UNIVERSITA' DEGLI STUDI DI
NAPOLI FEDERICO II

Facoltà di Ingegneria
Corso di Studi in Ingegneria Informatica

Elaborato d'esame

Reti di Calcolatori I

Anno Accademico 2017/2018

professore
Prof. Antonio Pescapè

studente
Carmine Cesarano
Matr. N46/2883

Sommario

INTRODUZIONE

Questo studio riguarda l'utilizzo del tool TIE, progettato e sviluppato dal gruppo di ricerca COMICS dell'Università degli Studi di Napoli " Federico II" , e l'analisi dei risultati forniti dal tool.

In una prima fase preparatoria è stato generato del traffico utilizzando applicazioni mobili su terminali Android, e catturato tramite 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 operata tramite procedure automatizzate sviluppate in software.

1 CLASSIFICAZIONE DEL TRAFFICO

La classificazione del traffico di rete consiste nell'associare ad una sequenza di pacchetti scambiati tra due host e due rispettive porte a livello di trasporto, la presunta applicazione che li ha generati.

Per la classificazione del traffico possono essere utilizzate diverse metodologie: *port-based, deep packet inspection, heuristics-based.*

1.1 Port Based

Questo primo approccio permette di identificare l'applicazione che genera un determinato flusso di traffico utilizzando la porta sorgente e destinazione del livello di trasporto, considerando pacchetti IP consecutivi con la stessa 5-tupla [*protocollo, indirizzo sorgente, porto sorgente, indirizzo destinazione, porto destinazione*] come appartenenti allo stesso flusso.

La costante crescita di nuove applicazioni Internet che fanno uso di porte di livello di trasporto casuali o comunque non assegnate dallo IANA ha evidenziato le lacune di questo approccio. [2]

1.2 Deep Packet Inspection

Un secondo 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, esso è utilizzato oggi solo come riferimento per valutare l'accuratezza di nuove tecniche sperimentali. [1]

1.3 Heuristic analysis

Il traffico viene classificato sulla base delle proprietà statistiche del flusso dei pacchetti che viene modellizzato come un processo casuale. Ogni flusso di traffico è associato ad un insieme di attributi (ad esempio lunghezza dei pacchetti o tempi di inter-arrivo), e tramite l'analisi dei valori di tali attributi, il classificatore può attribuire al flusso in esame l'applicazione d'appartenenza più probabile. [2]

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 (ground truth): una possibile soluzione è quella di registrare varie tracce di traffico ed usare tecniche di ispezione manuale dei payload offline per etichettare i dati di training. [2]

2 TIE: TRAFFIC IDENTIFICATION ENGINE

Quello che si propone il tool di classificazione TIE è di costituire un **multi-classification system** (MCS), ovvero un sistema che mette a disposizione più tecniche di classificazione. Le diverse tecniche (DPI, statistical, port-based) sono implementate all'interno del tool come **plugin** indipendenti, il che permette la loro comparazione e combinazione.

- Un oggetto comune nella Traffic Classification è il " **biflusso**, definito come una sequenza di pacchetti che condividono i valori della quintupla [protocol, source IP, source port, destination IP, destination port] e dove destinazione e sorgente possono essere scambiati tra loro.

Il tool, di default (cioè senza specificare un'opzione diversa), divide il traffico in " **sessions**" di tipo biflusso che poi classifica separatamente. Altri tipi di session selezionabili sono " flusso" o " host" (tramite l'opzione " *–q type*").

