Backfire-Monitor

Analisi e Filtraggio Tramite Tecniche di ARP-Spoofing

Gestione di Rete 2017/2018,
Samuele Sabella

Mentre le tecniche tradizionali di monitoraggio e filtraggio del traffico di rete prevedono l'adozione di più router e l'introduzione di devices che si frappongano tra questi, *Backfire-Monitor* consente ad un qualsiasi host di filtrare e analizzare i pacchetti in uscita dalla rete. Per far ciò viene sfruttata la mancanza di autenticazione nel protocollo ARP, utilizzato per la traduzione di indirizzi IP in indirizzi di livello Collegamento, per eseguire un attacco di ARP poisoning e dirottare il traffico verso il dispositivo utilizzato per il monitoraggio. Tramite questo approccio è quindi possibile, anche se con alcune limitazioni, evitare l'acquisto di apparecchiature specifiche con costo molto elevato [1].

1 Guida all'utilizzo

Come prima funzionalità il programma permette di filtrare il traffico della rete tramite apposita politica di sicurezza, specificata dal file *"policy.txt"*, in cui ciascuna riga costituisce un filtro [1].
Tutto il traffico che non rispetti almeno uno dei filtri viene bloccato mentre i restanti pacchetti vengono analizzati per poi essere reinoltrati al router/switch. *Backfire-Monitor* rimane quindi trasparente all'utente finale, testimone solamente del rallentamento della rete.
Un esempio di politica di sicurezza che blocchi tutto il traffico non diretto verso *Facebook* o *StackExchange* è il seguente:

```
dns
dst host facebook.com
dst host stackexchange.com
```

Per quanto riguarda il monitoraggio, l'implementazione attuale consente solo la scoperta degli host presenti nella rete e la percentuale di pacchetti che questi inviano [2], tuttavia è possibile registrare altre informazioni estendendo la funzione *Analyze* definita nel file *"analyzer.h"*. Questa viene chiamata ogniqualvolta un pacchetto ricevuto viene ammesso dalla politica di sicurezza e prende come argomenti il pacchetto catturato e la sua lunghezza in Byte.
Una volta analizzato, il pacchetto viene poi reinoltrato al router con indirizzo mittente impostato a *broadcast*. In questo modo si evita che il router associ l'indirizzo MAC

[1] Per dettagli sul formato dei filtri visitare https://www.tcpdump.org/manpages/pcap-filter.7.html
[2] Percentuale calcolata in base al numero totale di pacchetti inoltrati dal programma al router

dell'host originale all'interfaccia tramite cui la macchina con *Backfire-Monitor* in esecuzione è connessa, consentendo quindi al mittente di comunicare correttamente con il destinatario [2].
Una volta compilato tramite il comando *make*, per essere avviato correttamente *Backfire-Monitor* ha bisogno delle seguenti informazioni:

- Indirizzo mac del gateway a cui reinoltrare i pacchetti catturati, specificato dall'opzione obbligatoria *-a*.
- L'interfaccia di rete da utilizzare per catturare e inoltrare i pacchetti, specificata tramite l'opzione obbligatoria *-d*.
- L'indirizzo IP del host target dello spoofing, con valore di default *192.168.1.1*, configurabile tramite l'opzione *-i*.

Nell'esempio sotto riportato il programma è stato avviato da una macchina con sistema operativo *MacOS* per analizzare il traffico di una rete domestica con indirizzo MAC del router *10:13:31:c3:b2:2*.

```
$ sudo ./main.o -d en0 -a 10:13:31:c3:b2:2

  Tot packets: 196
  Device:      MAC/D0:81:7A:C0:3F:F4   IP/192.168.1.156   %(0.9592)
  Device:      MAC/E8:B4:C8:FB:7D:BD   IP/192.168.1.106   %(0.0357)
```

2 Implementazione

Per l'implementazione sono state utilizze due librerie: *"libpcap"* per la cattura e l'invio di pacchetti; la libreria di terze parti *"uthash"* per memorizzare e gestire le informazioni degli host nella rete.
Il diagramma di stato sotto riportato mostra il funzionamento generale del programma.

