IP Mining: Extracting Knowledge from the Dynamics of the Internet Addressing Space

Pedro Casas, Pierdomenico Fiadino, and Arian Bär
Telecommunications Research Center Vienna - FTW
{surname}@ftw.at

Abstract—**Going back to the Internet of one decade ago, HTTP-based content and web services were provided by centralized or barely distributed servers. Single hosts providing exclusive services at fixed IP addresses was the standard approach. Current situation has drastically changed, and the mapping of IPs to different content and services is nowadays extremely dynamic. The adoption of large CDNs by major Internet players, the extended usage of transparent content caching, the explosion of Cloud-based services, and the decoupling between content providers and the hosting infrastructure have created a difficult to manage Internet landscape. Understanding such a complex scenario is paramount for network operators, both to control the traffic on their networks and to improve the quality experienced by their customers, specially when something goes wrong. Using a full week of HTTP traffic traces collected at the mobile broadband network of a major European ISP, this paper studies the associations between web services, the hosting organizations/ASes, and the content servers' IPs. By mining correlations among these, we extract useful insights about the dynamics of the IP addressing space used by the top web services, and the way content providers and hosting organizations deliver their services to the mobile end-users. The extracted knowledge is applied on two specific use-cases, the former on hosting and service delivery characterization, the latter on automatic IP-based HTTP services classification.**

Keywords—IP Addressing Space; HTTP Traffic; Content Delivery Networks; Traffic Classification and Analysis

I. Introduction

A big share of today's Internet ecosystem is shaped by the success and influence of the most popular services running on top of HTTP (e.g., video and audio streaming, social networking, on-line gaming, etc.). HTTP is doubtlessly the dominating content delivery protocol in today's Internet, accounting for more than 75% of the residential customers traffic [1], [2]. HTTP-based services such as YouTube and Facebook are forcing the Internet to shift the content as close as possible to the end-users, which in turn is modifying the way content is hosted, addressed, and delivered. The very last few years have seen an astonishing development in Content Delivery Networks (CDNs) technology and Cloud Services provisioning platforms. It is therefore not surprising that todays' Internet content is largely delivered by major CDNs like Akamai or Google CDN, and traditional services are now running on Cloud platforms such as Amazon EC2.

In this complex scenario, content and services are no longer located in centralized delivery platforms, owned by single organizations, but are distributed and replicated across the Internet and handled by multiple players. Understanding issues such as HTTP traffic composition, usage patterns, content location, hosting organizations, and addressing dynamics is highly valuable for network operators. The application areas are multiple, spanning network planning and optimization (e.g., content caching), traffic engineering (e.g., traffic differentiation/priorization), network measurements (e.g., CDN traffic/network characterization), trend analysis and service profiling (e.g., heavy-hitter applications), just to name a few.

In this paper we study the addressing dynamics of the top Internet services running on HTTP. Using a full week of HTTP traffic traces collected at the mobile broadband network of a major European ISP, we study the associations between services, the hosting organizations, and the IPs assigned to the servers providing the content. The complete dataset consists of more than half a billion of passively observed HTTP flows, aggregated in a per-hour basis. For each flow, the dataset contains the contacted URL, the contacted IP address (i.e., IP of the server), the total bytes exchanged with the server IP, and a timestamp. The dataset includes the name of the organization owning the server IP, extracted from the MaxMIND ASes databases [24]. In addition, the Full Qualified Domain Name (FQDN) is automatically extracted from the contacted URL, which is used to deduce the corresponding service being accessed at the server IP, using HTTPTag [19].

HTTPTag is a flexible on-line HTTP classification system based on pattern matching and tagging, which associates a set of labels or *tags* to each observed HTTP flow, based on the contents and service being requested. This association is performed by simple regular expressions matching, applied to the `host` field of the corresponding HTTP flow's header (i.e., host name of the contacted server). HTTPTag currently recognizes and tracks the evolution of more than 280 services and applications running on top of HTTP, including for example tags such as `YouTube`, `Facebook`, `Google` (i.e., Google Search), `Twitter`, `Zynga`, `Gmail`, etc. Due to the highly concentrated traffic volume on a small number of heavy hitter applications, current list of services spans more than 70% of the total HTTP traffic volume on the 3G network of a leading European provider.

Our traffic analysis targets two specific use-cases: (i) characterization of the top HTTP-based services, their provisioning, and the underlying hosting servers, and (ii) automatic HTTP services classification based on IP addresses. In the first use-case, we identify the top web services running on HTTP and shed light on the way they are delivered by the underlying hosting organizations and CDNs, including a characterization of the number of server IPs used to deliver each service, the placement of the servers, the identification of load balancing techniques, and the temporal provisioning of resources. The goal of the second use-case is to evaluate the feasibility of using a minimalist approach for classifying HTTP flows on the fly, relying exclusively on the IP address of the server hosting the requested content. Such an approach is extremely lightweight and can be applied for on-line high speed classification. The question we try to answer is to which extent such an approach can provide useful results? The reason behind using IP addresses is simple: mappings between services and IPs are reasonably stable in time. Services running on top of HTTP are provided by companies with delivery infrastructures that tend to be either very stable in time, or in the case of CDN-based distribution, use well-known IP ranges with stable dynamics. After all, even if content is potentially served from multiple different datacenters, it is reasonable to accept that the number of datacenters serving some specific content varies slowly.

The remainder of the paper is organized as follows: section II presents a brief state of the art in the field of HTTP traffic analysis, CDN characterization, and automatic traffic classification. In section III we describe the labeling technique applied by HTTPTag to identify services running on top of HTTP, additionally providing some initial results on the analysis of the top web services present in the mobile traffic traces. Section IV tackles the first use-case on HTTP services provisioning, characterizing the top HTTP-based services and the underlying hosting infrastructure. In section V we introduce and evaluate the services classification approach, including its global accuracy and the per-service recall and precision for the top services in the traces. Section V concludes this work.

II. Related Work

The study and characterization of the Internet traffic hosted and delivered by the top content providers has gained important momentum in the last few years [3]–[7]. In [3], authors show that most of today's inter-domain traffic flows directly between large content providers, CDNs, and the end-users, and that more than 30% of the inter-domain traffic volume is delivered by a small number of content providers and hosting organizations, being Google the largest and fastest growing contributor to inter-domain traffic. According to [4], the top 10 organizations handle 65% of the total web traffic in a major European ISP, including companies such as Google, Akamai, Limelight, and Level3. Several studies have focused on CDN architectures and CDN performance, analyzing features such as CDN size, servers' location, and latencies to content among other [5]–[7]. In particular, [6] focuses on user-content latency analysis at the Google CDN, [7] provides a comprehensive study of the Akamai CDN architecture, and [5] characterizes the performance of both Akamai and Limelight in terms of server availability and delay.

