider, sulla rete aziendale per averne una migliore gestione e manutenzione.
Attivare il server web sul firewall potrebbe essere impossibile per incompatibilità di sistema operativo ma, anche nel caso in cui non esistesse questa incompatibilità, l'installazione di ulteriori servizi sul firewall è totalmente inaccettabile. Il firewall, come unico baluardo tra noi e "il nemico" deve essere totalmente sicuro e ogni servizio aggiuntivo tenderebbe a ridurre questo fattore di sicurezza.
Il server viene quindi agganciato alla rete interna e gli viene assegnato l'indirizzo intranet 192.168.0.1 .

Come si rende visibile il server ad Internet visto che ora ha un indirizzo intranet irraggiungibile dalla Rete?

Se è vero che l'installazione del servizio web sul firewall era inaccettabile è però vero che tentare di mappare la porta web del firewall con la porta web del server 192.168.0.1 non preclude assolutamente la sicurezza della rete.
L'idea è proprio questa: esternamente il server web verrà visto attivo sulla porta 80 (web) dell'indirizzo 194.222.12.2 ma tutti i pacchetti diretti su questa porta verranno poi dirottati dal firewall sulla porta 80 del server 192.168.0.1 .
Come segugi, seguiamo il pacchetto TCP/IP di richiesta di una pagina del nostro sito e il relativo pacchetto di risposta. Il pacchetto di richiesta diretto verso la porta 80 del server 194.222.12.2 entra dalla scheda esterna eth0 del firewall.
Nel processo di PREROUTING il firewall verifica che questo pacchetto corrisponde ad una determinata regola e ne modifica il destinatario (DNAT) sostituendo l'indirizzo di destinazione con il 192.168.0.2 .
La tabella di routing vede ora un pacchetto destinato a questo host interno e lo instrada sulla scheda eth1 permettendo al pacchetto di giungere sul server web realmente presente sulla rete.
Il server web interno processa la richiesta e prepara un pacchetto di risposta.
Questo pacchetto torna quindi sul firewall che riconosce questo come pacchetto di risposta relativo al pacchetto che poco prima lui aveva modificato e modifica questo pacchetto sostituendo all'indirizzo di sorgente intranet il proprio indirizzo e lo inoltra all'host remoto.

Si riporta di seguito il risultato dal punto di vista dell'host remoto.



La configurazione dell'iptables per ottenere questo risultato è molto semplice

```
-t nat -A PREROUTING -p tcp -d 194.222.12.2 --dport 80 -j DNAT
  --to 192.168.0.1
```

Come si vede, non è stato necessario specificare la porta di destinazione in quanto la mappatura delle porte è la stessa. Se internamente il server web era attivato su una porta non standard e quindi differente dalla porta 80 si sarebbe dovuto aggiungere questa informazione alla riga di configurazione del DNAT.
Ovviamente quello che funziona per un servizio funziona anche con molti. Si possono cosi avere più server interni e mapparli esternamente utilizzando il solo indirizzo IP del firewall. Se però si cerca di esportare due server dello stesso tipo, per esempio due server web attivati su due macchine differenti, solo uno dei due potrà essere visto esternamente presente su firewall sulla porta web standard. L'altro dovrà essere mappato su una porta xx non standard e sarà visibile esternamente solo tramite l'url http://www.miosito.it:xx/

5.5 Trasparent proxy: come dirottare i pacchetti TCP/IP

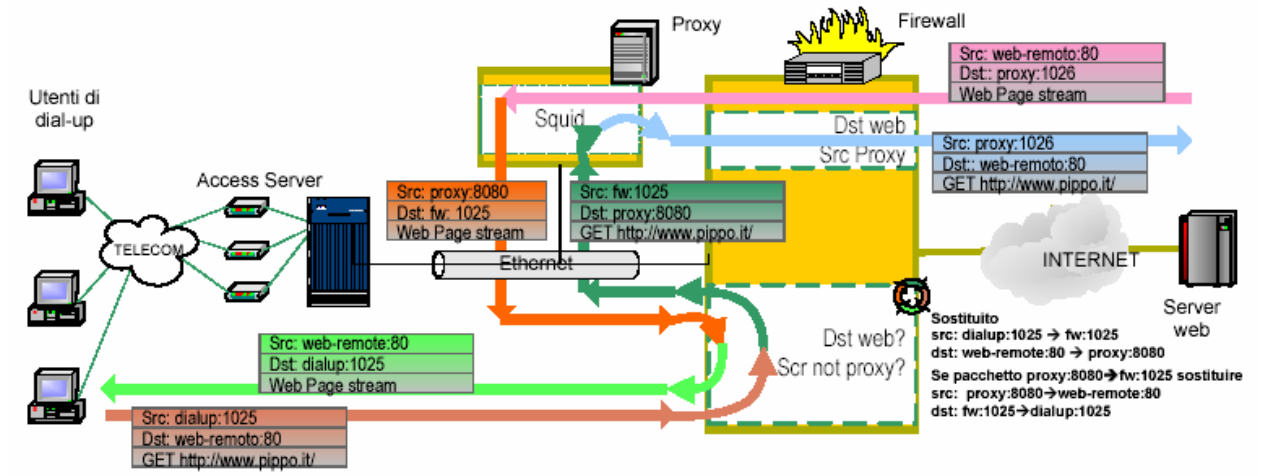
Una forma specializzata di DNAT è la così detta redirezione dei pacchetti che permette, in modo trasparente, di dirottare alcune tipologie di dati su di una singola macchina per motivi di monitoraggio o di trasparent proxy.
Analizziamo il perché si debba e si voglia eseguire tale operazione considerando il trasparent proxy di tipo web applicato ad un ISP che offre collegamenti in dial-up ad una utenza privata e/o commerciale. L'ISP ha, presso il suo data center, un proxy server che, se utilizzato dagli utenti, intercetta le richieste web facendosi carico della ricezione delle pagine e dell'invio all'utente con il vantaggio di salvarsi in locale le pagine ricevute. Se una successiva richiesta fa riferimento a una pagina già presente sull'hard disk locale viene inviata questa all'utente con ovvi risparmi in termini di banda uscente del provider e in termini di risposta all'utente visto che la pagina arriva da una locazione "più vicina".

