Gestione Reti - Twitch.tv dissector

Info

- Edoardo Alberto Dominici
- 501230

Il progetto consiste nell'implementazione di un dissector per *Twitch.tv*. Ovvero un modulo per nDPI che consente di capire dato un singolo pacchetto oppure una serie di pacchetti (*flow*) se lo stesso *flow* appartiene al servizio offerto da *Twitch.tv*. Il dissector sara' implementato come sottoprotocollo di HTTP. *Twitch.tv* e' un servizio di streaming video che e' tra le 15 applicazioni che utilizzano piu' banda nel mondo e quarta negli US. E' possibile usufruire del servizio esclusivamente tramite l'interfaccia web (`www.twitch.tv`) oppure la sua applicazione per smartphone.
Se andiamo ad analizzare il traffico da e verso *Twitch.tv* si puo' comporre di diverse parti :

- Contenuto pagina web
- Download stream
- Upload stream
- Chat

Eccetto per il contenuto della pagina web che e' identico a quello di un'altro qualsiasi sito web verranno visti gli altri 3. Per identificare il traffico della pagina e' sufficiente aggiungere come hostname `.twitch.tv` e la maggior parte del traffico verra' catturato. Dico la maggior parte poiche' ogni elemento della pagina web fa riferimento ad altri servizi, ad esempio analizzando il traffico si vedono diversi flow rivolti a CDN che fungono da cache per il contenuto statico e dinamico (script). Ne vengono citati alcuni in *Altri metodi di riconoscimento*.

Download stream

La visualizzazione dello stream tramite l'interfaccia web e' possibile accendendo all'URL `www.twitch.tv/username` , dove `username` non e' altro che il nome del canale che si vuole seguire, ottenendo cosi' una gerarchia piatta, senza l'esistenza di sottocanali o altro. Come si vedra' nell'ultima parte sulla chat, ogni canale corrisponde a un canale IRC. Per la visualizzazione del contenuto viene utilizzato un web player Flash, vi sarebbe anche un player HTML5 via HLS, tuttavia dal lato desktop nessun browser moderno lo supporta ed e' li solamente in relazione ad iOS : *When streaming video content over a cellular network with a duration lasting longer than 10 minutes, your application must use HTTP Live Streaming.* . Essendo Flash ancora supportato ed embeddato nativamente in Chrome (share piu' alto) non vi sono problemi di non supporto. Avvenendo tutto tramite il player flash, dopo che questo viene scaricato ed avviato lo script si connettera' a un host *Twitch.tv* che servira' lui il video.
Il trasferimento del video avviene tramite protocollo HLS. HLS e' l'acronimo di HTTP Live Streaming e come e' possibile intuire si basa su http per il funzionamento e su normali richieste `GET` . In particolare possono essere di due tipi :

Richiesta frammento

Un frammento e' una serie di frame di una specifica durata, con un certo formato e bitrate e hanno estensione `.ts` (*transport stream*). Twitch solitamente utilizza frammenti della durata di circa 4s, che pero' puo' variare in base ad appunto formato e bitrate. La richiesta di un frammento solitamente e' una `GET` sulle linee di:
[NOTA] Con formato viene intesa qualita' (bassa / media / alta / sorgente)

```
GET /hls-835440/mmorpgrs_14890958752_258065284/medium/index-0000004960-IEKB.ts HTTP/1.1
Host: video24.fra01.hls.ttvnw.net
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/43.0.2357
X-Requested-With: ShockwaveFlash/18.0.0.160
Accept: */*
Referer: http://www.twitch.tv/mmorpgrs
Accept-Encoding: gzip, deflate, sdch
Accept-Language: it-IT,it;q=0.8,en-US;q=0.6,en;q=0.4
```

L'indirizzo della `GET` e' ottenuto dalla *richiesta playlist* (spiegata successivamente). Per poter identificare che si tratta di *Twitch.tv* sono due gli elementi che sono presenti in ogni richiesta :

