UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II

Scuola Politecnica e delle Scienze di Base
Corso di Laurea in Ingegneria Informatica

Elaborato d'esame
Reti di Calcolatori I
A.A. 2018/2019

Raccolta e analisi di tracce di traffico di applicazioni mobili

Professore
Prof. Antonio Pescapè

Studenti	*Matricola*
Coppola Vincenzo	*N46/3356*
Della Torca Salvatore	*N46/3208*

INDICE

INTRODUZIONE

L'obiettivo di questo studio è quello di analizzare tracce di traffico che sono state generate da applicazioni mobili.
Uno degli strumenti utilizzati per lo studio è il tool TIE, progettato e sviluppato dal gruppo di ricerca COMICS dell'Università di Napoli "Federico II".

In una prima fase preparatoria è stato generato, utilizzando applicazioni mobili su terminali ANDROID (Xiaomi Mi5 e Nexus 7), e catturato del traffico attraverso un'interfaccia fisica con il sistema di cattura nel laboratorio ARCLAB.

La seconda fase, che riguarda la classificazione del traffico e l'analisi dei risultati, è stata eseguita tramite procedure automatizzate realizzate in software.

1. CLASSIFICAZIONE DEL TRAFFICO

Classificare il traffico di rete vuol dire associare ad un flusso di traffico l'applicazione che l'ha generato.
Negli ultimi anni, visto il notevole utilizzo di smartphone e dispositivi portatili, il traffico mobile ha subito una crescita esponenziale.
Per una TC (Traffic Classification) è necessario identificare:

1. le *classi di traffico,* che definiscono il livello di dettaglio della classificazione;
2. gli *oggetti di classificazione*, che possono essere pacchetti, connessioni TCP o UDP, flussi o biflussi.

Le metodologie utilizzate per la classificazione del traffico possono essere *port-based, payload-based, heuristics-based* o *deep packet inspection*.

1.1 Port-Based

Questo approccio è basato sull'assegnazione dei numeri di porto effettuata dallo IANA*(Internet Assigned Numbers Authority)* e sulla conoscenza dei porti utilizzati comunemente dalle applicazioni.
Permette di identificare l'applicazione che genera un determinato flusso di traffico utilizzando la porta sorgente e destinazione del livello trasporto, considerando pacchetti IP consecutivi con la stessa quintupla *[protocol, source address, destination addres, source port, destination port]* come appartenenti allo stesso flusso.
Tuttavia, la costante crescita di applicazioni Internet che fanno uso di porte di livello trasporto casuali o non assegnate dallo IANA, ha determinato la crescita di lacune in tale approccio.

1.2 Heuristics-based

Il traffico viene classificato sulla base delle proprietà statistiche del flusso dei pacchetti; ogni flusso di traffico è associato ad un insieme di attributi e, tramite l'analisi dei valori di tali attributi, il

classificatore può attribuire al flusso in esame l'applicazione di appartenenza più probabile.
Un classificatore basato su un approccio di tipo statistico può essere realizzato con algoritmi di apprendimento automatico *(MACHINE LEARNING)*.
In questo approccio è necessario però fornire all'algoritmo un sufficiente numero di flussi di appartenenza certa alle classi di interesse, in modo da fornire una base di verità affidabile che è definita *GROUND TRUTH:* una possibile soluzione è quella di registrare varie tracce di traffico ed usare tecniche di ispezione manuale dei payload off-line per etichettare i dati di training.

1.3 Deep Packet Inspection

Questo tipo di approccio consiste invece nel riconoscere le applicazioni tramite l'ispezione del payload di ciascun pacchetto. Anche se DPI è considerato l'approccio più accurato, a causa della sua complessità computazionale, dei problemi di privacy e del crescente utilizzo di tecniche di crittografia e offuscamento, quest'approccio è oggi utilizzato solo come riferimento per valutare l'accuratezza di nuove tecniche sperimentali.