Regarding Internet Traffic Classification (TC) and analysis, there has been an extensive research activity during the last decade [10]. Commonly deployed traffic classification methods rely on port and payload-based analysis techniques. These techniques present important limitations that highly reduce their effectiveness, particularly due to the emergence of new dynamic applications and the widespread use of encryption, tunneling, and protocol obfuscation. Standard classification approaches rely on Deep Packet Inspection (DPI) techniques, using pattern matching and statistical traffic analysis [11]. Probably the most popular approach for TC exploited in recent years by the research community is the application of Machine Learning (ML) techniques [12]. A standard non-exhaustive list of supervised ML-based approaches includes the use of Bayesian classifiers [13], linear discriminant analysis and k-nearest-neighbors [14], decision trees and feature selection techniques [15], and support vector machines [16]. Unsupervised and semi-supervised learning techniques have also been applied for traffic analysis and classification [17].

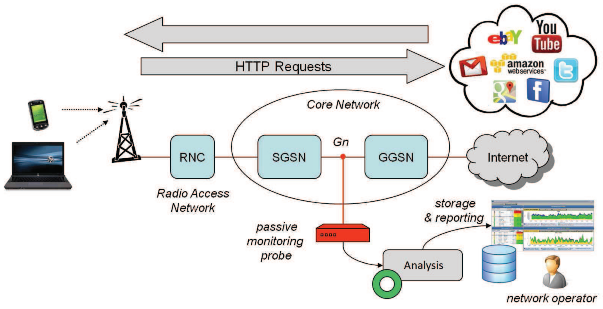


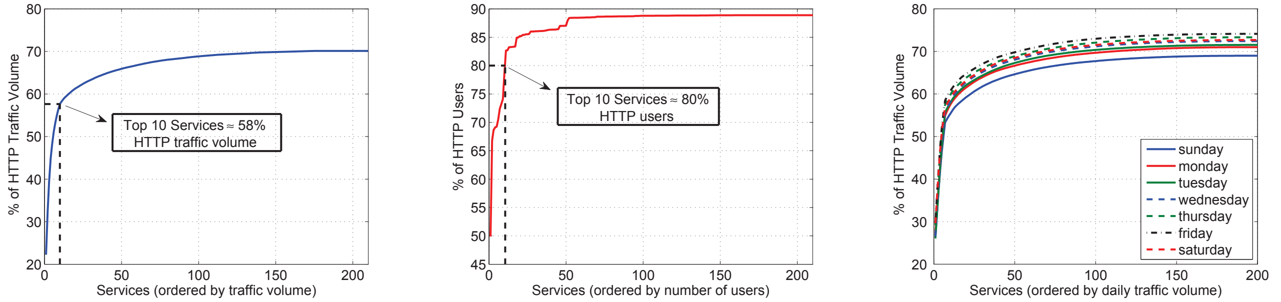
Figure 1. Passive HTTP traffic analysis in an operational 3G Network. HTTP flows observed at the Gn interface are analyzed and tagged on the fly.

In the specific case of HTTP traffic, classification and analysis has been the focus of many recent studies [2], [8], [9], [19]–[21]. In [20], authors use payload-based analysis heuristics to classify 14 different HTTP classes. In [19] we use pattern matching techniques applied to the `host` field of HTTP headers to recognize more than 280 applications and services running on top of HTTP. In [2], [9], authors use DPI techniques to analyze the usage of HTTP-based applications on residential connections, showing that HTTP traffic highly dominates the total downstream traffic volume. Authors in [8] study the extension of HTTP content caching in current Internet, characterizing HTTP traffic in 16 different classes using port numbers and heuristics on application headers. Recently, the authors of [21] provide evidence on a number of important pitfalls of standard HTTP traffic characterization techniques which rely exclusively on HTTP headers, showing for example that around 35% of the total HTTP volume presents a mismatch in headers like `Content-Type`, extensively used in previous studies.

In this paper we present the characterization and analysis of a full week of HTTP traffic traces collected at the 3G network of a major European ISP. The analysis spans both the web services and their underlying hosting organizations, as well as the possibility of using only IP addresses for classifying HTTP traffic flows. We acknowledge that, despite the large size of our traffic dataset, we analyze packets from a single mobile vantage point, which is far from providing a complete view of the global IP address space and HTTP services.

III. HTTPTag: Mapping Server IPs to Services

In this section we provide a description of HTTPTag, which is used to label the traffic dataset presented in this paper. HTTPTag works with passively captured packet data. Figure 1 shows the deployment of HTTPTag in the operational 3G Network that serves as vantage point for our study. Packets are captured on the Gn interface links between the GGSN and SGSN nodes, using the METAWIN passive monitoring system [23]. HTTP packets are detected and analyzed on the fly: every new HTTP transaction is parsed and the contacted host

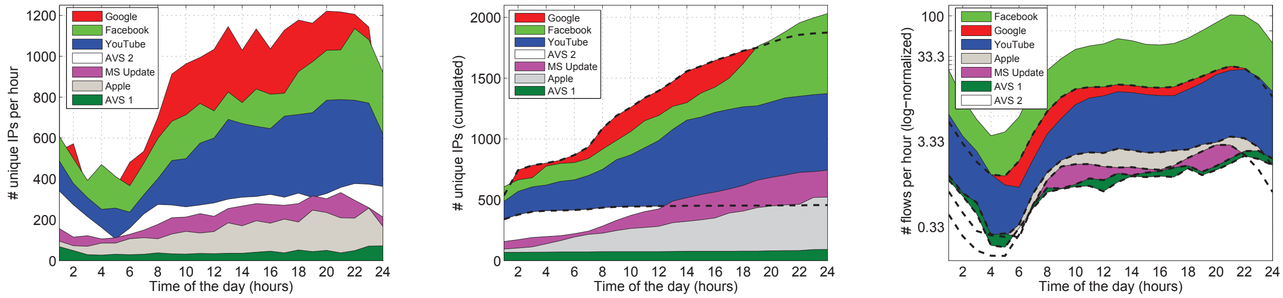


(a) HTTP traffic volume per service.

(b) Unique HTTP users per service.

(c) Daily HTTP traffic volume per service.

Figure 2. HTTP traffic classification using HTTPTag. HTTPTag labels more than 70% of the overall HTTP traffic volume caused by more than 88% of the web users. The top 10 services w.r.t. volume account for almost 60% of the overall HTTP traffic, and the top 10 services w.r.t. popularity are accessed by about 80% of the users. In (c), HTTPTag is able to label between 69% and 74% of the total HTTP volume on the studied traces, for the complete week.