- Host line
- Referer line

La prima contiene l'indirizzo dell'host che, anche se gli indirizzi sono quasi identici non va confuso con gli *ingestion servers* visti nella sezione successiva. Essendo un pacchetto su porta 80 HTTP si occupa di catturarlo di default e lo andrebbe ad identificare come suo se noi non ne prendessimo il possesso. Le opzioni sarebbero quindi di aggiungere una serie di case in base all'hostname direttamente all'interno del dissector http per verificare l'indirizzo del server. Tuttavia e' sufficiente aggiungere alla lista degli host `.ttvnw.net` poiche' fino a prova contraria (testando i server) gli URL si compongono come:

```
[video id che lo contiene].[server].hls.ttvnw.net
```

(Nota : si sarebbe potuto aggiungere anche `.hls.ttvnw.net`).
[NOTA] : Video id e' come se fosse un indice per server, non ve ne e' uno per utente. Il server inoltre indica uno dei raggruppamenti di server (e range IP), esempio `fra01` e' riferito a Frankfurt. Non e' possibile aggiungere l'indirizzo IP nella lista di network host poiche' il lookup per IP viene fatto in `guess_undetected_protocol` e non in altre parti del codice. Quindi, visto che ci troviamo comunque in un caso di un pacchetto HTTP ben formato verrebbe riconosciuto come `NDPI_PROTOCOL_HTTP` .
Una volta ricevuto l'indice per il frammento, viene fatta un'altra richiesta `GET` che questa volta avra' il vero compito di scaricare il video in formato `Content-Type: video/mp2t` che poi verra' combinato con gli altri all'interno del player.

Richiesta playlist

Una playlist e' un file salvato come `name.M3U8` (estensione del formato `.m3u` utilizzato per le playlist MP3) che contiene una lista di `index files` chiamati anche *segmenti*. ciascuno dei quali contiene un tag precedente al nome che identifica la sua durata. Sono ordinati ed e' possibile capire a quale frammento di tempo si riferiscono calcolandolo in base all'*elapsed seconds*.
Facciamo un esempio : Questo e' un frammento di una playlist di uno stream Twitch.tv contenuto all'interno di una risposta HTTP a una `GET` :

```
#EXTM3U
#EXT-X-VERSION:3
#EXT-X-TARGETDURATION:5
#ID3-EQUIV-TDTG:2015-06-16T12:34:41
#EXT-X-TWITCH-ELAPSED-SECS:19818.754
#EXT-X-TWITCH-TOTAL-SECS:19846.754
#EXT-X-MEDIA-SEQUENCE:4955
#EXTINF:4.000,
index-0000004955-j1F2.ts
#EXTINF:4.000,
index-0000004956-nroQ.ts
#EXTINF:4.000,
index-0000004957-STdO.ts
#EXTINF:4.000,
index-0000004958-Jo8A.ts
#EXTINF:4.000,
index-0000004959-zrfA.ts
#EXTINF:4.000,
index-0000004960-IEKB.ts
#EXTINF:4.000,
index-0000004961-I0br.ts
```

In questa playlist sono quindi indicizzabili `4 * 7` secondi di filmato, questo e' anche possibile capirlo dalla differenza di `#EXT-X-TWITCH-TOTAL-SECS` - `#EXT-X-TWITCH-ELAPSED-SECS` . Questo significa che il frammento n-esimo conterra' il video dal secondo `ELAPSED-SECS + n * 4` per i successivi 4 secondi. (Questo esempio ovviamente tiene conto dell'eguaglianza in durata dei frammenti. Il web player richiede periodicamente una playlist per vedere se qualche vi e' qualche frammento aggiornato, in caso positivo inizia a scaricarlo, questo vuol dire che nella richiesta alla playlist successiva lui si e' trovato tutti i frammenti precedenti shiftati di una posizione indietro con un nuovo *index* in cima alla coda e avendo perso l'ultimo che era presenta nella richiesta precedente. La richiesta della playlist viene fatta allo stesso server che contiene il video infatti gli indici non sono altro che nomi di file all'interno dello stesso server. Essendo i due esempi di HLS presi da uno stesso stream e' possibile vedere come nella richiesta

HTTP del frammento si trovi `GET /hls-835440/mmorpgrs_14890958752_258065284/medium/ index-0000004960-IEKB.ts` che non e' altro che il sesto frammento nella richiesta della playlist. Essendo la richiesta di una playlist esattamente identica a quella di un frammento eccetto per il tipo di file richiesto, per la sua identificazione e' possibile utilizzare lo stesso metodo visto precedentemente. Anzi, se il dissector era gia' in funzione quando uno stream e' stato aperto, e' molto piu' probabile che sia stata questo il tipo di richiesta che lo identifichera' come HTTP/*Twitch.tv*.

