Identifying Sessions to Websites as an Aggregation of Related Flows

Luis Miguel Torres, Eduardo Magaña, Mikel Izal and Daniel Morato
Departamento de Automática y Computación,
Universidad Pública de Navarra,
Pamplona, Navarra, Spain.
Email: [luismiguel.torres, eduardo.magana, mikel.izal, daniel.morato]@unavarra.es

***Abstract*—In the field of traffic classification, previous efforts have been centered on identifying applications (HTTP, SMTP, FTP, etc) rather than the actual services that they provide (e-mail, file transfer, video streaming, etc.). Nowadays, however, a single application as HTTP can provide multiple services for the end-user. Network traffic for a web-based service is composed by a characteristic pattern of flows rather than characteristic individual flows. In this paper we study the traffic of different web-based services (webmail, social networks, video streaming and online newspapers) as summarized by Netflow-type records. A method that clusters flows that belong to the same web session using only the data provided by those records is also proposed. The obtained clusters will contain more information about the service whose traffic they comprise, and thus it will be easier to assign those clusters to the right service.**

***Index Terms*—web and Internet services, web sites, web mining, clustering methods.**

I. Introduction

As of today, a wide variety of services are provided through the Internet (e-mail, video streaming, P2P file sharing, social networks, etc). Identifying the traffic generated by them is interesting in order to gather data about the use of the networks, prioritize important services or block undesired ones. Traditionally service and application were terms that could be used interchangeably. Because of this, most proposals in the literature deal with application identification as this was enough to find the services in the traffic. This is not true anymore as some applications now provide multiple services. For example, web traffic can be related to a myriad of different services from e-mail to video streaming as the web browser is more and more used as the sole interface with which the user accesses the Internet.

Identifying applications has been traditionally possible by simple port number inspection. Nowadays, however, this is not a reliable technique as some new applications use random (or deliberately misleading) ports in order to avoid firewalls or detection. Signature-based methods and, more recently, behaviour-based ones are now popular. The former yield good results but need constant updates, fail against encryption and raise privacy concerns; the latter are somewhat less reliable and difficult to tune but offer a broader spectrum of detection and their popularity is rising. Our work is orientated towards behaviour-based techniques as we believe that their advantages outweigh their limitations.

In any case, most proposals in the literature try to classify flows into applications by their individual characteristics. Nevertheless, most applications open multiple related connections. Individually, some of them may not be distinctive enough but together they can offer an accurate signature of the application's operation. Because of this, the performance of the classification process could be improved if the objective was to classify not individual flows, but groups of them that belong to the same service. The objective of this paper is, precisely, to present a method that attempts to group flows that belong to the same web sessions into clusters that are easier to classify.

For the moment, we have decided to limit our study to web traffic that, besides traditional web browsing, is nowadays associated with a myriad of other services. As, in the following sections, we will refer only to web traffic, all flows will be TCP. We define a session as a collection of TCP flows generated by the web browser while the user is accessing a specific service. For example, a webmail session would span all the connections opened by the web browser from the moment the user opened the login webpage of the mail service until he or she closes the browser, the tab or opens a different website in the same tab.

When working with web traffic it is important to notice that recent studies [1] show that its characteristics have greatly changed from the ones described thoroughly in the 1990s [2]. This is partially the result of the introduction of HTTP 1.1 (persistent connections and pipelining have made obsolete the notion that every connection comprises a single request/response pair). But, the truth is that webpages have evolved drastically over the last years affecting the profile of web traffic. Because of dynamic content, web analytic tools, content distribution networks (CDN) or client-side processing, the server used by a website will change over time, connections may stay open and thus occur concurrently to others that belong to a different session, different websites may open connections to the same servers or have unexpected traffic profiles. All of this makes clustering flows that belong to the same web session far from a trivial task.