Quando il protocollo di trasporto è TCP, i biflussi tipicamente approssimano le connessioni TCP. Questa euristica è semplice e leggera, tuttavia sono necessarie delle informazioni aggiuntive per identificare le connessioni più accuratamente. Altre euristiche opzionali basate sui flag sono state dunque introdotte per seguire lo stato delle connessioni TCP [3]:

- se il primo pacchetto di un biflusso TCP non contiene un flag SYN, allora questo viene trascurato (utile per filtrare le connessioni iniziate prima della cattura)
- la creazione di un nuovo biflusso è forzata se il pacchetto TCP ricevuto contiene solo un flag SYN
- un biflusso è forzato a scadere se viene rilevato un flag FIN in entrambe le direzioni
- il timeout di inattività è disabilitato sui biflussi TCP (essi scadono solo se il FIN è rilevato)

Inoltre, sempre di default, il tool opera in **modalità** " offline" e cioè fornisce le informazioni riguardanti la classificazione solo quando la sessione termina. Altre modalità selezionabili sono quella " ciclica" o " realtime" (opzione " *-m mode*").

Il plugin utilizzato nell'ambito di questo studio è " **nDPIng**" che opera una classificazione di tipo DPI basata su diverse features quali il contenuto e la dimensione del payload, la dimensione dei pacchetti, il numero di righe del payload e i certificati SSL.

3 CATTURA DEL TRAFFICO

In laboratorio è stato generato del traffico da un terminale Android collegato tramite interfaccia fisica al sistema di cattura. Per migliorare la precisione della classificazione sono state installate solo le applicazioni utili alla generazione del traffico (nel caso in esame: Google Play Store, Facebook e Messenger).

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 *scrittura* e *lettura* su **socket**, e viene fatto usando una specifica versione di **strace** che porta in output un file " strace.out" .

Al 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" .

4 DESCRIZIONE DEL PROGRAMMA RDCS

Lo script sviluppato (RDCS.sh) si propone di classificare il traffico catturato in laboratorio utilizzando il tool TIE, e validare la classificazione prodotta. In una prima fase della validazione si verifica che l'etichetta assegnata a ciascun flusso sia corretta; in seguito vengono fornite informazioni aggiuntive o correttive alla classificazione.

Il programma è costituito da un unico script scritto in linguaggio bash, ed eseguibile su una qualsiasi distribuzione Linux, che prevede tre opzioni di funzionamento:

1) **Classificazione:** utilizza il tool TIE per classificare i biflussi;

2) **Convalida**: confronta la classificazione con informazioni reperite dai file di log;

3) **Raffinamento**: corregge la classificazione o fornisce informazioni aggiuntive.

4.1 Preparazione ed esecuzione

Come detto in precedenza, lo script deve essere eseguito su sistema operativo Linux in quanto sono stati utilizzati comandi Linux per analizzare e processare i file.

Sono necessarie delle operazioni preliminari:

1) Aggiungere alla variabile **PATH** il percorso della cartella contenente il tool TIE
   echo export PATH=" $ PATH" :/TIE_path/bin » ∼ /.bashrc
   Per verificare l'aggiunta del percorso è possibile eseguire: echo $ PATH

2) Individuare il percorso della cartella contenente i file di cattura

3) Utilizzare il programma RDCS.sh

Lo script è eseguibile da terminale utilizzando la seguente sintassi:

./RDCS.sh [opzione][data_path]

4.2 Opzioni di funzionamento

CLASSIFICAZIONE [-t]

Per utilizzare correttamente questa opzione, l' argomento [data_path] deve essere una directory contenente le subdirectory dove sono salvate le tracce di traffico in formato " pcap" catturate in laboratorio. Queste tracce vengono classificate tramite il tool TIE usando nello script la seguente sintassi:

./TIE -a ndping_1.0 -r <path_to_pcap_file> -d <output_path>

- L'opzione " -r" permette di leggere il traffico da una traccia in formato pcap;
- L'opzione " -a" abilita solo i plugin specificati nella lista;
- L'opzione " -d" permette di specificare il path del file di output;
- Senza specificare l'opzione " -m" il tool opera di default in modalità offline;
- Senza specificare l'opzione " -q" il tool opera di default una classificazione su sessioni di tipo biflusso.