(a) Unique server IPs per hour.

(b) Cumulative number of unique server IPs.

(c) Normalized number of flows per hour.

Figure 3. Evolution of unique server IPs and normalized number of flows for the top 7 services on a single day. Google Search, Facebook and YouTube dominate the IP space and account for the majority of the flows. Thanks to Akamai, Facebook is the most IP-distributed service, using more than 2000 different IPs on a single day.

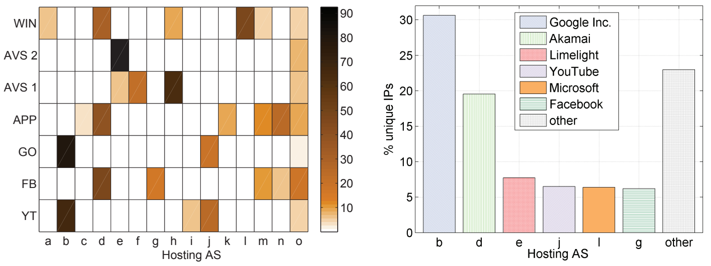
name is compared against a set of defined regular expressions or *patterns* describing different services and applications. If a matching pattern is found, the transaction is assigned to the corresponding service. To preserve user privacy, any user related data (e.g., IMSI, MSISDN) are removed on-the-fly, and payload content beyond HTTP headers is discarded.

HTTPTag uses TicketDB [22], a fast and scalable parallel database system tailored to meet the requirements of network monitoring in 3G networks. For every new HTTP transaction analyzed by HTTPTag, a summary ticket is stored and indexed in TicketDB, providing long term traffic analysis capabilities. Each ticket contains a timestamp, the IP address of the contacted server, the requested URL, volume stats (i.e., transferred bytes up/down), and the corresponding service resulting from the pattern matching step. As such, for every observed HTTP flow, HTTPTag provides a mapping or association between the hosting IP address and the corresponding service.

To improve pattern matching speed, patterns are ordered by probability of occurrence, which are computed from the history of successful matches. HTTPTag tagging approach is based on manual definition of tags and regular expressions, which might a priori impose scalability issues. Indeed, there are millions of websites on the Internet and it would be impossible to define enough patterns to classify every possible requested URL. However, the well known mice and elephants phenomenon also applies to HTTP-based services, and limiting the study to the most popular services already captures the majority of the traffic volume and users in the network. While the initial definition of tags is a time-consuming task, regular expressions identifying applications tend to remain stable in time, basically because they are associated to the name of the application itself and thus recognized and used by the end-user. This is specially true for popular services, which carry the most of the traffic. HTTPTag does not currently recognize HTTPS traffic, since the requested URLs are encrypted. An on-going extension of HTTPTag to solve this issue is to rely on DNS queries analysis, similar to the approach introduced in [18]. HTTPS analysis is out of the scope of this paper.

Figures 2(a) and 2(b) depict the distribution of HTTP traffic volume and number of users covered by HTTPTag in a standard day. Using about 380 regular expressions and 280 tags (i.e. services) manually defined, HTTPTag can classify more than 70% of the overall HTTP traffic volume caused by more than 88% of the web users in the studied network. Note that a small number of heavy hitter services dominate the HTTP landscape: the top 10 services w.r.t. volume account for almost 60% of the overall HTTP traffic, and the top 10 services w.r.t. popularity are accessed by about 80% of the users. These results reinforce the hypotheses behind HTTPTag: focusing on a small portion of the services already gives a large traffic visibility to the network operator.

Figure 2(c) shows the total daily HTTP volume labeled by HTTPTag on the week of traces used in the study. The week corresponds to 7 days during the second quarter of 2012, from



(a) Services & hosting organizations .

(b) Share of server IPs per organizations.

Figure 4. Distribution of the server IPs used by the top 7 services among the top hosting organizations.

Org. (AS num.)	id	Org. (AS num.)	id	Org. (AS num.)	id
Hotmail (12076)	a	Swiftwill (30361)	f	Apple (714)	k
Google (15169)	b	Facebook (32934)	g	Microsoft (8075)	l
Omniture (15224)	c	Level 3 (3356)	h	TeliaNet (1299)	m
Akamai EU(20940)	d	YouTube (36040)	i	Verizon (701)	n
Limelight (22822)	e	YouTube (43515)	j	other	o

Table I. TOP HOSTING ORGANIZATIONS AND ASES IN TERMS OF NUMBER OF UNIQUE IPS OF THE TOP 10 SERVICES (NON-ORDERED LIST).

Service	#/16	#/24	# IPs	top-subnet /24	Org. (AS num.)
YT	10	51	1373	74.125.232.0	Google (15169)
FB	62	140	2031	2.20.182.0	Akamai EU (20940)
GO	9	73	1875	74.125.232.0	Google (15169)
APP	35	71	522	80.239.149.0	TeliaNet (1299)
AVS 1	23	71	92	204.160.106.0	Level 3 (1299)
AVS 2	6	13	456	87.248.217.0	Limelight (22822)
WIN	41	200	743	2.20.182.0	Akamai EU (20940)

Table II. NUMBER OF IPS AND BLOCKS HOSTING THE TOP 7 SERVICES. THE TOP /24 SUBNETWORKS ARE DEFINED IN TERMS OF NUMBER OF HTTP FLOWS DELIVERED.

Sunday to Saturday. HTTPTag is able to label between 69% and 74% of the total daily HTTP volume on the studied traces. The study performed in the following sections considers only the labeled HTTP flows, and the approximately remaining 30% of unlabeled HTTP volume is ignored.

As previously mentioned, the top 10 services (in terms of traffic volume) flagged in figure 2(a) are responsible for almost 60% of the total daily HTTP volume during the whole evaluation week, which represents about 85% of the labeled services in terms of traffic volume. The list of most important services volume-wise includes services such as Apple (i.e., App Store and iTunes - APP), Facebook (FB), YouTube (YT), Google (i.e., Google Search - GO), two well-known Adult Video Streaming services AVS 1 and AVS 2, and Microsoft Windows Update (WIN) among others.

