I protocolli Ethernet e ARP

Obiettivo

Studio dei protocolli di rete Ethernet e ARP.

Procedura 1

- Avviare Wireshark e iniziare la fase di monitoraggio della rete;
- Avviare il browser e selezionare l'indirizzo: http://www.unical.it/portale/
- Arrestare il monitoraggio della rete.

Dalla schermata di Wireshark è possibile individuare il pacchetto di richiesta (GET) inviato dal vostro browser.

Domande:

Analizzando il pacchetto di richiesta (GET) inviato dal vostro Browser:

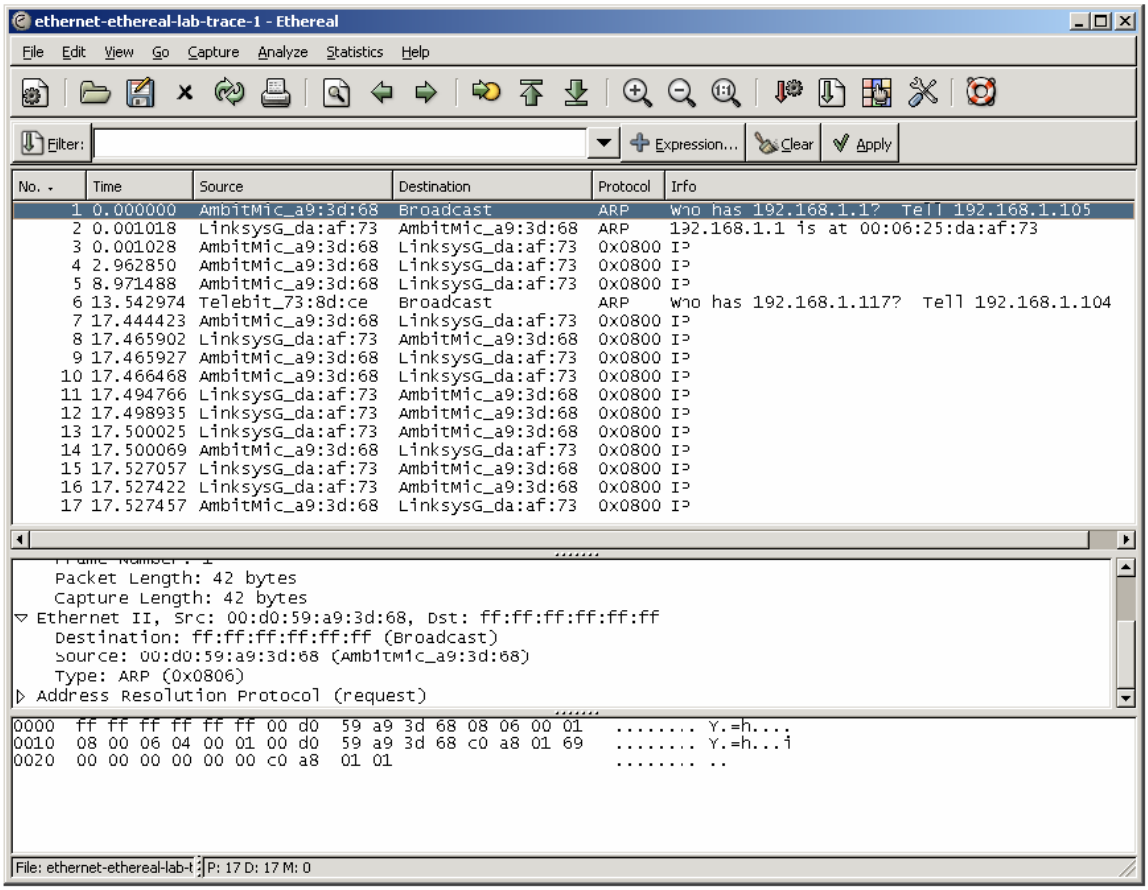
- Qual'è l'indirizzo Ethernet a 48 bit della vostra macchina?
- Qual'è l'indirizzo Ethernet a 48 bit della macchina destinazione? Questo indirizzo è il reale indirizzo della macchina destinazione? Motivare opportunamente la risposta.

Analizzando il pacchetto di risposta alla precedente richiesta:

- Qual'è l'indirizzo Ethernet a 48 bit della vostra macchina?
- Qual'è l'indirizzo Ethernet a 48 bit della macchina destinazione/sorgente? Questo indirizzo a quale macchina fa riferimento? Motivare opportunamente la risposta.