2. TIE: TRAFFIC CLASSIFICATION ENGINE

Il tool di classificazione TIE ha come obiettivo costituire un ***multi-classification system (MCS)***, ossia un sistema che mette a disposizione più tecniche di classificazione, che sono implementate come **plug-in** indipendenti.
Nella ***Traffic Classification*** ha importanza fondamentale il concetto di ***biflusso***.
Un biflusso è una sequenza di pacchetti che condividono i valori della quintupla [protocol, source IP, source port, destination IP, destination port] e in cui destinazione e sorgente possono essere scambiati tra loro.
Il tool TIE, di default, divide il traffico in "***session***" di tipo biflusso, che vengono poi classificate separatamente.
Quando il protocollo di livello trasporto è ***TCP (Trasmission Control Protocol)***, i biflussi tipicamente approssimano le connessioni TCP. Questa euristica è molto semplice ed efficiente ma sono necessarie delle informazioni aggiuntive per identificare le connessioni più accuratamente.
Inoltre, di default, il tool TIE opera in modalità offline ed è quindi in grado di fornire le informazioni riguardanti la cattura solo quando la sessione è terminata.
Il plug-in utilizzato in questo studio è "***nDPIng***" che opera una classificazione di tipo DPI basata su diverse *features* come il contenuto e la dimensione del payload, la dimensione dei pacchetti o i certificati ***SSL***.

3. WIRESHARK

Wireshark è un software per l'analisi di protocolli o ***packet sniffer*** distribuito sotto una licenza OpenSource e utilizzato per l'analisi del traffico generato in rete.
Le funzionalità di Wireshark sono molto simili a quelle di ***tcpdump***, ma con un'interfaccia grafica e un numero maggiore di funzionalità di ordinamento e filtraggio.
Grazie a questo software l'utente è in grado di osservare tutto il traffico presente sulla rete: Wireshark riesce a comprendere la

struttura di diversi protocolli di rete, eventuali incapsulamenti, riconoscere i singoli campi e interpretarne il significato.

3.1 Caratteristiche e funzionalità di Wireshark

- È possibile analizzare dati acquisiti in tempo reale e anche dati precedentemente salvati su file di cattura;
- È possibile analizzare i dati sia tramite interfaccia grafica sia da riga di comando (utilizzando il terminale) grazie al programma **tshark**;
- I dati catturati su file possono essere facilmente modificati, convertiti o filtrati, tramite comandi bash;
- È possibile scomporre e analizzare centinaia di protocolli di comunicazione;
- È possibile filtrare i dati da visualizzare.

Su molte piattaforme, catturare traffico da un'interfaccia di rete, richiede adeguati permessi di amministrazione: per questo motivo Wireshark viene spesso eseguito da ***root*** anche su piattaforme che non lo richiedono.
Durante la cattura di traffico di rete in tempo reale vengono utilizzate le *routines* di un numero molto elevato di scompositori di protocollo e quindi, in caso di bug anche su una singola routine, si potrebbero presentare dei seri problemi di sicurezza.

3.2 Tshark

Tshark è la versione a linea di comando di Wireshark.
È un analizzatore di protocolli di rete che permette di poter catturare pacchetti (***datagram***) da una rete o leggerne da un file di cattura precedentemente salvato.
Il formato della cattura di Tshark è il ***pcap*** (che viene utilizzato anche da ***tcpdump***).

4. CATTURA DEL TRAFFICO

Nel laboratorio ARCLAB è stato generato del traffico utilizzando due dispositivi Android (Xiaomi Mi5 e Nexus 7) collegati tramite interfaccia fisica al sistema di cattura.
Per avere una migliore precisione della cattura sono state installate solo le applicazioni necessarie per la generazione del traffico; nel caso in esame le applicazioni sono *Spotify e Dropbox* per la 1^a cattura, *Duolingo e Onefootball* per la 2^a.
La sessione iniziale di cattura è stata effettuata durante il *download* dell'applicazione e la successiva *registrazione* ad essa.
Le sessioni di cattura successive sono state invece effettuate durante l'*utilizzo* dell'applicazione, facendone un uso vario e completo cercando di generare quanto più traffico possibile.
Durante la sessione, al livello del sistema operativo, vengono misurate le attività di sistema monitorando le ***system call*** chiamate e i segnali ricevuti dai processi in esecuzione sul terminale.
In particolare, vengono raccolte quelle system call riguardanti la scrittura e la lettura su socket; tale operazione viene effettuata usando una specifica versione di ***strace*** che genera in output un file "***strace.log***".
A livello di rete vengono registrati tutti i pacchetti dati inviati e ricevuti usando una versione specifica di ***tcpdump*** che salva tutto il traffico in un file ***"traffic.pcap"***.

