# «Back|Track-[IT]

www.backtrack.it

(c) 2009 swami
flavio.baldassi@gmail.com

# *Netcat*

***

Oggi parleremo di uno dei più famosi tool riguardanti la sicurezza informatica, il suo nome è netcat ed è anche soprannominato il coltellino svizzero delle reti (vi lascio immaginare il perché). Netcat è uno strumento molto potente e molto facile da usare, queste due caratteristiche ne fanno uno dei tool più usanti. Le sue funzioni possono andare dal portscanning, file transfer, banner grabbing , remote administration, reverse shell fino al port redirection.

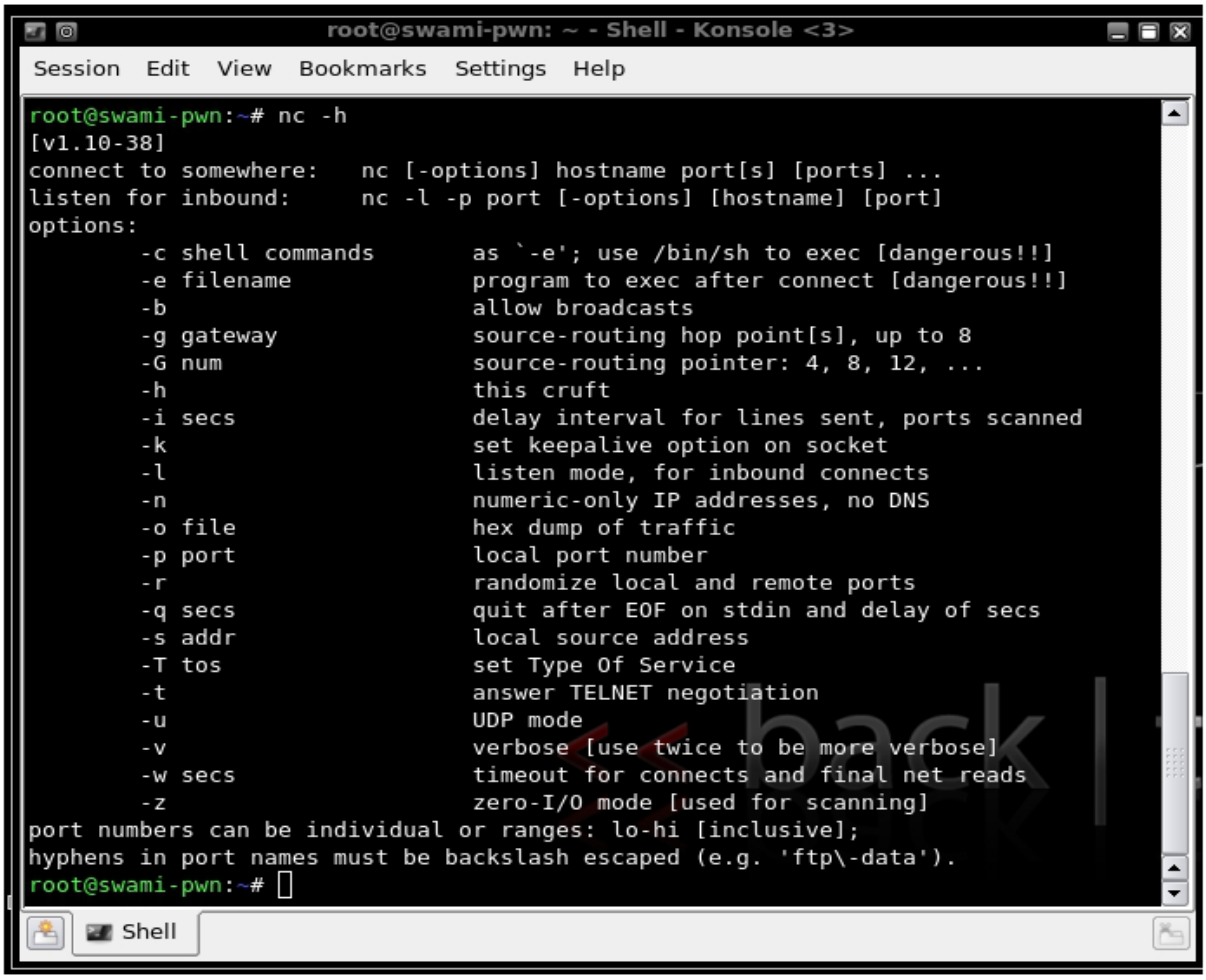
***

La versione 1.10 di netcat è installata di default in Bactrack4_PreFinal mentre la versione gnu netcat no. In questa breve guida andremo ad esaminare prima la versione 1.10 per poi passare alla versione riscritta gnu che aggiunge alcune funzionalità.