Il comando descritto, con il plugin **ndping** abilitato, permette di creare un *ground truth file* basato sulla deep packet inspection. Viene quindi operata una classificazione sulla base del payload dei pacchetti.

L'esecuzione di tale comando porta in output un file "class.tie" contenente per ogni riga una tupla associata ad un determinato biflusso del file pcap, con la corrispondente classificazione (app_id, app_detail)

Ad ogni singola classificazione è assegnato un valore di affidabilità dal classificatore stesso (campo "confidence"). Nel caso in cui sia attivo un unico plugin, il campo può assumere valore "0" o "100" con semantica rispettivamente "non classificato" o "classificato" .

VALIDAZIONE [-v]

L'argomento [data_path] deve corrispondere al path della cartella "output" generata con l'opzione [-t]. Si opera, con questa opzione, una validazione delle classificazioni precedentemente ottenute. Confrontando i file di

log (*strace.log*) con i file di classificazione (*class.tie*) è possibile valutare qualitativamente e quantitativamente l'operato del tool TIE.

Sono aggiunte ai file *class.tie* le colonne ***app_log*** e ***validation***.

Vengono processate riga per riga le 5-tuple **[ip_src, ip_dst, prt_src, prt_dst, proto]** contenute nel file *class.tie* e, per diminuire la complessità computazionale, vengono validati tutti i biflussi che hanno un valore di affidabilità pari a 100 (campo " confidence"). Per tutti gli altri biflussi non classificati, nella colonna **validation** viene aggiunta la stringa " NC" .

Si cerca una corrispondenza tra le 5-tuple del file class.tie e quelle contenute nel file *strace.log*. Nel caso di corrispondenze trovate, viene recuperato dalla prima colonna nel file log il nome del processo che ha generato il biflusso inerente alla tupla in esame; questo è inserito nella colonna **app_log** della riga in esame nel file tie.

Nella stessa riga del file class.tie la colonna **app** (già esistente) individua l'applicazione che, secondo il plugin di classificazione, ha generato quel biflusso.

Prendendo in esame quest'ultima colonna (*app*) e confrontandola con la colonna aggiunta (*app_log*) è possibile determinare se il plugin ha classificato correttamente il biflusso. Se viene riconosciuto un nome simile tra le due stringhe, nella colonna **validation** viene aggiunta la stringa " OK" , altrimenti viene aggiunta la stringa " N" (fatta eccezione per " Google" che viene associato manualmente al processo " com.android.vending").

Alcuni biflussi classificati da TIE possono non corrispondere ad alcuna tupla all'interno del file di log. La convalida di tali classificazioni non è attuabile sulla base delle sole informazioni recuperabili dal log. Per tali righe si aggiunge nella colonna di validazione la stringa " NV" (non validabile).

Le righe che corrispondono a biflussi di tipo non TCP vengono trattate diversamente dato che solitamente corrispondono a richieste DNS. Nella colonna di validazione è aggiunta la stringa " NOT_TCP" .

Esempio di validazione:

confidence	**app_details**	**proto**	**app_log**	**Validation**
0	Other TCP	6 (TCP)	com.facebook.katana	**NC**
100	Facebook	6 (TCP)	com.facebook.katana	**OK**
100	Google	6 (TCP)	com.facebook.katana	**N**
100	Facebook	6 (TCP)	-	**NV**
100	DNS	17 (UDP)	**-**	**NOT_TCP**

RAFFINAMENTO [-r]

Vengono ulteriormente esaminati tutti e soli i biflussi che hanno ricevuto, tramite l'opzione [-v], una validazione con valore " NC" , " NV" , " N" o " NOT_TCP" .

Tramite il programma tshark viene esaminato il file traffic.pcap e vengono estrapolate, per ogni biflusso, informazioni relative al proprietario dell'IP (campo ip.geoip.asnum), le informazioni relative ad eventuali **richieste dns** (campi dns.a, dns.qry.name) e l'**SNI** (campo ssl.handshake.extensions_server_name).