Il tallone di Achille di questo sistema sta nella frase "se utilizzato dall'utente". Infatti l'utilizzo o meno del proxy è a discrezione dell'utente che, sul browser del proprio computer, può attivare o meno l'utilizzo del proxy dell'ISP.

Ovviamente il vantaggio massimo con il proxy si ha se tutti lo utilizzano in modo massimizzando così la probabilità che una pagina di richiesta venga trovata in locale e non debba essere recuperata dal sito remoto.

Il trasparent proxy è quell'operazione di inviare in ogni caso tutte le richieste web al server proxy indipendentemente dal fatto che l'utente abbia o meno attivato l'utilizzo del proxy sul suo computer.

Come funziona a livello di flusso dati la cosa?



Il pacchetto di richiesta di una pagina web remota da parte di un utente locale attraversa il firewall che, applicando una determinata regola di DNAT, sostituisce all'indirizzo di destinazione quello del proxy. Il pacchetto viene quindi instradato dalla tabella di routing sulla stessa interfaccia da cui era entrato in quanto il server proxy si trova, rispetto al firewall, dalla stesso lato dell'utente, il lato protetto.
Il proxy prende in consegna la richiesta web e recupera la pagina in locale o da remoto in base alla disponibilità della stessa sull'hard disk. Una volta ottenuta la pagina di richiesta invia la risposta all'utente prendendo questa informazione dall'header del pacchetto.
Aggiungiamo un'osservazione, prima di mostrare la riga di configurazione dell'iptables che esegue l'operazione di DNAT per il trasparent proxy: il software proxy va istruito del fatto che sta per essere usato come trasparent proxy e che il pacchetto di richiesta che gli arriva non è un pacchetto di richiesta proxy ma è direttamente un pacchetto http e quindi deve modificare il protocollo di interpretazione del pacchetto ricevente per capire qual è l'informazione remota da cercare. Il server proxy SQUID ha per esempio questa funzionalità di utilizzare anche il protocollo http impostandosi come server in trasparent proxy.
Concludiamo questa discussione mostrando come appare la riga di configurazione del trasparent proxy su iptables:

```
-t nat -A PREROUTING -I eth1 -p tcp --dport 80
  -s ! xxx.xxx.xxx.xxx -j REDIRECT --to xxx.xxx.xxx.xxx:8080
```

dove xxx.xxx.xxx.xxx è l'indirizzo del server su cui è attivo il server proxy collegato alla porta 8080.

5.6 IPTables

IPTables è incluso nei kernel Linux dalla serie 2.4. La particolarità di iptables è quella di poter manipolare i pacchetti in diversi "punti" della nostra macchina. Infatti ci sono 3 tabelle: FILTER, NAT e MANGLE.

Filter è composta da tre catene predefinite:

- INPUT
- OUTPUT
- FORWARD

le quali controllano i pacchetti che sono diretti all'userspace (i programmi), escono o transitano dall'userspace.

Nat è utilizzata per i pacchetti che stanno per creare una nuova connessione (i pacchetti SYN) anche qui ci sono tre catene:

- PREROUTING
- OUTPUT
- POSTROUTING

Prerouting serve a stabilire le regole sui pacchetti che giungono prima della "routing decision" (l'analisi del pacchetto e la determinazione se è destinato alla macchina stessa o è da forwardare ad un'altra macchina), Output si riferisce ai pacchetti che sono in uscita e sono stati generati in locale, Postrouting regola i pacchetti che stanno per lasciare la macchina.