Linux(gnu-netcat): http://netcat.sourceforge.net/download.php
Windows: http://joncraton.org/files/nc111nt.zip

Prima di iniziare a descrivere le sue molteplici funzioni diamo un'occhiata al comando help di netcat, cosi da avere un'idea di cos'è in grado di fare. Digitaimo nella nostra shell il comando **nc -h.**

Il risultato dovrebbe essere il seguente:



***

## File transfer

È molto semplice inviare un file da e verso un host con netcat. Per prima cosa dobbiamo mettere in ascolto il destinatario su una porta non occupata da un'altro servizio, poi andiamo ad indicare il nome ed l'estensione del file che andremo a salvare (il nome può essere diverso da quello del mittente ma l'estensione no. Solo sotto windows ). Per finire dobbiamo settare il timeout cioè per quanto tempo proverà a connettersi all'indirizzo ip tramite il protocollo TCP.

Opzioni da usare:
-l = mette in ascolto per un'eventuale connessione.
-p = porta sulla quale si andrà in ascolto.
-v = quando inizia la connessione invia piccole informazioni (usato due volte per più informazioni).
-n = non userà i dns per tradurre l'indirizzo ip.
-w = timeout di connessione espresso in secondi.

Ora passiamo all'azione, per far si che il trasferimento funzioni Topolino (destinatario) deve mettersi in ascolto prima che Paperino (mittente) invii il file.

**Topolino: nc -lvp [numero_porta] -w [secondi] > [file] (es. # nc -lvp 5000 -w 5 > file.txt)**

Una volta creato il file possiamo inviarlo.

**Paperino: nc -vvn [indirizzo_ip_destinatario] [numero_porta] < [file] (es. # nc -vvn 192.168.2.8 5000 < testo.txt )**

Se tutto è andato per il verso giusto al termine della connessione il mittente dovrebbe ricevere il seguente messaggio che sta ad indicare il numero totale dei byte inviati.

**sent [dimensione_in_byte], rcvd 0**

Un'altra interessante caratteristica è che se non siamo sicuri quale sia la porta destinataria possiamo inviare il file in un range di porte, per esempio dalla 5000 alla 5050, basta solamente scrivere al posto di **5000** il nuovo range cioè **5000-5050**. Con questa opzione netcat proverà a connettersi dalla porta 5000 alla 5050 del destinatario fino a quando non troverà quella effettivamente in ascolto.

NB: Tutte queste informazioni passano in chiaro attraverso la rete.

