TECHNISCHE UNIVERSITÄT WIEN
Vienna | Austria

Dissertation

Traffic Characterization, Anomaly Detection and Diagnosis in Internet Scale Services

submitted in partial fulfillment of the requirements for the degree of
Doktor der technischen Wissenschaften
(Doctor of Technical Sciences)

to the Technische Universität Wien (TU Wien)
Faculty of Electrical Engineering and Information Technology

by

Pierdomenico Fiadino, M.Sc.
student number 1128703

Vienna, December 10th 2015

advisors: **Univ.Prof. Dr.-Ing. Tanja Zseby**
TU Wien, Austria

Dr. Pedro Casas
Telecommunications Research Center of Vienna, Austria

examiner: **Prof. Dr. Benoit Donnet**
University of Liège (ULg), Belgium

Dissertation

Traffic Characterization, Anomaly Detection and Diagnosis in Internet Scale Services

ausgeführt zum Zwecke der Erlangung des akademischen Grades
eines **Doktors der technischen Wissenschaften**

eingereicht an der
Technische Universität Wien
Fakultät für Elektrotechnik und Informationstechnik

von

Pierdomenico Fiadino, M.Sc.
Matrikelnummer 1128703

Wien, 10. Dezember 2015

Betreuung: **Univ.Prof. Dr.-Ing. Tanja Zseby**
Technische Universität Wien, Österreich

Dr. Pedro Casas
Forschungszentrum Telekommunikation Wien (FTW), Österreich

Gutachter: **Prof. Dr. Benoit Donnet**
University of Liège (ULg), Belgium

a mio padre

«Scusate la lunghezza di questa lettera» scriveva
un francese (o una francese) del gran settecento
«poiché non ho avuto tempo di farla più corta».

Leonardo Sciascia

Acknowledgements

I would like to start expressing my deepest gratitude to Dr Pedro Casas. It is customary to open the acknowledgments mentioning one's own supervisor. This time, however, it is not just about the *etiquette*. I could spend a thousand words to say how much Pedro has helped me during my PhD with his careful supervision, but the truth is that he has been much more than a supervisor for me: I consider him a mentor and an example to follow, professionally and personally. He has taught me to carry on with optimism and face problems with a smile.

A huge thanks to Dr Alessandro D'Alconzo for the central role he has had for me in these years, with his experience and his methodical guidance. Without his help, this work would not have been possible.

My most sincere gratitude goes to Prof Tanja Zseby for her precious supervision. Her valuable and constructive suggestions greatly contributed to the quality of this work.

I am very thankful to Dr Joachim Fabini for his many suggestions and the interesting discussions we have had in the last year.

I would also like to thank my former colleagues Danilo Valerio, Mirko Schiavone, and Arian Bär for their support, and, most of all, for the fun we have had. Thanks to all my other colleagues at FTW that contributed to create the most exiting and happiest working environment one could possibly imagine. I am sure we will forever have fond memories of the time we have spent together.

Thanks a lot to my friend Julia Schrodi for her support in translating the abstract in German. I know I should have been able to do it by my self at this point, but, as the saying goes, "life is too short to learn German".

Finally, a special thanks to my small family, Mireia and Pau, the source of my motivation and happiness ☺

Abstract

Rewinding the clock of the Internet to a decade ago, network traffic was largely dominated by peer to peer (P2P) file sharing and web services were provided by centralized or barely distributed platforms. Today the situation has drastically changed: the most popular services rely on web technologies, while highly dynamic and distributed Content Delivery Networks (CDNs) rule the Internet's landscape. The explosion of cloud-based services, the ever-growing volume of video streaming traffic, and the large user-base of online social networks call for sophisticated load balancing and caching techniques to optimize the usage of the underlying transport network, as well as the end-user experience.

As a result, current Internet traffic patterns are characterized by a much higher dynamism, posing serious challenges to network operators. Understanding today's traffic has become a daunting task, making traffic engineering, network optimization, and trend analysis arduous processes. The picture is complicated by the growing occurrence of unexpected anomalies which potentially impact the interests of the involved stakeholders, from the end-user's experience to the network planning enforced by providers.

In the light of this Internet scenario, we claim that traditional network analysis techniques need to be revised to better capture and explain current and future traffic dynamics. This thesis brings three major contributions to the field of network traffic monitoring and analysis.

The first contribution regards the analysis and characterization of Internet scale services and large-scale provisioning systems. We have not only analyzed and dissected rarely explored services and popular CDN infrastructures using both passive and active measurements, but also proposed multiple novel techniques to unveil their traffic patterns in both normal operations and during anomalies, even when they run on encrypted protocols.

The second contribution targets the automatic detection of network and traffic anomalies in modern services, where we have proposed novel anomaly detection techniques as well as extended previous proposals to self-adapt to current Internet dynamics and flag relevant anomalies. In particular, anomalies impacting both the experience of the end users as well as the performance of the network have been discovered through the proposed techniques. The detection performance of our system was compared against well-known solutions, such as entropy-based detectors, showing outperforming results in several cases.

The last contribution focuses on the diagnosis of the detected issues. We have provided a framework to unveil the root causes behind the flagged anomalies, relying on Machine Learning techniques and on the combined analysis of symptomatic and diagnostic passive measurements. We have also devised the design of a more advanced approach that relies on the analysis of both passive and distributed active measurements to iteratively investigate the anomalies.

To provide strong evidence on the relevance of our contributions, the presented studies were validated using real large-scale traffic measurements from different operational networks, including both cellular and fixed-line. Taking together the ensemble of the contributions, this thesis offers a holistic approach for network operators to efficiently monitor Internet scale services and interpret unexpected network traffic behaviors.

Zusammenfassung

Wenn man die Uhren des Internets um ein Jahrzehnt zurückdreht, war der Netzwerk Datenverkehr hauptsächlich von "peer to peer" (P2P) file sharing geprägt und Internet Dienste wurden von zentralisierten oder kaum verbreiteten Plattformen angeboten. Heute hat sich die Situation drastisch geändert: Die beliebtesten Dienste basieren auf webbasierten Diensten, während dynamische und weit verbreitete "Content Delivery Networks (CDNs)" die Internetlandschaft prägen. Die explosionsartige Verbreitung von "cloud-based-services", das stetig steigende Volumen von Videostreaming Datenverkehr, und die riesige Anzahl an Nutzern von sozialen Netzwerken, fordern gut durchdachte Lastverteilungsstrategien und fortgeschrittene Caching-Techniken um die Anwendung für die zugrundeliegenden Transportnetzwerke sowie die Anwendung für den Endverbraucher zu optimieren.

Als Folge dieser Entwicklung zeigt Datenverkehr heutzutage eine weitaus höhere Dynamik auf, welche eine ernsthafte Herausforderung für die Netzwerkbetreiber darstellt. Diesen Datenverkehr zu untersuchen und zu verstehen ist zu einer schwierigen Aufgabe geworden und die Bereiche Traffic Engineering, Netzwerk Optimierung und Trendanalyse erfordern aufwendige Prozesse. Diese Bereiche werden außerdem durch das erhöhte Auftreten von unerwarteten Anomalien erschwert, welche die Interessen der beteiligten Akteure vom Netzwerkplaner bis zum Endverbraucher beeinflussen können.

Angesichts dieses Internetszenarios, ist eine Überarbeitung traditioneller Netzwerkanalyse Techniken erforderlich, um aktuelle und zukünftige Dynamiken des Internetverkehrs besser erfassen zu können. Diese Arbeit liefert drei Hauptbeiträge auf dem Gebiet der Verkehrsbeobachtung und -analyse von Internetdaten.

Ein erster Beitrag betrifft die Analyse und Charakterisierung von Internetdiensten und weit verteilten Systemen zu Bereitstellung von Diensten, dazu haben wir neue bisher wenig erforschte Dienste und weit verbreitete CDN Infrastrukturen unter Anwendung von aktiven und passiven Messungen analysiert und verschiedene neue Methoden vorgeschlagen, um deren Muster im Internetdatenverkehr bei normalem Verhalten und bei Anomalien zu untersuchen, auch wenn diese verschlüsselte Protokolle nutzen.

Der zweite Beitrag dieser Arbeit hat die automatische Detektion von Anomalien in modernen Internet Diensten als Ziel. Dafür haben wir neue Techniken zur Erkennung von Anomalien vorgeschlagen, sowie bestehende Techniken erweitert, um eine automatischen Anpassung an aktuelle Internet Dynamiken zu realisieren und relevante Anomalien zu markieren. Durch diese Techniken wurden insbesondere Anomalien entdeckt, welche sowohl die Servicequalität beim Endnutzer (Quality of Experience) als auch die Performanz des Netzwerks beeinflussen. Die Detektionsleistung unseres Systems wurde mit existierenden Arbeiten verglichen und konnte diese in vielen Fällen übertreffen.

Der letzte Beitrag befasst sich mit der Diagnose der erkannten Probleme. Wir haben ein Rahmenwerk vorgeschlagen, das sich auf maschinelle Lerntechniken und auf die kombinierte

Analyse von symptomatischen und diagnostischen passiven Messungen stützt, um die Grundursachen hinter den markierten Anomalien zu enthüllen. Wir haben außerdem ein Design für einen weiter fortgeschrittenen Ansatz entwickelt, welcher auf der Analyse von passiven und verteilten aktiven Messungen basiert, um Anomalien iterativ zu analysieren.

Um die Relevanz der Beiträge zu untermauern wurden die Studien unter Anwendung von realen Internetverkehrsmessungen in verschiedenen Netzwerken, sowohl von Mobilfunk- als auch von Festnetzanbietern, überprüft.

Damit liefert diese Arbeit einen ganzheitlichen Ansatz für Netzbetreiber, um Internetdienste effizient zu steuern und unerwartetes Verhalten von Internetdatenverkehr zu analysieren und zu interpretieren.

Contents

List of Figures

List of Tables

List of Acronyms

2LD	2nd Level Domain Name
3LD	3rd Level Domain Name
ACC	Acceptance
AD	Anomaly Detection
ADSL	Asymmetric Digital Subscriber Line
ADT	Average Download Throughput
ANN	Artificial Neural Network
APN	Access Point Name
AS	Autonomous System
ASN	Autonomous System Number
AVS	Adult Video Service
C2P	Customer To Provider
CA	Classification Accuracy
CCDF	Complementary Cumulative Distribution Function
CDF	Cumulative Distribution Function
CDN	Content Delivery Network
DDoS	Distributed Denial of Service Attack
DNS	Domain Name System
DPI	Deep Packet Inspection
ENKLd	Entropy Normalized Kullback-Leibler Divergence
EWMA	Exponential Weighted Moving Average
FDT	Flow Download Time
FPR	False Positive Rate
FQDN	Fully Qualified Domain Name
GA	Global Accuracy
GGSN	Gateway GPRS support node
GPRS	General Packet Radio Service
HTTP	Hypertext Transport Protocol
HTTPS	Secure Hypertext Transport Protocol
IANA	Internet Assigned Numbers Authority
ICMP	Internet Control Message Protocol
ID	Identifier
IM	Instant Messaging
IMEI	International Mobile Station Equipment Identity
IMSI	International Mobile Subscriber Identity
IP	Internet Protocol
ISP	Internet Service Provider
IXP	Internet eXchange Point
KLd	Kullback-Leibler Divergence