IV. UNDERSTANDING HTTP SERVICES PROVISIONING

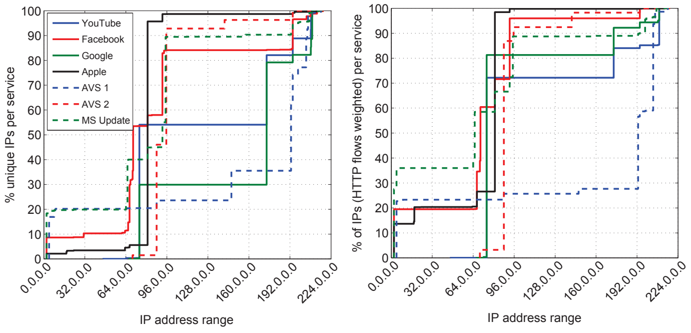
In this section we focus on the hosting and service delivery analysis. This includes a characterization of the number and temporal provisioning of the server IP addresses used for each service, the placement of the hosting servers, and the identification of load balancing techniques. To limit the number of services to study, the analysis is performed exclusively for the aforementioned top 7 services, which account for the majority of the HTTP traffic volume.

Let us begin by analyzing the number of unique server IP addresses used to deliver each of these services on a single day. Figures 3(a) and 3(b) depict the evolution of the number of unique server IPs per hour and the accumulated number of unique server IPs on a single day, whereas figure 3(c) plots the number of HTTP flows per hour (values are normalized to avoid disclosing sensitive business-related absolute values). For 6 out of the 7 services (i.e., all except AVS 1), there is a clear correlation between usage and number of unique server IPs delivering the corresponding content. The changes observed in the unique number of IPs being used by Google Search, Facebook, and YouTube are impressive, going from about 250 IPs per service at 5 am to up to 1200 in the case of Google Search. These three services are provided by large CDNs (i.e., Google CDN for Google services and Akamai for Facebook), which justifies the large number of unique server IPs being used during the day. Thanks to Akamai, Facebook is the most IP-distributed service, using more than 2000 different IPs on a single day. The number of unique IPs serving the video streaming service AVS 1 remains almost constant in time and is below 100 all over the day, suggesting a very stable delivery infrastructure.

Using the MaxMIND ASes databases [24] we explore now how distributed are these unique IPs in terms of the different organizations owning them. Figure 4(a) shows the fraction of unique IPs per service hosted by the list of organizations and ASes described in table I. The organization labeled as "other" (i.e., id o) consists mainly of ISP ASes which cache the content at the edge of their own networks.

As expected, Google Search and YouTube IPs are mainly hosted by Google Inc. ASes, Facebook IPs are mainly hosted by Akamai and Facebook ASes, and Windows Update IPs are mainly hosted by Microsoft ASes. For example, in the case of Facebook, it is well known that the static content is hosted by Akamai, whereas Facebook ASes host the dynamic content [4]. Almost all of the AVS 2 IPs are hosted by Limelight, and this organization is additionally hosting only a small fraction of AVS 1 IPs, with no other service being hosted there. This concentration of IPs on an almost exclusive organization explains the high classification accuracy obtained for AVS 2 flows in section V. In all the cases, a small fraction of the IPs used to deliver the services belong to ASes caching the content. Figure 4(b) depicts the top ordered organizations in terms of unique IPs providing the studied services. Google and Akamai are clearly the most distributed organizations in terms of IPs providing the top HTTP services. Limelight is the third CDN in our traces, in this case mainly providing the AVS 2 content.

Table II provides a summary on the number of IPs and *potential* /16 and /24 sub-networks or IP blocks hosting the studied services. The term potential comes from the fact that we only consider an aggregation of IPs using /16 and /24 net-masks for counting purposes, but we are actually not sure if the corresponding subnetworks are configured as such. The table also reports the top /24 subnetworks in terms of number of delivered flows, together with the corresponding AS and hosting organization. We can appreciate that the three services hosted by Akamai (i.e., Facebook, Apple, and Windows Update) are highly distributed in terms of disjoint IP blocks. This is coherent with the fact that the Akamai CDN deploys a highly distributed architecture with many thousands



(a) IP range of unique server IPs.

(b) Weighted IP range.

Figure 5. Distribution of the IP range associated to the tagged services on a single day. AVS 1 is highly distributed in terms of different IP blocks, whereas AVS 2 is mostly served from a small number of blocks.

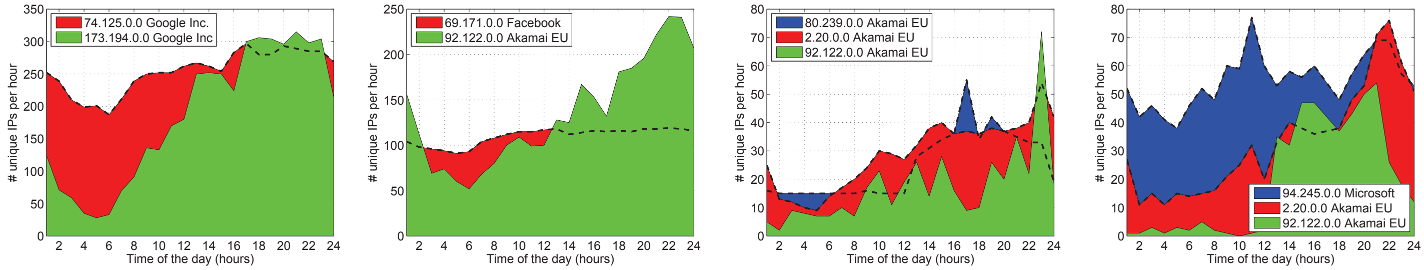
of servers (e.g., more than 27.000 in 2008 according to [5]) following the *enter deep into ISPs* approach [5], by deploying content distribution servers inside ISP POPs. The idea behind such an approach is to get as close as possible to the end users, improving user-perceived performance in terms of both delay and throughput. Such a design results in a large number of server clusters scattered around the globe. On the other hand, the AVS 2 service is the most concentrated one in terms of IP blocks, having around 450 different IPs scattered around 6 /16 IP blocks. As shown in 4(a), AVS 2 is mainly hosted by Limelight, which follows a completely different architectural design to that of Akamai; Limelight follows the *bring ISPs to home* approach [5], building large content distribution centers at only a few key locations and connecting these centers using private high speed connections.

A very interesting observation from figure 4(a) is that many IPs delivering different services are usually hosted by the same organization. For example, Akamai hosts content from Facebook, Apple, and Windows Update, whereas both YouTube and Google Search belong to Google and YouTube ASes. Specially in the case of Facebook and Windows Update, the majority of their flows are served from the /24 Akamai block 2.20.182.0/24, and the same happens to Google Search and YouTube, being served by IPs in the /24 Google block 74.125.232.0/24.