- SNI sta per Server Name Indication e consiste in un'estensione del protocollo di rete TLS con cui un client indica a quale *hostname* si sta tentando di connettersi all'inizio del processo di handshake.

Per una corrispondenza univoca delle tuple tra il file " pcap" e il file " tie" sono state utilizzate come chiavi i campi **frame.time_epoch** (pcap) e **t_start** (tie).

Il refining del file è stato effettuato come segue; per un determinato biflusso del file tie la cui validazione non ha ricevuto esito " OK" :

- Se l'analisi con tshark ha fornito informazioni relative al proprietario dell'indirizzo IP di destinazione (o sorgente) del flusso in esame, allora sulla relativa riga del file class.tie viene aggiunta quella stringa nella colonna **net_name**;

- Se il programma **dig** ha restituito una risoluzione per l'indirizzo IP di destinazione (o sorgente) del flusso, allora sulla relativa riga viene aggiunta la stringa nella colonna **name_resolution**

- Se l'analisi ha determinato un valore per il campo " dns.a" o " dns.qry.name" vuol dire che il flusso in esame si riferisce ad una richiesta (ai server di Google) di risoluzione nome.

Nell'ultimo caso, il tool TIE solitamente fornisce il nome " Google" come classificazione. Invece l'analisi della traccia pcap fornisce l'alias del server per il quale è stata richiesta la risoluzione e il relativo indirizzo. Questo nome viene inserito nella colonna " **dns**" .

Al termine di questa fase è effettuato un conteggio delle classificazioni che erano risultate non validabili tramite i file di log (categoria " NV") e ora risultano *validate* considerando le informazioni aggiunte sul **net_name**.

Esempio di classificazione validata con raffinamento:

app_details	confidence	app_log	validation	net_name
Facebook	100	-	NV	Facebook, Inc.

4.3 Modifica o aggiunta di classificazione

Tutte le informazioni aggiunte in fase di raffinamento (net_name, name_resolution, dns) possono essere utili alla validazione della classificazione o al fine di analizzare l'indirizzamento del traffico ma sono poco significative in funzione di una riclassificazione dei biflussi. Considerando che un'applicazione può generare traffico anche verso *servizi di terze parti*, risulta impossibile classificare un biflusso conoscendo solo i server di destinazione o le eventuali richieste di risoluzione nomi.

L'unico dato a disposizione utile a riclassificare un biflusso è il log dell'applicazione salvato nel file " strace.log" . Per tutti i biflussi che non hanno ricevuto una classificazione (campo *validation*=" NC") viene prelevato il nome del processo che ha generato quel determinato biflusso dal file di log, riformattato ed aggiunto alla colonna **new_classification**.

Questa operazione è effettuata nell'ambito dell'*opzione di raffinamento*.

Esempio di riclassificazione:

app_details	confidence	app_log	validation	net_name	new_classification
Other TCP	0	com.facebook.katana	NC	Amazon.com	Facebook

5 ANALISI DEI RISULTATI

Al termine dell'esecuzione dello script RDCS è prevista la scrittura e la stampa automatica di un report che schematizza i risultati di classificazione, validazione e raffinamento.

Di seguito è riportata una sezione del report inerente il traffico analizzato:

ven 22 dic 2017, 00.56.37, UTC

VALIDATION REPORT
tempo di elaborazione: 0 min 14 sec
biflussi processati: 738
biflussi classificati: 412
biflussi non classificati: 326

biflussi OK: 41
biflussi N: 0
biflussi NV: 150
biflussi NOT_TCP: 221

REFINING REPORT
tempo di elaborazione: 1 min 15 sec
biflussi VAL: 138
OK: la classificazione della tupla è stata validata con esito positivo confrontando con il file di log.
N: le informazioni fornite dal file di log non sono sufficienti per validare la classificazione della tupla.