Mangle serve a manipolare i pacchetti in entrata o in uscita infatti ha due catene:

- PREROUTING
- OUTPUT

la prima modifica i pacchetti in entrata prima della decisione del routing, la seconda interviene sui pacchetti in uscita dalla macchina prima del routing decision. Nelle singole catene occorre stabilire delle regole tramite le quali il firewall decide la sorte del pacchetto. Le regole di una catena hanno un target, ovvero un'istruzione che stabilisce cosa fare del pacchetto. Esistono cinque target fondamentali:

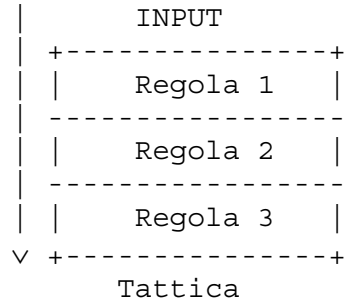
- ACCEPT
- DROP
- QUEQE
- RETURN
- REJECT

Il primo lascia passare il pacchetto, drop scarta il pacchetto senza dare avvisi al mittente, queqe accoda il pacchetto verso l'userspace, return fa analizzare il pacchetto alla catena chiamante (se return è chiamato da una catena predefinita, la sorte del

pacchetto è determinato dalla policy della catena chiamante), reject rifiuta il pacchetto. E' importante dire che come target si può utilizzare una catena definita dall'utente.

Precedentemente è stato accennato alla policy di una catena, per chiarire il concetto occorre fare una piccola premessa sul funzionamento di un firewall.
Il filtraggio sotto Linux 2.4 si basa come nelle versioni precedenti sui concetti di catene (chain), obiettivi (target) e tattiche (policy). Ciascuna catena consiste di una tattica (policy), e di una o più regole che stabiliscono quali pacchetti sono accettati e quali no. Una regola si compone in genere di una serie di caratteristiche che servono ad individuare un pacchetto e da un obiettivo (target) che indica cosa fare del pacchetto se la regola è soddisfatta. L'obiettivo corrisponde ai target sopra descritti.

Una catena si può rappresentare in questo modo (qui prendiamo ad esempio la catena INPUT):



I pacchetti che giungono dalla rete e che sono destinati alla catena INPUT verranno filtrati dalle regole della catena. Se la regola 1 non è soddisfatta si passa alla regola 2, se non è soddisfatta nemmeno la 2 si passa alla successiva, la prima regola soddisfatta accetta, rifiuta o scarta il pacchetto, se non viene soddisfatta nessuna regola si applica la policy che dice a sua volta se accettare, scartare o rifiutare il pacchetto. Generalmente si utilizzano due filosofie di policy, brutalmente: "accetta tutto tranne..." o "rifiuta tutto tranne...".

Le opzioni che si possono utilizzare nello scrivere uno script di configurazione di iptables sono:

- Creare una nuova catena (-N)
  Sintassi:
  `iptables -N nuova (max. 30 caratteri)`
- Cambia la tattica (policy) delle catene predefinite (-P)
  Sintassi:
  `iptables -P INPUT DROP`
- Mostra elenco delle regole di una catena (-L)
  Sintassi:
  `iptables -L`

- Cancella una catena vuota, senza regole (-X)
  Sintassi:
  `iptables -X nuova`
- Svuota una catena di tutte le sue regole (-F)
  Sintassi:
  `iptables -F OUTPUT`
- Azzera i contatori (pacchetti e byte) di tutte le regole di una catena (-Z)
  Sintassi:
  `iptables -Z INPUT`

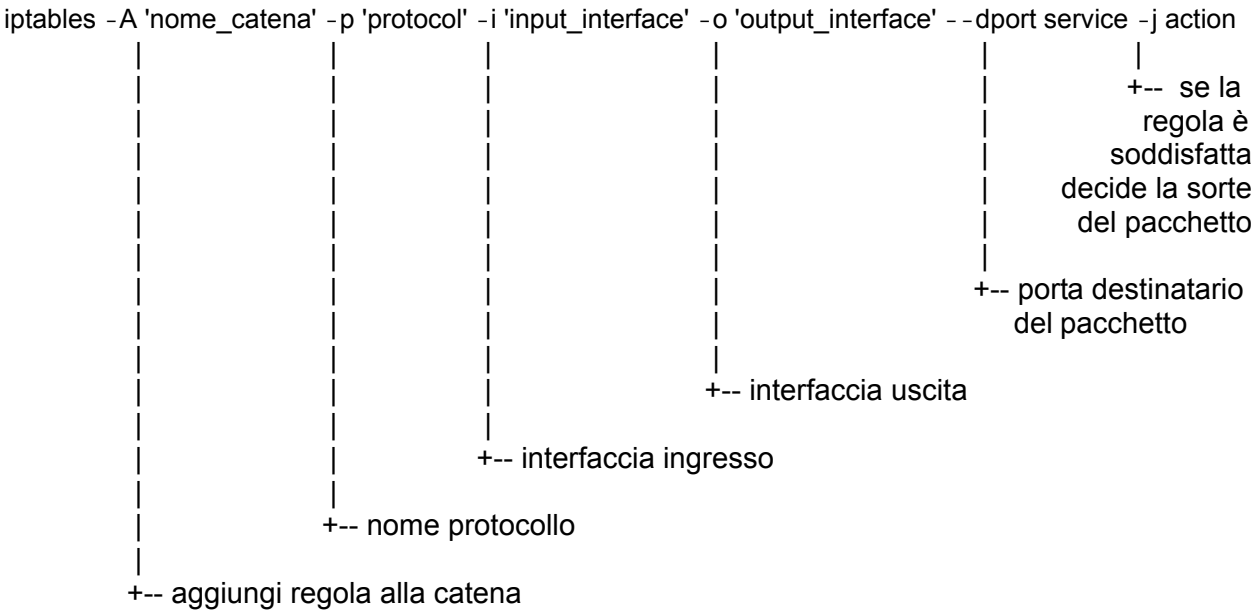
Le ultime 4 opzioni, se non viene specificata una catena, agiscono su tutte le catene:

```
# Rimuovi tutte le regole della catena OUTPUT
iptables -F OUTPUT

# Rimuovi tutte le regole presenti in TUTTE le catene
iptables –F
```