To further explore the ranges of used IPs, figure 5 depicts the distribution of the IP address ranges associated to the different top 7 services on a single day. Figure 5(a) depicts the distribution of IPs without considering the actual number of HTTP flows being served by each IP, whereas 5(b) weights each of the IPs by the number of flows delivered. The separation between blocks of IPs is remarkable, being AVS 1 the most notorious case. Indeed, according to table II, AVS 1 has only 92 unique IPs delivering its content, which are distributed along 23 different /16 IP blocks. Figure 5(b) shows the aforementioned blocks used by Akamai for Facebook and Windows Update, and by Google CDN for Google Search and YouTube. The highly concentrated group of IP blocks used by Limelight to deliver AVS 2 are also noticeable, with the block 87.248.217.0 serving the majority of the flows.

We move on to the analysis of the temporal evolution of the IPs used by some selected services and CDNs. Figure 6 depicts the temporal evolution of the number of hourly unique IPs per service, for some selected /16 blocks. Let us first focus on YouTube and Facebook, depicted in figures 6(a) and 6(b) respectively. Two /16 blocks are plotted in each case, the former remains reasonably stable during time in terms of number of unique IPs, the latter presents a big increase in the number of used IPs when traffic load increases. In the case of YouTube, the number of IPs in the block 74.125.0.0/16 varies between around 200 and 300 IPs, whereas the variation in the block 173.194.0.0/16 is between 50 and 300 different IPs approximately. Such differences suggest different location of content or different server roles at different blocks, load balancing techniques, or both. In the case of Facebook, the Facebook block 69.171.0.0/16 has an almost constant number of active IPs being accessed during the day, whereas the Akamai block 92.122.0.0/16 presents strong variations, reflecting once again different provisioning policies; in particular, Facebook servers might be continuously active due to specific service requirements (e.g., Facebook servers handle all the control metadata of Facebook sessions). Figures 6(c) and 6(d) show similar behaviors for 3 different IP blocks used by Apple and Windows Update, with some additional and very interesting *spiking* activity consisting of short periods of time with large increases in the number of IPs being contacted. For example, in the case of Apple, the Akamai block 92.122.0.0/16 presents a spiking behavior every a couple of hours in the afternoon, with a markedly change from 20 to 70 unique IPs in one single hour, at 23:00hs. Windows Update also presents spiking behavior out of the high-load time period, with an important increase of active IPs between 10:00hs and 12:00hs in the Microsoft block 94.245.0.0/16. Such changes reflect both the flexibility of Akamai to handle crowds with an increasing number of IPs, and the probable scheduling of certain activities in specific services (e.g., specific Microsoft software updates).

The last part of this section is devoted to the identification of CDN servers location and load balancing policies. Similar to [4], we consider the Round Trip Time (RTT) to the hosting servers as a measure of the servers distance from the vantage point. The RTT to any specific IP address consists of both the propagation delay and the processing delay, both at destination as well as at every intermediate node. Given a large number of RTT samples to a specific IP address, the minimum RTT values are an approximated measure of the propagation delay, which is directly related to the geographical location of the underlying server. It follows immediately that IPs showing similar min RTT values are most probably located at similar locations, whereas IPs with very different min RTTs are located in different locations (e.g., datacenters in different countries). RTT values are obtained from active measurements, performed during the complete week of measurements, using a standard ping tool. In order to identify the min RTT values, all the IPs assigned by HTTPTag to a specific service during each measurement hour are periodically pinged. In particular, every unique IP is pinged with trains of 100 IMCP echo request packets every 10 minutes, resulting in a total of 6 individual values of min RTT per hour and per IP. We are very aware that obtaining such min RTT measurements by active probing is not always the best approach, as many servers would simply not answer to an echo request, ICMP packets can be altered or differently treated by the ISP or the CDN, the content provider might make use of IP Anycast in its network, just to name a few of the possible shortcomings. In order to reduce the



(a) YouTube IPs.

(b) Facebook IPs.

(c) Apple IPs.

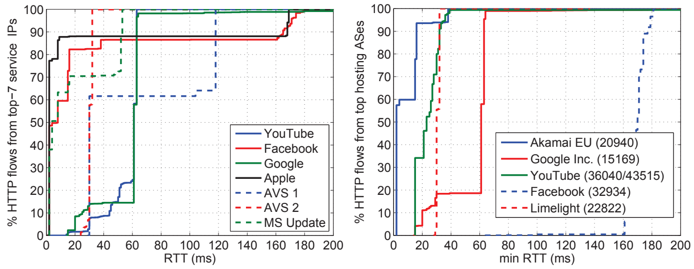
(d) Windows Update IPs.

Figure 6. Temporal evolution of number of hourly unique server IPs per service, for selected /16 blocks of IPs. The number of unique IPs used by Akamai to deliver different services from different IP blocks is highly dynamic during the day, and presents big changes under high-load or other on-demand situations.

impacts of such shortcomings, we filter out all the inconsistent results providing different min RTT values at different hours of the day.

Figures 7(a) and 7(b) depict the distribution of min RTT values per service and per hosting organization/AS respectively. Frequencies are weighted by the number of flows coming from each specific IP during one single day of measurements. Modes or steps in the distributions suggest the existence of different geographically separated hosting locations. Figure 7(a) shows that a large fraction of the Facebook, Apple, and Windows Update flows come from servers probably located in the same city of the vantage point, as min RTT values are below 5ms. These three services are largely provided by Akamai, thus results are very in-line with the min RTT values depicted for Akamai IPs in figure 7(b). Indeed, more than 60% of the Akamai HTTP flows come from servers *inside the ISP*, justifying the aforementioned low min RTT values. Apple flows seem to be served from three markedly different locations, given the three modes clearly visible in the CDF. Two of them are probably located in the same country of the vantage point, as min RTT values are below 10ms, whereas the third location is located outside Europe (i..e, min RTT $>$ 160ms), probably in the US due to Apple and Verizon IPs. The AVS 2 service seems to be mainly served from two locations in Europe (min RTT $\approx$ 30ms), perfectly matching the results depicted in figure 7(b) for the Limelight CDN. The two marked and very similar modes for Limelight min RTT in 7(b) reinforce the comments on the *bring ISPs to home* approach. A deeper analysis of the underlying IPs with the MaxMIND GeoIP data [24] reveals Limelight IPs in Italy and UK. AVS 1 is served from three different locations, including a Limelight CDN datacenter in Europe and two locations outside Europe, with at least one of them being Level3 according to table I. According to 7(b), most of the Facebook flows provided by the Facebook AS come from the US, and a very marginal fraction comes from inside Europe, more precisely Ireland according to manual inspection with MaxMIND. Interestingly, most of the YouTube flows come from servers under Google ASes and not YouTube ASes, which will have a major impact in the classification confusion matrix between Google Search and YouTube flows in section V.