5. SCRIPT PER L'AUTOMAZIONE DELL'ANALISI

Lo script per l'automazione dell'analisi (script.py) è stato sviluppato utilizzando il linguaggio di programmazione ***PYTHON***, con una aggiunta di istruzioni in linguaggio ***BASH***.
Per automatizzare l'esecuzione delle istruzioni in linguaggio *bash*, è stato necessario importare il modulo *subprocess*; attraverso l'istruzione *suprocess.run("cmd", stdout="nomefile.txt", shell=True);* si specifica con *cmd* il comando bash da eseguire, con *stdout="nomefile.txt"* il file sul quale si vuole salvare l'output restituito dal comando e con *shell=True* il comando *cmd* viene interpretato come istruzione in linguaggio bash.
Lo script ha come obiettivo classificare il traffico catturato in laboratorio utilizzando il tool TIE, e validare la classificazione che è stata prodotta.
La validazione, in una prima fase, consiste nel verificare che l'etichetta assegnata a ciascun flusso sia corretta; successivamente vengono poi fornite informazioni aggiuntive e/o correttive alla classificazione.
Il programma può essere eseguito su una qualsiasi distribuzione Linux (è stato realizzato utilizzando il SO Linux Mint 19).

5.1 Preparazione ed esecuzione dello script

Come detto in precedenza, a causa della presenza di comandi BASH realizzati in Linux, lo script può essere eseguito solamente su distribuzioni Linux.
Prima di realizzare lo script è stato necessario effettuare delle operazioni preliminari:

```
1 sudo apt-get install build-essential libpcap-dev
2 wget http://tinyurl.com/api-homework-tie -O tie-1.2.1-with-ndping.tgz
3 tar -xzf tie-1.2.1-with-ndping.tgz
4 cd TIE-1.2.1
6 cd src/
7 make
8 cd ../bin
```

Queste operazioni sono necessarie per poter installare il tool di classificazione TIE.
Una volta installato il tool, per poter classificare una traccia di traffico con estensione . pcap

```
9  ./TIE -a ndping_1.0 -r <path_to_pcap_file>
10 cat output/class.tie
```

In questo modo è stato prodotto un file di testo con estensione .tie

Per eseguire lo script bisogna aprire il terminale ed accedere alla cartella in cui si trova il file traffic.tie ed eseguire l'istruzione ***python3 script.py***.
Il lancio di questa istruzione porta all'esecuzione dello script che genera un file ***traffic.gt.tie*** che contiene le colonne del file tie, relative all'id, source ip, destination ip, source port, destination port e protocol, più una serie di colonne aggiuntive che sono state realizzate per effettuare la validazione/correzione del risultato fornito da tie.

5.2 Descrizione dettagliata dello script

Per prima cosa viene importato il modulo *subprocess* che è necessario per poter eseguire istruzioni in linguaggio bash, direttamente da script, utilizzando il linguaggio *python:*

- la prima subprocess viene utilizzata per creare il file "*traffic.gt.tie*" che deve essere utilizzato per la "correzione";
- la seconda subprocess serve a scrivere il dns di ogni quintupla analizzata da tshark all'interno di un file "*outputdns.txt";*
- la terza subprocess serve a scrivere il *Server Name Indication* di ogni quintupla all'interno di un file "*outputsni.txt*";
- l'ultima subprocess viene utilizzata per creare il file "*outputwhois.txt"* che dovrà contenere i whois risolti.

Viene aperto il file "*traffic.tie*" per poter prelevare le quintuple contenenti informazioni relative al traffico.

Poiché le prime undici righe di ogni file generato da TIE sono righe identificative per quel file, sono state copiate così come sono nel file "traffic.gt.tie".
Del file "traffic.tie" non vengono considerate le *righe* dove il campo "ip_dest" è "8.8.8.8" perché non si tratta di traffico dati ma solamente di risoluzioni del DNS.
Delle righe considerate si prende prima in considerazione il campo "*confidence":*

- se è 0 la quintupla è *non classificata;*
- se è 100 vuol dire che TIE è riuscito a "classificare" quel traffico, ed è quindi necessario effettuare altri controlli.

Si va allora a prelevare il *"package"*, se è presente, effettuando un controllo sugli ip e sui porti contenuti all'interno delle socket del file "strace.log".
Bisogna ora controllare che, all'interno del file "outputdns.txt", sia presente una quintupla il cui campo "ip_dest" sia uguale a quello della quintupla di interesse: se si verifica questa corrispondenza si preleva il DNS per quella quintupla.
In maniera del tutto identica, sono stati prelevati gli *"sni"* che sono stati precedentemente salvati nel file "outputsni.txt".
Infine è stato necessario ottenere informazioni sulla compagnia intestataria degli indirizzi ip: per fare ciò è stata utilizzata l'interrogazione whois.