- Si possono anche visualizzare le informazioni riguardanti i pacchetti sottoposti alle regole con l'opzione -v
- Aggiungi in fondo alla catena una nuova regola (-A)

  Sintassi:



dove

- nome_catena : nome della catena alla quale applicare le regole
- protocol : tipo di protocollo del pacchetto
- input_interface : interfaccia dalla quale entra il pacchetto (ppp0 → modem, eth0→ scheda di rete)
- output_interface : interfaccia dalla quale deve uscire il pacchetto (ppp0 → modem, eth0 → scheda di rete)

- dport: porta destinataria del pacchetto, bisogna inserire il nome del servizio della porta.
- action: target della regola

Esempio:

```
iptables -A FORWARD -p tcp -i eth0 -o eth1 --dport telnet -j DROP
```

Questa regola impedisce a tutti i pacchetti TCP diretti alla LAN che utilizzano il Telnet di passare (non lascia fare telnet su di una macchina della rete dall'esterno).

- Inserisci una nuova regola alla catena (-I)
  Sintassi: vedi precedente modificando l'opzione -A con -I
  ```
  iptables -I INPUT -p tcp --dport telnet -j REJECT
  ```
- Sostituisci una regola presente in una determinata posizione (-R)
  Sintassi: sostituisci la seconda regola con la nuova
  ```
  iptables -R INPUT 2 -p icmp --icmp-type ping -j DROP
  ```
- Cancella una regola presente in una determinata posizione (-D)
  Sintassi: Cancella la precedente regola
  ```
  iptables -D INPUT -p icmp --icmp-type ping -j DROP
  ```
- Cancella la terza regola della catena input
  Sintassi:
  ```
  iptables -D INPUT 3
  ```

Operazioni di selezione:
Le seguenti opzioni (alcune già accennate) permettono di realizzare delle regole per il filtraggio dei pacchetti

- interfaccia input e output (‘-i’ e ‘-o’)
  Sintassi:
  ```
  iptables -A FORWARD -i ppp0 -o eth0 -p tcp -j REJECT
  ```
- E' possibile identificare un gruppo di interfacce utilizzando il carattere +
  Sintassi:
  ```
  iptables -A FORWARD -i ppp0 -o eth+ -p tcp -j REJECT
  ```
- Viceversa si possono identificare tutte le interfacce diverse da una (per indicare tutte le interfacce diverse da eth0 bisogna digitare ! eth0)
  Sintassi:
  ```
  iptables -A FORWARD -i ! ppp0 -j ACCEPT
  ```
- indirizzo sorgente e destinazione (‘-s’ e ‘-d’)
  Sintassi:
  ```
  iptables -A INPUT -s $RETE_LOCALE -d $RETE_ESTERNA -p icmp -j ACCEPT
  ```
- Protocollo tcp (‘-p tcp’ o ‘--protocol tcp’)
  Sintassi:
  ```
  iptables -A INPUT -p tcp -j ACCEPT
  ```

E' possibile specificare inoltre le opzioni '--tcp-flags', '--tcp-option', e '--syn' oltre a '--sport' e '--dport'.

- Porta sorgente e/o destinazione ('--sport' e '--dport')
  Sintassi:
  ```
  iptables -A FORWARD -p tcp -d 192.168.x.x --dport telnet -j ACCEPT
  ```
- E' possibile specificare anche un intervallo di porte, come ad es. 23 (Telnet), 21 (FTP), ecc...
  Sintassi:
  ```
  iptables -A INPUT -p tcp --dport ftp:telnet -j ACCEPT
  ```
- flags tcp ('--tcp-flags')
  Sintassi:
  ```
  iptables -A INPUT -p tcp --tcp-flags ALL SYN,ACK -j DROP
  ```

  I flag sono SYN, ACK, FIN, RST, URG, PSH.

  La prima stringa è la maschera (ALL) (flag da esaminare), la seconda indica quali dovrebbero risultare impostati.

- Pacchetti di richiesta connessione ('--syn')
  Sintassi:
  ```
  iptables -A INPUT -p tcp --syn -j DROP
  ```

  '--syn' si può utilizzare solo con il protocollo TCP, ed è molto utile per impedire i tentativi di connessione verso il firewall o la propria rete locale.

- Protocollo udp ('-p udp' o '--protocol udp')
  Sintassi:
  ```
  iptables -A INPUT -p udp -j ACCEPT
  ```
- Protocollo icmp ('-p icmp' o '--protocol icmp')
  Sintassi:
  ```
  iptables -A INPUT -p icmp -j ACCEPT
  ```
- E' possibile specificare inoltre l'opzione '--icmp-type' per indicare il tipo ICMP (ad es. '--icmp-type destination-unreachable', '--icmp-type 3', '--icmp-type 3/3').
- Per una lista completa utilizzare 'iptables -p icmp –h'.