KPI ..... Key Performance Indicator
LWL ..... Locally-Weighted-based Learning
ML ..... Machine Learning
MLP ..... Multi-Layer Perceptron
MOS ..... Mean Opinion Score
MP3 ..... MPEG-2 Audio Layer III
MSISDN ..... Mobile Subscriber ISDN Number
NB ..... Naive Bayes
NO ..... Neighbor Operator
OS ..... Operating System
OSN ..... Online Social Network
P2P ..... Peer 2 Peer
PDF ..... Probability Distribution Function
PEP ..... Performance Enhancement Proxies
POP ..... Point of Presence
QoE ..... Quality of Experience
RAT ..... Radio Access Technology
RCA ..... Root Cause Analysis
RF ..... Random Forest
ROC ..... Receiver Operating Characteristic
RST ..... Reset flag (in TCP)
RTT ..... Round Trip Time
SGSN ..... Serving GPRS support node
SLA ..... Service Level Agreement
SQL ..... Standard Query Language
SSL ..... Secure Socket Layer
SVM ..... Support Vector Machines
SYN ..... Synchronize packet (in TCP)
TAC ..... Type Allocation Code
TC ..... Traffic Classification
TCP ..... Transmission Control Protocol
TMA ..... Traffic Monitoring and Analysis
TLS ..... Transport Layer Security
TPR ..... True Positive Rate
TSP ..... Temporal Similarity Plot
TTL ..... Time To Live
UDM ..... User Defined Measurement
UDP ..... User Datagram Protocol
US ..... United States
VBR ..... Variable Bit Rate
VD ..... Video Duration
VP ..... Vantage Point
VPT ..... Video Played Time
WWW ..... World Wide Web
XML ..... Extensible Markup Language
XMPP ..... eXtensible Messaging and Presence Protocol

1. Introduction

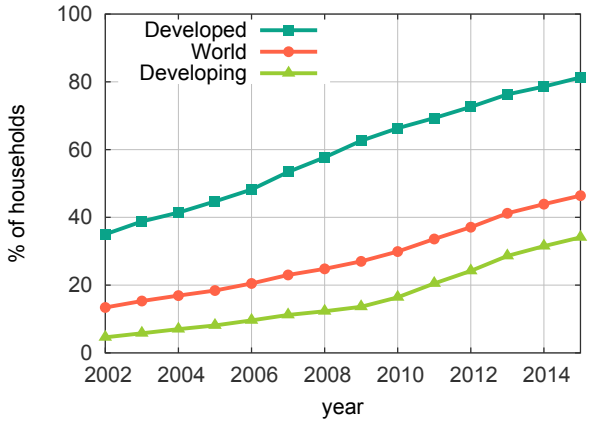
> The Internet is the first thing
> that humanity has built that
> humanity doesn't understand,
> the largest experiment in anarchy
> that we have ever had.
>
> *Eric Schmidt*
> *executive chairman of Alphabet Inc*

The Internet is a global infrastructure that connects billions of devices for the sake of exchanging information. There are no organizations or administrative entities operating or controlling it on a global scale, but it is rather an interconnection of locally managed and independent networks. Its diversity provides it with great flexibility and good resiliency, and has driven rapid innovation at the edge. Indeed, in the last two decades, after the advent of the World Wide Web (WWW) and the interest of the general public, we have witnessed an astonishing growth of the Internet in terms of infrastructure, traffic volume, and number of users. According to the International Telecommunication Union (ITU), it is estimated that more than 46% households world-wide have Internet access in 2015, with this fraction peaking up to 82% in developed countries [1] (cfr. Figure 1.1(a)). It is expected that these figures will rapidly increase, as large shares of population in developing countries are getting access to the Internet, reducing the world digital divide. The fast adoption of mobile broadband (cfr. Figure 1.1(b)) is maybe one the main driving factors of this growth, easing the deployment of access infrastructures and reducing costs. To support the Internet usage increase, there has been a collateral boost in the distribution and complexity of the overall infrastructure: for example, the number of Autonomous Systems assigned by IANA has reached almost 50.000 in 2015 [1], more than 20 times larger than eighteen years ago (cfr. Figure 1.1(c)).

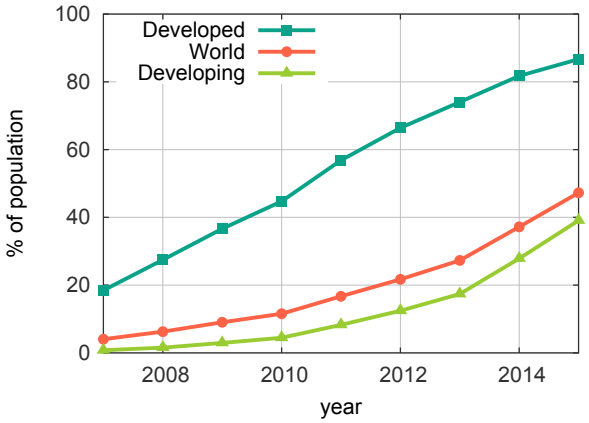
The independent and distributed nature of the Internet is for sure its main value. Even though there have been —and there are– many attempts to control it, and in some cases to censor it, it still offers an unprecedented communication mean, allows access to an unlimited number of resources and information, opens up to opportunities for participation and, ultimately, democracy and freedom.

1.1. A Tangled Internet

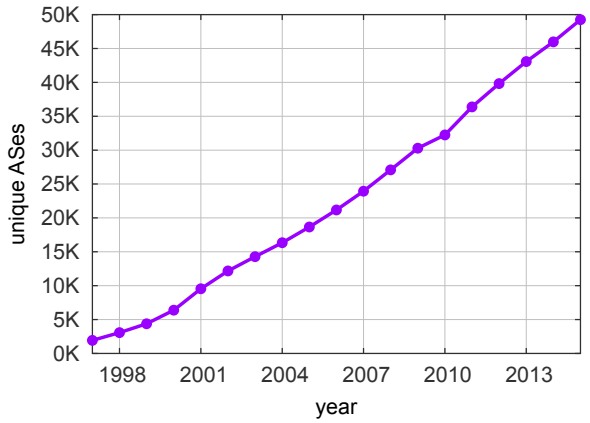
However, this large "experiment in anarchy" does not come without costs. The same independent and distributed nature, that allows Internet's very own existence as we know it, poses



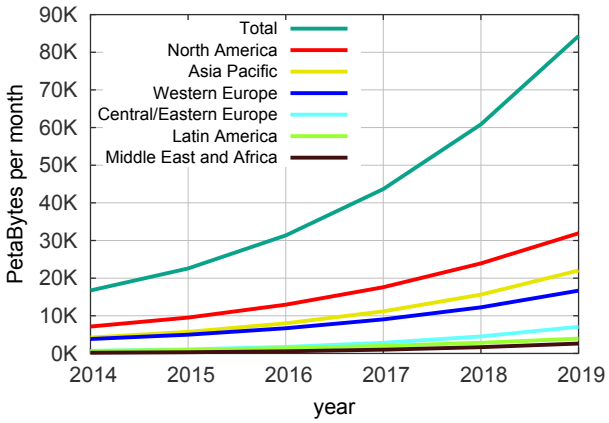
(a) Households Internet access. Data source: [1]



(b) Mobile broadband access. Data source: [1]



(c) Number of unique ASes. Data source: [2]



(d) CDN traffic volume forecast. Data source: [3].

Figure 1.1.: Internet statistics: trends and forecasts.

serious challenges to network operators. The Internet has shown to be fragile to problems arising from interactions among networks and to misbehaving terminal nodes. Understanding the cause of performance degradation, service failures, or the presence of anomalous traffic behaviors such as abusive traffic, flash crowds, or routing outages has become a daunting task, made even more challenging by the fast and constant deployment of new services and applications. As the number of users and network applications raises, content and service providers have to deploy increasingly complex provisioning infrastructures to cope with the big user demands for reliability and quality. The great success of specialized Content Delivery Networks (CDN) confirms this trend: according to recent reports [3], the CDN North American market was estimated worth $1.95 billions in 2013, and it is expected to reach $12 billions by 2019 [4], with total traffic volume served by CDNs multiplied by four (cfr. Figure 1.1(d)). Today, the general approach for Internet scale services is to push contents as close as possible to users, adopting sophisticated load balancing techniques to do so. Popular services rely on distributed data centers to ensure high availability, scalability and low

latency. As a consequence, it has become extremely difficult for operators to track the usage of popular applications, as their traffic patterns are highly dynamic and volatile. This reduces the degree of freedom for operators to take the correct countermeasures when issues arise, limiting the effectiveness of network management and operational activities. The Internet is often a large, obscure black box, and network operators and application providers lack today the necessary mechanisms to drill down to the cause of problems therein.

1.2. The Role of Network Measurements

In the years, a long list of measurement technologies and tools have been developed in order to support network operators in their daily business. Some of the basic and ubiquitous active probing tools, such as ping and traceroute, or passive packet analyzers like Wireshark [5], just to name the most popular ones, are generally enough for providing basic insights on the root-cause of simple network issues, but their effectiveness is very limited, or totally inexistent, in more complex cases. Large scale provisioning systems make the traffic characterization and troubleshooting more challenging every day; in this scenario, the diagnosis of anomalies in large-scale services require better tools and algorithms than what operators have traditionally applied. This has driven a great interest in the field of network data analytics in order to mitigate these hurdles. In the last years, a lot of work has been dedicated to develop more sophisticated measurement and analysis platforms to understand and characterize the complexity of the Internet through a higher degree of network visibility. Passive probes for large-scale monitoring [6, 7], distributed active measurement frameworks [8, 9, 10] and elaborated systems for network traffic analysis based on active measurements [11, 12] have been designed with this goal in mind.

Nonetheless, today we still face the lack of a flexible monitoring approach, and corresponding algorithms, that can guide network operators in the process of understanding and troubleshooting problems in complex networks as the Internet. Many of the problems that we see in current monitoring systems and algorithms for detecting and diagnosing network anomalies derive from the same problem: the lack of a structured approach that allows flexibility and adaptability. Indeed, while the problem of network anomaly detection has received a lot of attention, understanding the nature and causes of these anomalies is still in most cases a manual task. Root cause analysis is usually left to network operators, who use their knowledge and experience to analyze the traffic flows where the anomaly was flagged in search of events that can explain it, or that give some hints on which additional measurements should be performed to better understand the problem. This manual process is time-consuming, error-prone, and can be prohibitive if the number of events and the size of the data to analyze are too big, which is more and more the case. In light of this, after more than a decade of research in the field of network monitoring and analysis, there is still fertile ground and open research questions.

The research community is still very active in trying to fill these gaps. Projects like mPlane [13] are currently aiming at rising the bar in our understanding of the Internet by moving forward the state of the art in network monitoring and analysis. In particular, mPlane has the goal to design and develop a distributed and *ubiquitous* approach to network measure-

ments. Its key idea is to provide a framework for setting-up measurements in a flexible way, through the *hybridation* of active and passive monitoring schemes, and ultimately allowing the deployment of advanced iterative troubleshooting algorithms. The work presented in this dissertation has been carried out in the context provided by the mPlane project and the Traffic Monitoring and Analysis (TMA) research community in general.

1.3. The Uphill Race to Network Monitoring

There are a number of reasons why there are still unanswered questions in the field of network measurements today, but probably the first and most prominent one is the limited accessibility of datasets. Traffic monitoring and analysis relies mostly on passive probing Vantage Points (VP) located in large scale networks. However, the access to such monitoring data is difficult for the research community, as network operators are reluctant to disclose sensitive information outside their boundaries. The reasons should be sought in both the privacy and business spheres. On the one hand, there are legitimate concerns that prevent operators from releasing sensible details that violate the privacy of their customers (which could be somehow solved by adopting data anonymization and aggregation techniques). On the other hand, ISPs are afraid of revealing information that could prejudice their competitiveness on the market.

Due to these reasons, they tend to adopt a very narrow perspective, reducing the chances of collaborations with neighbor ISPs, competitors, and research institutions. However, this attitude can be counter-productive, mostly for access providers, as they become hostage of obscure network policies deployed by third-party content providers and are down-graded to mere *dump pipes* [14]. While we wait for a federated approach to measurements, in which all the involved parties partially disclose information for the sake of network optimization, troubleshooting, and network neutrality verification, we can simply resign to the current state of things. Note that the monitoring framework proposed by the mPlane project provides the instruments for achieving a distributed measurement scheme, laying the fundamental stone for the solution of this problem, at least from the technical point of view.