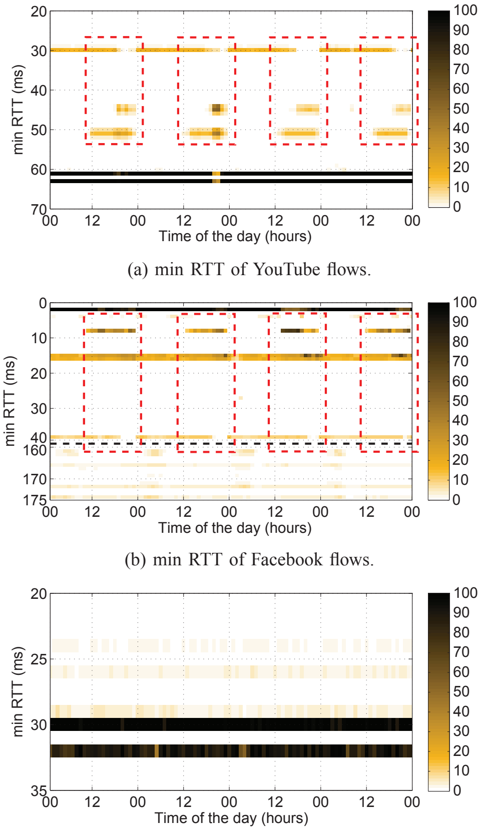
To conclude with this part of the study, we analyze now the temporal evolution of the min RTT for some selected services, aiming to show evidence on load balancing techniques employed by the Google CDN, Akamai, and Limelight. Figure 8 depicts the hourly evolution of the min RTT for different service flows during 4 consecutive days, from Monday to Thursday, including YouTube (mainly Google CDN), Facebook (mainly Akamai), and AVS 2 (mainly Limelight). Each column in figure 8 depicts the CDF of the min RTT of all the corresponding service flows, using a heatmap-like plot (i.e., the darker the color, the more concentrated the CDF in that value). Figure 8(a) plots the results for YouTube flows. The majority of the flows are delivered from the two Google locations depicted in figure 7(b) at 61ms and 63ms, about 15% of the flows are served from a third location at 30ms, and the remaining flows are served from different locations at around 44ms and 51ms. The interesting observation is that markedly min RTT shifts occur every day at exactly the same time slots, showing a min RTT periodic pattern. These temporal patterns are flagged by dotted rectangles. Such traffic shifts suggest either some regular content access pattern (i.e., users access the same contents every day at the same time-slots), periodical network congestion events, or much more likely, the presence of load balancing techniques which permit the CDN to serve the content from different locations according to some internal decision policies. Similar patterns can be observed for the Facebook static content hosted by Akamai as depicted in figure 8(b); we mention the static content as the min RTT values correspond to Akamai servers, i.e., RTT $<$ 40ms. Both results suggest that Google CDN and Akamai make use of internal load balancing policies to serve the content from their different hosting locations. Finally, figure 8(c) depicts the same analysis for the AVS 2 service. As expected, most of the flows are served from the two previously mentioned Limelight locations at 30ms and 32ms. However, in this case there are



(a) min RTT per services.

(b) min RTT per hosting organization.

Figure 7. Distribution of min RTT per service and per hosting organization. A big share of Facebook, Apple, and Windows Update flows come from servers located in the same city of the vantage point. More than 60% of the Akamai HTTP flows come from servers "inside the ISP", with min RTT values smaller than 5ms.



(a) min RTT of YouTube flows.

(b) min RTT of Facebook flows.

(c) min RTT of AVS 2 flows.

Figure 8. Daily min RTT for YouTube, Facebook, and AVS 2. Google CDN and Akamai make use of internal load balancing policies to serve content from different hosting locations.

no observable temporal patterns, suggesting that Limelight is not applying load balancing techniques in Europe, at least not for provisioning the corresponding service.

V. HTTP Traffic Classification using IPs

We shall now shift the analysis towards the second use-case tackled in this paper: automatic classification of HTTP flows based solely on the IP address of the server being contacted. The approach we follow is straightforward; in a nutshell, given a specific service S_i to be identified, we build a set of k_i well-known IP addresses $IP_i = \{ip_i(1), ip_i(2), \ldots, ip_i(k_i)\}$ hosting S_i, using the associations $A_i = \{S_i.IP_i\}$ between server IPs and services provided by HTTPTag on a certain *learning* period. Given a list of m services $S_{i,\{i=1..m\}}$ to classify and an HTTP flow f_{new} coming from IP address ip_{new}, we apply the following classification rule: $\mathcal{F}(f_{new}) = S_i \leftrightarrow ip_{new} \in IP_i$.

As we showed in previous section, given the widespread usage of third-party hosting organizations serving the content of multiple services (e.g., Akamai), the big number of companies hosting multiple services in the same locations (e.g., Google CDN), and the ISPs content caching policies, multiple different services S_i might be associated to the same server IP address, which actually means that the m sets IP_i are not necessarily disjoint sets. We shall refer to this IP sets intersection issue as IP *hosting collisions*. Such collisions are observed both in figure 5 and 7. For example, figure 5(a) shows that about 8% of the Facebook IPs are in the same range of about 17% of Windows Update IPs and 3% of Apple IPs, and that about 16% of the IPs used by AVS 1 also intersect with Windows Update IPs. There are also IP hosting collisions between Google Search and YouTube, AVS 1 and AVS 2, and among Facebook, Apple, and Windows Update on a different IP range. In this case, the previous classification rule would associate f_{new} to all those services mapped to ip_{new}. To solve this multi-classification issue and decide for one single output, we use a simple random selection approach, in which the decided service is randomly chosen among the potential ones. Such a straightforward decision approach could be improved by heuristics, for example by adding weights to the candidate services based on different criteria (e.g., size of the IP sets), but we shall keep it simple in this paper.