Our clustering algorithm uses only the information that we would get from a NetFlow type record [3]. These flow records provide manageable data and are collected in many links and networks throughout Internet, which makes a real-

978-1-4673-1389-6/12/$31.00

time implementation of our system more feasible, although it sacrifices the possibility of using the more detailed information of a packet trace. In particular, our method relies on the starting and ending timestamps of the flows and the server IP addresses. We do not intend to achieve just one cluster of flows per session because, as previously stated, sessions are very variable in length (depending on the service and the particular user's behaviour) and they often occur concurrently. Rather than that, we will try to group the flows of a session in the fewest clusters while avoiding clustering flows of different sessions.

II. Related Work

We center this section around behaviour-based classification as the method we present in this paper falls in this category. In 2008, Nguyen and Armitage [4] published a thorough survey of different machine learning-based techniques that attempt to assign connections to applications. They distinguished between supervised methods [5], [6], [7], [8] which classify connections into a predetermined number of applications and clustering techniques [9], [10], [11] in which that number is decided by the algorithm. Both points of view have their advantages and drawbacks: supervised methods require more knowledge about the behaviour of the set of applications whose traffic is going to be identified and usually yield good results in this limited set. Clustering techniques discover groups of flows that are similar and may be related to the same application. They are less specific but they can detect new or unknown applications.

When it comes to services rather than applications, there are less precedents as it is a newer field. In the case of web traffic, some work has been done in characterizing the traffic of certain services. For example, the video streaming platform of Youtube.com has gathered a lot of attention [12], [13], [14] but most studies focus on the social characteristics (popularity of videos, etc.) or study the effect of the video codecs in the per packet statistics of the traffic generated. Social networks have also received interest: four of the most popular were studied in [15] by comparing the activity of their users with other HTTP activity. There is also an interesting comparison between normal HTTP traffic and the one generated by certain interactive AJAX-based services in [16]. More similar to our approach, Schatzmann *et al.* [17] try to design a method able to distinguish webmail flows from other HTTPS connections using NetFlow records. They discover that periodicities found in the creation of new connections can be characteristic of specific services and that webmail and POP/SMTP/IMAP servers tend to be in the same networks (thus having near IP addresses). Anyway, identifying the wide variety of services embedded in an application as the web is still a scarcely studied field.

III. Data Collection

We need traffic traces in which every flow is labeled as belonging to a session in order to study the features these flows share and test the inferred clustering method. Traces this thoroughly labeled are very hard to come by. In fact, even labeling a trace manually (by inspecting the payloads of the packets) might be impossible as the payloads are usually anonymized or do not contain enough information to assign the flow to a session.

Because of this, we have prepared a traffic capture and labeling system that can be easily distributed among multiple test users in order to obtain the traces we need. This system involves a portable virtual machine (Windows XP) installed on a USB flash drive that can be executed in any Windows system. The virtual machine offers a controlled environment where any traffic captured should be directly related with web navigation. A simple C# program acts as an interface and allows the user to choose a web browser, type the URL of the desired website and select the type of service that they consider that the website offers. The user is instructed to only follow internal links and not to open any new tabs or different websites. The program captures the generated traffic until the user closes the web browser window. Then flow records are obtained using Argus [18].

We are aware that this approach has a weakness in that it considerably limits the behaviour of the users so the captured traffic may not always correspond to the one they would generate in normal conditions. This is specially true for advanced users who switch faster between websites and usually have more than one tab or browser window open at the same time. Nevertheless, we believe that this is the only way we can correctly assign every flow to a specific web session.

Using this methodology we have captured more than 300 sessions of different websites and services. For this study we have selected four services: web mail (hotmail and gmail, 107 traces), video streaming (youtube, 24 traces), social networks (facebook, tuenti and myspace, 151 traces) and online local newspapers (elmundo and noticiasdenavarra, 46 traces). Each of the traces contains an individual session.

IV. Clustering Parameters

A. Flow Interarrival Times and Flow Overlap

It seems intuitive that flow interarrival times (i.e. the difference between the initial timestamps of two flows) should be useful to cluster flows from the same sessions. We expect to find groups of flows that belong to the same session whose opening timestamps are very near and which are separated from other groups by the longer periods of time related to user actions.