Protocollo HLS

Visto che Twitch.tv utilizza il protocollo HLS qualsiasi altra informazione puo' essere trovata all'interno del draft di proposta all'Internet Foundation.

Upload stream

Con upload stream viene intesa il momento in cui non sei tu ad usufruire del contenuto prodotto da altri, ma invece sei tu stesso a produrlo, overro mandando in stream un video proveniente dal tuo computer stesso o da altri device. Per diventare streamer vi e' bisogno di un software che si occupa di fare l'upload, esempi sono XSplit Broadcaster oppure Open Broadcaster Software. Teoricamente per cercare di capire se un certo pacchetto e' diretto a Twitch.tv con l'intenzione di essere mandato in stream dovremmo controllare il payload dello stream TCP ed vedere se qualche informazione ci riconduce a loro. Perfortuna non e' necessario analizzare il protocollo utilizzato (RTMP o variante) poiche' *Twitch.tv* ha una serie di *ingest server* pubblici. Il compito degli ingest server (dislocati in tutto il mondo) e' quello di ricevere i dati del singolo streamer e poi mandarli possibilmente ad altri server dove gli utenti si potranno connetttere e tramite HLS scaricare il video corrente.
La lista degli ingest server e' ottenibile dalla loro API pubblica. Il numero e' 22 al momento della scrittura, quindi abbastanza contenuto. Per riconoscerli, basterebbe teoricamente aggiungere i loro indirizzi ip alla lista contenuta in `ndpi_content_match.c.inc` affinche' vengano identificati arrivando al controllo per IP. Per far si che l'upload stream venga identificato ci basta quindi aggiungere la lista di indirizzi alla `host_protocol_list` :

```
{ Indirizzo in forma esadecimale, CIDR, protocollo }
```

Essendo indirizzi unici il CIDR e' sempre 32.

Chat

Per la chat *Twitch.tv* utilizza un protocollo IRC infatti e' possibile utilizzare un qualsiasi client IRC per connettersi. nDPI contiene gia' un dissector IRC quindi utilizzando un client di terze parti il traffico verrebbe riconosciuto correttamente. Trovandosi all'interno di uno stream, identificato come `www.twitch.tv/username` Il traffico della chat ricade all'interno di quello della pagina stessa eccetto per alcuni dati. All'interno della chat e' infatti possibile utilizzare emoticon ed altre immagini, queste pero' non sono date dai server di Twitch.tv, per comodita' vengono scaricate (sempre se possibile) dal CDN di *Twitch.tv* / *Justin.tv*, il cui hostname e' `static-cdn.jtvnw.net` che contiene tutti i contenuti statici del network di *JustinTV* che era il nome precedente di *Twitch.tv*. Le richieste di questi contenuti cadono sotto HTTP (80) e sono richieste `GET` normali. e.g. :

```
GET /emoticons/v1/47303/1.0 HTTP/1.1
Host: static-cdn.jtvnw.net
Connection: keep-alive
Accept: image/webp,*/*;q=0.8
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/42.0.2311.152
Referer: http://www.twitch.tv/isamuxlive/chat?popout=
Accept-Encoding: gzip, deflate, sdch
Accept-Language: en-US,en;q=0.8
```