In this thesis, I had the – far from being given for granted – chance to access a number of passive measurement campaigns carried out in the 2012-2014 time span at the operational networks of Nation-wide operators (both cellular and fixed-line). This allowed me to produce results that reflect real-world operational conditions. But, as a consequence of the aforementioned problems, I am not allowed to provide the datasets used for supporting the described analysis. Nevertheless, the structured description of the methodology I provide – which is one of my main contributions – is general enough to be used in different contexts and offers the guidelines for the reproducibility of results in other networks.

The second hurdle related to network monitoring is more technical. In addition to a deep knowledge of advanced data mining techniques, the study of large datasets needs adequate technologies and hardware, especially when the correlation of diverse data sources is required. Luckily, the progress of big-data technologies is filling this gap, providing instruments that let data scientists to focus on the actual data analytics, rather than the technical aspects. In particular, in this thesis I have used a specialized data-warehouse system tailored for traffic

monitoring that greatly facilitates the analysis of data streams. I believe that the new data-processing technologies will contribute, in the short term, to advance the state of the art in the field of network monitoring and analysis.

1.4. Research Questions and Contributions

While some of the obstacles presented above are beyond our reach, in this thesis I tackle some of the network monitoring open issues. As a general approach, I have tried to identify and combine state of the art solutions in a systematic approach to answer the three main research questions listed below. I do not want to re-invent the wheel, but rather couple well-known techniques with novel approaches to shed light on some obscure Internet dynamics, with the final goal of designing new approaches to detect and diagnose large scale network anomalies.

(Q_I) Classification of Services in an Encrypted World: How to Classify Encrypted Applications? Traffic classification is a prosperous field with a large literature. In the past years, the main effort was concentrated on distinguishing application layer protocols, mostly driven by the ISPs' concerns about specific services, such as P2P file exchange. However, as most of the applications beyond traditional web surfing have already moved on top of HTTP (video streaming, online social networks and gaming, web mail, text/audio/video chat, file transfer, etc.), the interest is shifting rather to the differentiation of Web services. This is not a straight-forward task, also due to the increasing use of end-to-end encryption and the *HTTPS-everywhere* scheme. We deal with the problem of differentiating Web services in Chapter 3.

(Q_{II}) Understanding Network Data: How to Track Large Scale Provisioning Systems and Detect Unexpected Events? After being able to distinguish flows belonging to different services, it is of vital importance for ISPs to understand where their traffic is flowing from, i.e., they need to have a clear knowledge of large-scale service provisioning systems, in order to enforce an efficient network planning, optimization and trend analysis. The static relation between a service and a (small) set of hostnames is an abandoned paradigm by now: specialized CDNs allow an efficient decoupling of services and serving IP addresses, deploying highly dynamic load balancing policies. In this scenario, understanding data flows, in particular focusing on the most popular large scale provisioning systems is still an open question. To be more specific, the main sub-problems are: (i) who are the big players in today's Internet?, in terms on generated traffic volume and number of users; (ii) how (geographically and topologically) distributed are their hosting infrastructures?; (iii) how and by using which traffic features can we track the dynamic load balancing policies and study their evolution?; (iv) does the access technology (e.g., cellular vs fixed-line) induce any difference in the traffic patterns?; (v) what are the flow characteristics and how they relate to user perceived Quality of Experience?. I address these and other minor points in Chapter 4. Unfortunately, the problem of understanding network data is not limited to this: the study of network traffic patterns is made more difficult in presence of anomalies. It is, in fact,

very hard to distinguish normal changes induced by CDNs policies from the ones provoked by anomalous behaviors. I discuss a number of case studies in Chapter 5, where I try to answer the following sub-questions: (vi) what are the traffic features that better describe the characteristics of the anomaly?; (vii) what are the effects of the anomaly? (e.g., outage, performance degradation); (viii) which of the involved stakeholders is negatively affected by it?.

(Q_{III}) Towards a More Structured Approach for Anomaly Detection and Diagnosis: How to Automatize the Detection and Diagnosis of Anomalies in Current Internet? Literature is rich on techniques for detecting anomalous events. Most of the existing algorithms tackle security or performance degradation issues, which generally correspond to transient effects. As said before, today's Internet is characterized by highly dynamic traffic patterns, with frequent changes that are not necessarily related to service or network anomalous conditions, questioning the applicability of traditional anomaly detection approaches. To this extent, I investigate which existing detection algorithm is able to unveil anomalies occurring in modern large scale provisioning systems and how it could be improved. The detection of anomalies is only the first step in the troubleshooting process. Our goal is to go further and propose an approach for *diagnosing* them. By diagnosis, I mean the procedure of observing the *symptoms* of a potential anomalous and/or harmful behavior and provide a report stating its causes and effects. Network anomalies are often characterized by many different aspects: in order to understand them, a diagnosis system should monitor and correlate a number of traffic features. Usually, this is done by manual inspection of expert operators, resulting in a slow and error-prone procedure. Today we still miss a structured approach to speed-up a complete root cause analysis process. I tackle the detection and diagnosis of anomalies in Chapters 6 and 7, where I provide the design of a framework that addresses this gap by correlating multiple anomaly detector instances and using supervised machine learning techniques to automatically diagnose anomalies.

1.5. Thesis Outline

This thesis is organized in eight Chapters. Figure 1.2 depicts the distribution of the contents, research questions, and the interaction among them. In detail, the organization of this thesis goes as follows.

(Chapter 2) "Research Context, Tools and Methodologies" This Chapter describes some concepts for supporting the reader of this thesis. In particular it provides the research context (i.e., the mPlane project which is the framework for this work) and an overview of the tools and methodologies adopted.

(Chapter 3) "Traffic Classification Techniques" We introduce three novel traffic classification techniques based on hostname pattern-matching. This Chapter also shows practical examples of the long-term tracking capabilities of the classifiers. The techniques developed here are a fundamental building block for the analysis described in the rest of the thesis.

(Chapter 4) "Characterization of Traffic from Major Internet Services" This Chapter is devoted to a deep characterization of the traffic generated by the top hosting companies and Internet services. By using passive traces collected at the network of Major European operators and applying both novel and well-known methodologies in the field of TMA, we shed light on the complex traffic characteristics and dynamics that regulate the provisioning systems of popular applications.

(Chapter 5) "Large-scale Network Anomalies" While the previous Chapter was focusing on the *normal operations* of popular services and their provisioning systems, this Chapter is dedicated to the study of their anomalous behaviors. Here we study a number of anomaly case studies that impact remote services, access operators and end-users. These case studies provide the foundation for the design of anomaly detection and diagnosis algorithms.

(Chapter 6) "Advanced Anomaly Detection Techniques" Here we introduce a novel automatic detection and diagnosis framework for network anomalies. Using the lessons learned in previous Chapter, we compare and evaluate two detection algorithms using both real and semi-synthetic datasets. The main contributions of this Chapter are (i) a framework for structured and systematic anomaly detection and diagnosis, (ii) an evolution of a distribution-based detection algorithm, (iii) a comparison of the de-facto anomaly detection system to the proposed techniques, questioning its supremacy for anomaly detection, and (iv) a system for generating synthetic datasets that preserve real-traffic statistical characteristics.

(Chapter 7) "Towards Automatic Diagnosis of Anomalies" This Chapter describes the final module of the diagnosis framework previously introduced. This component relies on Machine Learning techniques to classify the detected anomaly, additionally correlating pattern deviations to automatically provide anomaly reports. We also present some possible future research directions aimed at improving the diagnosis in an iterative fashion.

(Chapter 8) "Conclusions" We draw here the final conclusions of the thesis.

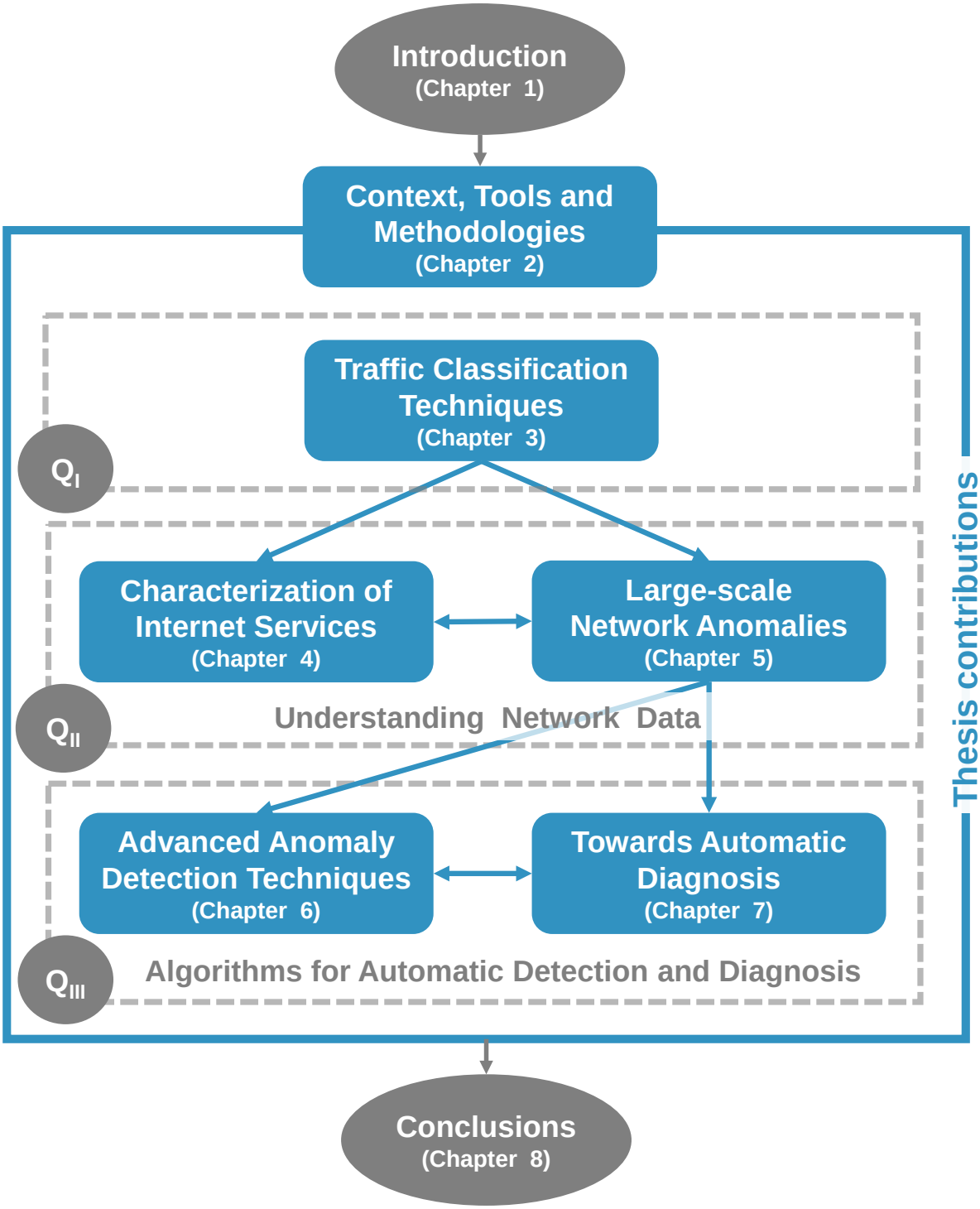


Figure 1.2.: Diagram of the thesis outline, showing Chapters and their interconnections.

2. Research Context, Tools and Methodologies

This Chapter gives some guide-lines to support the reader of this thesis. We start by introducing mPlane, the project that provided a framework for this work in Section 2.1. This work is, in fact, positioned in the field of Traffic Monitoring and Analysis (TMA), which is the common background of the project consortium members. Getting familiar with the project organization and goals clarifies the research context and the questions that this work tries to answer.

Section 2.2 provides an overview of the tools and the methodologies employed in this work. Some of them are novel and part of the contributions of this thesis.