***

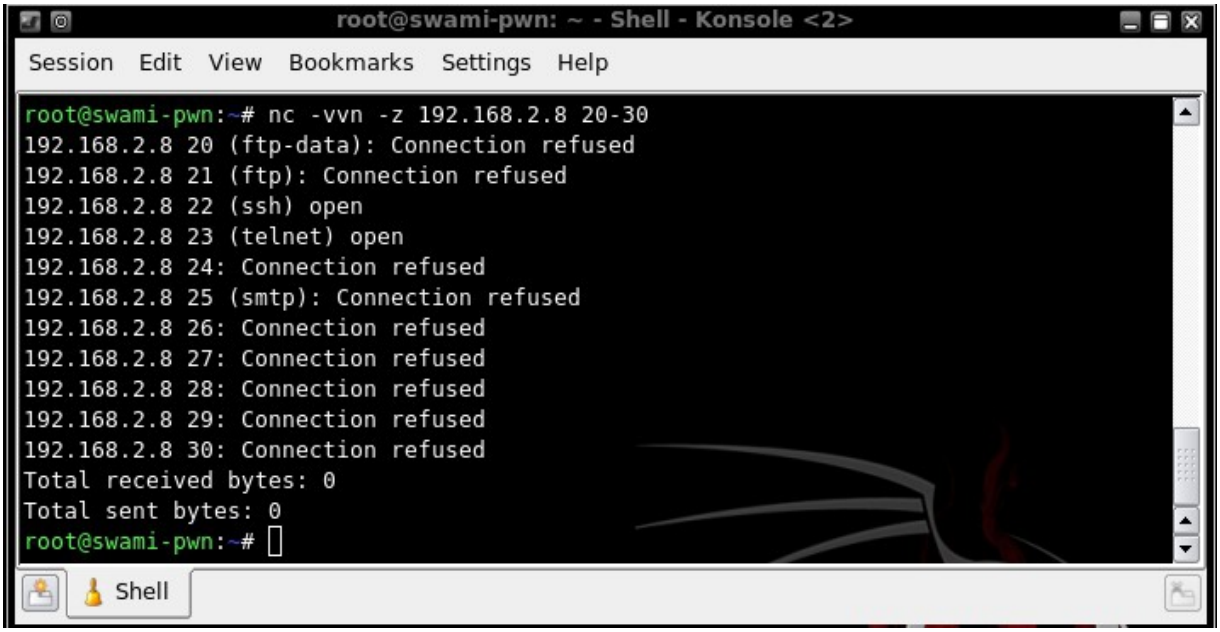
## Portscanning

Come detto all'inizio della guida con netcat possiamo utilizzare anche questa funzionalità per determinare quali/e porte siano aperte/chiuse e quale servizio opera su di esse/a in un determinato host. Per fare ciò basta utilizzare l'opzione **-z** (zero-I/O mode), la quale semplicemente invia una pacchetto TCP composto solamente dal flag SYN settato a uno. Se la porta presa in esame è aperta netcat completerà il three-way handshake altrimeti riceverà un pacchetto TCP con il flag RST cioè rifiuterà il tentativo di connessione ( porta chiusa ).

Opzioni da usare:
-v = quando inizia la connessione ti invia piccole informazioni (usato due volte per più informazioni)
-n = non traduce l'ip
-z = effettua "I/O" sulle porte tramite TCP

Ora vediamo un esempio pratico.

**Mittente: nc -vvn -z [indirizzo_ip_destinatario] [range_porte] (es. # nc -vvn -z 192.168.2.8 20-30)**




Il testo tra parentesi indica quale processo è in ascolto su quella porta o altrimenti quale servizio netcat si aspetta che ci sia secondo lo standart IANA .

Se invece vogliamo fare uno scan delle porte tramite protocollo UDP possiamo utilizzare la stessa sintassi con solo l'aggiunta dell'opzione **-u**.

Possiamo aggiungere l'opzione **-r** per far si che la scansione delle porte venga effettuta in maniera random.

***

## Banner Grabbing

*I* browser e i web server comunicano tramite il protocollo HTTP mediante richieste e risposte. Queste richieste sono formate da un metodo ed altri parametri necessari a compiere la richiesta. Il protocollo HTTP permette diversi metodi:

1- GET = richiede al server l'invio della risorsa indicata.
2- POST = invia delle informazione al server racchiuse nel body del pacchetto.
3- HEAD = analogo a GET solo che restituisce l'header della risposta HTTP.
…

Banner grabbing o “agguantare il banner” è una tecnica di enumerazione che ci permette di identificare i servizi attivi in una determinata porta . L'identificazione in questo caso avviene mediante la richiesta dell'header http ( metodo HEAD ) e il successivo invio da parte del server dell'header http di risposta nel quale sono contenute delle informazioni basilari (es. server montato, versione server e protocollo http utilizzato, sistema operativo utilizzato ecc...).
Se ora stiamo usando backtrack possiamo provare ad identificare il nostro server web apache installato di default, altrimenti possiamo prendere come riferimento un qualsiasi server web (es. www.backtrack.it ).

