Firewall Piercing mini-HOWTO

François-René Rideau, fare@tunes.org

v0.3b, 27 novembre 1998

Direttive per l'utilizzo del protocollo ppp usando telnet per operare in modo trasparente attraverso un firewall Internet. Traduzione a cura di *Stefano di Sandro <stedis@radiolink.net>* , ultima revisione 24 Gennaio 2000.

Indice

1 Varie

1.1 LIBERATORIA

LEGGI QUESTA SEZIONE: È IMPORTANTE !!!

Qui di seguito declino tutte le responsabilità per queste informazioni. Qualunque sia il modo in cui il loro utilizzo possa ritorcersi contro di voi, non è colpa mia. Se non siete in grado di comprendere i rischi cui vi accingete a sottoporvi facendo ciò che qui è scritto, non fatelo. Se userete queste informazioni e ciò permetterà a balordi teppisti di penetrare le difese dei computer della vostra azienda compromettendo voi, il vostro impiego e i miliardi della vostra azienda, beh sono problemi vostri. Non venite a piangere da me.

1.2 Copyright

1.3 Ringraziamenti

Sebbene abbia riscritto quasi tutto tranne la liberatoria, sono in debito con Barak Pearlmutter `mailto:bap@cs.unm.edu`

per il suo Term-Firewall mini-HOWTO: penso che vi fosse la necessità di un mini-HOWTO sul firewall piercing, e nonostante qualche difetto, il suo mini-HOWTO si è rivelato un modello e un incoraggiamento.

2 Introduzione

2.1 Premessa

Dal momento che amministratori di sistema e semplici utenti hanno diversi diritti e doveri, può accadere che un utente si trovi protetto dietro un firewall, che è in grado di attraversare, ma in maniera complicata. Questo mini-HOWTO spiega un metodo generale e versatile per utilizzare gli strumenti di Internet, senza che apparentemente si debba attraversare alcun firewall, facendo uso di un emulatore IP all'interno di una sessione telnet.

Tutto è liberamente inspirato al Term-Firewall mini_HOWTO di Barak Pearlmutter `mailto:bap@cs.unm.edu` , che si affidava sia all'antico e abbandonato Term (a suo tempo un grande programma), che ai dettagli di una implementazione di telnet non proprio standard, fatta di codice obsoleto e non portabile.

2.2 Problemi di sicurezza

Se l'amministratore del vostro sistema ha predisposto un firewall, è naturale che possa aver avuto le sue buone ragioni, così come è probabile che voi abbiate sottoscritto l'accordo di non aggirarlo. D'altro canto, l'eventuale possibilità di utilizzare telnet verso l'esterno (che è un requisito affinché quanto ci accingiamo a spiegare funzioni) significa che vi è stato concesso di connettervi con sistemi remoti e, se potete effettuare il login su alcuni di questi sistemi, significa che qualcuno ve l'ha a sua volta permesso.

Quindi l'utilizzo dei varchi legali in un firewall diventa estremamente *conveniente* per permettere a qualunque programma remoto di comunicare con la nostra macchina utilizzando i normali protocolli. Nel caso opposto avremmo bisogno di programmi speciali o modificati (e ricompilati) facenti uso di proxy dai compiti particolari, le cui configurazioni possono essere opera di amministratori sprovveduti o incompetenti. Oppure potremmo dover installare un certo numero di convertitori per poter accedere a ciascuno dei normali servizi (come la posta elettronica) attraverso le strade consentite dal firewall (come il web).

Inoltre l'uso di un emulatore IP che operi a livello utente come SLiRP, può essere anche utile per prevenire attacchi dall'esterno in grado di perforare il firewall nuovamente in senso inverso, a meno che non siate voi a permetterlo esplicitamente (oppure che l'attacco sia condotto in modo abile e astuto o che l'intruso abbia acquisito i privilegi di root o, infine, che sia in grado di spiarvi sull'host remoto).

Sia come sia, il metodo qui presente dovrebbe essere *relativamente* sicuro. Tutto dipende dalle particolari circostanze nelle quali vi troverete quando lo metterete all'opera e io non posso darvi alcuna garanzia. Molti sono gli aspetti intrinsecamente insicuri in una qualunque connessione internet e non dipendono esclusivamente dall'uso di questo metodo; non assumete a priori di essere al sicuro a meno che non ne abbiate delle valide ragioni e cercate di criptare sempre l'informazione.

Per concludere: non usate questo metodologia se non sapete cosa state facendo. È meglio che rileggiate la liberatoria.

