Wireshark on the wire (DNS, TCP, HTTP, SMTP)

N.B. Per esercitarsi è possibile utilizzare i file di cattura allegati. In alternativa, è possibile catturare del traffico lanciando Wireshark e navigando il web (causando dunque del traffico HTTP e DNS), inviando email da un programma di posta client (causando dunque del traffico SMTP) ecc…

Esercizio 1

Analisi del traffico HTTP.

Per catturare del traffico HTTP è sufficiente collegarsi ad un qualsiasi sito internet tramite un browser.
Siti raccomandati:

- http://jrobots.sourceforge.net/jjr_tutorials.shtml
- http://anh.cs.luc.edu/python/hands-on/3.1/handsonHtml/graphics.html

1) Osservare tutto il traffico catturato nella lista dei pacchetti in Wireshark.
   a) Sono presenti protocolli diversi da HTTP ?
   b) Riordina alfabeticamente la colonna dei protocolli e stendi una lista di tutti quelli di cui non conosci il significato (dai poi uno sguardo veloce a casa per capire che ruolo ha ciascuno di essi).
   c) Infine, utilizzando la barra dei filtri, seleziona solo i pacchetti di tipo **HTTP**.
2) Selezionare il primo pacchetto HTTP etichettato come comando **GET /** ed effettuare alcune analisi su di esso.
   a) Hints:
      i) filtrare i soli pacchetti http
      ii) Selezionare tutti i pacchetti http il cui tipo di richiesta è **GET**
      iii) Selezionare tutti le richieste http il cui URL riferito è pari a un certo testo "testodacercare"
      iv) Trovare lo user agent associato alla richiesta
      v) Trovare i cookies associati alla richiesta
      vi) Verificare se esistono **external referer.** Se si, in che layer hai guardato per trovare la richiesta ?
3) Osserva gli indirizzi IP sorgente e destinazione.
4) Filtra tutti i pacchetti il cui **IP sorgente** è l'indirizzo della tua macchina.
   a) Seleziona un qualsiasi pacchetto HTTP e controlla la porta sorgente e la porta di destinazione
      i) La porta di destinazione trovata è quella che ti aspettavi?
      ii) Prova ora a filtrare tutti i pacchetti tcp la cui porta destinazione è 443
      iii) Cosa appare nella colonna dei protocolli di Wireshark? Perché?
5) Filtra ora tutte le http request
   a) Trova la richiesta precedente e successiva a quella selezionata (se presenti) e l'eventuale risposta
6) Trovare tutte le connessioni HTTP:
   a) Il cui indirizzo sorgente è pari all'ip del tuo computer;
   b) La cui porta destinazione è 80;
   c) Il cui request method è GET;
   d) Il cui uri **contiene** la parola "string" (sostiture *string* con il nome del sito web visitato)
7) Sia dato il primo risultato mostrato usando il filtro del punto precedente. Individuare la response corrispondente
   a) Qual è la Response Phrase e il relativo Status Code ?
   b) Quali sono i campi che si differenziano tra una request e una response HTTP ?

Esercizio 2

Obiettivo
In questo esercizio analizzeremo che cosa succede sui livelli trasporto e applicazione durante una sessione (insieme di connessioni multiple) **HTTP** e una conversazione **SMTP**.

Procedura 1: HTTP.

1. Fate partire Wireshark.
2. Aprite un browser web indicando un indirizzo non usato recentemente (per evitare che il valore DNS si trovi in cache) ma non premete ENTER.
3. Fate partire una sessione di cattura frame con Wireshark.
4. Premete ENTER nel browser.
5. Attendete che la pagina sia completamente caricata. Determinate l'indirizzo IP del vostro compagno di banco ed effettuate un comando ping <indirizzoIPdelvostrocollega>. Per sapere l'indirizzo IP di una macchina, digitate ***ipconfig*** (Windows) oppure ***ifconfig*** (Linux).
6. Salvate la pagina web come riferimento e infine terminate e poi salvate la cattura Wireshark.