2.1. A Measurement Plane for the Internet

The three years work that led to this thesis has been carried in the context of the European FP7 research project mPlane, from 2012 to 2015. mPlane has the goal of designing *"an Intelligent Measurement Plane for Future Network and Application Management"* [13]. mPlane proposes to advance the state of the art in the field of Internet measurements. The complex nature of Internet, in fact, still poses serious challenges when it comes to understanding traffic dynamics, mostly when something goes wrong.

The key idea is to embed measurements as a network capability, i.e., a *measurement plane*. It operates at a large range of scales and employs active, passive and hybrid probes in a distributed fashion. Given the amount of data produced by measurements, another core element is the design of intelligent analytics adopting on-the-edge big-data technologies and data mining techniques. The project defines the communication standards among the involved components (e.g., probes, repositories, etc.) also offering a reference implementation [15]. This architecture allows to deploy *intelligent reasoning algorithms* that iteratively drill down into the cause of anomalies, determining the conditions leading to given issues, and supporting the understanding of problem origins.

Illuminating the Internet dynamics with this approach brings benefits to all the involved actors. Network operators could get higher and more fine-grained visibility on their networks, adopting ad-hoc measurements and deploying Root Cause Analysis (RCA) algorithms. Content providers would have the instruments for detecting and assessing performance degradations and Quality of Experience (QoE) impairments. Finally, customers could enforce the application of Service Level Agreements (SLA). Most of these problems are currently addressed with ad-hoc manual analysis. In most cases they are time consuming, error prone and lack of generality. mPlane, and the development of a *reasoner* in particular, sets the

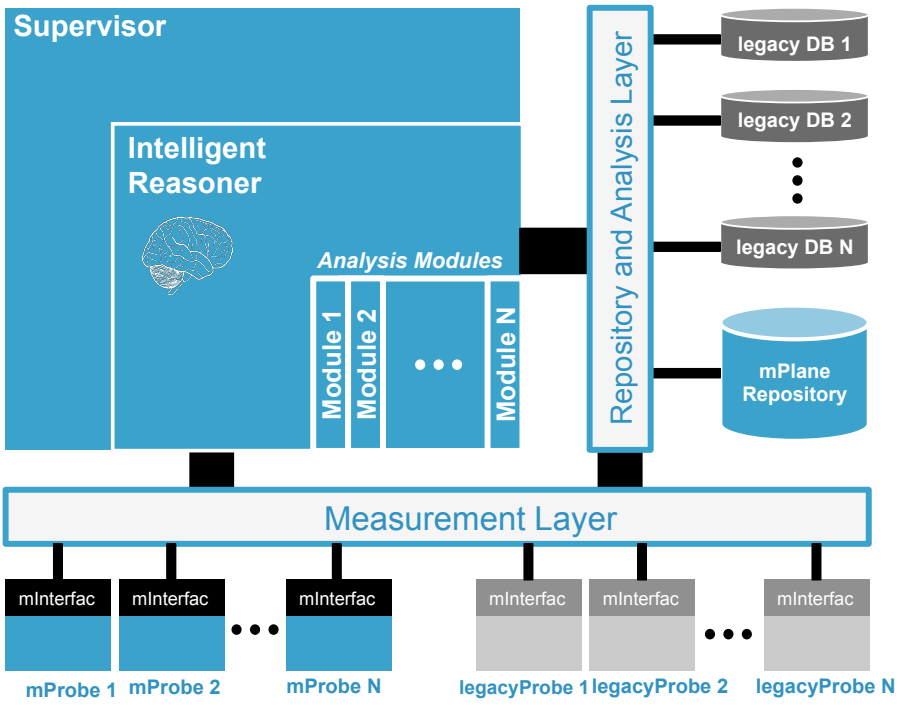


Figure 2.1.: The mPlane distributed measurements framework organized in three main layers. The Measurement Layer gives an interface to network probes. The Repository and Analysis Layer gives and interface to storages and basic analytics. Finally the Management Layer orchestrates the framework. Based on the Figure originally published in [13].

first mile stone for a structured, iterative and truly automatic approach to the problem of measuring the Internet.

2.1.1. Related Measurement Platforms

A distributed measurement platform similar to mPlane was originally presented in [16], where the authors envisioned an infrastructure, i.e., a *"network oracle"*, for the coordination of measurement and database technologies, aimed at supporting queries on the network status on a global scale and in real-time. In a similar direction, [17] proposes a pervasive measurement system that, in addition, is able to automatically troubleshoot network problems.

For what concerns existing operational platforms for coordinated active measurements, there is a number of projects worth mentioning. Archipelago [10], for instance, targets topology measurements through traceroute. Similarly, Routeviews [18] tackles real-time updates on global routing status. A more general measurement infrastructure is the one represented by RIPE Atlas [8], which allows to deploy large-scale ping, traceroute, DNS, and HTTP active measurements campaigns. RIPE Atlas has been used to produce some of the results presented in this thesis and has been integrated in mPlane. Similar to RIPE Atlas, PlanetLab [9] allows to set up active measurements with an higher degree of freedom,

but with a smaller number of available probes. TopHat [19] is an extension to PlanetLab that supports an efficient probe selection depending on the requested type of measurements. Finally, M-Lab [20] is a closed platform that supports performance estimations and makes some of the collected measurements publicly available.

These frameworks are valuable measurement instruments, that, in some cases, allow to access distributed computing resources. However, they miss the *intelligence* needed to extract valuable knowledge. In particular, while they provide the measurement infrastructure to collect network data from different Vantage Points (VPs), they do not easily allow to deploy analysis algorithms to make sense of it. They also do not foresee the integration with a storage layer for supporting data aggregation and historical analysis and only focus on active probing. In a nutshell, by themselves they are still far from the flexible monitoring plane envisioned in [16] and [17].

2.1.2. System Architecture

The architecture proposed by the project is organized in three main layers, the Measurement Layer, the Repository and Analysis Layer and the Management Layer, as depicted in Figure 2.1.

Measurement Layer This layer offers a flexible interface to passive (e.g., Tstat [21]), active (e.g., standard ping and traceroute tools) or hybrid probes. It consists both on new programmable mPlane probes and existing measurement technologies. For the latter, the project aims to write software proxies to make them mPlane-compatible, as it has been done for Tstat, for instance. The proxies allow mPlane components to trigger on-demand analysis or accessing results from existing measurement campaigns. An interesting example is the one represented by RIPE Atlas [8]. Atlas is an existing distributed active measurement platform which have been integrated in the mPlane greatly enriching the scale of available measurements.

Repository and Analysis Layer The measurements produced by the first layer are then collected in a standard way and pre-processed. Similarly to the measurement layer, this one is composed by both novel and existing storage and processing technologies (e.g., Hadoop, MongoDB, SQL databases). This layer is, indeed, not to be considered a mere storage for measurement results, but it includes analysis capabilities. In particular it produce online traffic aggregations to support the RCA process. DBStream, [22] a streaming data-warehouse developed in the context of mPlane, is a clear example of this. It is capable of importing measurements and run continuous query to produce on-line aggregations and materialized views. Most of the results produced for this thesis are actually based on custom DBStream jobs.

Management Layer This component is the key element of the mPlane framework. It controls probes and repositories and iterate on the obtained results to drill down on specific network issues. The distributed nature of the measurement and repository/analysis layers

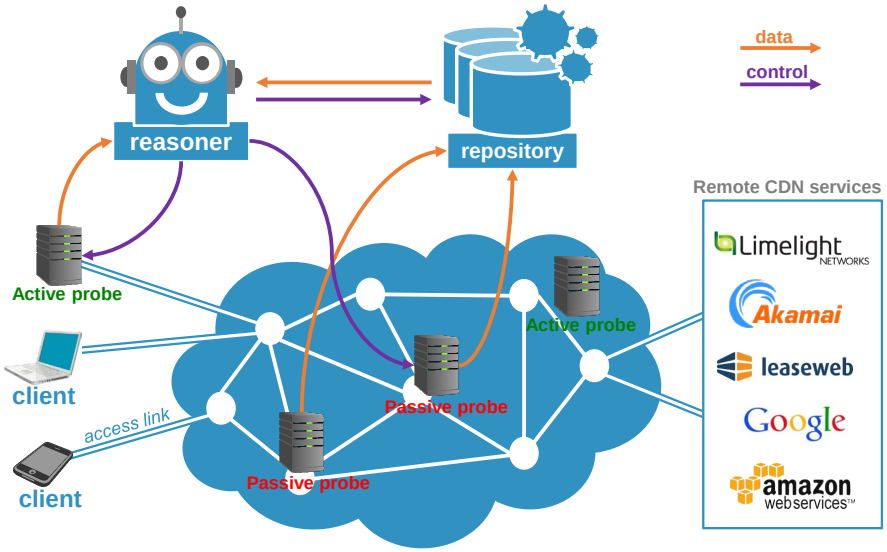


Figure 2.2.: Example of an intelligent reasoner for the automatic detection and diagnosis of network anomalies. The reasoner controls both active/passive probes and repositories through the mPlane management layer to iteratively drill down on the causes of network problems. Based on the Figure originally published in [13].

allows the supervisor to gain a large-scale visibility and mine correlations among network phenomenons. The iterative analysis is supported by an *intelligent reasoner* and specific analysis modules (e.g., anomaly detection modules). The results of Chapter 7 represent an example of a reasoner devoted to the diagnosis of network anomalies.

2.1.3. Anomaly Detection and Diagnosis in mPlane

The distributed mPlane architecture is designed to be capable of tackling many different use-cases and monitoring scenarios. In the course of the project, the consortium has proposed a list of sample case studies [23], designed so as to demonstrate the advantage of using an mPlane-like approach to measurements. These use cases include: estimation of content and service popularity for network optimization, web content promotion and curation, active measurements for multimedia content delivery, quality of experience for web browsing, verification of service-level agreements, anomaly detection and root cause analysis.

The results described in this thesis are part of the work done on the latter. The goal is to continuously monitor the network traffic on a large-scale level in order to not only detect, but also diagnose anomalous events that impact the involved stake-holders, from the remote services to the end-users. The focus is on particularly popular Internet applications such as social networks, video streaming services and instant messaging systems. The study of this services, however, imposes serious challenges due to their sophisticated and dynamic provisioning systems. Indeed, distributed Content Delivery Networks (CDNs) such as Amazon Web Services, Akamai, SoftLayer, Limelight, Google CDN, etc., have become the standard approach to cope with the huge and ever increasing demands of users in terms of reliability

and expected quality.

The problem of achieving the automatic diagnosis of anomalies in this complex scenario calls for the adoption of this distributed and structured measurement approach, such as the one mPlane proposes. More specifically, a supervisor, which is able to orchestrate ad-hoc measurements and trigger deeper analysis upon request, would offer all the instruments for designing an intelligent component, i.e., a *reasoner*, that allows an efficient Root Cause Analysis (RCA), such as the one sketched in Figure 2.2. An example of workflow for diagnosis reasoner is described in the following:

(i) passive and active monitoring: deployment of passive probes at several Vantage Points (VP). The gathered data allows to monitor large-scale services (if the probes are located at PoPs aggregating large populations of customers) and/or end-customer connections (if the probes are located at a specific server, for examples an FTP server used at target for throughput measurements). The latter case results in an hybrid measurement scheme as the passively collected information can be correlated with ad-hoc active measurements. In addition, deployment of active probes in an ISP network to periodically run latency (using the standard ping tool) and speed-test measurements (using a monitored FTP server as target, as explained above). All the collected measurements are fed to a smart repository, such as DBStream.

(ii) anomaly detection: run multiple instances of a specific analysis module for the detection of anomalies on different traffic features. An example of an anomaly detection module developed in mPlane is ADTool, which is part of the contributions of this thesis. If an alarm on a monitored traffic feature is raised, the Supervisor receives a notification.

(iii) correlation of multiple data-source and reactive monitoring: when the Supervisor is triggered by the previous step, it runs a *correlation analysis* to investigate which features are involved in the anomaly (i.e., present time-correlated changes). If needed, the Supervisor can also instantiate ad-hoc active measurements or retrieve existing passive measurements from different VPs to increase the visibility on the issue. When all the required information has been collected and correlated, it produces a signature of the anomaly to support the troubleshooting (e.g., it compares the signature against a catalog of known anomaly patterns).