Funzionalità avanzate:
Fino ad ora sono state utilizzate delle istruzioni abbastanza semplici che per la maggior parte erano compatibili con ipchains, in questa sezione vengono trattate regole che forniscono maggiore protezione ma che esistono solo in iptables e con le apposite opzioni del Kernel compilate (o installate come moduli).

Possiamo, ad esempio, controllare lo stato dei pacchetti, come abbiamo descritto nel paragrafo 5.2 a pagina 74.

E' possibile ottenere ciò inserendo nelle regole l'opzione avanzata 'state', la quale è contenuta nel modulo ip_conntrack e serve a tenere traccia delle connessioni e a stabilire se un pacchetto appartiene a una di esse o meno. Per connessioni si intendono anche quelle UDP, ICMP che non necessitano di un meccanismo di richiesta/conferma ("three-way handshake").

Una volta caricato il modulo ip_conntrack, appena viene stabilita una nuova connessione esso provvede a prendere nota delle informazioni più importanti (indirizzo sorgente, indirizzo destinazione, porta sorgente, porta destinazione, ...) e a memorizzarle.

Quindi è possibile interrogare il "connection tracking" per conoscere che tipo di pacchetto è stato ricevuto (NEW, ESTABLISHED, RELATED, INVALID) e decidere l'azione da intraprendere (ACCEPT, DROP, …).

A questo scopo è necessario specificare l'opzione '-m state --state' seguita da una o più degli stati:

```
iptables -A FORWARD –d 62.0.0.3 –m state –-state ESTABLISHED,RELATED –j ACCEPT
```

Allo stesso modo è possibile controllare i flags dei pacchetti, per decidere se accettarli o meno:

```
iptables -N tcp_flags
iptables -A tcp_flags -p tcp --tcp-flags ALL NONE -j DROP
iptables -A tcp_flags -p tcp --tcp-flags SYN,RST SYN,RST -j DROP
iptables -A tcp_flags -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP
```

l'opzione alla prima riga crea una nuova tabella, la prima regola blocca i pacchetti senza flags, la seconda quelli del tipo SYN+RST e la terza i SYN+FIN.

La scrittura di un semplice script:
Ora che sappiamo le principali opzioni di iptables possiamo cimentarci nella scrittura di un semplice script.

Per semplicità mettiamo il path di iptables nella variabile IPTABLE e in IP il nostro IP

```
IPTAB=/usr/local/sbin/iptables
IP=`ifconfig ppp0 | grep inet | cut -d : -f 2 | cut -d\ -f 1`
```

Definiamo ora le policy della catena filter:

```
$IPTAB -P INPUT DROP $IPTAB -P FORWARD DROP $IPTAB -P OUTPUT ACCEPT
```

Ora stabiliamo le regole della tabella nat:

```
$IPTAB -t nat -A PREROUTING -i ppp0 -s 127.0.0.0/8 -j DROP
$IPTAB -t nat -A PREROUTING -i ppp0 -s 192.168.0.0/16 -j DROP
$IPTAB -t nat -A PREROUTING -i ppp0 -d ! $IP -j DROP
```

Con la prima regola scartiamo tutti i pacchetti che dicono di provenire dalla nostra macchina (127.x.x.x) (il numero dopo ‘ / ’ è la maschera) perchè questi dovrebbero viaggiare solo sull'interfaccia virtuale (lo) e non sul modem (ppp0).
La seconda regola elimina tutti i pacchetti provenienti dalla rete locale poiché sono sull'interfaccia del modem.
La terza scarta tutti i pacchetti che non sono rivolti a noi (dato che un router non li invierebbe certo a noi questi pacchetti sono sospetti). Questa istruzione attiva un controllo sulla sanità del pacchetto scartando i pacchetti ambigui.

Vengono create due catene alle quali verranno rimandati i pacchetti:

```
$IPTAB -N ppp_in
$IPTAB -N services
```

e definite le seguenti regole:

```
$IPTAB -A INPUT -i lo -j ACCEPT
$IPTAB -A INPUT -i ppp0 -j ppp_in
```

La prima lascia passare tutti i pacchetti local-to-local sull'interfaccia lo.
La seconda rimanda tutti i pacchetti che provengono dal modem alla catena ppp_in precedentemente creata.

```
$IPTAB -A ppp_in -p icmp --icmp-type ! echo-request -j ACCEPT
```

Quest’ultima linea lascia passare tutti i pacchetti ICMP tranne le echo-request (i ping). Questa regola consente comunque l’invio di pacchetti di ping perché siamo nella catena chiamata da INPUT, quindi un ping sul nostro IP non produrrà alcun risultato.

```
$IPTAB -A ppp_in -p tcp --dport 6000:6010 -j DROP
$IPTAB -A ppp_in -p udp --dport 6000:6010 -j DROP
$IPTAB -A ppp_in -p tcp --dport 7000:7010 -j DROP
$IPTAB -A ppp_in -p udp --dport 7000:7010 -j DROP
$IPTAB -A ppp_in -p tcp --dport 1:1024 -j services
$IPTAB -A ppp_in -p udp --dport 1:1024 -j services
```