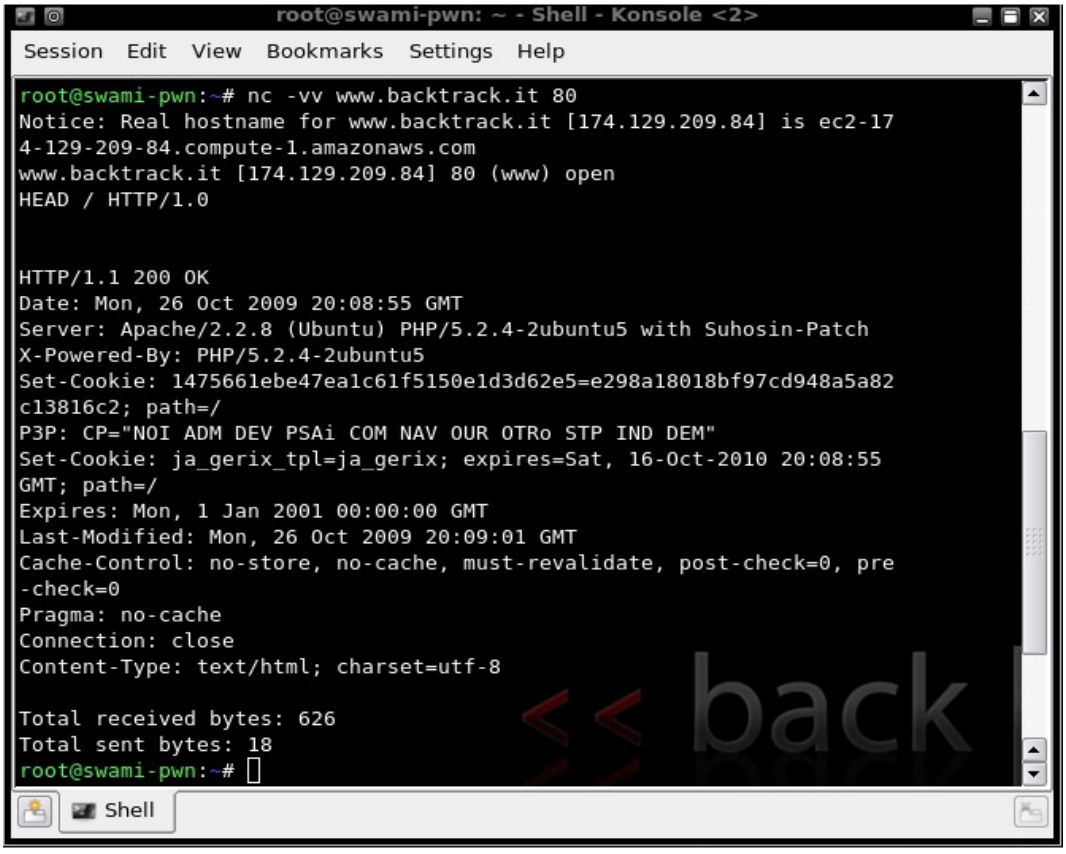
Opzioni che andremo ad usare:
-v = quando inizia la connessione invia piccole informazioni (usato due volte per più informazioni)

Usando netcat possiamo effettuare una connessione usando il protocollo TCP verso la porta 80 di un qualsiasi server web, dopo aver fatto ciò chiediamo al server di inviarci il suo header http attraverso il comando **HEAD / HTTP/1.0**. Una volta inviata la richiesta lui ci invierà l'header http di risposta.

Ora mettiamo in pratica quello che abbiamo appena detto:

Se stiamo usando backtrack diamo il seguente comando da shell **# /etc/init.d/apache2 start** (possiamo controllare che il web server sia effettivamente avviato attraverso il comando **# netstat -ant | grep 80** ) per avviare apache, ora con netcat diamo il comando **# nc -vv <indirizzo_ip> <numero_porta>** (es. **# nc -vv localhost 80** ) **.**

Se invece non abbiamo un web server installato basta che inviamo la richiesta ad un qualsiasi server web (es. **# nc -vv www.backtrack.it 80**), dopo aver effettuato la richiesta vi dovrebbe dire che la porta 80 (http) è aperta. Eseguiamo la richiesta dell'header http con il comando **# HEAD / HTTP/1.0** e premiamo due volte invio.
Il server web risponderà alla richiesta inviando l'header di risposta che dovrebbe essere “simile” alla mia:



Come già detto in precedenza con questo metodo si riescono a scoprire le informazioni basilari su un server web come:

– Sistema Operativo
– Tipo di web server
– Versione del web server
– Versione del modulo php

Non fidatevi di tutti i banner che trovate perchè gli amministratori più furbi camuffano i banner cosi da rendere più difficile l'identificazione.

***

## Remote Administration (Bind Shell)

Una della più importanti caratteristiche di Netcat è quella di poter prendere un file eseguibile .exe e reindirizzarlo sulla porta desiderata, una volta fatto ciò un host sapendo su che porta è stato reindirizzato il file .exe può eseguire sulla propria macchina il file in questione.
Facciamo un esempio tra Paperino (server windows che indirizza il file cmd.exe) e Topolino (client linux che "eseguirà" il file .exe).