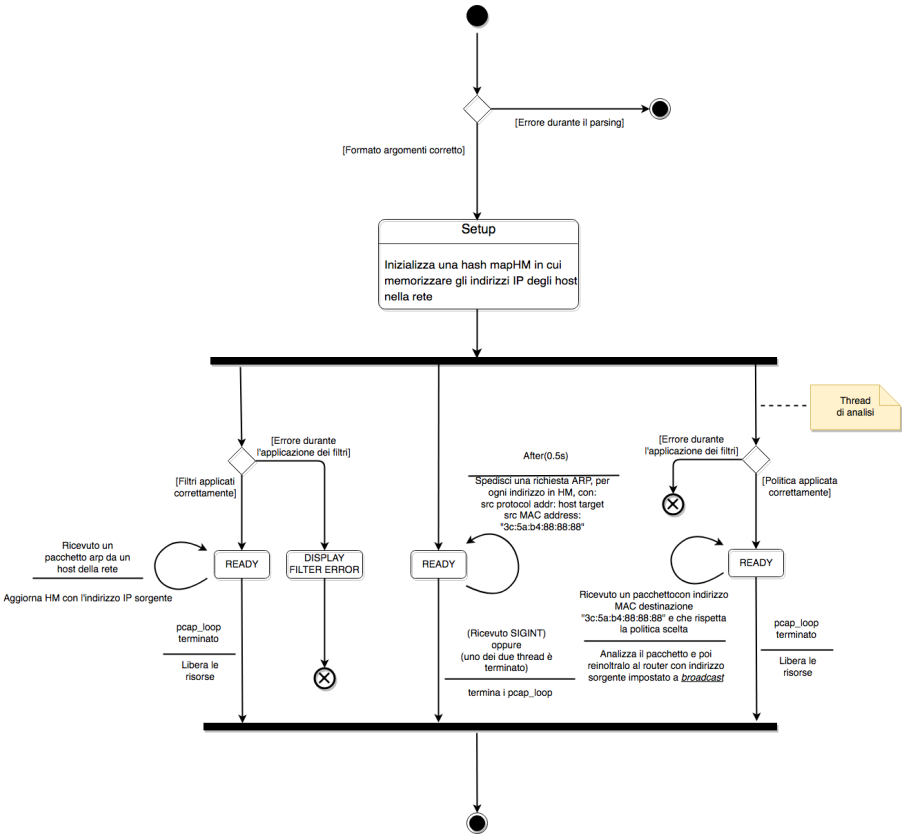

Fig.1

L'applicazione sfrutta tre thread. Il primo che filtra i messaggi ARP dei device nella LAN per aggiornare una hash map con gli indirizzi IP sorgente dei mittenti. Il secondo che ogni 0.5 secondi genera un messaggio di richiesta ARP per ogni indirizzo presente nella hash table. Supponendo che l'indirizzo obiettivo dello spoofing sia *192.168.1.1*, la cache del dispositivo con indirizzo *192.168.1.7* viene avvelenata utilizzando messaggi di richiesta così definiti:

SPOOFED ARP REQUEST		
HTYPE: 0x0001 (Ethernet)		PTYPE: 0x0800 (IPv4)
HLEN: 6	PLEN: 4	OPER: 0x0001 (request)
Source Hardware Address: 3c:5a:b4:88:88:88		
Source Protocol Address:192.168.1.1		
Target Hardware Address: ff:ff:ff:ff:ff:ff (broadcast)		
Target Protocol Address:192.168.1.7		

Fig.2

Una volta che le cache dei dispositivi sono state avvelenate creando l'associazione tra l'indirizzo target dello spoofing e l'indirizzo MAC *"3c:5a:b4:88:88:88"*, i dispositivi della rete inoltrearanno i pacchetti a questo indirizzo. Il terzo thread intercetta questi pacchetti, li analizza per poi reinoltrarli al router. L'utilizzo di un indirizzo MAC fittizio permette di distinguere il traffico diretto verso il dispositivo sorgente dell'attacco dal traffico dei dispositivi la cui cache è stata infettata con successo.

References

[1] Bitbridge: How does firewalla intercept traffic. https://www.firewalla.com/bitbridge-firewalla-intercept-traffic/.

[2] Behrouz A. Forouzan e Firouz Mosharraf. Computer networking: a top-down approach. page 316, 2000.