NV: la classificazione non è validabile tramite il file di log in cui non vi sono riferimenti alla tupla.
NOT_TPC: biflussi con protocollo non TCP non validabili tramite log.
VAL: classificazioni precedentemente non validate con i file di log e ora validate tramite l'analisi dei file pcap

5.1 Biflussi classificati, non classificati e riclassificati

In primo luogo è necessario notare che, tra quelli processati, 412 biflussi su un totale di 738 (circa il 56%) sono stati classificati utilizzando il plugin **nDPIng** (conteggiati sulla base del valore del campo " confidence").

Tutti i biflussi non classificati, che corrispondono quindi al 44% di quelli analizzati, vengono riclassificati sulla base delle informazioni del log come descritto nel paragrafo 4.3.

Considerando che il plugin è basato sulla *DPI* e quindi sull'ispezione del payload è ragionevole che il traffico indirizzato a server di terze parti non sia stato classificato.

Di seguito alcuni esempi di biflussi non classificati:

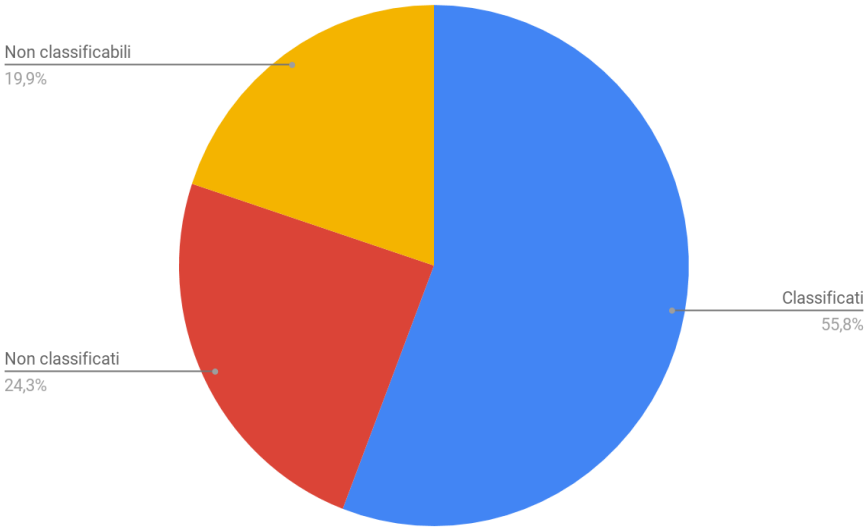
app_details	**confidence**	**app_log**	**validation**	**net_name**
Other TCP	0	com.facebook.orca:notification	NC	Google Inc.
Other TCP	0	com.facebook.katana	NC	Amazon.com, Inc

Di seguito sono indicate le occorrenze, tra i biflussi non classificati, delle reti di destinazione:

```
NET OCCURRENCES FOR NC BIFLOW
    117 Amazon.com, Inc.
      7 Consortium GARR
     97 Facebook, Inc.
      3 Fastly
     82 Google Inc.
     20 NOTFOUND
```

Tra queste, le 97 occorrenze di " Facebook, Inc" sarebbero dovute essere classificate come " Facebook" e le 82 di " *Google Inc*" come " Google" o " DNS" ; in totale sono 179.

È possibile affermare dunque che il failure rate del plugin, sul traffico analizzato, è circa del 24% (179 su 738).