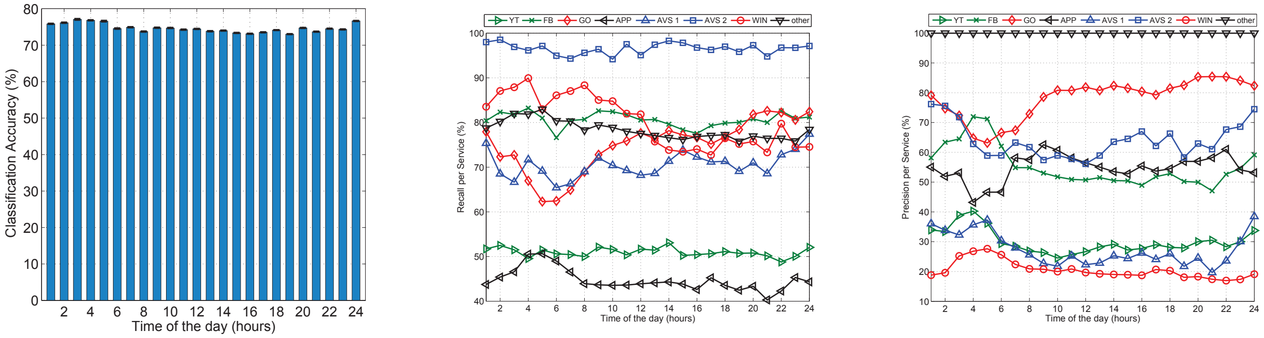
To test the classification performance achieved for each of the analyzed top 7 services, we divide the complete week of labeled HTTP flows in $n = 8$ *classes*: the first 7 correspond to the top 7 services, whereas the 8th class corresponds to all the rest of the labeled flows and will be referred to as the `other` class. Using the labeled traffic flows from Monday as learning dataset, we construct 7 IP sets IP_i containing all the unique IPs per service observed during the day. The size of each of this sets is available from table II: $\{\#IP_i\} = \{1373, 2031, 1875, 522, 92, 456, 743\}$. The classification associated to the class `other` is simply done by a complementary decision rule: if according to $\mathcal{F}(f_{new})$, flow f_{new} is not assigned to any of the top 7 services, then it is assigned to the `other` class.

To asses the classification performance of the aforementioned approach, we employ three traditionally used performance metrics in the traffic classification literature: the Classification Accuracy (CA), the Recall (R$_i$), and the Precision (P$_i$) per class:

$$\text{CA} = \frac{\sum_{i=1}^{m} TP_i}{n}, \quad \text{R}_i = \frac{TP_i}{TP_i + FN_i}, \quad \text{P}_i = \frac{TP_i}{TP_i + FP_i}$$

where TP_i corresponds to the number of correctly classified flows in class i (i.e., number of true positives), and FN_i and FP_i correspond to the number of false negatives and false positives in class i. The classification accuracy indicates the percentage of correctly classified flows among the total number of flows n. Recall R$_i$ is the number of flows from class i correctly classified, divided by the total number of flows in class i. It measures the per-class accuracy. Precision P$_i$ is the percentage of flows correctly classified as belonging to class i among all the flows classified as belonging to class i, including true and false positives. It measures the fidelity (i.e., variance of the classification error) of the classifier regarding each particular class.

Figure 9 depicts the classification performance achieved in the learning day (i.e., Monday), on an hourly basis. Given the random decision process used in case of IP hosting collisions, the algorithm is run 20 consecutive times, and the provided results correspond to the obtained average values. Figure 9(a) depicts the classification accuracy for the 8 defined classes, including the error variance bounds resulting from the 20



(a) Classification Accuracy.

(b) Recall per Service.

(c) Precision per Service.

Figure 9. Classification performance achieved in the learning day. The overall classification accuracy is remarkably high and stable during the day, rounding about 75% of correctly classified HTTP flows. More than 60% of all the Facebook, Adult Video, Google Search, and Windows Update HTTP flows are correctly classified.

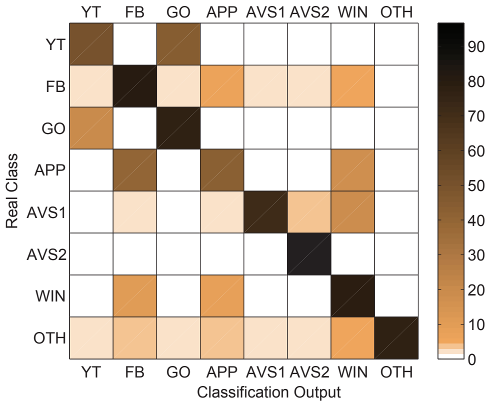


Figure 10. Confusion matrix for traffic classification. Many YouTube flows are classified as Google Search. Windows Update flows are misclassified as Facebook and Apple, given the previously mentioned IP hosting collisions within Akamai.

consecutive runs, which are negligible. The overall classification accuracy is remarkably high and stable during the day, rounding about 75% of correctly classified HTTP flows. These a-priori excellent results achieved by only using IP addresses can be in fact misleading, because we are considering the other class inside the classification process, which contains a much larger number of unique IPs. Figure 10 shows the confusion matrix for the classification results. Many YouTube flows are classified as Google Search, and vice versa. Windows Update flows are misclassified as Facebook and Apple, given the previously mentioned IP hosting collisions within Akamai. Similar behavior is observed for Apple. As previously observed, the AVS 2 service is accurately classified with a very low false negatives rate.

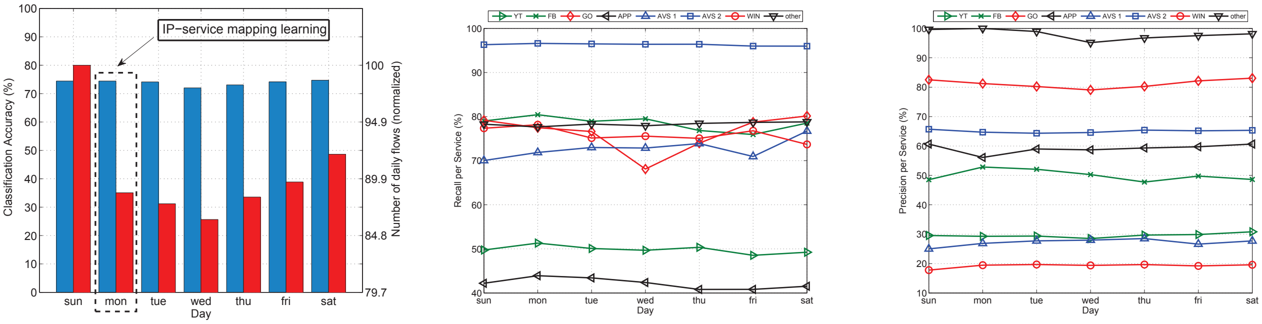
Let's focus now on the per service recall and precision, depicted in figures 9(b) and 9(c) respectively. The recall or per-service classification accuracy is still remarkably high and stable during the day, with more than 60% of all the Facebook, Adult Video Streaming, Google Search, and Windows Update HTTP flows correctly classified. Specially in the case of the AVS 2 service, recall is as high as 98%, and both Facebook and Windows Update HTTP flows are identified with a per-class accuracy above 80%. However, YouTube flows are poorly classified, and the recall achieved is between 40% and 50%. The main reason for these poor results comes directly from the IP hosting collisions associated to Google CDN and Akamai, as many of the YouTube and Apple flows are classified as Google Search and Facebook or Windows Update flows respectively, as depicted in figure 10.