Opzioni che andremo ad usare:
-l = mette in ascolto per un'eventuale connessione
-v = quando inizia la connessione invia piccole informazioni (usato due volte per più informazioni )
-p = porta settata su listen
-e = programma da eseguire dopo la connessione

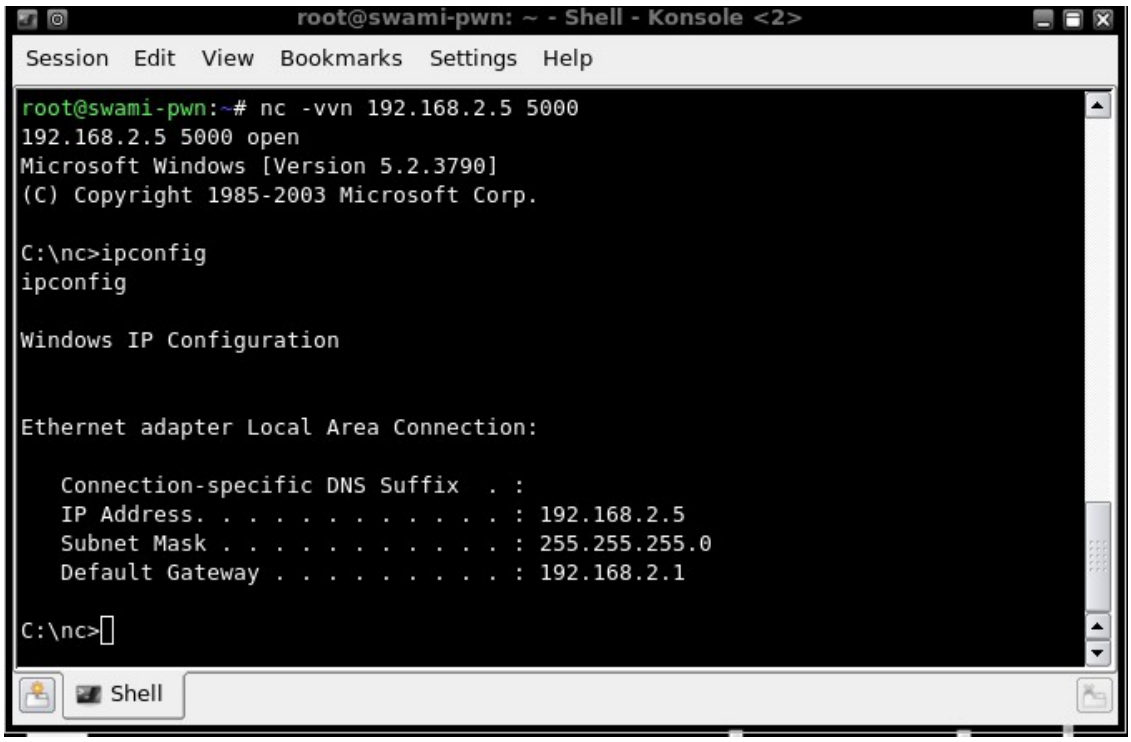
Paperino è il nostro bel server che vuole essere amministrato da remoto, e per poterlo fare dobbiamo indirizzare il prompt dei comandi su una porta.

**Paperino:** **#nc -lp <numero_porta> -e <programma_da_eseguire>** (es. # **nc -lp 5000 -e cmd.exe** )

Una volta saputa la porta Topolino non deve far altro che connettersi a quella porta.

**Topolino:** **# nc -vvn <indirizzo_ip> <numero_porta>** (es. # **nc -vvn 192.168.2.5 5000** )

Ora Topolino è in "possesso" del prompt dei comandi di Paperino, quindi ha piena libertà di movimento. Se Paperino al suo comando aggiunge l'opzione **-L** netcat rimarrà in ascolto su quella porta anche dopo la disconnessione di Topolino.




NB: Tutte queste informazioni passano in chiaro attraverso la rete.

***

## Reverse shell

*R*everse shell è la tecnica "inversa" di quella vista prima, cioè questa volta Topolino (linux) ha bisogno di essere amministrato da remoto e quindi per far ciò dobbiamo usare più o meno la stessa tecnica dell'amministrazione remota.