Fig. 1 shows the CCDF of the difference between starting timestamps of consecutive flows. In this case, data from the different websites has been grouped in four bigger service categories (webmail, social network, video streaming and news). The complementary cumulative distribution functions of the interarrival times are calculated for each trace and then aggregated by service. We can see that the interarrival times of almost 50% of consecutive flow pairs that belong to same session are under 100ms. This is a very small period of time to think that both flows in the pair do not belong to the same session. It is also good that the distribution is similar for all

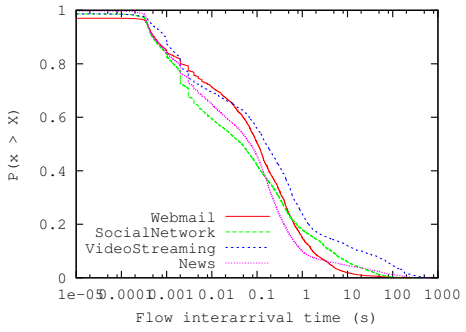

Fig. 1. Flow interarrival times

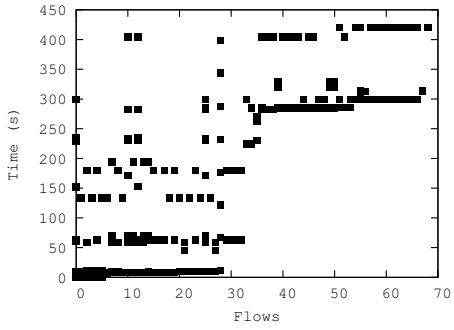

Fig. 2. Flow overlap in an example trace

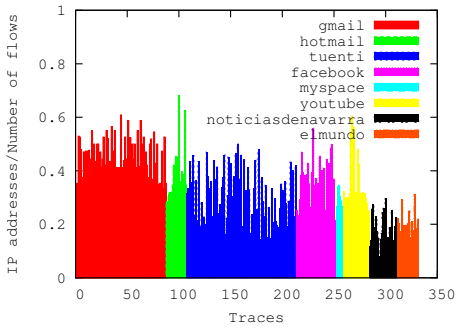

Fig. 3. Number of different IPs/number of flows

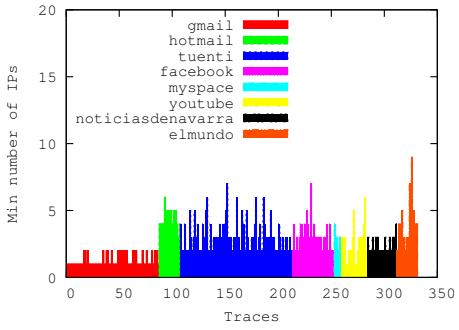

Fig. 4. Heavy IPs

the services as it indicates that this parameter does not depend on the type of webpage being loaded.

This coincidence of starting timestamps can also be observed, in some cases, for the timestamps of the last packet of the flows. This happens because web browsers tend to leave some connections open until the user closes the browser, the tab or loads a different website in the same tab. In order to show this behaviour we provide an example in Fig. 2 where the flows of a facebook.com trace are represented as a collection of points in the one second intervals where traffic is sent or received. Most traces yield graphics similar to these. A majority of flows open at the same timestamps, which signal the start of the session and the moments when the user opens new pages within the visited website. Some of them are left open for the duration of the session while others are closed before by the browser but, usually, the closing timestamp of a flow coincides with others of the same session. Anyway, as it can be seen in the figure, even though browsers may leave connections open, this does not mean that they are active all the time. In fact, we have calculated that, for all the traces, more than 60% of the flows are inactive in more than 80% of the one-second intervals of their length. Moreover, traffic tends to be concentrated at the beginning of the flows.