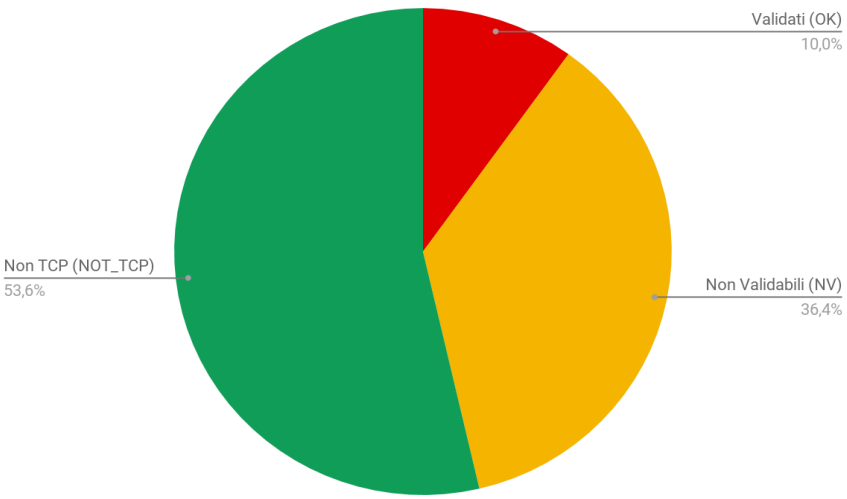


5.2 Biflussi validati, non validati, non validabili, non TCP

Come descritto nel paragrafo 4.2, lo script effettua una validazione della classificazione sulla base dei file di log.

I biflussi la cui classificazione è stata validata con esito positivo tramite il log risultano essere circa l'10% dei classificati (41 su 412), mentre circa il 36% (150 su 412) dei biflussi risultano non essere validabili utilizzando il log.

All'ultima categoria, invece, corrispondono i biflussi con protocollo diverso dal TCP e rappresentano circa il 53% (221 su 412) dei flussi totali classificati.



5.3 Richieste DNS

La maggior parte dei biflussi " non TCP" corrisponde a traffico DNS e consiste in richieste di risoluzione di nomi ai server di Facebook o ad alcuni server di terze parti. Di seguito sono riportare le occorrenze delle risoluzioni richieste:

```
5   2.android.pool.ntp.org          1   media1.giphy.com
2   account.cyngn.com               1   media2.giphy.com
2   android.clients.google.com      1   media3.giphy.com
13  api.facebook.com                1   media.tenor.co
4   b-api.facebook.com              1   m.facebook.com
5   b-graph.facebook.com            17  mqtt-mini.facebook.com
4   b-www.facebook.com              6   play.googleapis.com
1   cdn.fbsbx.com                   3   portal.fb.com
17  edge-mqtt.facebook.com          1   r2---sn-b5gg-ca9e.gvt1.com
4   external-mxp1-1.xx.fbcdn.net    1   r3---sn-b5gg-ca9e.gvt1.com
29  graph.facebook.com              1   rupload.facebook.com
3   h.facebook.com                  1   safebrowsing.googleapis.com
1   lh3.ggpht.com                   14  scontent-mxp1-1.xx.fbcdn.net
1   lh4.googleusercontent.com       11  scontent.xx.fbcdn.net
1   lh5.googleusercontent.com       1   shopvac.cyngn.com
1   lh6.googleusercontent.com       15  static.xx.fbcdn.net
4   lithium.facebook.com            3   stats.cyanogenmod.org
2   lookaside.facebook.com
```

5.4 Raffinamento della classificazione

Come descritto nel paragrafo 5.2, tutti i biflussi che non hanno ottenuto una validazione con valore " OK" vengono analizzati con diverse tecniche, quali risoluzione DNS inversa o ispezione dei pacchetti tramite tshark.

In particolare vengono analizzate le seguenti categorie di biflussi:

biflussi con validazione " N"	0
biflussi con validazione " NV"	150
biflussi con validazione " NOT_TCP"	221
biflussi non classificati " NC"	326
totale	679

Notiamo che è necessaria una ulteriore analisi per circa il 92% dei biflussi totali esaminati (679 su 738).

Nel valutare questa percentuale dobbiamo però considerare che una grande parte dei biflussi rianalizzati corrisponde a biflussi già classificati ma che non hanno riscontri nei file di log (categoria NV).