Procedura 2

- Pulire la cache di ARP (Windows: arp –d *; Linux: arp –d *);
- Avviare la fase di monitoraggio del traffico di rete con Wireshark;
- Chiedere al browser di caricare la pagina http://www.wintriks.it;
- Appena la pagina viene caricata, arrestare la procedura di acquisizione di Wireshark.



Domande:

- Individuare le richieste ARP inviate sulla rete!
- Quali sono i valori che identificano la sorgente e la destinazione dei pacchetti?
- Individuare i pacchetti di risposta (.. is at ..) e analizzare la loro struttura.
- Visualizzare la cache ARP (Windows: arp –a, Linux: arp)

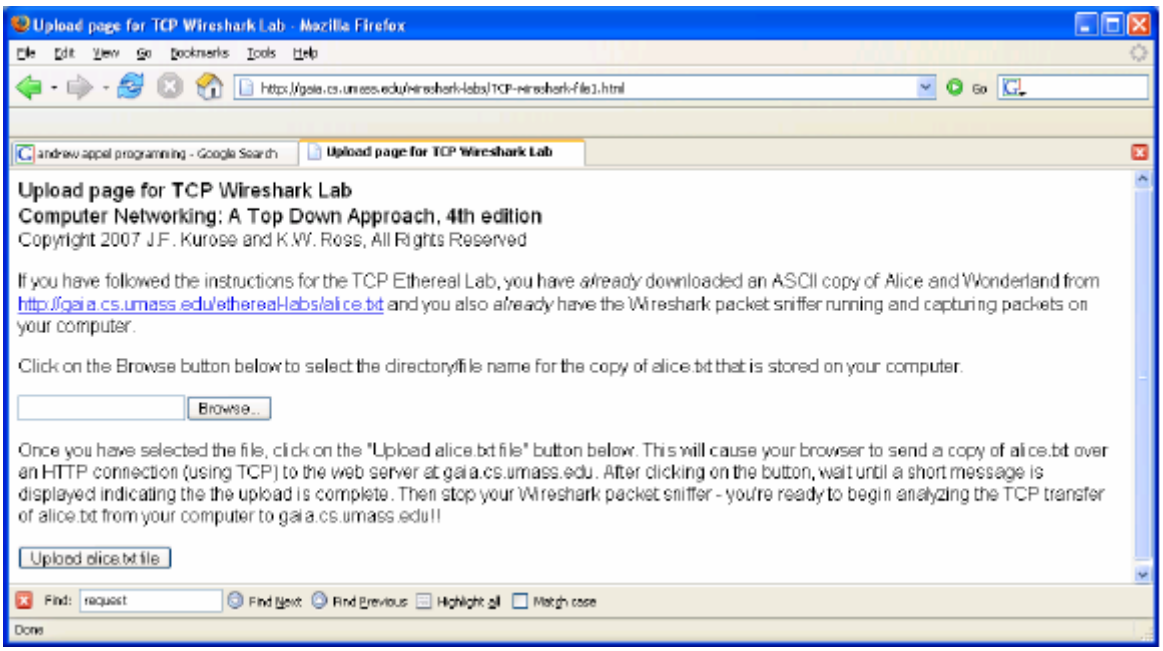
Il protocollo TCP

Obiettivo

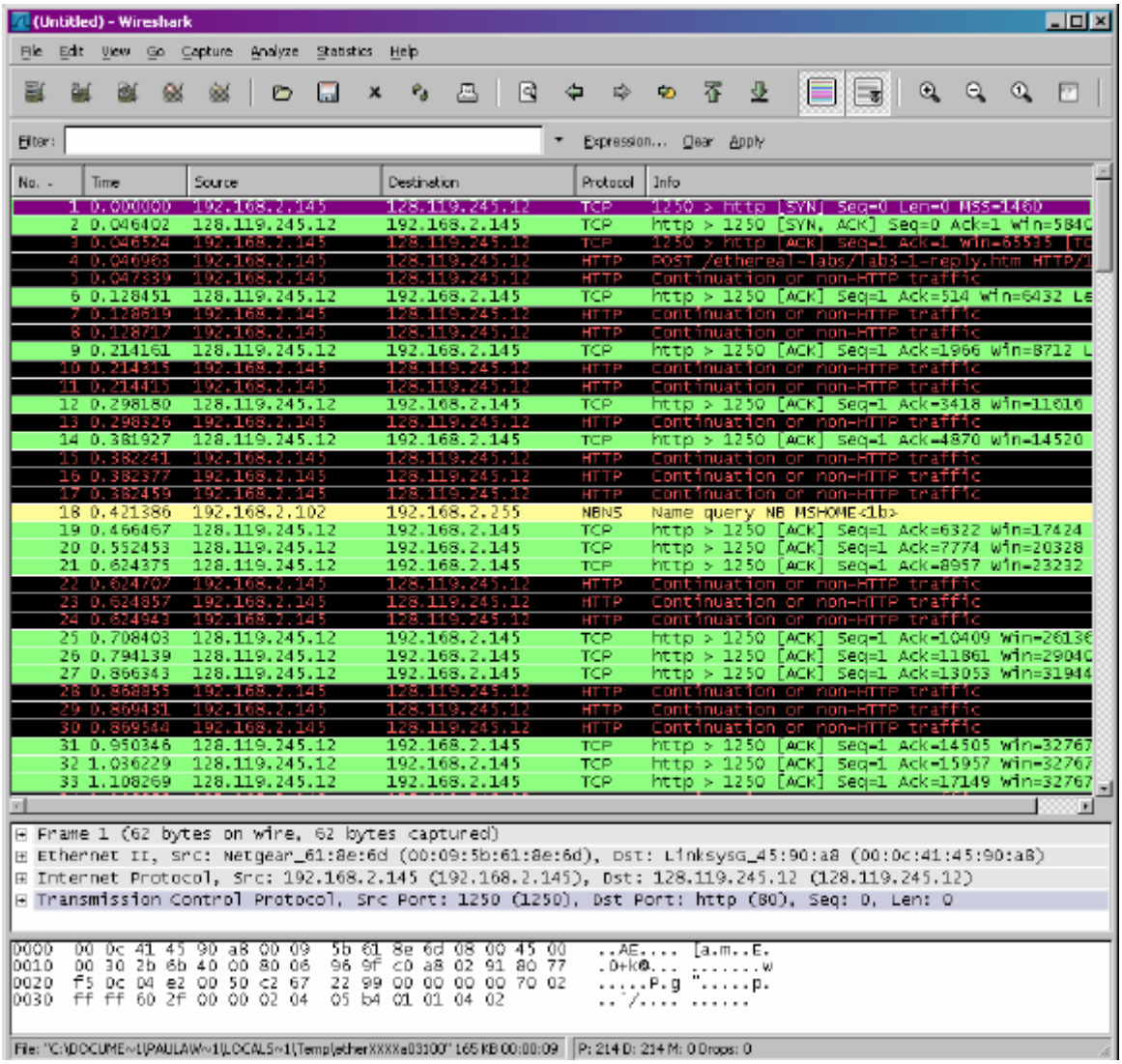
Studio del funzionamento del protocollo TCP, in particolare analisi della "traccia" di segmenti TCP scambiati tra il vostro computer ed un server remoto.

Procedura

- Aprire il web browser all'indirizzo http://gaia.cs.umass.edu/wireshark-labs/alice.txt, recuperare il file di testo *Alice in Wonderland* e memorizzarlo sull' hard disk.
- Andare su http://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html. Dovrebbe apparire una schermata di questo tipo:



- Cliccare su *Browse* per inserire il nome del file memorizzato sul vostro computer contenente il testo di *Alice in Wonderland*, ma senza cliccare su *Upload alice.txt* file.
- Adesso avviare Wireshark e iniziare ad ascoltare il traffico di rete.
- Ritornare al web browser e cliccare su *Upload alice.txt file* per caricare il file sul server *gaia.cs.umass.edu.server*. Una volta caricato il file, verrà visualizzato un breve messaggio di congratulazioni.
- A questo punto terminare la sessione di cattura di pacchetti su Wireshark. La finestra di Wireshark dovrebbe essere simile a quella riportata di seguito:



Se non vi è possibile catturare questi pacchetti con Wireshark allora potete scaricare un file di cattura dall'indirizzo https://www.mat.unical.it/informatica/Reti_di_Calcolatori?action=AttachFile&do=get&target=tcp-trace. Memorizzate questo file sul vostro computer e apritelo da Wireshark attraverso il comando *Open* del menu *File*.