Opzioni che andremo ad usare:
-l = mette in ascolto per un'eventuale connessione
-p = porta settata su listen
-e = programma da eseguire dopo la connessione
-v = quando inizia la connessione invia piccole informazioni (usato due volte per più informazioni)

Paperino è un server windows quindi non deve far altro che restare in ascolto su una porta concordata fra i due.

**Paperino: # nc -l -p <numero_porta>** (es. **# nc -l -p 5000** )

Topolino è un client linux quindi deve solo inviare la shell (bash) testuale di linux.

**Topolino: # nc -v <indirizzo_ip> <numero_porta> -c <shell_path>** (es. **# nc -v 192.168.2.8 5000 -e /bin/bash** )

Ora Paperino usando i comandi appositi della shell può amministrare il computer di Topolino.

NB: Tutte queste informazioni passano in chiaro attraverso la rete.

***

## Capturing network Traffic

Grazie alle opzioni -x e -o possiamo loggare il traffico che sta transitando in una determinata porta, per esempio supponiamo che Paperino sia in ascolto nella porta 80 e Topolino tramite il suo browser faccia una richiesta http. Una volta eseguita la richiesta potremmo visualizzarla nel file nel quale abbiamo loggato il traffico.

Opzioni che andremo ad usare:
-l = mette in ascolto per un'eventuale connessione.
-p = porta settata su listen.
-x = mostra l'hexdump dei dati in entrata e uscita.
-o = scrive l'output del traffico sul FILE (implica -x).

Esempio:

**Paperino # nc -lp <porta> -x -o <output_file> (es. nc -lp 80 -x -o request )**

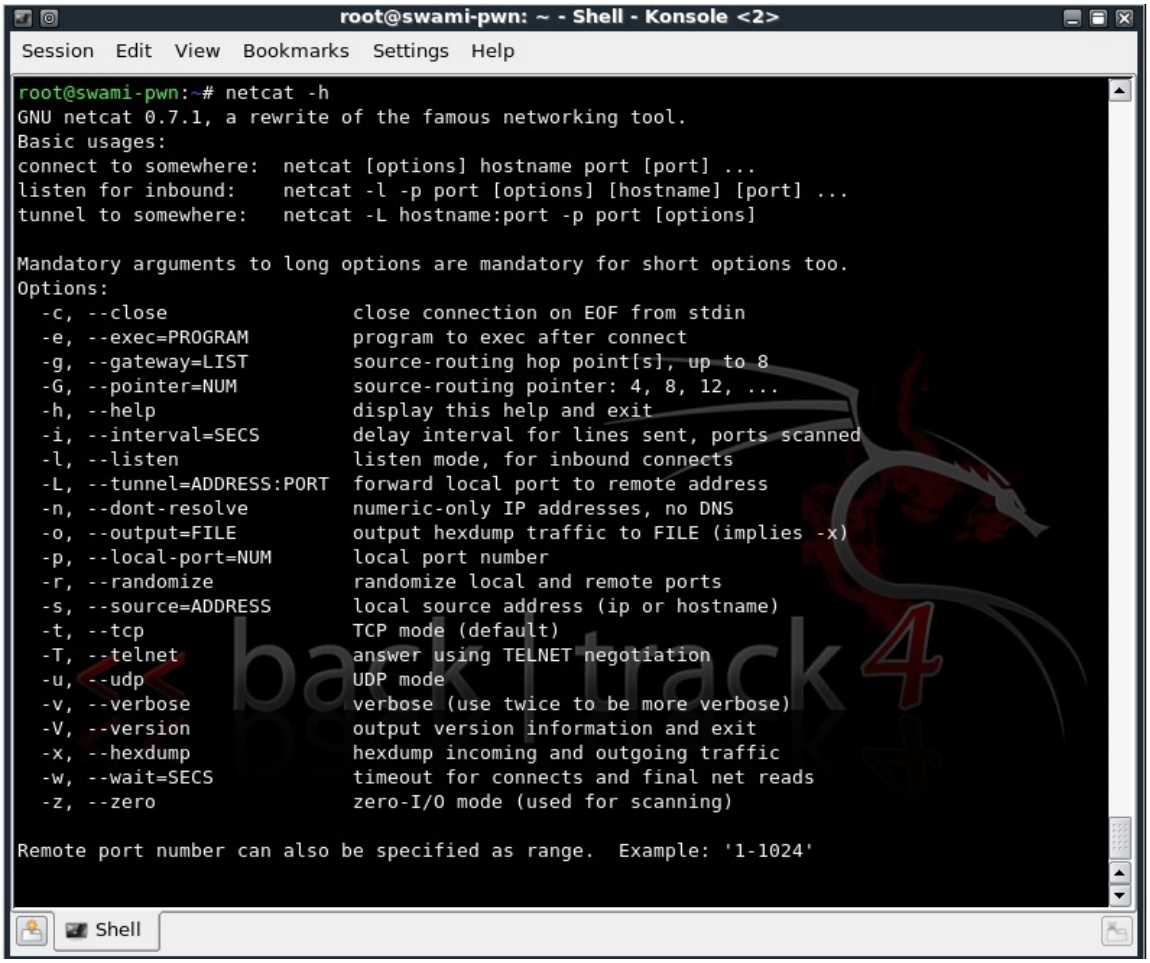
Una volta che Topolino ha fatto la richiesta potremmo visualizzarla nel file request.