Protocol Analysis Questions
Analizzando l'elenco dei frame catturati, rispondete alle seguenti domande.

1. ***Protocolli catturati***
   - Esaminate la colonna dei protocolli catturati nella finestra di Wireshark, stendete un elenco di tutti i protocolli catturati, che dovrebbero contenere DNS, TCP, HTTP, ICMP. Ci sono delle sigle di protocolli a voi sconosciute? Completate la lista con una breve descrizione di ogni protocollo incontrato (sconosciuto e non)
   - Ci sono tra i frame catturati, dati di cui **non siete** destinatari? Stendete una lista degli indirizzi IP destinatario non corrispondenti all'indirizzo IP della vostra interfaccia di rete primaria.
2. ***Frame IP e UDP***
   - Cercate il primo frame 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 frame DNS
     b) Identificate il numero di porta sorgente e destinazione. Qual è la porta destinazione?
3. ***DNS***
   - Esaminate il payload (I dati) del primo frame DNS inviato.
   - Dove sta l'informazione che dice se il messaggio è una query o una risposta?
     a) Il corpo della query cosa dice?
     b) Qual è il codice (ID) della query?
     c) 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 frame? 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 quei frame che trasportano dei comandi "HTTP GET". Identificate quante connessioni sono state usate.
     a) Confermate che i numeri di sequenza e acknowledgement sono quelli che vi aspettavate.
     b) Guardate i flag, sapete spiegare il perché di tali valori?
     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 frame in arrivo dal server? Che numero di acknowledgement?

Procedura 2: SMTP.

1. Fate partire una cattura Wireshark.
2. Aprite la vostra soluzione per la prima esercitazione che è stata condotta per il modulo di Reti (Esercizio 2, implementazione del comando *sendMail*), o in eseguite i sorgenti disponibili qui: https://www.mat.unical.it/ianni/SOR-Web/codice/esercitazioneSocket/smtp/smtp.py
3. Provate a invocare *sendMail* indicando come mittente il VOSTRO indirizzo email e come destinatario iannir@mat.unical.it. Usate come mail server SMTP *ml.mat.unical.it* porta 25.
4. Fermate la cattura.
5. Individuate in Wireshark i frame relativi alla conversazione *SMTP* appena svolta. Sapete individuare l'IP del mail server SMTP che viene contattato ?
6. Qual è il suo nome? E qual è l'***IP*** del client che vi si connette?
7. Cambiate ora l'indirizzo mittente in sorcio@sorcino.it e lanciate di nuovo il programma. Questa volta dovreste leggere un errore da console. Perché? Usate Wireshark per capire come si è esattamente svolta la conversazione SMTP.

Analisi di pacchetti UDP.

Per acquisire i pacchetti UDP aprire il browser e collegarsi ad un sito internet non presente in cache o, più semplicemente, catturare il traffico prodotto da una conversazione **Skype**, una riunione **Microsoft Teams** o una conferenza **Zoom**.

1. Selezionare un pacchetto e determinare il numero e i nomi dei campi presenti nell'intestazione del pacchetto.
2. Selezionare tutti i pacchetti UDP che hanno come porta di destinazione la numero 53.

   A che protocollo corrisponde ?
3. Selezionare un qualsiasi pacchetto dalla lista mostrata
   a. Trovare la porta sorgente
   b. Trovare la lunghezza del pacchetto **UDP** e successivamente dell'intero **Frame**.
      i. Le 2 lunghezze sono uguali ?
      ii. Perché ?

      iii. Il pacchetto selezionato è una query o una response dns? Da dove si evince?
4. Trovare il pacchetto di risposta della prima query dns

   Che tipo di query è ?