2.3 Altri requisiti

Si dà per scontato che sappiate cosa state facendo; che sappiate impostare una connessione di rete; che possediate un account di shell su entrambi i lati del firewall; che possiate usare telnet (o ssh, o equivalenti) da un account all'altro; che possiate far girare un emulatore IP su entrambi i lati della connessione; che possediate programmi in grado di lavorare su un'emulazione IP. Si noti come qualunque programma possa usare la connnessione, in questo caso è il pppd l'emulatore locale che colloquia con il kernel di Linux; altri emulatori, come Term, necessitano di essere ricompilati e collegati a speciali librerie.

Parlando di emulatori IP, il demone pppd si trova in qualunque buona distribuzione di Linux o sito ftp; e lo stesso vale per SLiRP. Se l'account sulla shell remota vi consente di eseguire programmi soltanto a livello utente, SliRP è la soluzione da adottare per la connessione.

2.4 Scaricare il software

La maggior parte del software descritto sarà disponibile nella vostra distribuzione o enventualmente tra i contrib; tranne gli ultimi due tutti si possono trovare come pacchetti rpm. Nel caso vogliate recuperare l'ultima versione dei sorgenti o degli eseguibili (dopo tutto non è detto che su entrambi i lati della connessione si trovi un sistema Linux) utilizzate gli indirizzi elencati di seguito:

- `SLiRP` si trova a

  `http://blitzen.canberra.edu.au/slirp` oppure

  `ftp://www.ibc.wustl.edu/pub/slirp_bin/` .

- `zsh` la trovate presso

  `http://www.peak.org/zsh/` .

- `ppp` è scaricabile da

  `ftp://cs.anu.edu.au/pub/software/ppp/` .

- `fwprc` e `cotty` sono invece a

  `http://www.tunes.org/~fare/files/fwprc/` .

3 Capire il problema

Capire un problema è la prima metà del percorso che porta alla sua soluzione.

3.1 Dare un nome alle cose

Se volete che questo medoto funzioni, è obbligatorio che abbiate un'idea di come funziona, così, nell'eventualità che qualcosa vada storto, saprete dove andare a mettere le mani.

D'ora in avanti avranno aggettivo locale sia la macchina che inizia la connessione, sia i programmi e i file che si trovano in essa; dunque, tutto quello che si trova dall'altra parte sarà remoto.

3.2 Il problema

Il nostro fine è collegare l'ingresso e l'uscita di un emulatore IP locale rispettivamente all'uscita e all'ingresso di un emulatore IP remoto. I canali di comunicazione con i quali gli emulatori interagiscono sono device diretti (come nel caso usuale del pppd) o il tty corrente. Questo ovviamente non si verifica con una sessione telnet. In quest'ultimo caso la situazione è complicata perché, quando lanciate l'emulatore locale da riga comando, il tty corrente è collegato all'utente non a una sessione remota. Inoltre quando apriamo una nuova sessione, sia essa locale o remota, su un nuovo terminale, dobbiamo sincronizzare l'avvio e la connessione di entrambi gli emulatori IP altrimenti la spazzatura prodotta in uscita da una delle due sessioni rappresenterà un comando per l'altra sessione con il risultato di produrre altra spazzatura.

3.3 Difficoltà aggiuntive

Per ottenenere la massima facilità d'uso, l'emulatore IP locale deve fornire un IP al kernel per le operazioni di rete, per tale ragione si usa il pppd. Comunque, il pppd è limitato abbastanza da accettare dati attraverso una sola voce all'interno della directory /dev o attraverso il terminale corrente (tty); una coppia di pipe sarebbe stata molto più naturale. Tutto funziona correttamente per quanto riguarda il pppd remoto, visto che quest'ultimo può usare il tty della sessione telnet; ma per il pppd locale è un problema perché non è in grado di lanciare la sessione telnet per effettuare la connessione e quindi dovremo provvedere ad aggiungergli attorno uno strato di software.

Telnet si comporta *quasi* corretamente con una coppia di pipe, solo che si ostinerà sempre a effettuare il controllo di I/O (ioctl) sul tty corrente, con il quale interagisce; usare telnet senza un tty è causa inoltre di corse critiche che faranno fallire la connessione su macchine lente (fwrpc 0.1 funziona perfettamente su un P/MMX 233, una volta su sei su un 6x86-P200+ e mai su un 486DX2/66).

[Nota: se trovo quel bischero (probabilmente qualcuno del MULTICS, sebbene debba esserci stata gente UNIX stupida abbastanza da copiare l'idea) che ha inventato il principio dei dispositivi tty in base al quale si legge e si scrive dallo stesso pseudo-file, invece di poter disporre di una pulita coppia di pipe, lo strangolo!]

4 La soluzione

4.1 Il principio

Il programma per il firewall-piercing, `fwprc`, farà uso di un proxy tty, `cotty`, che apre due dispositivi pseudo-tty, invoca alcuni comandi su ciascuno di questi slave e, senza più smettere, copia ogni carattere che viene battuto, nel tty che serve da ingresso per l'altro comando. Un comando sarà la connessione telnet al sito remoto e l'altro sarà il locale demone pppd. Il pppd può quindi aprire e controllare la sessione telnet con il più classico degli script di chat.