Queste regole bloccano tutti i pacchetti destinati alle porte del X11 e Xfont Server perché i pacchetti su queste porte potrebbero danneggiare i rispettivi servizi.
I pacchetti diretti alle porte riservate (inferiori alla 1024) vengono controllati dalla catena services.

```
#DNS
$IPTAB -A ppp_in -p udp --sport 53 -j ACCEPT
$IPTAB -A ppp_in -p tcp --sport 53 -j ACCEPT

#smtp
$IPTAB -A ppp_in -p tcp --sport 25 -j ACCEPT
```

```
#nntp
$IPTAB -A ppp_in -p tcp --sport 119 -j ACCEPT

#telnet
$IPTAB -A ppp_in -p tcp --sport 23 -j ACCEPT

#ftp (dati e linea di controllo)
$IPTAB -A ppp_in -p tcp --sport 20 -j ACCEPT
$IPTAB -A ppp_in -p tcp --sport 21 -j ACCEPT

#irc
$IPTAB -A ppp_in -p tcp --sport 6666:7000 -j ACCEPT
$IPTAB -A ppp_in -p udp --sport 6666:7000 -j ACCEPT

#talk
$IPTAB -A ppp_in -p tcp --sport 518:519 -j ACCEPT
$IPTAB -A ppp_in -p udp --sport 518:519 -j ACCEPT
#http e https
$IPTAB -A ppp_in -p tcp -m multiport --sport 80,443 -j ACCEPT

#pop3
$IPTAB -A ppp_in -p tcp --sport 110 -j ACCEPT

#imap
$IPTAB -A ppp_in -p tcp -m multiport --sport 143,220 -j ACCEPT
```

Queste regole fanno passare i pacchetti relativi a: DNS, posta in uscita, News, Telnet in uscita, FTP, IRC, Talk, Web, server Pop3, server Imap2 e Imap3.

Il modulo multiport serve a scrivere più porte non consecutive nella stessa regola.

```
$IPTAB -A services -p tcp --dport 113 -j REJECT
$IPTAB -A services -j DROP
```

La prima regola indica di non accettare i pacchetti sulla porta 113, ma la macchina che li ha generati (solitamente un server IRC) verrà avvisata che il servizio non è disponibile. La seconda regola scarta tutti gli altri pacchetti senza avvisare nessuno. Se invece volessimo lasciare aperti dei servizi occorre inserire tra le due regole (l'istruzione DROP deve essere sempre l'ultima) quest'ultima:

```
$IPTAB -A services -p tcp --dport 80 -j ACCEPT
```

Così facendo il nostro server web sarà accessibile dall'esterno.

```
$IPTAB -N conn_state
$IPTAB -N tcp_flags
```

Queste tabelle verranno utilizzate per il controllo dello stato del pacchetto e i flags del pacchetto.

```
$IPTAB -A conn_state -m state --state INVALID -j DROP
$IPTAB -A conn_state -m state --state RELATED,ESTABLISHED -j ACCEPT
```

La prima regola scarta i pacchetti che non appartengono a nessuna connessione o che non sono dei SYN.
La seconda invece lascia passare i pacchetti appartenenti o relativi ad una connessione.

Per i pacchetti SYN la decisione viene lasciata alla catena ppp_in.

La seguente catena verifica i flags dei pacchetti per decidere se accettarli o meno.

```
$IPTAB -A tcp_flags -p tcp --tcp-flags ALL FIN,URG,PSH -j DROP
$IPTAB -A tcp_flags -p tcp --tcp-flags ALL ALL -j DROP
$IPTAB -A tcp_flags -p tcp --tcp-flags ALL SYN,RST,ACK,FIN,URG -j DROP
$IPTAB -A tcp_flags -p tcp --tcp-flags ALL NONE -j DROP
$IPTAB -A tcp_flags -p tcp --tcp-flags SYN,RST SYN,RST -j DROP
$IPTAB -A tcp_flags -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP
$IPTAB -A tcp_flags -p tcp --tcp-flags FIN FIN -j DROP
```

Le prime tre regole servono ad impedire gli scan Xmas tree di nmap, la quarta blocca i pacchetti senza flags, la quinta blocca i pacchetti SYN+RST, la penultima blocca i pacchetti SYN+FIN.
L'ultima, contro gli scan FIN, è sensata solo se è presente anche la catena conn_state e se viene chiamata per prima.

I SYN che tornano da una connessione TCP verranno riconosciuti come parte della connessione e fatti passare.

Queste due catene dovranno essere inserite all'inizio della catena ppp_in con queste istruzioni:

```
$IPTAB -I ppp_in 1 -j conn_state
$IPTAB -I ppp_in 1 -j tcp_flags
```

La catena conn_state rende superflua la catena ppp_in in quanto un normale utente non vorrà permettere connessioni sul proprio computer.