6. ANALISI DEI RISULTATI

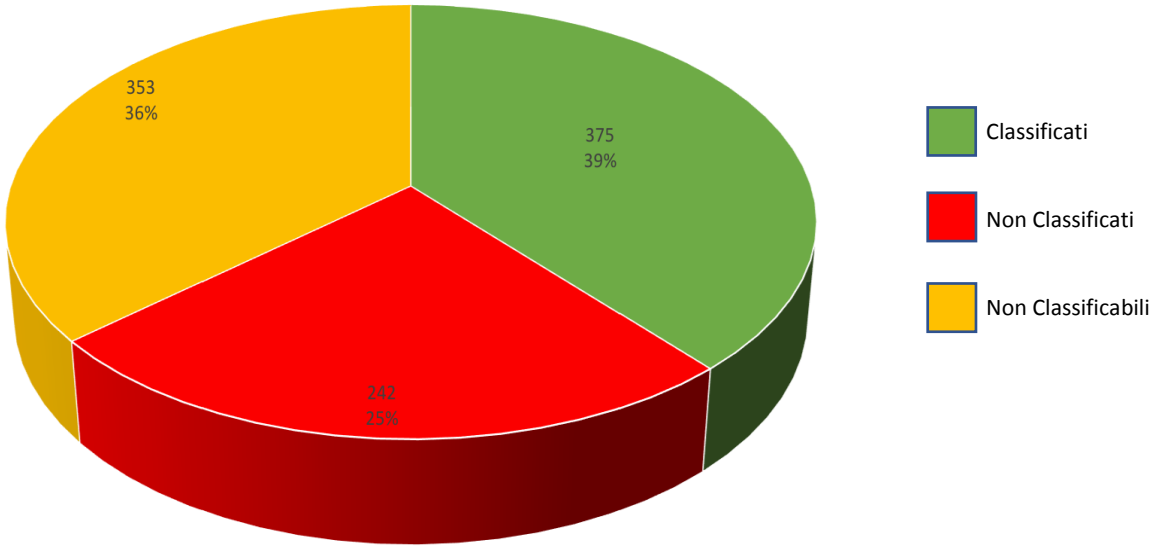
L'esecuzione dello script fornisce come risultato il file "traffic.gt.tie" che contiene le principali colonne del file "traffic.tie" più altre colonne che sono aggiunte grazie alle istruzioni eseguite dallo script "script.py".

I risultati forniti dallo script sono stati inseriti in un foglio di lavoro EXEL:

- in verde sono stati evidenziati i biflussi classificati da TIE che hanno avuto un riscontro positivo con la classificazione effettuata da TSHARK;
- in giallo sono stati evidenziati i biflussi che TIE è riuscito a classificare ma per i quali TSHARK non è riuscito ad individuare alcune informazioni di interesse, come *package, sni…;*
- in rosso sono stati evidenziati i biflussi che TIE non è riuscito a classificare e per i quali non è quindi possibile avere informazioni.

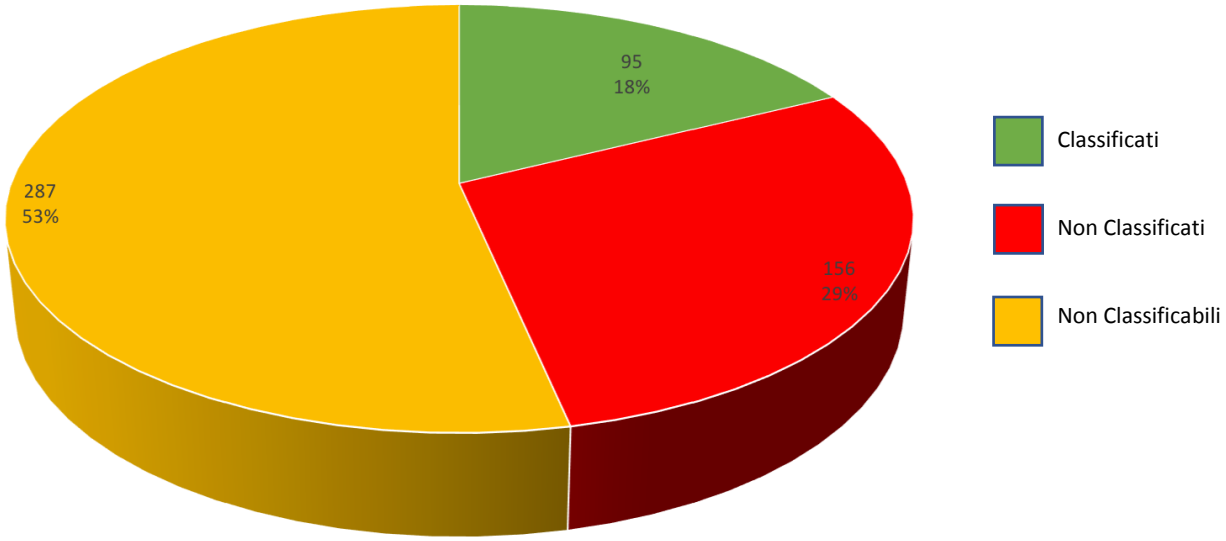
109	192.168.20.105	92.122.125.80	6	42610	80	HTTP
48	216.58.198.46	192.168.20.105	6	443	44581	OtherTCP
71	192.168.20.105	216.58.198.46	6	47626	443	Google