As we will show, this thesis addresses these three points, among other contributions. In particular, (i) it firstly deals with the problem of understanding network data by deeply characterizing popular services and the anomalies they are affected by, focusing on passive measurements (cfr. Chapter 4 and 5). Thanks to this characterization, (ii) we also design analysis modules for the detection of anomalies (cfr. Chapter 6). Finally (iii) we design a linear diagnosis process, based on the analysis and correlation of passive measurements (cfr. Chapter 7). The *reactive monitoring* approach, as originally envisioned in [24], is still an open research topic. However, we believe that the preliminary results on the iterative diagnosis scheme presented in Chapter 7 provide an interesting starting point for future research.

2.2. Analysis Tools and Methodologies

In this Section we give an overview of the tools and methodologies that have been adopted to support this work, both well-known and novel. Where specified, they are part of the contributions of this thesis.

Large-scale Passive Measurements The results presented in this thesis are mostly based on the analysis of passive network traces, which have been collected at Vantage Points (VPs) located in operational networks. The great advantage of inference from passive measurements is that results reflect the actual usage of the network by real users. Furthermore, it allows to gain statistics at a large scale, potentially from millions of users, hence with a high degree of statistical significance. On the other end, it is usually difficult to have access of such datasets due to privacy and business related issues. In our studies, we had access to two VPs at the core of a fixed-line and a cellular network belonging to two Major European ISPs, respectively. For the capture of datasets, we relied on two distinct passive probes, both well-known in the research community: **Tstat** [7, 21] and **Metawin** [6]. Tstat (TCP STatistic and Analysis Tool) is a traffic sniffer developed by the Polytechnic University of Turin. The datasets captured by Tstat and used in this thesis correspond to a residential ADSL access line composed of thousands of customers in Europe collected in 2013. METAWIN (MEasurements and Traffic Analysis in WIreless Networks) is a probe developed by the Telecommunications Research Center of Vienna (FTW) specifically designed for cellular networks, capable of handling the complete 3GPP stack. For our studies, we relied on the traffic captured by METAWIN at the `Gn` interface of an operational Nation-wide HSPA network in Europe, during several capturing campaigns carried out in the years 2012-2014.

Active Measurements Some hosting infrastructure and service characterizations presented in Chapter 4 and Chapter 5 additionally rely on active measurements. Differently from passive techniques, the concept of active measurements consists in injecting traffic in the network. Popular tools such as ping or traceroute rely on the Internet Control Message Protocol (ICMP) protocol to check the reachability of remote hosts, calculate the Round Trip Time (RTT) and the path. In this thesis we use ping to observe the RTT from our location to the servers hosting the applications under study. This is particularly useful for estimating the *distance* of servers, discovering the hosting infrastructure and its temporal dynamics.

Geo-distributed Active Measurements One of the main problems of active measurements is that they provide a local view on traffic. Indeed, it is impossible to generalize the results obtained from a single measurement point. To overcome this limitations, we rely on RIPE Atlas [8]. Atlas is a large measurement network composed of geographically distributed probes used to measure Internet connectivity and reachability, maintained by RIPE NCC. The framework allows to set up distributed active measurement campaigns through the definition of User Defined Measurements (UDMs). In this thesis, it has been used to complement the analysis presented in Section 4.6 to verify the validity of the conclusions obtained by a single

VP on a world-wide scale. Furthermore, we present in Section 7.8 some preliminary results on an iterative diagnosis process exploiting this measurement scheme. In this work, we relied on an open source command-line interface to the framework, i.e., **Atlas Toolbox**, developed by the author of this thesis [25].

Hybrid Measurements Some Internet services use encrypted connections, therefore the first step to analyze their functioning in the wild is to better understand its inner working. To this end, we rely on the manual inspection of hybrid measurements, i.e., we actively generate traffic for the specific service at end terminals and, at the same time, we passively observe the traffic at the gateway. An example of this approach is presented in Section4.6, where we characterize WhatsApp. In that particular case, we focus on the capture of DNS requests to uncover the server naming scheme used by WhatsApp. For the live capture we relied on the popular tool **Wireshark** [5], an open source packet analyzer.

GeoIP Datasets To support the characterization of Internet services and content providers (cfr. Chapter 4) we relied on a GeoIP dataset provided my **MaxMind** [26]. Such datasets are periodically regenerated to offer and updated mapping between IPv4 addresses and their topological and geographical location, i.e., Autonomous System Number (ASN) a city-level estimation of their position. Given that the city-location accuracy of this kind of datasets is questionable [27], we usually cross check the validity of the location using active measurements, such as traceroute and ping. The localization of IP addresses is not only important for the characterization of Internet services' hosting infrastructures, but also for the study of large scale changes in the provisioning of popular applications, such as the ones presented in Section 5.3 and 5.6.

Stream Data-Warehouse All datasets used in this thesis have been analyzed using custom Perl scripts running on top of **DBStream** [22, 28]. DBStream is an open source streaming data-warehouse system developed by the Telecommunications Research Center of Vienna (FTW) in the context of the mPlane project. Its main goal is to process data from multiple sources as they are produced, create aggregations, and store query results for further processing by external analysis modules or visualization. The system is particularly useful in dealing with online network monitoring.

Traffic Classifiers The analysis based on the fixed-line datasets relies on the classification capabilities of Tstat, which we have used in particular for producing the results of Section 4.4. For what concern the METAWIN-based datasets captured in cellular networks, we classified web services using the pattern-matching based classifiers **HTTPTag** [P1] and **HTTPTag2** [P14], which are part of the contributions of this work and will be fully described in Chapter 3.

Distribution Distance Metric During this work, we resort on the comparison of traffic feature distributions, for example for building graphical tools (such as the TSP, presented below) and in our distribution-based detection algorithm (cfr. Section 6.6). We adopted the Kullback-Leibler divergence (**KLd**) as our metric of choice, as it has been proved to be

very effective in the field of traffic monitoring and analysis [29]. As the KLd is not strictly a distance metric per se, we use its entropy-normalized symmetric version, indicated later as **ENKLd**. We refer the reader to Section 6.6 for a more detailed description.

Temporal Similarity Plots (TSP) It is a powerful graphical tool, originally proposed in [29], that allows pointing out the presence of temporal patterns and (ir)regularities in distribution time-series by graphical inspection. In a nutshell, a TSP is a symmetrical heatmap-like plot, in which the value $\{i, j\}$ reflects how similar are the two distributions at time t_i and t_j. By construction, the TSP is symmetric around the 45° diagonal, and it can be interpreted either by columns or by rows. For example, if we read the TSP by rows, for every value j in the y-axis, the points to the left [right] of the diagonal represent the degree of similarity to past [future] distributions. In this thesis, we used the KLd distance metric for building the TPS. The TSP is particularly useful for graphically pointing out temporal patterns in the provisioning systems of Internet services and their changes of settings, as done in Section 5.3.2 and 4.5.5. We refer the interested reader to [29] for a detailed description of the TSP tool.

Entropy-based Detectors It is a popular and extensively studied approach for detecting anomalies in network traffic. It works by flagging abrupt changes in the time series of the empirical entropy of certain traffic features, which is often an evidence of the presence of an anomaly. The change detection is based on algorithms such as Exponential Weighted Moving Average (EWMA). Despite the popularity, we found its inapplicability in a number of scenarios. We refer the reader to Chapter 6 for the complete description and evaluation.

Distribution-based Detectors One of the goals of this thesis is to design and evaluate novel anomaly detection algorithms. We will later present a statistical detector based on the Kullback-Leibler divergence. In a nutshell, the non-parametric algorithm computes the degree of similarity between the empirical distribution of traffic features to a set of (anomaly-free) distributions in a dynamic "observation window", which describe the "normal" behavior. The algorithm is based on a previous work [29], but has been greatly improved and re-adapted as a contribution of this thesis. We refer to Chapter 6 for a description and evaluation of the algorithm. In Appendix A we provide an overview of **ADTool**, an implementation of the algorithm developed by the author of this thesis. ADTool has been used as one of the main components for the anomaly detection and diagnosis reasoner in the context of the mPlane project.

Quality of Experience (QoE) models The aftermath of network anomalies is, in some cases, the degradation of the end-user QoE. Assessing the effects of such degradations is a difficult task as it requires to map network statistics to human expectations. Towards this end, we rely on QoE models produced in controlled laboratory experiments. This experiments consist in simulating different network conditions and observing user reactions in presence of service impairments. By this, it is possible to build a model that estimate the QoE impacts of the anomalies observed passively on large-scale networks. These models encode acceptance thresholds in correspondence to different values of Key Performance Indicators (KPIs), such

as the down-link throughput. Some QoE considerations based on this approach are presented in Sections 4.4.4, 4.6.4, while in Sections 5.6, 6.9 we study a QoE anomaly by applying such models at a large-scale.

Social Data Analysis For the study of an outage of a popular Internet service in Section 5.5, we correlate the outage characteristics, such as the drop of traffic volume, with the reaction of users on Online Social Networks (OSN) such as Twitter (see an example in Section 5.5). Even if social network data are employed in a number of different analytics, from the best of our knowledge we are the first ones to correlate such information with network measurements to complement the study of anomalies. We believe that this is an interesting starting point for a possible research direction in the field of QoE, as one could resort to OSN as an alternative ground-truth for measuring the user satisfaction and/or user reactions in presence of service impairments.

Machine Learning Techniques The diagnosis process that will be described in Chapter 6 and 7 terminates with the generation of anomaly fingerprint which are in turn classified. We decided to build the classifier based on standard C4.5 decision tress, and compare its performance to that obtained through five standard supervised-learning-based approaches previously used in the literature: Multi-Layer Perceptron (MLP), Artificial Neural Networks (ANN), Naive Bayes (NB), Random Forest (RF), Support Vector Machines (SVM), and Locally-Weighted-based Learning (LWL). As for the tools, we rely on **Weka** (Waikato Environment for Knowledge Analysis), a popular open source suite of machine learning software [30].

2.3. Characterization Methodology at a Glimpse

The general approach of this work, both for the characterization of Internet services' provisioning systems and for the design of diagnostic algorithms, lies on study of the correlations of a number of traffic features. To this end, we rely on different kinds of measurements: mainly passive, supported by additional active and hybrid probing. The adopted methodologies and algorithms are, in some cases, well-established in the traffic monitoring and analysis community, having been built in over a decade of operational experience. In addition to those, we have designed novel techniques aimed at unveiling original perspectives on some traffic dynamics. Chapters 4 and 5 are rich of examples of advanced mining of traffic characteristics. When tackling the characterization of an Internet service, we use the tools and methodologies described above, resulting in a structured approach that can be sketched as follows:

(i) Understanding server naming scheme. It consists in manually checking the set of Fully Qualified Domain Names (FQDN) of a target service and consequently represent it in a concise regular expression. This is usually done by using hybrid measurements, as done in Section 4.6, for instance.

(ii) Classification. By using the *regex* previously built, we use our own classification tools (HTTPTag) to extract all the relevant flows or HTTP tickets from a passively captured

dataset. The classification is achieved by pattern matching of requested FQDN in DNS queries or in the GET field of HTTP tickets (when available). The complete procedure is described in Chapter 3.

(iii) Provisioning system discovery. We extract the list of remote server IP addresses used by a Web service that are visible in the passive datasets. Note that using large-scale network traces, we can have a quite general perspective of the IP list (limited, of course, to the geographical location of the VP). Using active measurements to obtain the RTT towards servers is a common way to disclose groups of close hosting locations and datacenters. In addition, MaxMind-like datasets allow to estimate their geographical (Nation-level) and topological (AS-level) location, which provides details on the hosting companies involved in a service provisioning system. This information could be used by ISPs to understand traffic sources and consequently enforce efficient network planning decisions and peering agreements with their neighbors.