When it comes to evaluate the per-service precision, the achieved results are much less encouraging, and show in all the cases that many of the flows are assigned to classes sharing similar IP ranges. The recall obtained for Google flows is still pretty high and above 80% from 9 am onwards, but results for YouTube, AVS 1, and Windows Update show a big number of false positives associated to these services. As expected, the precision for the other class is of 100% during the complete learning day, which comes directly from the applied classification technique for this specific class.

The final analysis consists in the classification performance evaluation on the complete week of traffic traces, using the IPs of Monday as learning data. Figure 11 depicts the per-day accuracy, recall and precision achieved in the 7 days of the study. Figure 11(a) shows that the classification accuracy is remarkably stable during the full week, clearly suggesting that the sets of IPs delivering the different services are stable in time, at least in a weekly-basis. The figure additionally shows the normalized number of analyzed flows per day, to have an idea of the volume variations during the week. Figures 11(b) and 11(c) additionally present the daily recall and precision for the full week, showing once again that classification performance is very stable in time. In fact, achieved results remain almost unchanged from those obtained during the training day, achieving a classification accuracy close to 75%.

VI. Concluding Remarks

In this paper we have addressed the problem of extracting useful knowledge from the dynamics of the Internet addressing space, specially targeting the characterization of the top web services, their hosting organizations and the way services are delivered to the end-users, as well as the problem of HTTP traffic classification from network measurements. Using a full week of HTTP traffic traces collected at the mobile broadband network of a major European ISP, we have investigated the associations between services and the IPs assigned to



(a) Daily Classification Accuracy and num Flows.

(b) Daily Recall per Service.

(c) Daily Precision per Service.

Figure 11. Classification performance achieved in the analyzed week of HTTP traffic. The classification accuracy is stable during the complete week, and around 75% of the HTTP daily flows are correctly classified.

the servers providing the content, additionally evaluating the performance of a very simplistic approach to classify the top services accessed by the users of the studied network.

Among our main findings regarding web services and their hosting/delivery servers, we have shown how dynamic and distributed are current major CDN players like Google and Akamai, providing not only large numbers of servers or IPs at very distributed locations, but also making use of load balancing techniques to shift HTTP flows among their preferred hosting locations. We have also shown evidence on the more static approach followed by other CDNs like Limelight, reflecting a different philosophy for CDN architectures.

Regarding HTTP flows classification, we have shown that despite its simplicity, the IP-based approach is able to classify the HTTP flows of the top services with a classification accuracy as high as 75%. However, we have also seen that the classification recall and precision are highly impacted by IP hosting collisions, seriously impacting its performance as a robust traffic classifier. Still, results obtained for some of the analyzed services like Google Search, Facebook, Windows Update, and AVS services were encouraging, achieving a daily per-class accuracy above 70% in all the cases, with precision values above 65% for Google Search and AVS 2. This paper has therefore provided evidence on the possibilities of using such a minimalist approach for recognizing the top HTTP services in terms of end-user consumed traffic volumes, offering a practical and very flexible solution for traffic aware networking.

Acknowledgments

This work has been performed in the framework of the EU-IP project mPlane, funded by the European Commission under the grant 318627.

References

[1] A. Gerber et al., “Traffic Types and Growth in Backbone Networks”, in *OFC/NFOEC*, 2011.

[2] G. Maier et al., “On Dominant Characteristics of Residential Broadband Internet Traffic”, in *ACM IMC*, 2009.

[3] C. Labovitz et al., “Internet Inter-domain Traffic”, in *ACM SIGCOMM*, 2010.

[4] V. Gehlen et al., “Uncovering the Big Players of the Web”, in *PAM*, 2012.

[5] C. Huang et al., “Measuring and Evaluating Large-Scale CDNs”, in *ACM IMC*, 2008.

[6] R. Krishnan et al., “Moving Beyond End-to-End Path Information to Optimize CDN Performance”, in *ACM IMC*, 2009.

[7] E. Nygren et al., “The Akamai Network: A Platform for High-Performance Internet Applications”, in *ACM SIGOPS* 44(3), 2010.

[8] J. Ermanet al., “Network-Aware Forward Caching”, in *WWW*, 2009.

[9] J. Erman et al., “HTTP in the Home: It is not just about PCs”, in *ACM CCR* 41(1), 2011.

[10] A. Dainotti et al., “Issues and Future Directions in Traffic Classification”, in *IEEE Network*, 2012.

[11] A. Finamore et al., “Experiences of Internet Traffic Monitoring with Tstat”, in *IEEE Network* 25(3), 2011.

[12] T. Nguyen et al., “A Survey of Techniques for Internet Traffic Classification using Machine Learning”, in *IEEE Comm, Surv. & Tut.*, 2008.

[13] A. Moore et al., “Internet Traffic Classification using Bayesian Analysis Techniques”, in *ACM SIGMETRICS*, 2005.

[14] M. Roughan et al., “Class-of-Service Mapping for QoS: a Statistical Signature-Based Approach to IP Traffic Classification”, in *ACM IMW*, 2004.

[15] N. Williams el al., “A Preliminary Performance Comparison of Five Machine Learning Algorithms for Practical IP Traffic Flow Classification”, in *ACM CCR*, vol. 36 (5), 2006.

[16] S. Valenti et al., “Accurate, Fine-Grained Classification of P2P-TV Applications by Simply Counting Packets”, in *TMA*, 2009.

[17] P. Casas et al., “MINETRAC: Mining Flows for Unsupervised Analysis & Semi-Supervised Classification”, in *ITC*, 2011.

[18] I. Bermudez et al., “DNS to the rescue: Discerning Content and Services in a Tangled Web”, in *ACM IMC*, 2012.

[19] P. Fiadino, et al., “HTTPTag: A Flexible On-line HTTP Classification System for Operational 3G Networks”, in *IEEE INFOCOM*, 2013

[20] W. Li et al., “Classifying HTTP Traffic in the New Age”, poster, in *ACE SIGCOMM*, 2008.

[21] F. Schneider et al., “Pitfalls in HTTP Traffic Measurements and Analysis”, in *PAM*, 2012.

[22] A. Bär et al., “Two Parallel Approaches to Network Data Analysis”, in *LADIS*, 2011.

[23] F. Ricciato, “Traffic Monitoring and Analysis for the Optimization of a 3G network”, in *IEEE Wireless Communications*, vol. 13(6), 2006.

[24] MaxMIND GeoIP Databases, available at http://www.maxmind.com. [Accessed on 20-03-13]