In brief, we have seen that flows that belong to the same session tend to overlap in time but, as they are not active during most of their length, the overlap is not really inherent to web traffic, only the coincidence of starting times is. As it will be discussed in Sec. V, we do use ending timestamps of flows in our clustering method although not in a direct way which could lead to bad classification.

B. Shared and Near IP Addresses

Server IP addresses are also intuitively a good parameter for flow clustering. As we stated previously, during the load of a webpage multiple connections to the same server may be opened and is easy to cluster them by their shared IP address. Moreover, even when objects are downloaded from different servers it is to be expected that, in some cases those servers will be in the same networks (if they belong to the same company or institution) so their IP addresses will be near.

In Figs. 3 and 4, traces have been ordered by service and website and a different shade has been assigned for the traces of each website in order to facilitate comparison. Fig. 3 shows the number of different accessed IPs in a trace divided by the number of flows in that trace (two variables that are strongly correlated). This ratio is usually smaller than 0.5 which means that, on average, at least two connections are opened to the same IP addresses in the traces. Nevertheless, usually multiple connections are open to a small number of IPs which concentrate the majority of traffic while the rest receive just one flow. Fig. 4 shows the minimum number of different IP addresses which concentrate at least 80% of the total traffic of each session. In general, this number is low (mean = 2.5, median = 2) and its dependency on the total number of IP addresses present in the trace is not strong.

We have seen how multiple connections are opened to the same IP addresses for each session. To represent near IP addresses in a session we calculate the IP offset as the difference

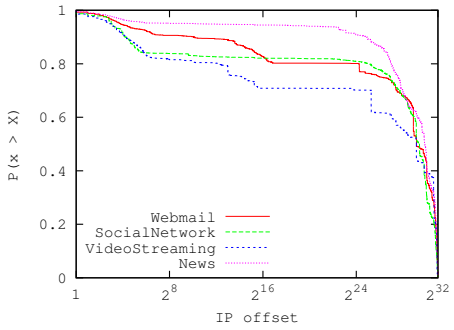

Fig. 5. CCDF of the IP offset for the four services

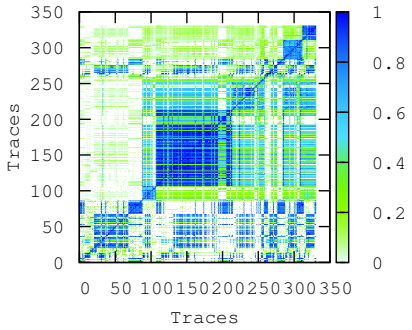

Fig. 6. Ratio of flows that have near IPs between traces

between a pair of IP addresses expressed as unsigned integers. In Fig. 5 the IP offset between pairs of destination IP addresses present in the traces is represented. The IP offset is calculated for all pairs of each trace and then the results are aggregated. As we see, for video streaming and social networks, around 20% of IP pairs have an offset lower than 256, the size of a class C network. It is necessary, now, to study if near IP addresses happen in different sessions. In Fig. 6, the colour of each dot represents how many of the flows of a particular trace (divided by the total number of flows in that trace) have near server IP address to the ones of flows in other trace. The nearness threshold chosen is, again, 256. Traces are ordered as they were in the Fig. 3. As expected, traces of the same website share a lot of IPs. However, it also happens between traces of different websites as flows from certain services (e.g. Google Analytics) may be present in both. Moreover, some websites may download content from or be hosted in servers located in the same networks (CDNs).

V. Clustering Method

In this section we explain the operation of our clustering method. Our objective is to group the flows that belong to the same sessions in the smallest number of clusters while trying to avoid mixing flows from different sessions in the same cluster. The only information used for each flow is the times of capture of its first and last packet, its size in bytes, the total number of packets in the flow and the server IP address (we only consider flows opened by the client). All this data, as stated previously is collected by Argus from pcap traces and is normally provided by NetFlow (IPFIX).