Domande:

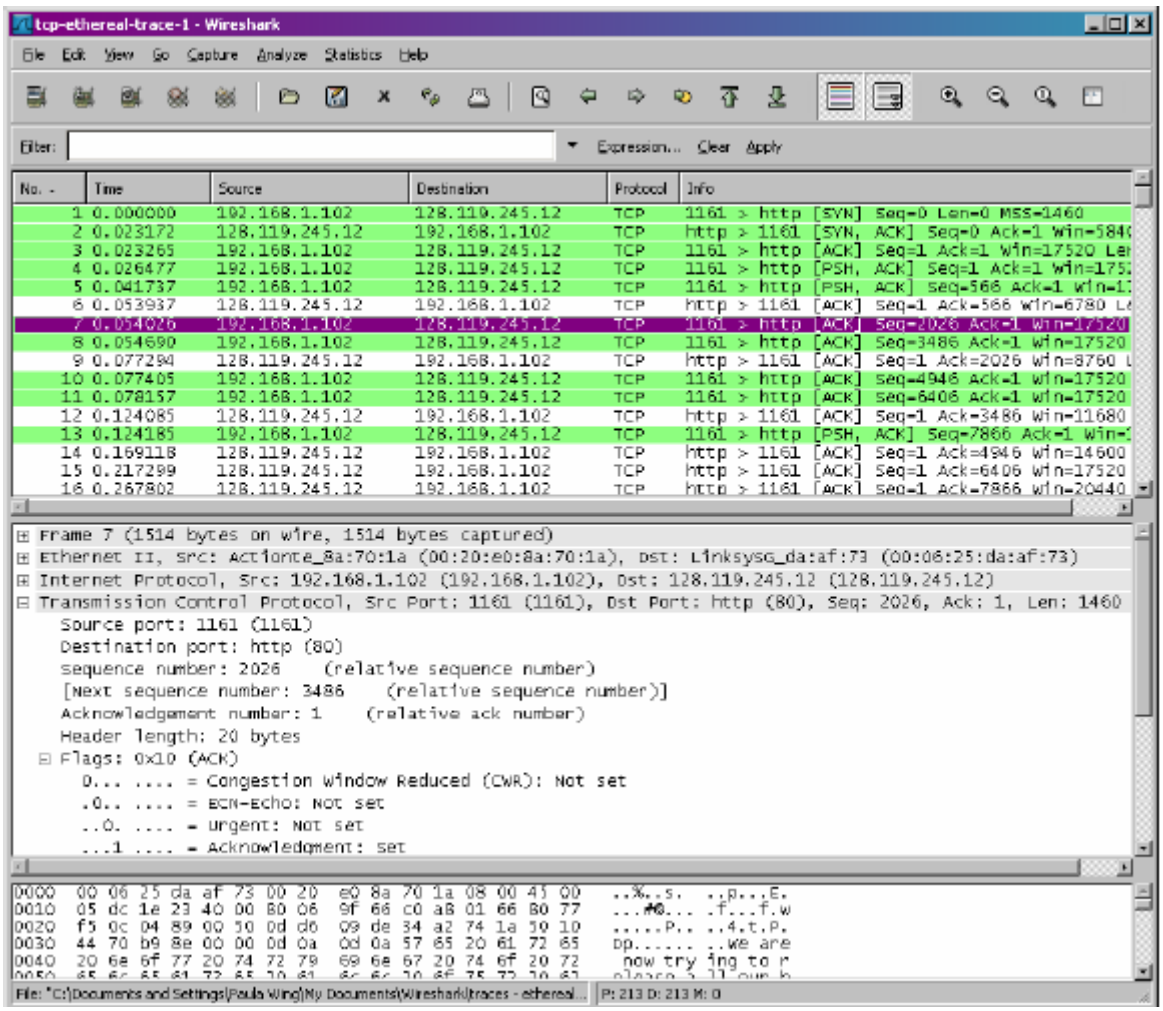
Filtrate i pacchetti visualizzati nella finestra di Wireshark inserendo la parola "tcp" nella casella testuale per la specifica del filtro. Ciò che dovreste vedere è una serie di messaggi TCP e HTTP scambiati tra il vostro computer ed il server *gaia.cs.umass.edu*.