In totale il numero di biflussi che sono stati analizzati è pari a 970, e di questi solamente il 39% è stato classificato correttamente; il 36% dei biflussi è non classificabile mentre il restante 25% dei biflussi non è stato classificato.

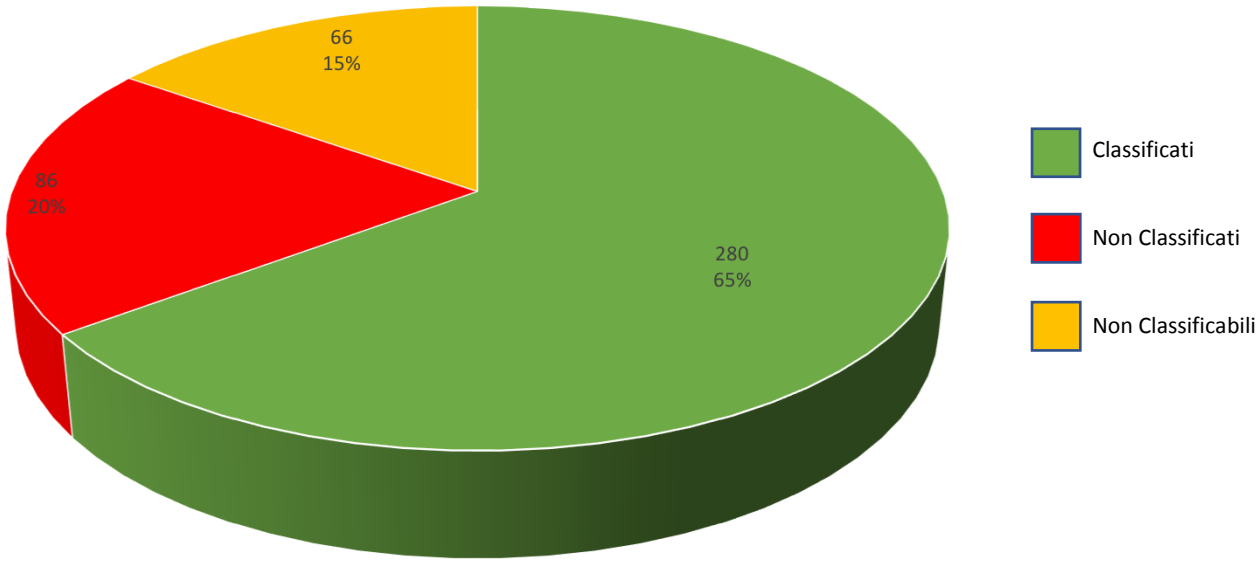


In particolare di questi 970 biflussi, 538 sono stati generati dall'utilizzo dell'applicazione *SPOTIFY*, mentre 432 dall'applicazione *DROPBOX*.

Dei 538 biflussi generati da Spotify, 95 sono stati validati correttamente, 156 non sono stati classificati e per i restanti 287 non è stato possibile fornire alcuna informazione a causa della presenza di campi UNKNOWN.



L'applicazione Dropbox ha generato 432 biflussi, di cui il 65% (280 su 432) è stato classificato, il 20% (86 su 432) non è stato classificato e il 15% (66 su 432) non è classificabile.