Altre opzioni:
-g , --gateway=LIST source-routing hop point[s], fino a 8
-G, --pointer=NUM source-routing pointer: 4, 8, 12, ...
-V, --version mostra informazioni sulla versione ed esce
-T, imposta il tipo di servizio ( può essere uno tra “Minimize-delay”, “Maximize-Throughput”, “Maximize-Reliability”, o “Minimize-Cost”)
-b = Permette udp broadcast
-q = dopo che EOF è stato identificato , aspetta un specifico numero di secondi e poi esce

***

## GNU Netcat

La versione Gnu di netcat è sostanzialmente una riscrittura dell'omonimo con l'aggiunta di alcune feature, come la possibilità di reidirezione delle porte. Questa versione non è presente di default in Backtrack4_PreFinal quindi se la vogliamo utilizzare dobbiamo scaricarla dal link che ho idicato all'inizio.




***

## Port redirection

Un esempio di redirezione delle porte potrebbe essere il seguente; supponiamo che Topolino e Paperino siano nella stessa sottorete e che le policy del firewall non permettano a Topolino di connettersi ad internet mentre a Paperino si. Ora Paperino grazie all'opzione **-L** di netcat può far si che Topolino tramite lui si colleghi ad internet anche attraverso una porta differente dall'80.

Opzioni che andremo ad usare:
-v = quando inizia la connessione invia piccole informazioni (usato due volte per più informazioni)
-p = porta settata su listen
-L = inoltra una porta ad un indirizzo remoto

Esempio: Supponiamo che Topolino voglia connetersi al sito www.backtrack.it

**Paperino # nc -vv -L <host name>:<porta> -p <porta su cui reidirigere> (es. nc -vv -L www.backtrack.it:80 -p 3333 )**

Ora Topolino digitando sul suo browser l'indirizzo ip di Paperino e la porta 3333 può connetersi al sito backtrack.it, ma potete notare che dopo aver eseguito la richiesta netcat termina. Se volgiamo che netcat rimanga in esecuzione per più di una richiesta possiamo scrivere un semplice script che ci permette di far ciò.

```
while [[ 1 ]]; do
  nc -L www.baktrack.it:80 -p 3333
done
```

Finchè è vero eseguo le istruzioni dentro al while.

Se Paperino esegue questo script netcat rimarrà in esecuzione fino a quando non lo si interromperà con il comando ctrl+z.

Altre opzioni:
-c = chiudi la connessione all'EOF da standart input

***

## Cryptcat

Cryptcat è una speciale versione di netcat che ci permette di effettuare tutte le operazioni che ho elencato sopra in modo sicuro cioè cifrando i dati che inviamo in rete attraverso l'algoritmo twofish.

Link:
–Linux, Windows: http://sourceforge.net/projects/cryptcat/files/

Questa guida esplica una parte dei mille usi di netcat, quindi vi invito a “smanettare” un po' con questo software e vedrete che ne sarete soddisfatti.

# www.backtrack.it

***

***Questo documento è da ritenersi esclusivamente per scopi informativi / didattici, l' autore del testo e coloro che lo ospitano sul proprio spazio non sono responsabili delle azioni commesse da terze parti.***

***