Arpscan

Progetto gestione di rete
AA 2018-2019

Biancucci Francesco 545063

1 Introduzione

Il programma arpscan crea dei pacchetti ARP che vengono inviati a tutti gli ip della sottorete al fine di individuare i dispositivi connessi alla rete e verificare se tali dispositivi restano connessi o no. Inoltre se l'utente lo desidera il programma è in grado di associare un nome simbolico ad ogni dispositivo in base alla sua scheda di rete. Il software è stato sviluppato in c ed utilizza la libreria libpcap per inviare e ottenere i pacchetti. Sono state definite due librerie per l'implementazione delle hashtable in cui salvare i dati e una libreria in cui sono definite costanti e macro di controllo.

2 Funzionamento

Con l'opzione -h è possibile visualizzare un helper che specifica le varie opzioni utilizzabili. Gli altri parametri sono:

- -i è un parametro obbligatorio che definisce l'interfaccia di rete del nostro dispositivo dalla quale vogliamo inviare e ricevere pacchetti
- [-t] Timeout tra una scansione e un'altra (di default a 30 secondi)
- [-n] Numero di scansioni da effettuare (di default interrompibili con Ctrl+C)
- [-v] Verbose, associa il nome del costruttore alla scheda

Quindi una volta avviato il programma controlla la validità dell'interfaccia e delle altre opzioni se presenti, nel caso in cui sia presente l'opzione -v legge il file con le associazioni mac-nome e salva tali associazioni in un hashtable dove il mac è la chiave dopodichè inizia a forgiare ed inviare i pacchetti ARP per poi attendere la risposta che viene filtrata mediante un apposito filtro impostato in precedenza. Nel caso in cui ci sia una risposta l'ip e il mac del dispositivo vengono salvati in un hashtable insieme a dei flag che servono per capire se il dispositivo viene visto per la prima volta, se -v abilitato viene salvato il nome e se invece non si tratta della prima scansione e un dispositivo non risponde quel dispositivo viene marcato per essere eliminato nel caso in cui non risponda neanche alla scansione successiva.

3 Testing e indicazioni

Il software è stato sviluppato e testato su ubuntu 19.04 e oltre all'utilizzo di libpcap nel software per il testing si è utilizzato wireshark per verificare la ricezione corretta dei pacchetti. Per avviare il software dopo aver eseguito il comando make (si richiede che sia installato gcc): **sudo ./arpscan -i interfaccia [-t timeout] [-n numero loop] [-v]**

4 Bug noti

Il software nel caso di dispositivi wirless connessi alla rete come ad esempio cellulari non ne assicura la scoperta al primo loop nonostante l'invio di due pacchetti ARP ad ogni loop.

```
francesco@francesco-GL753VD:~/Scrivania/UNI/gestioneprog$ make
gcc -I.  -c -o arpscan.o arpscan.c
gcc -I.  -c -o hasht.o hasht.c
gcc -I.  -c -o hasht_DB.o hasht_DB.c
ar rvs lib.a hasht.o hasht_DB.o
ar: creazione di lib.a
a - hasht.o
a - hasht_DB.o
gcc  -I.  -L. -o arpscan arpscan.o lib.a hasht.h hasht_DB.h -pthread -lpcap -lm
francesco@francesco-GL753VD:~/Scrivania/UNI/gestioneprog$ sudo ./arpscan -i enp3s0 -t 10
[sudo] password di francesco:
Starting arpscan for 192.168.1.0/24
78:02:F8:F6:17:04 at 192.168.1.103 First time seen
84:26:15:33:24:B2 at 192.168.1.254 First time seen
B0:6E:BF:09:98:79 at 192.168.1.101 First time seen

done (3 host alive) scanned: 256 host

78:02:F8:F6:17:04 at 192.168.1.103 Alive
84:26:15:33:24:B2 at 192.168.1.254 Alive
B0:6E:BF:09:98:79 at 192.168.1.101 Alive

done (3 host alive) scanned: 256 host

^Cfrancesco@francesco-GL753VD:~/Scrivania/UNI/gestioneprog$ sudo ./arpscan -i enp3s0 -t 10 -n 4 -v
Starting arpscan for 192.168.1.0/24
78:02:F8:F6:17:04 XiaomiCo at 192.168.1.103 First time seen
84:26:15:33:24:B2 AdbBroad at 192.168.1.254 First time seen
B0:6E:BF:09:98:79 AsustekC at 192.168.1.101 First time seen

done (3 host alive) scanned: 256 host

78:02:F8:F6:17:04 XiaomiCo at 192.168.1.103 No response
84:26:15:33:24:B2 AdbBroad at 192.168.1.254 Alive
B0:6E:BF:09:98:79 AsustekC at 192.168.1.101 Alive

done (2 host alive) scanned: 256 host

78:02:F8:F6:17:04 XiaomiCo at 192.168.1.103 No response, going to remove
84:26:15:33:24:B2 AdbBroad at 192.168.1.254 Alive
B0:6E:BF:09:98:79 AsustekC at 192.168.1.101 Alive

done (2 host alive) scanned: 256 host

84:26:15:33:24:B2 AdbBroad at 192.168.1.254 Alive
B0:6E:BF:09:98:79 AsustekC at 192.168.1.101 Alive

done (2 host alive) scanned: 256 host
```