- Quali sono l'indirizzo IP ed il numero di porta TCP utilizzati dal computer (sorgente) che sta trasferendo il file a *gaia.cs.umass.edu*?
  Qual è l'indirizzo IP di *gaia.cs.umass.edu*? Su quale numero di porta sta inviando e ricevendo segmenti TCP per questa connessione?

Se siete stati in grado di creare il vostro file di cattura, senza scaricarlo dal sito del corso rispondete a questa domanda:

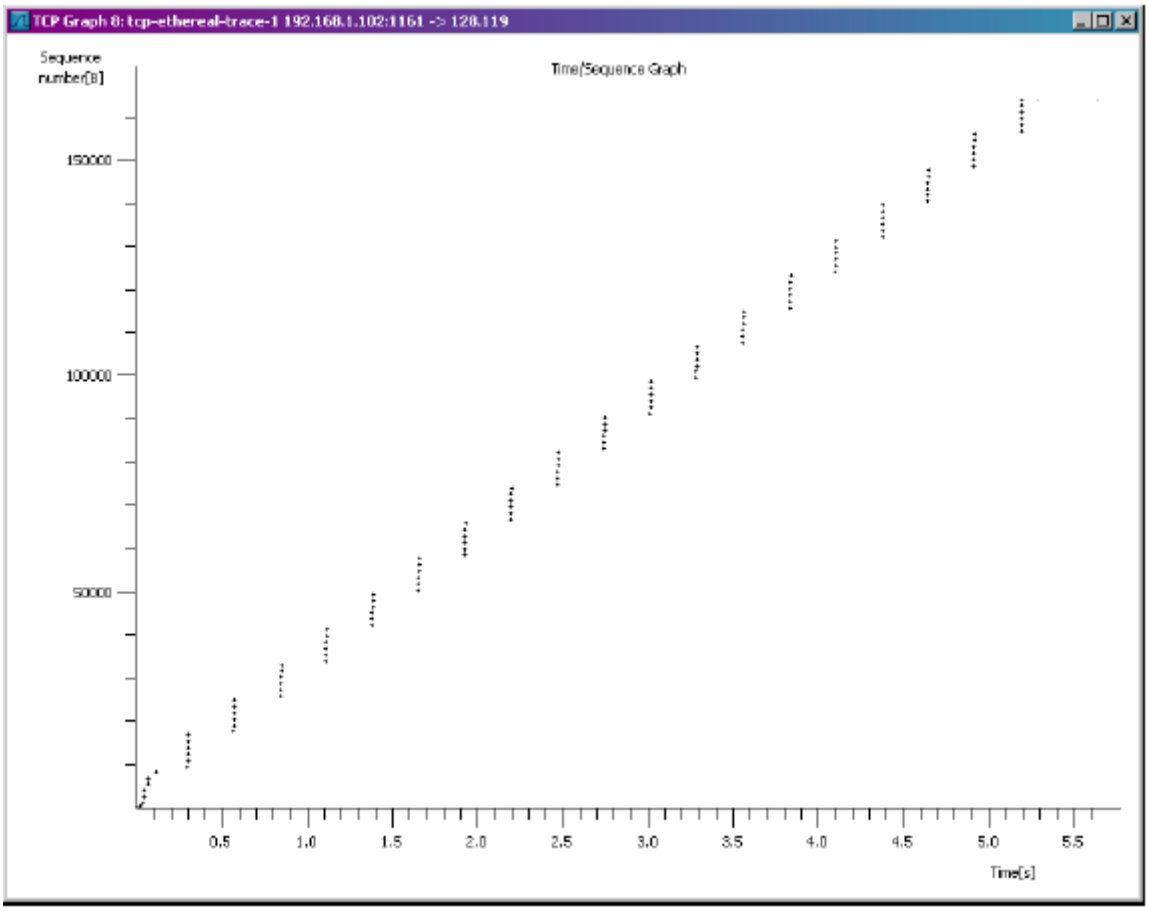
- Quali sono l'indirizzo IP ed il numero di porta TCP utilizzati dal vostro computer (sorgente) per trasferire il file a *gaia.cs.umass.edu*?