Esercizio 3

Passo 1. Si deve progettare un semplice script Python3, chiamato `searchProduct.py`, che può essere invocato con la seguente sintassi:

```
./searchProduct.py <website_link>
```

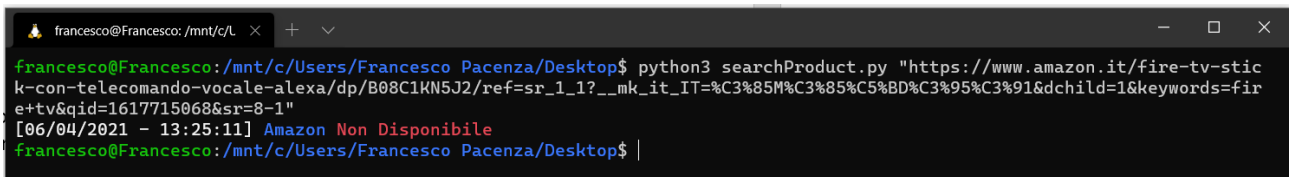
ad esempio:

```
./searchProduct.py https://www.amazon.it/fire-tv-stick-con-telecomando-vocale-alexa/dp/B08C1KN5J2/ref=sr_1_1?__mk_it_IT=%C3%85M%C3%85%C5%BD%C3%95%C3%91&dchild=1&keywords=fire+tv&qid=1617715068&sr=8-1
```

il comando deve interrogare il marketplace indicato nel link (Amazon nell'esempio di cui sopra) tramite una GET request e controllare se il prodotto cercato è disponibile o meno.

Lo script stamperà su stdout data, ora, sito web visitato ed eventuale disponibilità del prodotto.

Esempio:



Suggerimenti e osservazioni:

- È possibile effettuare get/post request facendo uso della libreria python3 urllib3 installabile tramite pip3
- Per effettuare il parsing della response http è possibile utlizzare la libreria BeautifulSoup4, anche'essa installabile tramite pip3 (pip3 install beautifulsoup4)
- Il sito web su cui effettuate la GET request potrebbe bloccare la vostra richiesta poiché il vostro user agent è anonimo. Per ovviare a questo problema è possibile settare lo user-agent manualmente impostando il parametro "headers" mentre effettuate la http request.
  - Esempio: headers = { 'User-Agent' : 'Mozilla/5.0 (Windows NT 6.1; Win64; x64)' }
- Piccoli esempi di utilizzo di urllib3 e beautifulSoup4:
  - https://urllib3.readthedocs.io/en/latest/user-guide.html
  - https://beautiful-soup-4.readthedocs.io/en/latest/

Esercizio 4

Passo 1. Si deve progettare un semplice script Perl, chiamato askScholar, che può essere invocato con la seguente sintassi:

```
askScholar <paroleChiaveArgomentoSeparateDaSpazi>
```

ad esempio:

```
./askScholar higgs boson
```

il comando deve interrogare il portale scholar.google.com con un URL nel formato

`https://scholar.google.com/scholar?hl=en&q=<paroleChiaveArgomentoSeparateda+>`

e ritornare in standard output il numero di risultati che viene indicato in output nella frase

"About N results"

Suggerimenti e osservazioni:

- Si può catturare l'output del comando shell wget, opportunamente invocato con "`wget -O - URL`" (stampa in standard output la risorsa appena recuperata)
- Il comando di cui sopra può ritornare una stringa vuota e/o un messaggio di errore dal server. Perché?
- Comando wget da usare: `wget -O output -U 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.103 Safari/537.36' 'https://scholar.google.com/scholar?hl=en&q=<paroleChiaveArgomentoSeparateda+>'`

Passo 2. Si modifichi lo script di cui al passo 1 per accettare da standard input un file una raccolta di diverse parole chiave, messe ciascuna su una linea separata.
Il nuovo script deve produrre in output una statistica, ordinata per valori decrescenti delle parole chiave con più risultati.

Alcuni dati di esempio sono disponibili su
`http://www.mat.unical.it/ianni/storage/scholartestsetSOR.csv`

Ad esempio, se il file di input contiene

TURING Alan
BASSOTTI Banda
MOUSE Mickey

L'output atteso potrebbe essere

TURING Alan	56600
MOUSE Mickey	32200
ASSOTTI Banda	353

Osservazioni: che cosa succede dopo aver interrogato in pochi minuti `scholar.google.com` con un certo numero di queries?