(iv) Flow characteristics. It consists in studying, for instance, the flow duration, size and throughput. This gives an indication of the network footprint of a service. Furthermore, a bleeding edge research topic is the correlation of these characteristics with QoE models to assess the quality level perceived by end users. Indeed, the "human factor" is quickly becoming a pillar in the field of network measurements. It is important to take QoE-models into account in assessing the impacts anomalies involving performance degradations (cfr. study of QoE impairments in Section 5.6). The QoE models are built in controlled lab experiments and then applied in the wild to check specific Key Performance Indicator (KPI) changes could result in degradation of quality.

(v) Unveiling temporal dynamics. For this, we employ novel approaches to study the evolution of the hosting infrastructures of Internet services over a number of different time scales (e.g., daily, weekly, monthly, longer-term). We use minRTT heatmaps and TSPs to graphically inspect the server dynamics and to discover the presence of peculiar events. This is a fundamental step, as the design of diagnostic algorithms has to take into account the traffic dynamics and variability to be robust against them and, most importantly, being able to distinguish *"physiological"* from anomalous changes.

(vi) Comparing different access technology patterns. This step consists in the comparison of the above mentioned characteristics as seen in Vantage Points located in different networks. Unfortunately, such a comparison is in general hard to achieve as it requires the access to parallel measurement campaigns. In this thesis, we had the chance to compare YouTube traffic as seen in a Nation-wide cellular and in a large-scale fixed line DSL network. This study is presented in Section 4.4 and highlights interesting peculiar characteristics of the two caching approaches used by the popular video streaming service YouTube.

As for the second part of this work, i.e., the design of a system for automatic detection and diagnosis of network anomalies, the correlation of different traffic features is the core element. Literature is rich of systems for detecting the presence of anomalies in specific traffic features. However, both in production and research environments, the common approach is to manually mine the presence of correlations to study an anomaly. Our goal is to provide a structured view of both existing and new mining techniques to get closer to the automation of these procedures.

3. Traffic Classification Techniques

Notice of adoption from previous publications

Parts of the contents of this Chapter have been published in the following papers:

[P1] **P. Fiadino**, A. Bär, P. Casas, "HTTPTag: A Flexible On-line HTTP Classification System for Operational 3G Networks", in *IEEE INFOCOM Poster/Demo Session*, 2013.

[P2] P. Casas, **P. Fiadino**, "Mini-IPC: A Minimalist Approach for HTTP Traffic Classification using IP Addresses", in *Wireless Communications and Mobile Computing Conference (IWCMC)*, 2013.

[P3] P. Casas, **P. Fiadino**, A. Bär, "IP Mining: Extracting Knowledge from the Dynamics of the Internet Addressing Space", in *The 25th International Teletraffic Congress (ITC2013)*, 2013

[P13] **P. Fiadino**, M. Schiavone, P. Casas, "Vivisecting WhatsApp through Large-Scale Measurements in Mobile Networks", extended abstract in *ACM SIGCOMM Poster/Demo Session*, 2014.

[P14] **P. Fiadino**, M. Schiavone, P. Casas, "Vivisecting WhatsApp in Cellular Networks: Servers, Flows, and Quality of Experience", *TMA*, 2015.

The author of this thesis provided major contribution to the design of the classification techniques (HTTPTag, HTTPTag2 and Mini-IPC) and the results obtained by using them. The implementation and integration with DBStream, the data-warehouse system in use, has been supported by the co-authors, in particular Arian Bär, the main developer of DBStream. The work has been supervised by Dr. Pedro Casas.

3.1. Introduction

HTTP is doubtlessly the dominating content delivery protocol in today's Internet. The popularity of services running on top of HTTP (e.g., video and audio streaming, social networking, on-line gaming, etc.) makes that more than 75% of today's residential customers traffic is accountable to HTTP [31, 32]. In addition, a big share of Internet ecosystem is shaped by the success and influence of the most popular web services: YouTube, Netflix, Facebook, and even Dropbox 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, replicated, addressed, and served. In this scenario, understanding HTTP traffic composition, usage patterns, and content location and distribution is highly valuable for network operators, with application in multiple areas such as network planning and optimization (e.g., content caching), traffic engineering (e.g., traffic differentiation/priorization), marketing analysis (e.g., heavy-hitter applications), just to name a few of them.

For instance, lets consider specific classes of services which are particularly critical for operators, such as automatically updated services or similar scheduled activities. Their synchronized behavior imposes serious challenges as the steep increase of traffic concentrated

in short periods of time could lead to an high consumption of the available resources. Identifying such services, and consequently understanding their traffic patterns, could help the operator to optimize and dimension network resources by forecasting traffic peaks. Another interesting example is the one represented by Video Streaming services, due to the large amount of traffic volume they generate. Understanding video traffic composition can help in deploying content caching servers and optimize internal routing to minimize costs and maximize the performance.

This Chapter addresses the classification of applications running on top of HTTP by presenting three novel techniques relying, respectively, on hostname pattern matching, DNS requests and IP addresses. Beyond its intrinsic importance for network operators, HTTP classification is clearly the staging post for understanding large-scale traffic dynamics and addressing their anomalies. As such, the tools resulting from the studies illustrated in this Chapter constitute the first and fundamental step for this thesis.

3.2. Related Work and Contributions

The field of automatic Internet Traffic Classification (TC) and analysis has been extensively studied during the last decade [33, 34]. Standard classification approaches rely on Deep Packet Inspection (DPI) techniques, using pattern matching and statistical traffic analysis [7, 35, 36, 37]

Probably the most popular approach for TC exploited in recent years by the research community is the application of Machine Learning (ML) techniques [38]. A standard non-exhaustive list of supervised ML-based approaches includes the use of Bayesian classifiers [39], linear discriminant analysis and k-nearest-neighbors [40], decision trees and feature selection techniques [41], and support vector machines [42]. Unsupervised and semi-supervised learning techniques have also been applied for traffic analysis and classification [43].

Work on TC for specific network management requirements, such as fast TC [44] and TC based on sampled traffic [45] has also been part of the long list of studies conducted in the past.

The shift of applications towards Web-based traffic in today's Internet has recently started a new wave in the study and development of TC techniques. In the specific case of Web and HTTP traffic, classification and analysis has been the focus of many studies [46, 31, 47, 48, 49, 50]. Authors in [46] study the composition and characteristics of modern Web traffic, using a large dataset spanning close to 200 countries worldwide. In [50], authors use payload-based analysis heuristics to classify 14 different HTTP classes. In [31, 48], 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 [47] 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 [49] 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.

network type	cellular (2G/3G)
monitoring system	METAWIN at Gn interface
ticket type	HTTP tickets
length	$\sim$ 6 months
time	Q4 2011 - Q2 2012

Table 3.1.: Dataset used for long-term tracking with HTTPTag

Finally, the large scale adoption of end-to-end encryption over HTTP has motivated the development of novel technique to classify applications distributed on top of HTTPS traffic [51, 52], relying on the analysis of DNS requests.

In this Chapter, we present three novel classification systems specifically designed, deployed and tested in cellular networks. From the best of our knowledge, we are the first to provide a description of a working traffic classification algorithm specifically designed and tested on an operational cellular network. Moreover, we avoid relying on standard DPI techniques, both for ethical and technical reasons.

3.3. System Architecture and Datasets

The long-term application tracking showed in this Chapter are based on the analysis of network traces passively captured at the core of an operational cellular network in Europe over 6 months between 2011 and 2012. The dataset characteristics are summarized in Table 3.1. For capturing and filtering the traffic, we rely on the METAWIN passive monitoring system [6]. Packets are captured on the Gn interface links between the GGSN and SGSN nodes at the core of a Nation-wide cellular network. The monitoring system produces flow-level traces and application-level tickets, such as HTTP and DNS. A ticket is the summary of a transaction: in the case of HTTP, it contains information related to the HTTP connection (e.g., timestamp, requested hostname, transferred up-/down-link volume, error codes, user-agent, etc.), while a DNS ticket contain the queries issued by users, the set of answers, the resolver IP addresses and a status code (e.g., query successful, timeout, etc.). To preserve user privacy, any user related data (e.g., IMSI, MSISDN) are removed on-the-fly, and payload content beyond HTTP headers is ignored.

These network traces and tickets are continuously fed to a streaming data warehouse system called DBStream [22, 53]. DBStream is a middle-ware layer on top of PostgreSQL specifically tailored to import, store and process data streams. It allows at the same time to run live analysis on imported data and execute long-term historical studies. The overall system architecture is depicted in Figure 3.1.

3.4. Hostname-based Classification (HTTPTag)

In this Section we show HTTPTag, described in our publication [P1], a flexible on-line traffic classification system for analyzing applications running on top of HTTP. Similar to

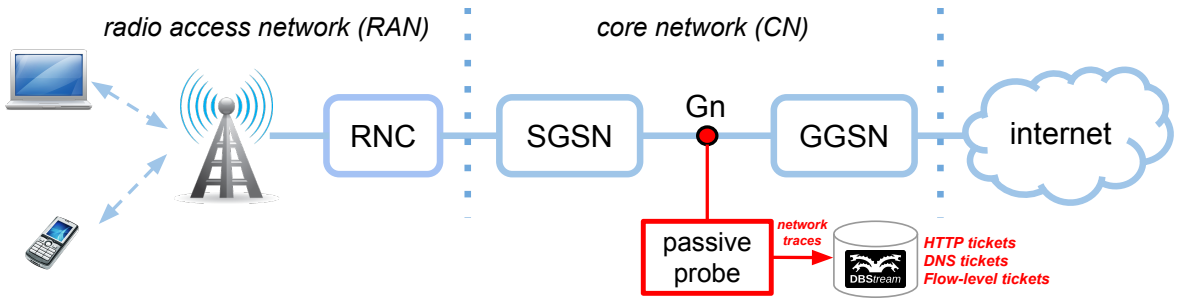


Figure 3.1.: Vantage Point (VP) deployment in an operational Mobile Network. A passive probe (METAWIN) collects network traces at the Gn interface and stores them in a stream data-warehouse (DBStream).

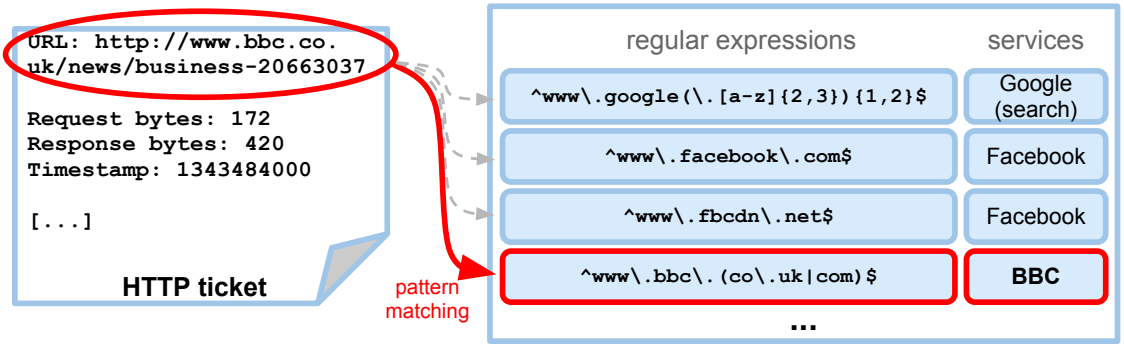
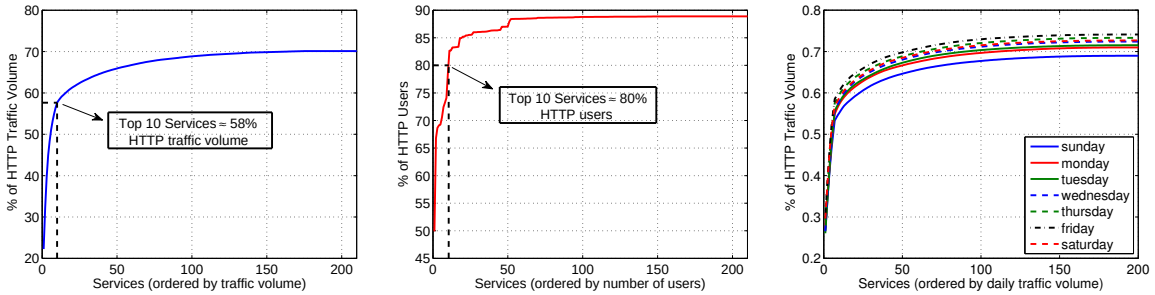


Figure 3.2.: Matching URLs and Hostnames with patterns and services. The regular expressions are ordered by probability of occurrence to improve the pattern matching speed.