Adesso andate su Analyze -> Enabled Protocols e poi deselezionate il box HTTP e cliccate su OK; dovreste vedere una schermata simile a quella riportata di seguito:



- Qual è il numero di sequenza del segmento TCP SYN che è stato usato per iniziare la connessione TCP tra il computer client e *gaia.cs.umass.edu*? Cos'è che identifica il segmento come segmento SYN?
- Qual è il numero di sequenza del segmento SYN/ACK inviato da *gaia.cs.umass.edu* al computer client in risposta al SYN? Qual è il valore del campo ACKNOWLEDGEMENT NUMBER nel segmento SYN/ACK? Come ha determinato *gaia.cs.umass.edu* quel valore? Cos'è che identifica il segmento come segmento SYN/ACK?
- Riabilitate la visualizzazione del protocollo IP dal menu Analyze/Protocols. Qual è il numero di sequenza del segmento TCP contenente il comando HTTP POST?
- Considerate il segmento TCP contenente il comando HTTP POST come primo segmento della connessine TCP. Quali sono i numeri di sequenza dei primi sei segmenti della connessione TCP (incluso il segmento contenente il comando HTTP POST)? In che istante è stato inviato ciascun pacchetto? Quando è stato ricevuto l'ACK per ciascun segmento? Data la differenza tra quando ciascun segmento TCP è stato inviato e quando il suo ACK è stato ricevuto, qual è il valore di RTT per ciascun dei sei segmenti? Qual è il valore di *EstimatedRTT* dopo la ricezione di ciascun ACK? Il valore di *EstimatedRTT* per il primo segmento è uguale alla misura di RTT, e poi per i segmenti successivi viene calcolato usando l'equazione vista a lezione.
- Qual è la lunghezza di ciascuno dei primi sei segmenti TCP?
- Ci sono dei segmenti ritrasmessi nel file di cattura? Cosa devi verificare per poter rispondere a questa domanda?
- Tipicamente quanti segmenti vengono confermati con un solo pacchetto che trasporta un ACK? Potete identificare i casi dove il ricevitore sta rispondendo con degli ACK a più di uno dei segmenti ricevuti?
- Qual è il throughput (byte trasferiti per unità di tempo) per la connessione TCP? Spiegate come calcolare questo valore

Adesso esaminiamo la quantità di dati inviati per unità di tempo dal client al server. Per farlo useremo una delle utilità grafiche di Wireshark per il protocollo TCP – il *Time-Sequence-Graph(Stevens)* – per rappresentare i nostri dati in maniera grafica.
Selezionate un segmento TCP e poi cliccate su *Statistics -> TCP Stream Graph → Time-Sequence-Graph(Stevens*). Dovreste vedere un grafico simile a quello seguente:



Qui ciascun punto rappresenta un segmento TCP inviato, ed è ottenuto incrociando il suo numero di sequenza con l'istante di tempo al quale è stato inviato.

- Usate il tool *Time-Sequence-Graph(Stevens)* per visualizzare il grafico (tempo, numeri di sequenza) dei segmenti che sono stati inviati dal client a *gaia.cs.umass.edu*. Potete identificare dove inizia e dove finisce la fase di slowstart di TCP? E dove termina la congestion avoidance? Commentate i modi in cui i dati misurati differiscono dal comportamento ideale di TCP che abbiamo studiato a lezione.
- Rispondete a ciascuno dei due quesiti precedenti per il file di cattura che avete generato quando avete trasferito un file dal vostro computer a gaia.cs.umass.edu.

Il protocollo UDP

Obiettivo

Studio del protocollo di trasporto UDP.

Procedura

Iniziare una sessione di cattura di pacchetti in Wireshark e poi fare qualcosa che causerà l'invio e la ricezione di pacchetti UDP da parte del vostro host. Che cosa potreste fare?
Come prima, se non riuscite a catturare dei pacchetti con Wireshark allora potete scaricare un file di cattura dall'indirizzo
https://www.mat.unical.it/informatica/Reti_di_Calcolatori?action=AttachFile&do=get&target=udp-trace.pcap. In tal caso memorizzate il file sul vostro computer ed apritelo con Wireshark.

Dopo avere terminato la sessione di cattura, impostate il filtro in modo che Wireshark vi mostri solo pacchetti UDP inviati e ricevuti dal vostro host. Scegliete uno di questi pacchetti ed espandete i campi UDP nella finestra di dettaglio .