In order to carry out the clustering, we have designed a two-step procedure. In the first step, we only attempt to cluster contiguous flows. We consider that two contiguous flows should be in the same cluster if they fulfill at least one of two conditions:

1) Their start timestamps are very near (so near that it is improbable that they are related to different user actions). In Fig. 1 we saw that the interarrival time of consecutive flows in the traces was less than 100ms for around 50% of flow pairs. As this is a time interval short enough to avoid confusion with different user actions we will use it.
2) They share the same destination IP address.

Given this, in this part of the procedure clusters only grow while one of this conditions is met by the last flow in the current cluster and the new candidate. When this does not happen, a new group is created for the new flow. The next one will attempt to join that new group.

The second part of our system is a process that is called whenever there are clusters in memory that are considered old. A cluster is old if the last packet captured that belongs to the cluster was captured more than 30 seconds ago (which in [19] is suggested as an average value between user actions). When a cluster is old, our system tries to join it with a newer cluster. We use two parameters to do this. On the one hand, we define the length of a cluster as the time elapsed between the capture of the first and the last packet in the cluster (i.e. the smallest of the start times of the flows in the cluster and the biggest of the end times). Consequently, we define the time overlap of two clusters as two times the period of time both of them exist divided by the sum of both their lengths. Therefore, the time overlap varies from 1 when the clusters' start and end times coincide exactly, to 0 when they are never present at the same time. On the other hand, we say that two IP addresses are near if, when expressed as unsigned integers, the difference between them is smaller than 256 (a class C network size). We now consider the amount of bytes in each cluster that belong to flows whose destination IP addresses are near to those of some of the flows in the other cluster. We define the shared-IP weight for each cluster as that amount of bytes divided by the size of each cluster. We join two clusters if:

- The time overlap between both of them is very high (95%).
- The time overlap is high (50%) and the clusters share at least one IP address.
- The time overlap is low but the shared-IP weight is high (50%).

Finally, if an old cluster could not be incorporated into a newer one, it is taken out of the system.

Service	N. of clusters	N. of clusters for 90% traffic
Webmail	5.88	1.65
Social net.	7.00	2.63
Video str.	13.20	3.20
Newspapers	39.32	6.46

TABLE I
MEAN N. OF CLUSTERS BY SERVICE

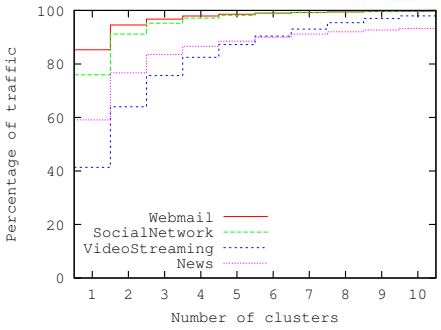

Fig. 7. Aggregated traffic in n clusters

VI. Results and Discussion

A. Individual Traces

Our first approach to testing our clustering method has been to use the collection of session traces described in Sec. III. Using the configuration specified previously, we have obtained the results that compose table 1. We provide, for each service, mean values of the number of clusters the method found in each trace and the number of clusters that include, at least 90% of the traffic of each trace. This is interesting as small clusters (most of them with just one flow) occur frequently but are not very significative. As we see, most of the traffic is grouped in few clusters specially considering that the captured sessions can be very long and comprise hundreds of different connections.

Results for the same traces are shown in a different way in Fig. 7. In it, we have aggregated, again by service, the traffic of the different traces in order to show how much of it is included in a specific number of clusters. Except for the video streaming service, the biggest of the clusters discovered in each trace contain, in average, more than 50% of the total traffic of the session. The case of video streaming is slightly different as during long youtube sessions, users normally watch multiple videos. If the connections carrying two videos (which are responsible for the biggest part of the load) are grouped in two different clusters, both will have a significative weight even if one of them is composed by a lot more (smaller) flows than the other. Anyway, the four curves grow fast and in average 90% of the traffic is grouped in 6 or 7 clusters in the worst case. It is worth noting that although table 1 and Fig. 7 show the same results, they are not calculated in the same way and may seem inconsistent. This happens because in table 1 all traces are given the same weight when calculating the means regardless of the total traffic in each of them. In Fig. 7, as the total traffic of the service is considered, bigger traces have a bigger effect than smaller ones.