Il firewall può essere sintetizzato in questo modo:

```
$IPTAB -P INPUT DROP
$IPTAB -F
$IPTAB -X
$IPTAB -t nat -A PREROUTING -i ppp0 -s 127.0.0.0/8 -j DROP
$IPTAB -t nat -A PREROUTING -i ppp0 -s 192.168.0.0/16 -j DROP
$IPTAB -t nat -A PREROUTING -i ppp0 -d ! $IP -j DROP
$IPTAB -N ppp_in
$IPTAB -N services
$IPTAB -N conn_state
$IPTAB -N flags
$IPTAB -N blocked
```

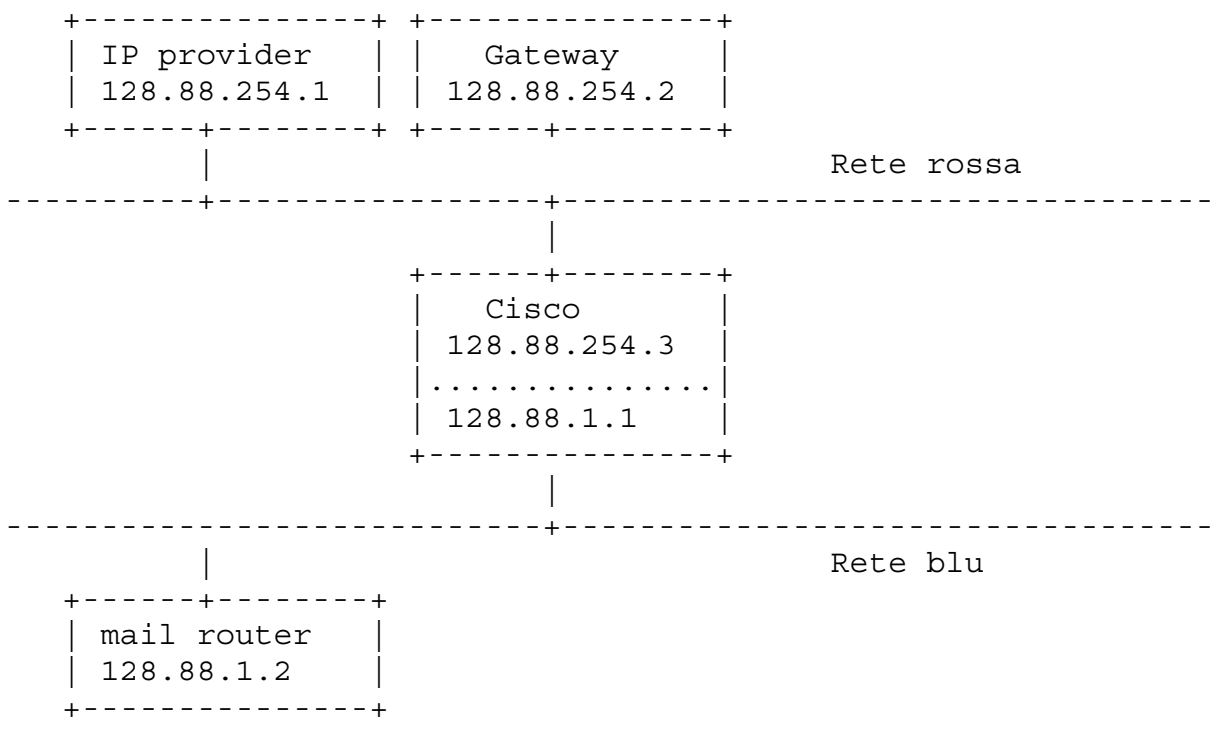
```
$IPTAB -A INPUT -i lo -j ACCEPT
$IPTAB -A INPUT -i ppp0 -j ppp_in
$IPTAB -A conn_state -m state --state INVALID -j DROP
$IPTAB -A conn_state -m state --state RELATED,ESTABLISHED -j ACCEPT
$IPTAB -A tcp_flags -p tcp --tcp-flags ALL FIN,URG,PSH -j DROP
$IPTAB -A tcp_flags -p tcp --tcp-flags ALL ALL -j DROP
$IPTAB -A tcp_flags -p tcp --tcp-flags ALL SYN,RST,ACK,FIN,URG -j DROP
$IPTAB -A tcp_flags -p tcp --tcp-flags ALL NONE -j DROP
$IPTAB -A tcp_flags -p tcp --tcp-flags SYN,RST SYN,RST -j DROP
$IPTAB -A tcp_flags -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP
$IPTAB -A tcp_flags -p tcp --tcp-flags FIN FIN -j DROP
$IPTAB -A services -j DROP
$IPTAB -A ppp_in -j blocked
$IPTAB -A ppp_in -j conn_state
$IPTAB -A ppp_in -j flags
$IPTAB -A ppp_in -p tcp --dport 1:1024 -j services
$IPTAB -A ppp_in -p udp --dport 1:1024 -j services
$IPTAB -A ppp_in -p icmp --icmp-type ! echo-request -j ACCEPT
$IPTAB -A ppp_in -p tcp --sport 20 -j ACCEPT
```

Appendice

A1. Regole logiche di filtraggio per il mio Cisco

Il seguente esempio mostra una possibile configurazione per usare il Cisco come un router di filtraggio. E' un esempio che mostra l'implementazione di una specifica linea di condotta.