Domande:

- Selezionate un pacchetto. Da questo pacchetto determinate quanti campi ci sono nell'intestazione(*header*) UDP. Individuate anche i nomi di questi campi.
- Dal campo content del pacchetto, determinate la lunghezza (in byte) di ciascuno dei campi dell'intestazione UDP.
- Il valore del campo Length indica la lunghezza di cosa? Verificatelo nel pacchetto UDP che avete catturato.
- Qual è il numero massimo di byte che può essere incluso in un payload UDP?
- Qual è il numero di porta più grande possibile?
- Qual è il numero di protocollo per UDP? Date le vostre risposte sia nella notazione esadecimale che in quella decimale.
- Cercate la parola "UDP" da Google e determinate i campi sui quali viene fatto il checksum UDP.
- Esaminate una coppia di pacchetti UDP in cui il primo pacchetto è inviato dal vostro host mentre il secondo pacchetto è una risposta al primo. Descrivete la relazione che c'è tra i numeri di porta nei due pacchetti.

Domanda extra:

- Catturate un piccolo pacchetto UDP. Verificate a mano il checksum di questo pacchetto. Mostrate il procedimento ed i passi che avete seguito per svolgere questo compito.

Il protocollo IP

Obiettivo

Studio del funzionamento del protocollo IP. In particolare analizzeremo i pacchetti IP generati durante una esecuzione del comando tracert (traceroot con linux). Ricordiamo che questo comando ha un comportamento differente tra linux e windows. Sotto linux, traceroot invia all'host di destinazione una serie di pacchetti UDP; sotto windows, il comando tracert invia messaggi ICMP. Questa distinzione è necessaria per comprendere le differenze tra i pacchetti rilevati usando due sistemi operativi differenti. Una similitudine nel funzionamento è comunque presente. Entrambi i comandi (tracert e traceroute) inviano pacchetti con un valore del campo TTL che varia da pacchetto a pacchetto. In generale la sequenza di pacchetti che viene inviata sulla rete ha il campo TTL modificato secondo il seguente schema:

- Il primo messaggio ha TTL=1;
- Il secondo ha TTL=2, e così via.

In questo processo, ogni router che viene attraversato decrementa di uno il valore del campo TTL fino ad arrivare a TTL=1. Quando a un router perviene un messaggio con TTL=1, il router risponde al mittente inviando un messaggio ICMP di errore (Time To Live exceeded). Con questa procedura, il sistema è capace di mappare i passi successivi che un pacchetto segue durante il suo percorso sulla rete per raggiungere la destinazione.

Procedura

- Fate partire Wireshark e iniziate ad ascoltare il traffico di rete.
- Dalla console dei comandi di Windows, eseguite il seguente programma "tracert hostname";
- Quando il comando tracert termina la sua esecuzione, interrompere lo sniffing della rete.

Dalla schermata di Wireshark è possibile individuare una serie di pacchetti ICMP di tipo "Echo Request" generati dalla vostra macchina. Inoltre, è possibile individuare una serie di pacchetti ICMP di tipo TTLexcedeed pervenuti alla vostra macchina dai vari router che i messaggi ICMP hanno attraversato.

Domande:

- Individuare il primo messaggio ICMP inviato dalla vostra macchina e espandere il protocollo IP corrispondente al pacchetto inviato. Qual'è l'indirizzo IP della vostra macchina?
- Da quanti byte è costituito l'header del pacchetto IP che state analizzando? Da quanti byte è composto il payload del pacchetto IP? Fornire una spiegazione della procedura per il calcolo del payload del pacchetto;
- Il pacchetto IP in analisi è frammentato? Come fare a stabilire se il pacchetto è frammentato?

Ordinare i pacchetti rilevati dalla rete secondo il mittente. Identificare i pacchetti inviati dalla vostra macchina con protocollo ICMP e rispondere alle seguenti domande:

- Quali sono i campi del pacchetto IP che cambiano tra un messaggio e l'altro?
- Quali sono i campi che restano costanti?
- Quali sono i campi che devono restare costanti e quali quelli che devono cambiare? Perché?
- Descrivi e giustifica i valori che vengono visualizzati nel campo Identification del pacchetto IP;

Identificare i messaggi ICMP di tipo TT-exceeded che sono pervenuti alla vostra macchina:

- Che valori assumono i campi Identification e TTL di questi pacchetti IP?
- I valori di questi campi restano invariati per tutti i pacchetti inviati dal router più vicino (first hop)? Perché?