4.2 fwprc

Ho realizzato un script auto-documentato per forare i firewall, `fwprc`, disponibile al mio sito `http://www.tunes.org/~fare/files/fwprc/` , insieme a `cotty` (che è necessario nelle versioni `fwprc` 0.2 e successive). Al momento di scrivere queste parole, le versioni più recenti sono `fwprc` 0.3a e `cotty` 0.3a.

Il nome fwprc è volutamente illeggibile e impronunciabile, al fine di confondere il paranoico amministratore di sistema che probabilmente è la causa del firewall che vi rompe le scatole (naturalmente, possono esistere anche firewall opportuni, e talvolta indispensabili; la sicurezza è tutta una questione di *corretta* configurazione). Se dovete pronunciare questa parola ad alta voce, fate in modo di dirla nel peggior modo possibile.

SFIDA! SFIDA! Mandatemi un file audio in formato `.au` con la registrazione digitale della vostra pronuncia di fwprc. La peggiore vincerà un aggiornamento gratuito e il suo nome nella pagina della versione 1.0 di fwprc.

Ho verificato il programma con svariate impostazioni configurandolo attraverso dei file di risorse. Ma naturalmente, per la legge di Murphy, a voi non funzionerà. Ritenetevi liberi di contribuire ai miglioramenti che potranno rendere la vita più facile alle persone che faranno le cose dopo di voi.

4.3 .fwprcrc

`fwprc` può essere personalizato attraverso il file `.fwprcrc` che deve essere disponibile su entrambi i lati della connessione. È anche possibile predisporre configurazioni diverse da usare alternativamente (per esempio, *io* lo faccio), ed è lasciato come esercizio al lettore.

Per cominciare, copiate la sezione appropriata di `fwprc` (la penultima) nel file `.fwprcrc` all'interno della vostra home directory. Poi rimpiazzate i valori delle variabili con quelli adatti alla vostra configurazione. Infine copiate il tutto anche sull'altro host e provate.

Il metodo di base prevede di usare il pppd localmente e slirp sulla macchina remota. Per modificarlo potreste ridefinire la appropriata funzione all'interno del vostro `.fwprcrc` con una linea del tipo:

```
remote_IP_emu () { remote_pppd }
```

Ricordate che SLiRP è più sicuro di pppd, ed è più facile accedervi, dal momento che non richiede privilegi di root sull'host remoto. Un'altra caratteristica di sicurezza consiste nel fatto che scarterà tutti i pacchetti che non provengono direttamente dalla macchina a esso connessa (tale caratteristica diventa un difetto se tentate di sfruttare questo metodo per realizzare il routing di una sottorete usando il mascheramento dell'IP). Le funzionalità di base di SLiRP sono piuttosto affidabili anche se l'ho trovato privo delle aggiunte promesse (quali la controllabilità a tempo di esecuzione), ma, dal momento che è un software libero, siete anche voi liberi di mettere le mani nel codice sorgente in modo da implementare tutte le funzionalità di cui possiate avere bisogno.

5 Piercing al contrario

5.1 Giustificazioni

Talvolta, solo da un lato del firewall è possibile lanciare una sessione telnet; ciò nonostante alcune forme di comunicazione restano possibili (tipicamante usando la posta elettronica). Forare il firewall è ancora possibile, escogitando un qualunque modo di innescare una connessione telnet dalla parte giusta del firewall verso l'altra.

`fwprc` include il codice per scatenare tali connessioni a partire da un messaggio di posta elettronica autenticato con PGP; tutto ciò di cui abbisognate è aggiungere `fwprc` come filtro per `procmail(1)` ai messaggi che fanno uso di tale protocollo, (le istruzioni sono incluse in `fwprc`). Notate comunque che per lanciare pppd con i privilegi appropriati dovrete creare da soli un suid wrapper per diventare root. Istruzioni incluse in `fwprc`.

La sola autenticazione di questa sorta di scintilla non significa aver predisposto una connessione sicura. Diventa davvero opportuno, in questo caso, fare uso di una Secure Shell (anche sopra telnet) per rendere sicuro il collegamento. E infine osservate attentamente ciò che accade tra l'innesco della connessione telnet e la ssh che ha luogo su tale connessione. Qualunque contributo in questa direzione è ben accetto.