Se il sistema di cattura avesse salvato informazioni su tutti i biflussi, e avessimo considerato dunque solo i biflussi con validazione " N" , quelli non classificati " NC" e quelli " NOT_TCP" , sarebbe stata necessaria un'ulteriore analisi solo per il 74% dei flussi totali (547 su 738).

Con le informazioni aggiuntive fornite in fase di raffinamento è stato possibile validare un altro blocco di classificazioni che non erano state validate tramite log (categoria " NV"). È emerso che in seguito al raffinamento altre **138 classificazioni sono state validate** con esito positivo.

6 SERVIZI DI TERZE PARTI

Studiare l'origine del traffico significa individuare la percentuale del traffico di rete che proviene dai server di proprietà del fornitore dell'app. Questa metrica è di particolare interesse per la privacy dell'utente, poiché è un'indicazione del controllo che il fornitore ha sui dati dell'app. [5]

Per quest'ultima analisi nello script si prevede di associare ad ogni biflusso l'applicazione che l'ha generato sulla base del campo *app_log* o, se quest'ultimo risulta vuoto, *new_classification* ed individuare l'origine di quel biflusso tramite il campo *net_name*. Di seguito un estratto del report:

```
TRAFFIC ORIGIN
      app     net
    4 -       Amazon.com, Inc.
  194 DNS     Google Inc.
   20 Facebook
  113 Facebook        Amazon.com, Inc.
    7 Facebook        Consortium GARR
  218 Facebook        Facebook, Inc.
    3 Facebook        Fastly
   85 Facebook        Google Inc.
    1 Google  Amazon.com, Inc.
   54 Google  Google Inc.
    3 HTTP    Facebook, Inc.
    6 MDNS
    1 NTP     Boxed IT Ltd.
    1 NTP     Electronic Communities Ltd
    1 NTP     Interoute Communications Limited
    1 NTP     Metrolink S.R.L.
    3 NTP     Prometeus di Daniela
    2 RTCP
    2 SSL_with_certificate    Amazon.com, Inc.
    2 SSL_with_certificate    Consortium GARR
    2 SSL_with_certificate    Facebook, Inc.
    3 SSL_with_certificate    Fastly
   12 STUN    Facebook, Inc.
```

Durante l'utilizzo dell'applicazione " Facebook" quasi il 50% del traffico proviene dai server di proprietà Facebook, mentre l'altra metà del traffico proviene da server di terze parti.Per classificare il traffico in diverse categorie (es. cloud piuttosto che ads network) è necessario identificare l'entità da cui proviene ed effettuare delle ricerche in rete.

- **CDN e Cloud Traffic:** Questa categoria include la percentuale del traffico che proviene dai server di CDN o dai cloud providers. Durante l'utilizzo di Facebook 3 biflussi provengono da *Fastly (CDN platform)* e 113 da *Amazon.com* (si può ipotizzare 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 Facebook/Messenger questo traffico è da attribuirsi ai servizi di Google con cui le applicazioni comunicano, quali maps, ads, analytics e Google App Engine. 85 biflussi associati all'app Facebook provengono da Google.

RIFERIMENTI

1. Alberto Dainotti, Antonio Pescape', and Carlo Sansone: *Early Classification of Network Traffic through Multi -classification*.

2. Rottondi Cristina. Tesi in: *Classificazione del traffico internet con attributi per-source*.

3. Alberto Dainotti, Walter de Donato, Antonio Pescapé, and Giorgio Ventre: *TIE: a community-oriented traffic classification platform*.

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

5. Xuetao Wei, Lorenzo Gomez, Iulian Neamtiu, Michalis Faloutsos: *ProfileDroid: Multi-layer Profiling of Android Applications*.

6. Giuseppe Aceto, Domenico Ciuonzo, Antonio Montieri, Antonio Pescapé: *Traffic Classification of Mobile Apps through Multi-classification*.