Un estratto della chat IRC :

```
CAP REQ :twitch.tv/tags twitch.tv/commands
..PASS blah
..NICK justinfan216548
..:tmi.twitch.tv CAP * ACK :twitch.tv/tags twitch.tv/commands
.:tmi.twitch.tv 001 justinfan216548 :Welcome, GLHF!
.:tmi.twitch.tv 002 justinfan216548 :Your host is tmi.twitch.tv
.:tmi.twitch.tv 003 justinfan216548 :This server is rather new
.:tmi.twitch.tv 004 justinfan216548 :-
.:tmi.twitch.tv 375 justinfan216548 :-
.:tmi.twitch.tv 372 justinfan216548 :You are in a maze of twisty passages, all alike.
.:tmi.twitch.tv 376 justinfan216548 :>
.JOIN #mushisgosu
..:justinfan216548!justinfan216548@justinfan216548.tmi.twitch.tv JOIN #mushisgosu
.:justinfan216548.tmi.twitch.tv 353 justinfan216548 = #mushisgosu :justinfan216548
.:justinfan216548.tmi.twitch.tv 366 justinfan216548 #mushisgosu :End of /NAMES list
.:jtv MODE #mushisgosu +o mushisgosu
.:jtv MODE #mushisgosu +o thedemiigod
.:jtv MODE #mushisgosu +o xanbot
.:jtv MODE #mushisgosu +o bauwsbot
.:jtv MODE #mushisgosu +o liekabauws
.:jtv MODE #mushisgosu +o lolgeranimo
.:jtv MODE #mushisgosu +o hitomicky
.:jtv MODE #mushisgosu +o mrs_atreides
.:jtv MODE #mushisgosu +o sophiasapphire
.@color=#0000FF;display-name=bejergesen;emotes=;subscriber=0;turbo=0;user-type= :bejergesen!bejergesen@
.@color=;display-name=BeCareful2014;emotes=25:9-13;subscriber=0;turbo=0;user-type= :becareful2014!becar
.@color=#5F9EA0;display-name=AddiiccT;emotes=;subscriber=0;turbo=0;user-type= :addiicct!addiicct@addiic
.@color=#8A2BE2;display-name=Risingdeath47;emotes=;subscriber=0;turbo=0;user-type= :risingdeath47!risin
.@color=;display-name=Zerantur;emotes=;subscriber=0;turbo=0;user-type= :zerantur!zerantur@zerantur.tmi.
```

Prima di tutto io sono acceduto alla chat come visitatore senza essermi loggato, quindi come se fossi anonimo, dandomi come user `justinfan216548` ed password `blah`, viene fatto il join al canale corrispondente allo streamer `mushigosu` e successivamente dopo avermi dato una lista di moderatori, ovvero utenti con privilegi maggiori inizia la lista di messaggi.

```
.@color=;display-name=BeCareful2014;emotes=25:9-13;subscriber=0;turbo=0;user-type= :becareful2014!becar
```

La maggior parte delle informazioni contenute dicono come formattare il messaggio, il colore del nome utente e quali emoticon utilizza, segue infinite il messaggio al canale dello streamer. Anche se utilizzasse qualche notazione particollare e' IRC standard (come mostrato dal link precedente).

Altre metodi di riconoscimento.

Essendo Twitch.tv il 152esimo sito web piu' visitato al mondo e 74esimo in US (6/16/2015). Utilizza come supporto uno o piu' CDN per assicurare un'esperienza migliore all'utente (cit). Per questo alcuni dei suoi indirizzi CDN da aggiungere alla lista di host name per essere identificato sono :

- `static-cdn.jtvnw.net` (Immagini principalmente)
- `www-cdn.jtvnw.net` (Script principalmente)

Ma sicuramente ve ne sono molti altri e identificarli richiederebbe piu' test, una volta trovati e' sufficiente aggiungerli alla lista degli host e verranno identificati correttamente sotto *Twitch.tv*. Molte di queste informazioni sono tuttavia temporanee e come tutte le liste di indirizzi IP/ hostname andrebbero aggiornate. Oltre a questo *Twitch.tv* ha una propria API pubblica con alcuni dei seguenti indirizzi :

- `api.twitch.tv`
- `spade.twitch.tv` (utilizzato da quanto trovato per ottenere il file `crossdomain.xml`)

Che fortunatamente ricadono sempre nella sottostringa `.twitch.tv` utilizzata come hostname per identificare il download del contenuto delle pagine web.