5.2 Ricevere il messaggio di innesco

Se un firewall vi circonda, il vostro messaggio potrebbe trovarsi in un server centrale che non consente il fitraggio con procmail o alcuna connessione telnet. Niente paura! Potete usare `fetchmail(1)` da eseguire in modalità demone per recuperare e trasferire la posta al vostro sistema linux che opera da clien, e/o aggiungere un job al servizio cron per automatizzare il recupero della posta ogni 1-5 minuti. Fetchmail inoltrerà la posta verso l'indirizzo locale usando `sendmail(8)` che, a sua volta, deve essere configurato per usare `procmail(1)` per il recapito. Se eseguite `fetchmail(1)` in background come demone, questi impedirà qualunque altra esecuzione di fetchmail: come nel caso dell'apertura di un `fwprc`. Naturalmente potreste far girare fetchmail come falso utente. Recuperi troppo frequenti possono non essere opportuni nei confronti del server, così come recuperi troppo sporadici possono obbligare a pesanti attese affinchè il messaggio venga letto e la connessione inversa stabilita. Io uso una frequenza di recupero di due minuti.

6 Note Finali

6.1 Altre impostazioni

Ci sono altri tipi di firewall, come quelli che non consentono le connessioni telnet. Dal momento che un continuo flusso di pacchetti attraversa un firewall e trasporta informazioni fuori e dentro di esso, è sempre possibile perforarlo; al solo prezzo di usare il punteruolo più in alto o più in basso.

In un caso estremamente semplice, potreste semplicemente lanciare `ssh` su un pty, e lavorare con il `pppd` nel tty slave. `cotty` 0.3a dovrebbe essere in grado di farlo, ma ancora nessuno ha modificato `fwprc` per effettuare il login. Potrebbe essere un buon esercizio per stanotte. Potreste voler applicare quanto visto con un firewall ostile, solo per costrure una "VPN" sicura (Virtual Private Network). Leggete VPN mini-HOWTO a proposito di questo.

Se dovete passare attraverso una linea a 7-bit, probabilmente userete SLIP al posto di PPP. Io non ho mai provato: le linee sono quasi tutte 8 bit al giorno d'oggi, ma non dovrebbe essere complicato.

Ora, se l'unica strada attraverso il firewall è un proxy WWW (di solito è il minimo per una rete connessa a internet) potreste scrivere un demone che registra il traffico di dati in ingresso e in uscita, e farlo girare durante le connessione HTTP, ottenendo una sorta di telnet-su-HTTP con il quale eseguire fwrpc. Potrebbe rivelarsi una soluzione lenta e non molto efficace ma sufficiente da permettervi di usare `fetchmail(1)`, `suck(1)` e altri programmi non interattivi.

Se volete prestazioni maggiori o se le uniche cose che passano inalterate attraverso il firewall sono cosucce di basso livello (richieste DNS, pacchetti ICMP, ecc.) allora il gioco si fa duro dal momento che dovrete mettere le mani sul primitivo stack IP usando (per esempio) i Fox project's packet-protocol functors. Otterrete una forma diretta di IP-su-HTTP, IP-su-DNS, IP-su-ICMP, o affini. Questi ultimi non richiedono soltanto un complesso protocollo, ma anche un'interfaccia al nucleo del sistema operativo ed entrambi sono costosi da implementare.

Ancora una cosa, se usate un demone HTTP per il Firewall-piercing, non dimenticate di fare in modo che serva pagine fasulle, in questo modo ingannerete i sospettosi amministratori dei firewall avversari.

6.2 Manutenzione dell'HOWTO

Ho sentito il bisogno di scriverlo, ma non ho tutto il tempo da dedicargli, ecco perché questo HOWTO è così scarno. E così resterà, a meno che non riceva abbastanza commenti che mi permettano di individuare le sezioni che devono essere migliorate. I commenti sono benvenuti. L'aiuto è benvenuto. Il mantenimento del mini-HOWTO è benvenuto.

A ogni modo, le sezioni che avete letto hanno messo in evidenza diversi problemi le soluzioni dei quali richiedono solo che qualcuno (voi?) vi dedichi un po' di tempo (oppure denaro, pagando altri che lo facciano in sua vece), sedendosi e scrivendole: nulla di davvero complicato, sebbene i dettagli possano apparire pesanti e difficili.

Non esitate a contribuire con altri problemi e possibilmente con altrettante soluzioni, a questo mini-HOWTO.

6.3 Copia extra della IMPORTANTE LIBERATORIA — CREDETEMI!!!

> **Di seguito declino ogni responsabilità per questa metodologia. Il fatto che possa ritorcersi contro di voi in un qualunque modo è un problema che non mi riguarda. Se non riuscite a comprendere i rischi inerenti il suo utilizzo, non utilizzatelo. Se, usando questa metodologia, permetterete a balordi vandali di penetrare nei computer della vostra azienda compromettendo voi, il vostro lavoro e i miliardi della stessa, non venite a piangere da me.**