In questo esempio, una compagnia ha una Classe B di indirizzi di rete 128.88.0.0 e sta usando 8 bit per le subnet. La connessione Internet è sulla sottorete rossa 128.88.254.0. Tutte le altre sottoreti sono considerate di fiducia o sottoreti blu.



Teniamo in mente i seguenti punti perchè ci aiuteranno a capire i pezzi della configurazione:

1. I Cisco applicano i filtri ai soli pacchetti in uscita.
2. Le regole sono testate in ordine e si fermano quando la prima occorrenza è stata trovata.
3. C'è una implicita regola di rifiuto alla fine di una lista di accesso che nega ogni cosa.

Detto ciò, passiamo al nostro esempio.

Le linee guida per l'implementazione sono:

- Niente non esplicitamente permesso è negato.
- Il traffico tra la macchina esterna che fa da gateway e l'host della rete blu è permesso.
- I servizi permessi sono originati dalla rete blu.
- Assegnare un range di porte alla rete blu per le informazioni di ritorno della connessioni dati FTP.

Ecco le nostre regole (i numeri di linea e la formattazione sono state aggiunte per leggibilità):

```
 1  no ip source-route
 2  !
 3  interface Ethernet 0
 4  ip address 128.88.1.1 255.255.255.0
 5  ip access-group 10
 6  !
 7  interface Ethernet 1
 8  ip address 128.88.254.3 255.255.255.0
 9  ip access-group 11
10  !
11  access-list 10 permit ip 128.88.254.2 0.0.0.0
     128.88.0.0 0.0.255.255
12  access-list 10 deny   tcp 0.0.0.0 255.255.255.255
     128.88.0.0 0.0.255.255 lt 1025
13  access-list 10 deny   tcp 0.0.0.0 255.255.255.255
     128.88.0.0 0.0.255.255 gt 4999
14  access-list 10 permit tcp 0.0.0.0 255.255.255.255
     128.88.0.0 0.0.255.255
15  !
16  access-list 11 permit ip 128.88.0.0 0.0.255.255
     128.88.254.2 0.0.0.0
17  access-list 11 deny   tcp 128.88.0.0 0.0.255.255
     0.0.0.0 255.255.255.255 eq 25
18  access-list 11 permit tcp 128.88.0.0 0.0.255.255
     0.0.0.0 255.255.255.255
```

Linea	Spiegazione
1	Benché non sia una regola di filtraggio, è buona cosa includerla qui.
5	Ethernet 0 è nella rete rossa. La lista estesa 10 di permessi di accesso sarà applicata all'output su questa interfaccia. Puoi anche pensare l'output dalla rete rossa come input nella rete blu.
9	Ethernet 1 è nella rete blu. La lista estesa 11 di permessi di accesso sarà applicata all'output su questa interfaccia.
11	Permette tutto il traffico dalla macchina gateway alla rete blu.

12-14 Permette connessioni originarie della rete rossa che entrano tra le porte 1024 e 5000. Ciò avviene per permettere alle connessioni dati in FTP di dialogare alla rete blu.

16 Permette l'accesso a tutti i pacchetti della rete blu alla macchina gateway.

17 Nega l'SMTP (porta TCP 25) mail alla rete rossa.

18 Permette a tutto l'altro traffico TCP di accedere alla rete rossa.

Nota: permessi e richieste nella connessione FTP avvengono su una porta diversa da quella dove vengono scambiati i dati (5000 è stato scelto come limite superiore).

Bibliografia

Building Internet Firewalls (first edition, November 1995)
di D. Brent Chapman e Elizabeth D. Zwicky - ed. O'Reilly

Firewalling and Proxy Server HOWTO (v0.80, Feb. 26, 2000)
di Mark Grennan - http://www.grennan.com/Firewall-HOWTO.html

Linux Firewall: metti una marcia in più nella tua rete (Feb. 24, 2002)
di Emiliano Bruni - www.ebruni.it

Internet e Reti di Calcolatori (seconda edizione, 2003)
di James F. Kurose e Keith W. Ross - ed. McGraw-Hill

Telnet Protocol Specification RFC 854
di Valentina Casola - Seconda Università degli studi di Napoli

Note per la distribuzione

Questo lavoro è stato realizzato da Vincenzo Capuano, Mario Guida, Domenico Brasiello e Giancarmine Manco. Il documento nasce nel giugno 2004 come approfondimento teorico del corso di Reti di Calcolatori della Seconda Università degli studi di Napoli.

Il materiale raccolto viene reso disponibile gratuitamente e può essere distribuito sotto qualsiasi forma (anche cartacea) alle seguenti condizioni:

- il materiale dovrà essere distribuito senza alcuna modifica ai suoi contenuti (salvo esplicita autorizzazione);
- niente sarà dovuto da chi lo acquisisce a colui che lo distribuisce, salvo rimborso di eventuali spese per il supporto che lo contiene.

E' possibile contattare gli autori all'indirizzo e-mail enzo@sitoserio.it

Gli autori, giugno 2004