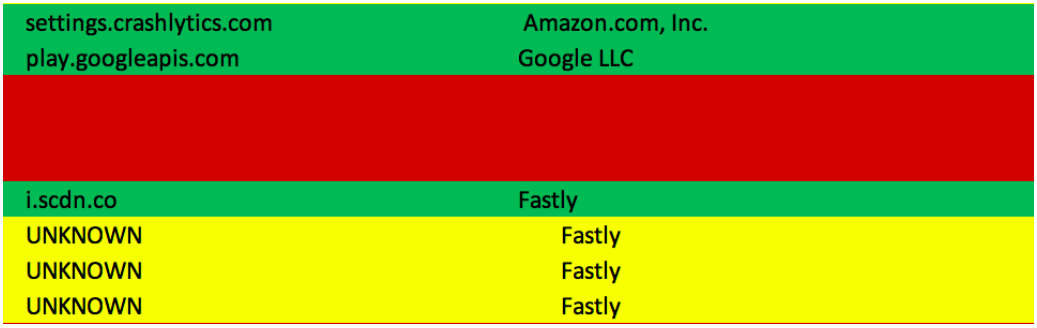


N.B: I flussi il cui campo protocollo è di tipo *OtherTCP, OtherUDP o UDP* sono stati considerati come **non classificati**.

7. SERVIZI DI TERZE PARTI

Analizzare il traffico vuol dire anche studiarne l'*origine,* cioè individuare quanto traffico di rete proviene effettivamente dai server di proprietà del fornitore di quella applicazione.
Attraverso queste informazioni è possibile "stimare" il livello di privacy dell'utente, dal momento che si tratta di un' indicazione del controllo che il fornitore ha sui dati della propria app.

Per effettuare quest'ultima analisi sono state eseguite delle istruzioni nello script con obiettivo di associare ad ogni biflusso l'applicazione che l'ha generato sulla base del campo *app_log*.



Questi sono alcuni dei campi che sono stati forniti dall'esecuzione del comando *WHOIS*.
Durante l'utilizzo dell'applicazione "Spotify", alcuni biflussi di traffico sono stati generati da "Facebook", altri da Amazon e altri ancora da Google.
La maggior parte dei biflussi proviene però da "*FASTLY*": si tratta di una *CDN Platform; è stata fondata nel 2011 ed è cresciuta fino a diventare un'importante CDN provider. Fastly offre un'ampia gamma di servizi, tra i quali "streaming media".*

È possibile quindi distinguere nelle classificazioni effettuate diverse categorie di applicazioni di terze parti:

- *CDN e Cloud Traffic:* questa categoria include la percentuale di traffico che proviene dai server di CDN o dai cloud providers; tra questi troviamo FASTLY e AMAZON.COM

  (si ipotizza che si tratti della piattaforma cloud Amazon AWS);

- *Google Traffic:* include tutto il traffico scambiato con i server di GOOGLE; durante le sessioni di utilizzo di Spotify e Dropbox questo traffico può essere attribuito ai servizi di Google che le app utilizzano come *maps, ads, analytics e Google App Engine*.

RIFERIMENTI

Alberto Dainotti, Antonio Pescapè, Carlo Sansone: *Early Classification of Network Traffic through Multi-Classification;*

Alberto Dainotti, Walter de Donato, Antonio Pescapè, Giorgio Ventre: *TIE: a community-oriented traffic classification platform;*

Antonio Pescapè, Antonio Montieri: *Cattura e Classificazione del Traffico Mobile*
https://www.docenti.unina.it/webdocenti-be/allegati/materiale-didattico/34010803

Documentazione TIE:
http://tie.comics.unina.it/doku.php?id=sections:documentation:start

Documentazione WireShark e TShark:
https://www.wireshark.org/docs/man-pages/tshark.html

https://codexsprawl.wordpress.com/2016/02/29/wireshark-un-programma-per-analizzare-la-rete/

Gestione dei file in Python:
https://www.python.it/doc/Howtothink/Howtothink-html-it/chap11.htm

https://www.html.it/pag/15617/lavorare-con-i-files/

Filtri disponibili in Wireshark:
https://linuxaria.com/article/more-fun-with-wireshark-filters?lang=it

Python-Check if a word is in a string:
https://stackoverflow.com/questions/5319922/python-check-if-word-is-in-a-string

How to get WHOIS info by IP in python 3:
https://stackoverflow.com/questions/24580373/how-to-get-whois-info-by-ip-in-python-3

How to run bash command in Python:
https://stackoverflow.com/questions/4256107/running-bash-commands-in-python