Trace 1

Website	Recall	Precision
gmail.com	43.2	100
facebook.com	100	94.3
youtube.com	100	79.2
noticiasdenavarra.com	100	100

Trace 2

Website	Recall	Precision
gmail.com	36.6	100
facebook.com	98.4	93.8
youtube.com	100	62.9
elmundo.es	97.6	100

TABLE II
TWO NORMAL WEB TRAFFIC TRACES

B. Normal Web Navigation

The next step is to test the method with traces of normal web traffic that contain multiple sessions of different services in order to check if the algorithm is able to distinguish between them. As stated in Sec. III, the biggest challenge here is labeling the traces indicating to which session each flow belongs. We have captured two traces of about thirty minutes duration each with traffic of the websites previously introduced. In these traces some different sessions are kept open at the same time as it will happen normally. We have labeled the connections manually using whois information, analyzing previous DNS requests and looking into HTTP headers. Nevertheless, even giving this much attention to each flow, in some cases the classification is doubtful. For example, it is difficult to know if some secure connections to Google servers belong to gmail or youtube sessions.

Table 2 shows the performance of our method when classifying traffic from those traces. We say that a flow is correctly classified when it has been aggregated into a cluster where the majority of flows belong to its same website. We then define recall as the percentage of correctly classified flows for a website against the total number of flows of the website. We also define precision as the percentage of flows that are correctly classified as belonging to a website against the total number of flows classified as belonging to that same website.

As we can see, results are generally good. There are some problems between gmail.com and youtube.com sessions as both of these services are hosted in Google servers. If a gmail session is open while visiting youtube, its connections tend to be included in youtube clusters. These problems are almost unavoidable with services that are so interrelated.

VII. Conclusions

In this paper we have presented a method to cluster flows that belong to the same web session using only the information provided by flow records. We believe that this is interesting as

the resulting clusters will contain more information about the session than the individual flows and thus will be more easily classified into the different services that are provided through the web. In order to design our method we have previously captured and studied multiple sessions to four popular web services. We have prepared a capturing and labeling system that can be easily distributed but has some limitations as it can only gather traces of individual sessions. We find that labeling the connections of a trace with multiple different sessions is complex even when doing it manually.

We have selected flow start and end timestamps and server IP addresses as the most interesting parameters in order to cluster the flows. Nevertheless, they all have their drawbacks which our method attempts to minimize: flows of concurrent sessions may start at the same time; when the connections are closed depends heavily on the web browser implementation; and server IP addresses may be shared by flows of different services if, for example, they are provided by the same company. Experimental results show that flows are grouped in a relatively small number of big clusters for each session and that recall and precision are generally good for the services tested.

Future lines of work will be to improve our capturing and labeling system so it is able to capture traces of normal web navigation (multiple sessions to different services in the same trace) with which we could tune and test our system more extensively. Moreover, we aim to characterize the obtained clusters in order to design a classification method able to assign them to the services whose traffic they comprise.

REFERENCES

[1] H. Weinreich, H. Obendorf, E. Herder, and M. Mayer, "Off the beaten tracks: exploring three aspects of web navigation," in *Proceedings of the 15th international conference on World Wide Web*, ser. WWW '06. New York, NY, USA: ACM, 2006, pp. 133–142.

[2] L. D. Catledge and J. E. Pitkow, "Characterizing browsing strategies in the world-wide web," *Comput. Netw. ISDN Syst.*, vol. 27, pp. 1065–1073, April 1995.

[3] E. B. Claise, "RFC 5101: Specification of the IP flow information export (IPFIX) protocol for the exchange of IP traffic flow information," January 2008.