Web Browsing (DNS, TCP, HTTP)

Obiettivo

In questo esercizio analizziamo che cosa succede sui livelli trasporto e applicazione durante una transazione HTTP.

Procedura

1. Avviare Wireshark.
2. Aprire un browser web con un indirizzo non usato recentemente (per evitare che il valore DNS si trovi in cache) ma non premere ENTER.
3. Fare partire una sessione di cattura pacchetti con Wireshark.
4. Premere ENTER nel browser.
5. Fermare la cattura solo quando la pagina è completamente caricata. Salvare il file di cattura.
6. Determinarel'indirizzo IPdel vostro compagno di banco ed effettuare un comando *ping* <indirizzoIPdelvostrocollega>. Per conoscere l'indirizzo IP di una macchina, digitate *ipconfig* (Windows) oppure *ifconfig* (Linux).
7. Salvare la pagina web come riferimento.

Protocol Analysis Questions

Analizzando l'elenco dei pacchetti catturati, rispondere alle seguenti domande.

1. ***Protocolli catturati***
   - Esaminate la colonna dei protocolli catturati nella finestra di Ethereal, stendete un elenco di tutti i protocolli catturati, che dovrebbero contenere DNS, TCP, HTTP, ICMP.
2. ***Pacchetti IP e UDP***
   - Cercate il primo pacchetto DNS inviato da voi
     a. Identificate l'indirizzo IP del mittente. Dovrebbe trovarsi nell'intestazione IP (che studieremo in dettaglio in seguito).
   - Guardate l'intestazione UDP del primo pacchetto DNS
     a. Identificate il numero di porta sorgente e destinazione. La porta destinazione corrisponde a 53? Che protocollo applicazione trasporta il datagramma UDP?
3. ***DNS***
   - Esaminate il payload (I dati) del primo pacchetto DNS inviato.
     a. Dove sta l'informazione che dice se il messaggio è una query o una risposta?
     b. Il corpo della query cosa dice?
     c. Quant'è il codice (ID) della query?
     d. Qual'è il TIPO della query? In che campo viene codificato?
   - Ora cercate la risposta alla query DNS fatta.
     a. Quali dovrebbero essere gli indirizzi IP di mittente e destinatario di questo pacchetto? Verificate che corrispondano a ciò che vi aspettate.
     b. Quanti byte occupa il datagramma di risposta? E' più piccolo o più grande del datagramma di query?
     c. Verificate che l'ID della risposta corrisponda con l'ID della query.
     d. Quante risposte ci sono in totale nel messaggio di risposta? Hanno tutte lo stesso TTL? Cos'è il TTL nelle risposte DNS?
4. ***HTTP GET***

- Trovate il pacchetto che trasporta il comando "HTTP GET".
  a. Confermate che i numeri di sequenza e acknowledgement sono quelli che vi aspettavate.
  b. Guardate i flag? Sapete spiegare perché i valori sono tali?
  c. Quant'è lungo il segmento TCP? Quanti byte sono invece i dati effettivi?

- Guardate ora il contenuto del comando GET.
    a. Confrontate i valori grezzi nel terzo riquadro in basso con i valori decodificati presentati nel secondo riquadro.
    b. Contate il numero di byte nel messaggio e verificate che questo numero corrisponda al campo lunghezza nell'intestazione TCP.
    c. Che numero di sequenza vi aspettate nel prossimo pacchetto in arrivo dal server? Che numero di acknowledgement?

5. HTTP Response

- Trovate il pacchetto di risposta.
- Quanto tempo è passato dalla cattura del messaggio GET fino al messaggio di risposta?
- Verificate se il server risponde con un messaggio HTTP di risposta oppure solo con un segmento TCP ACK. Verificate che il numero di sequenza e ack siano quelli che vi aspettavate.
- Ora considerate il messaggio che contiene i dati HTTP di risposta.
    a. Qual è la lunghezza del payload nel segmento TCP (payload = dati netti)?
    b. Controllate il valore dei flag e spiegate perché alcuni sono impostati.
    c. Che numero di ACK vi aspettate dal client?
- Considerate ora i dati http di risposta.
    a. Che codice di risultato avete?
    b. Evidenziate la sezione DATA nel secondo riquadro. Comparate il testo decodificato con i dati grezzi nel terzo riquadro.