[47, 48], HTTPTag focuses exclusively on HTTP traffic analysis. The approach adopted for the HTTP classification is based on *tagging*, i.e. associating a set of labels or *tags* to each observed HTTP request, 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 request's header. HTTPTag currently recognizes and tracks the evolution of more than 280 services and applications running on top of HTTP, including for example tags like `YouTube`, `Facebook`, `Twitter`, `Zynga`, `Gmail`, etc. Due to the highly concentrated traffic volume on a small number of heavy hitter applications, the current list of services spans more than 70% of the total HTTP traffic in the 3G network of a leading European provider.

3.4.1. HTTPTag Overview

HTTPTag works with HTTP tickets collected in DBStream, as previously explained. HTTP tickets are detected and analyzed on the fly: every new HTTP transaction is parsed and the contacted `hostname` (extracted from the URL) is compared against the defined regular expressions or *patterns*, see Figure 3.2. If a matching pattern is found, the transaction is



(a) HTTP traffic volume per service.

(b) Unique HTTP users per service.

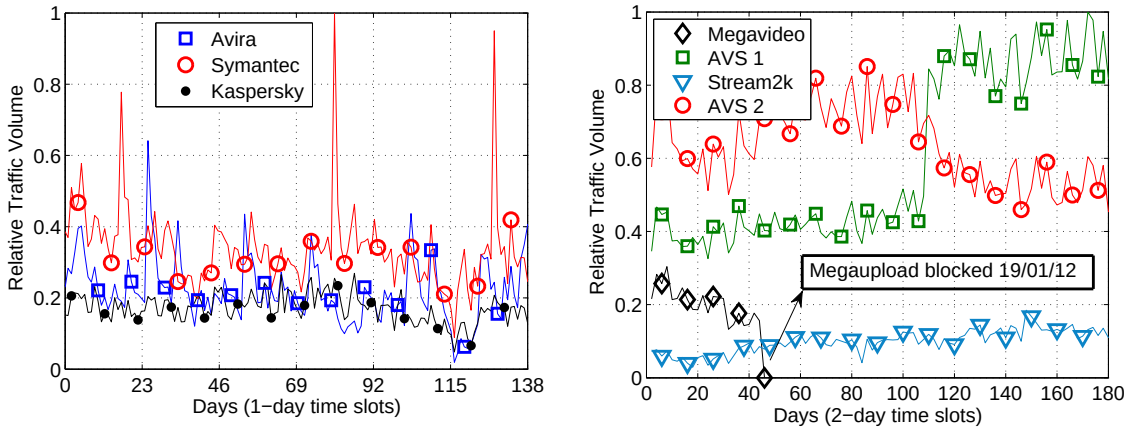
(c) Daily HTTP traffic volume per service.

Figure 3.3.: 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.

assigned to the corresponding service. 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 [54], and limiting the study to the most popular services already captures the majority of the traffic volume/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. We have discovered that an initial effort in classifying the top 50 sites combined with weekly updates ensures a high classification rate. HTTPTag provides a GUI-based exploring system to identify the top `hostnames` responsible for the largest non-classified traffic volume and number of visitors, easing the tagging of new services.

Figures 3.3(a) and 3.3(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 dominates 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, mostly overlapping.



(a) 5-months anti-virus services tracking.

(b) 6-months video services tracking.

Figure 3.4.: HTTPTag classification coverage and some long-term tracking examples revealing different events of interest in an operational 3G network.

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.

To verify the stability of these results over time, Figure 3.3(c) shows the total daily HTTP volume labeled by HTTPTag during a week of traces. The week corresponds to the first 7 days of April 2012, from Sunday the 1st till Saturday the 7th. HTTPTag is able to label between 69% and 74% of the total daily HTTP volume on the studied traces, depending of the different network usage patterns during weekends and working days. While the set of heavy hitter applications is stable over time, in fact, the distribution are slightly less skewed on Sunday inducing a small decrease in the classification rate.

3.4.2. Long-term results

We now show two examples of long-term tracking capabilities of the HTTPTag classification system applied in an operational network. To this end, we chose two popular types of applications, namely Antivirus Update and Video Streaming services. These classes are particularly relevant for ISPs due to, respectively, the potential issues caused by the synchronized behavior of terminals and the large traffic volume that they generate. Figure 3.4 depicts the long-term tracking capabilities of HTTPTag in these two case studies.

Antivirus Update Services In Figure 3.4(a) we track the traffic generated by three popular antivirus services (Symantec, Kaspersky, and Avira) over a four months period (from the 26/05/12 to 15/10/12). Analyzing the traffic patterns on a sufficiently long period gives a good image on the different approaches the three companies use to manage software and virus-definition updates. While `Kaspersky` shows a quite constant behavior, both `Symantec` and `Avira` present important peak volumes on specific update periods, which might heavily

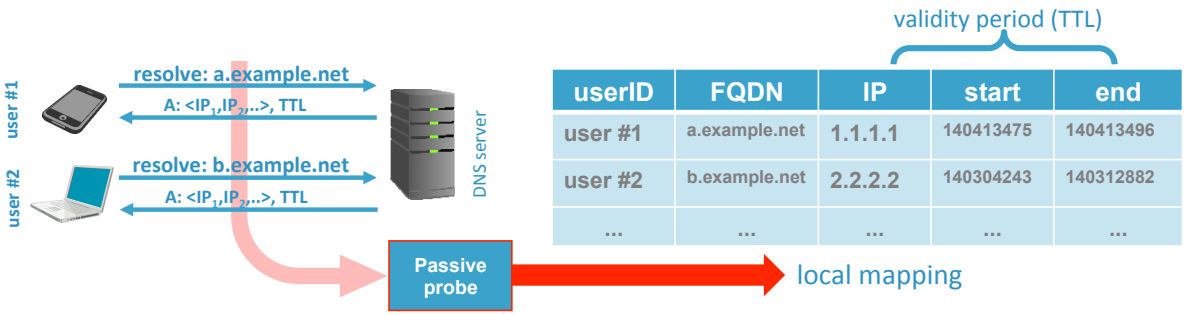


Figure 3.5.: Temporary IP-hostname mapping through DNS monitoring. HTTPTag's hostname-based approach is then applied on the hostname column.

load the network. This information could be directly used by the network operator to define routing, load balancing, or prioritization/shaping policies.

Video Streaming Services Figure 3.4(b) depicts a comparison of four video streaming services on a 6-months period (from the 1/12/11 to the 25/05/12): Megavideo, Stream2k, and two adult video services (AVS 1 and 2). After 46 days from the starting tracking day, `Megavideo` traffic completely disappears, which correlates to the well-known shut-down of the Megaupload services on the 19/01/12. Part of the video streaming volume provided by `Megavideo` was taken by a direct competitor, `Stream2k`, which shows a slow yet constant growth on the following months. Finally, we observe a drastic shift in the consumed volume from the two AVS services after 3 months and a half of steady traffic. We do not have a direct answer for this shift, but a change in the charging policy to access the content (e.g., free to subscription-based access) could explain such a variation. Having a complete picture of these popularity/usage modifications gives the operator the chance to better react to them (e.g., by defining specific content caching policies to reduce the load on the core links).

3.4.3. Leveraging DNS for HTTPS classification (HTTPTag2)

As seen, HTTPTag is capable of achieving a good classification rate, with more than 70% of the overall HTTP traffic volume caused by more than 88% of the web users in the traces previously studied. However it does not recognize HTTPS traffic: its prerequisite, in fact, is the availability of the requested hostname, which is encrypted in HTTPS. Recent studies show that the overall share of HTTPS is constantly increasing as popular web services are relying on it by default: it is estimated that HTTPS accounts for 25% of downlink and 80% of uplink traffic volume [55]. On the long term, this could lead to an overall decrease of the classified traffic volumes. To overcome this limitation, HTTPTag has been extended in order to leverage the analysis of DNS query for the classification of HTTPS traffic.

Our approach is depicted in Figure 3.5. Every time a user issues a DNS request for a Fully Qualified Domain Name (FQDN) `example.net`, HTTPTag creates an entry mapping this user to the server IPs provided in the DNS reply. Each entry is time stamped and contains the TTL replied by the DNS server. Using these mappings, all the subsequent flows between

this user and this remote IP are assumed to be flows related to `abc.example.net`. To avoid miss-classifications due to out-of-date mappings, every entry expires after a TTL-based time-out. To increase the robustness of the approach, the list of IPs is augmented by adding the list of server IPs signing the TLS/SSL certificates with the string `*.example.net`. Finally, HTTPTag is applied on the obtained local mapping to classify the associated services by means of the usual pattern-matching.

Note that, in this way, the pattern-matching scheme can be used to assign flows to a specific service not limiting the scope to HTTPS, but also, by extension to every application layer protocol that does not allow to easily extract the contacted hostname, because of encryption or by design. In the course of this thesis, we will extensively use this classification scheme to analyze different types of applications (cfr. XMPP-like chat service characterization in Section 4.6.

HTTPTag2 approach is similar to [51], however we also leverage the pattern matching feature to also map flows to services, not limiting the classification to FQDNs. Furthermore, HTTPTag has been designed as a module in DBStream, hence relying on the stream processing capabilities of the system for on-line classification.

3.5. A light-weight IP-based approach (Mini-IPC)

We now explore the possibility to exploit the remote service IP addresses as the only traffic feature for the HTTP classification. To this extent, we propose a light-weight classifier, Mini-IPC [P2], which is suitable for such HTTP services whose hosting infrastructure is particularly stable over time, or in case of CDN-based distribution, use well-known IP ranges with slow dynamics. Mini-IPC uses as *learning* input data the results provided by HTTPTag, described in the previous section.

3.5.1. Mini-IPC Overview

Mini-IPC classifies HTTP flows based solely on the IP address of the server being contacted. In a nutshell, given a specific service S_i to be identified, Mini-IPC builds 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 a downstream HTTP flow f_{new} coming from IP address ip_{new}, Mini-IPC applies the following classification rule: $\mathcal{F}(f_{new}) = S_i \leftrightarrow ip_{new} \in IP_i$.

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 datacenters (e.g., Google), and the ISPs content caching policies, multiple different services S_i might be associated to the same 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*. 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, Mini-IPC currently uses a 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). Please note that improving the classification process is out of scope. Our primary goal is to explore the applicability of this classification approach that, as we will see, is not suitable for nowadays CDN services.