[4] T. T. T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," *Communications Surveys & Tutorials, IEEE*, vol. 10, no. 4, pp. 56–76, 2008. [Online]. Available: http://dx.doi.org/10.1109/SURV.2008.080406

[5] M. Roughan, S. Sen, O. Spatscheck, and N. Duffield, "Class-of-service mapping for QoS: a statistical signature-based approach to IP traffic classification," in *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, ser. IMC '04. New York, NY, USA: ACM, 2004, pp. 135–148. [Online]. Available: http://doi.acm.org/10.1145/1028788.1028805

[6] A. W. Moore and D. Zuev, "Internet traffic classification using bayesian analysis techniques," *SIGMETRICS Performance Evaluation Review*, vol. 33, no. 1, pp. 50–60, 2005.

[7] Z. Li, R. Yuan, and X. Guan, "Accurate classification of the internet traffic based on the SVN method," in *ICC'07: Proceedings of the 2007 IEEE International Conference on Communications*, June 2007, pp. 1373–1378.

[8] H. Kim, K. Claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K. Lee, "Internet traffic classification demystified: myths, caveats, and the best practices," in *CoNEXT '08: Proceedings of the 2008 ACM CoNEXT Conference*. New York, NY, USA: ACM, 2008, pp. 1–12.

[9] A. McGregor, M. Hall, P. Lorier, and J. Brunskill, "Flow clustering using machine learning techniques," in *Passive and Active Network Measurement*, ser. Lecture Notes in Computer Science, vol. 3015. Springer Berlin, 2004, pp. 205–214.

[10] S. Zander, T. Nguyen, and G. Armitage, "Automated traffic classification and application identification using machine learning," in *LCN '05: Proceedings of the 30th IEEE Conference on Local Computer Networks*. Washington, DC, USA: IEEE Computer Society, 2005, pp. 250–257.

[11] L. Bernaille, R. Teixeira, and K. Salamatian, "Early application identification," in *CoNEXT '06: Proceedings of the 2006 ACM CoNEXT conference*. New York, NY, USA: ACM, 2006, pp. 1–12.

[12] M. Zink, K. Suh, Y. Gu, and J. Kurose, "Characteristics of youtube network traffic at a campus network - measurements, models, and implications," *Computer Networks*, vol. 53, no. 4, pp. 501–514, 2009.

[13] P. Gill, M. Arlitt, Z. Li, and A. Mahanti, "Youtube traffic characterization: a view from the edge," in *IMC '07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*. New York, NY, USA: ACM, 2007, pp. 15–28.

[14] B. Yu and H. Fei, "Multiscale analysis and modeling of user session traffic in social networks," in *ICCT 2008: Proceedings of the 11th IEEE International Conference on Communication Technology*, nov. 2008, pp. 85 –88.

[15] F. Schneider, A. Feldmann, B. Krishnamurthy, and W. Willinger, "Understanding online social network usage from a network perspective," in *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, ser. IMC '09. New York, NY, USA: ACM, 2009, pp. 35–48.

[16] F. Schneider, S. Agarwal, T. Alpcan, and A. Feldmann, "The new web: Characterizing AJAX traffic," in *Passive and Active Network Measurement*, ser. Lecture Notes in Computer Science. Springer Berlin, 2008, vol. 4979, pp. 31–40.

[17] D. Schatzmann, W. Mühlbauer, T. Spyropoulos, and X. Dimitropoulos, "Digging into HTTPS: flow-based classification of webmail traffic," in *Proceedings of the 10th annual conference on Internet measurement*, ser. IMC '10. New York, NY, USA: ACM, 2010, pp. 322–327. [Online]. Available: http://doi.acm.org/10.1145/1879141.1879184

[18] "ARGUS: Audit record generation and usage system," http://www.qosient.com/argus/.

[19] A. Bianco, G. Mardente, M. Mellia, M. Munafo, and L. Muscariello, "Web user-session inference by means of clustering techniques," *Networking, IEEE/ACM Transactions on*, vol. 17, no. 2, pp. 405 –416, april 2009.