3.5.2. Mini-IPC Evaluation

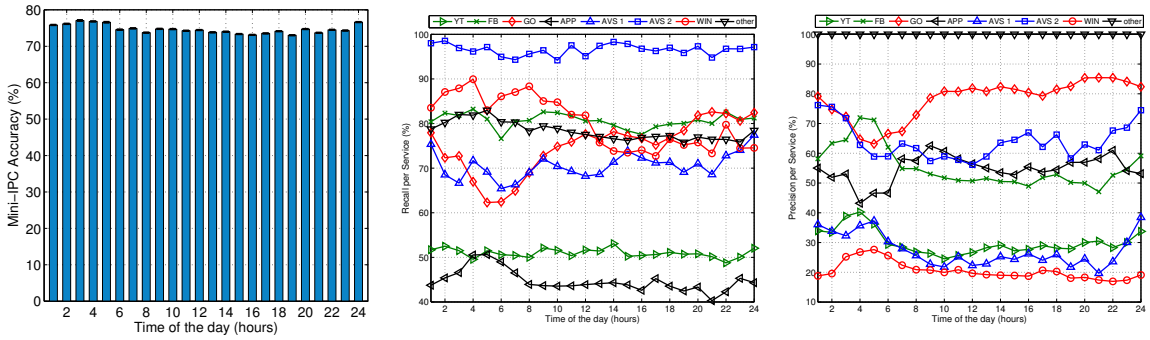
To test the classification performance achieved by Mini-IPC, we focus the attention of the top-7 services (in terms of traffic volume) depicted in Figure 3.3. These top-7 services are responsible for almost 60% of the total daily HTTP volume during the whole duration of the dataset (i.e., one week), which represents about 85% of the labeled services in terms of traffic volume. The ordered list of services in terms of volume includes `YouTube` (YT), `Facebook` (FB), `Google` (i.e., Google Search - GO), `Apple` (i.e., App Store and iTunes - APP), two well-known Adult Video Streaming services `AVS 1` and `AVS 2`, `Microsoft Windows Update` – `WIN`.This volume-based ordering corresponds to the traffic of Monday, but remains stable enough during the evaluated week. we divide the complete week of labeled HTTP flows in $n = 8$ services or *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. 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:

$$\mathsf{CA} = \frac{\sum_{i=1}^{m} TP_i}{n}, \quad \mathsf{R}_i = \frac{TP_i}{TP_i + FN_i}, \quad \mathsf{P}_i = \frac{TP_i}{TP_i + FP_i} \tag{3.1}$$

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 3.6 depicts the classification performance achieved in the learning day (i.e., Monday), on an hourly basis, using the output of HTTPTag as ground-truth. 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 3.6(a) depicts the classification accuracy for the 8 defined classes. The error variance bounds resulting from the 20 consecutive runs are negligible and not visible in the Figure. The overall classification accuracy seems remarkably high and stable during the day, rounding about 75%



(a) Classification Accuracy. (b) Recall per Service. (c) Precision per Service.

Figure 3.6.: 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.

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 3.7 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. In all cases, many flows are misclassified as belonging to the `other` class.

Let's focus now on the per service recall and precision, depicted in Figures 3.6(b) and 3.6(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 around 50%. The main reason for this poor results come directly from the IP hosting collisions associated to the Google CDN, as many of the YouTube flows are classified as Google Search according to Figure 3.7. 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 a complete week of traffic traces, using the IPs of Monday as learning data. Figure 3.8 depicts the per-day accuracy, recall and precision achieved in over 7 days. Figure 3.8(a) shows that the classification accuracy is remarkably stable during the full week, clearly suggesting that the

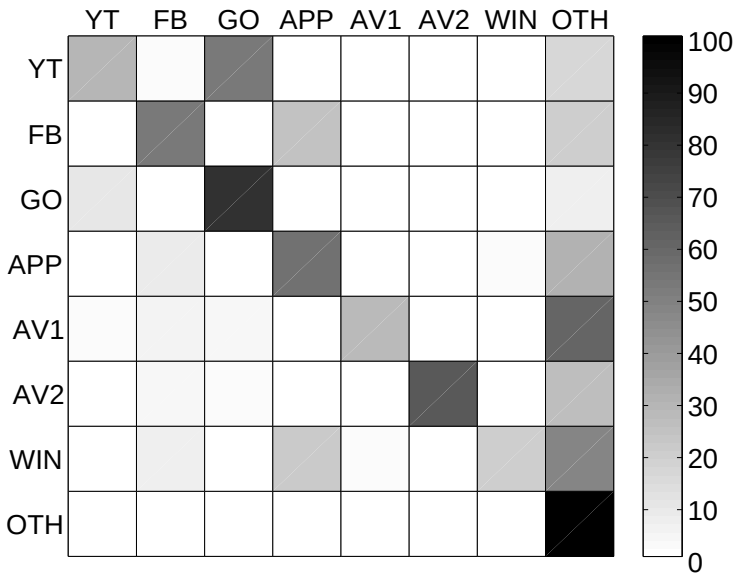


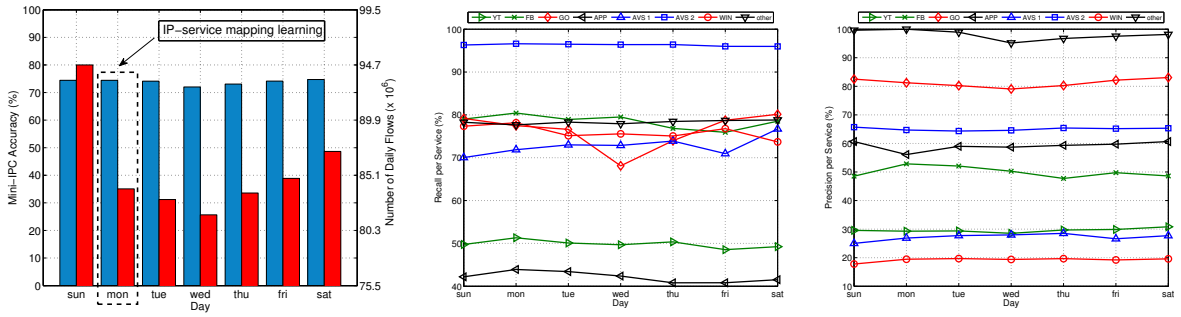
Figure 3.7.: Confusion matrix for traffic classification. Many YouTube flows are classified as Google Search. Windows Update flows are misclassified as Facebook and Apple, given the previously mentioned IP hosting collisions within Akamai. In all cases, many flows are misclassified as belonging to the `other` class.

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 3.8(b) and 3.8(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.

As a final consideration, we have seen that, despite its simplicity, this minimalist approach is able to classify the HTTP flows associated to the top-services with a rather decent accuracy of 75%. However, we have also seen that the classification recall and precision are highly impacted by IP hosting collisions, seriously impacting the performance of Mini-IPC as a robust traffic classifier. Still, results obtained for some of the analyzed services were encouraging, achieving a daily per-class accuracy above 70%. Mini-IPC could be a practical and very flexible solution for traffic aware networking, tackling those services with a stable or slowly changing hosting. As we will see in the next Chapter, some large-scale services with big user base (e.g. in the order of millions of users) are still characterized by simple infrastructures, as in the case of WhatsApp (cfr. Section 4.6).

3.6. Summary

In this Chapter we have addressed the problem of HTTP traffic classification from network measurements, exploring a hostname-based approach (HTTPTag, in Section 3.4). From this, we have proposed a variant that relies on the analysis of DNS traffic passively captured to



(a) Daily Classification Accuracy (blue) and num Flows (red).

(b) Daily Recall per Service.

(c) Daily Precision per Service.

Figure 3.8.: 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.

extend the classifier scope to HTTPS, as well (HTTPTag2, in Section 3.4.3). The use of encryption, in fact, makes it impossible to rely on the hostname as the sole traffic feature used for the classification, without leveraging additional information. To this extent, we proposed another pattern-matching classification scheme that also leverages DNS queries to achieve the classification of encrypted traffic). The latter approach is the basis of most of the analysis illustrated in the remainder of this thesis.

This Chapter also investigated the possibility of relying on a simple and lightweight classification approach (Mini-IPC, in Section 3.5). The evaluation of Mini-IPC demonstrated that it is sometime tricky to simply relying on IP addresses in the field of TC. Nevertheless, studying its limitations helped to start shedding light on the complexity of large-scale services hosting infrastructure and the dynamics of their addressing space, which is the main goal of the following Chapter.

4. Characterization of Traffic from Major Internet Services

Notice of adoption from previous publications

Parts of the contents of this Chapter have been published in the following papers:

[P2] P. Casas, **P. Fiadino**, "Mini-IPC: A Minimalist Approach for HTTP Traffic Classification using IP Addresses", in *Wireless Communications and Mobile Computing Conference (IWCMC)*, 2013

[P3] P. Casas, **P. Fiadino**, A. Bär, "IP Mining: Extracting Knowledge from the Dynamics of the Internet Addressing Space", in *The 25th International Teletraffic Congress (ITC2013)*, 2013

[P4] P. Casas, **P. Fiadino**, A. Bär, "Understanding HTTP Traffic and CDN Behavior from the Eyes of a Mobile ISP", in *Passive and Active Measurements Conference / Poster session (PAM2014)*, 2014

[P5] **P. Fiadino**, A. D'Alconzo, P. Casas, "Characterizing Web Services Provisioning via CDNs: The Case of Facebook", in *The 5th International Workshop on Traffic Analysis and Characterization (TRAC2014)*, 2014.

[P6] A. D'Alconzo, P. Casas, **P. Fiadino**, A. Bär, A. Finamore, "Who to Blame when YouTube is not Working? Detecting Anomalies in CDN-Provisioned Services", in *The 5th International Workshop on Traffic Analysis and Characterization (TRAC2014)*, 2014.

[P7] P. Casas, **P. Fiadino**, A. Bär, A. D'Alconzo, A. Finamore, M. Mellia, "YouTube All Around: Characterizing YouTube from Mobile and Fixed-line Network Vantage Points", *The European Conference on Networks and Communications (EuCNC2014)*, Bologna, Italy, 2014.

[P8] P. Casas, A. D'Alconzo, **P. Fiadino**, A. Bär, A. Finamore, "On the Analysis of QoE-based Performance Degradation in YouTube Traffic", in *10th International Conference on Network and Service Management (CNSM2014)*, 2014.

[P9] P. Casas,**P. Fiadino**, A. Sackl, A. D'Alconzo, "YouTube in the Move: Understanding the Performance of YouTube in Cellular Networks", in *Wireless Days 2014 Conference (WD2014)*, Rio de Janeiro, Brazil, 2014.

[P10] P. Casas, A. D'Alconzo, **P. Fiadino**, A. Bär, A. Finamore, T. Zseby, "When YouTube doesn't Work – Analysis of QoE-relevant Degradation in Google CDN Traffic", in *IEEE Transactions on Network and Service Management*, vol. 11, no. 4, 2014.

[P11] **P. Fiadino**, A. D'Alconzo, A. Bär, A. Finamore, and P. Casas, "On the Detection of Network Traffic Anomalies in Content Delivery Network Services", in *Teletraffic Congress (ITC), 2014 26th International*, 2014.

[P12] M. Schiavone, P. Romirer-Maierhofer, **P. Fiadino**, P. Casas, "Diagnosing Device-Specific Anomalies in Cellular Networks", in *ACM CoNEXT Student Workshop*, 2014.

[P13] **P. Fiadino**, M. Schiavone, P. Casas, "Vivisecting WhatsApp through Large-Scale Measurements in Mobile Networks", extended abstract in *ACM SIGCOMM*, 2014.

[P14] **P. Fiadino**, M. Schiavone, P. Casas, "Vivisecting WhatsApp in Cellular Networks: Servers, Flows, and Quality of Experience", *in Traffic Monitoring and Analysis (TMA 2015)*, 2015.

[P15] **P. Fiadino**, P. Casas, M. Schiavone, A. D'Alconzo, "Online Social Networks Anatomy: on the Analysis of Facebook and WhatsApp in Cellular Networks", *in IFIP Networking 2015*, 2015.

[P16] P. Casas, **P. Fiadino**, M. Schiavone, "QoMOSN - On the Analysis of Traffic and Quality of Experience in Mobile Online Social Networks", in *European Conference on Networks and Communications (EuCNC 2015)*, 2015.

[P17] P. Casas, M. Varela, **P. Fiadino**, M. Schiavone, H. Rivas, R. Schatz, "On the Analysis of QoE in Cellular Networks: from Subjective Tests to Large-scale Traffic Measurements", *in International Wireless Communications & Mobile Computing Conference - TRAC (IWCMC 2015)*, 2015.

The author of this thesis provided major contribution to the analysis of network traces that produce the results. Arian Bär and Mirko Schiavone supported the interaction with the streaming data-warehouse system DBStream. Dr Pedro Casas supported the QoE-based studies. The work has been supervised by Dr. Pedro Casas, Dr. Alessandro D'Alconzo and Prof. Tanja Zseby.

4.1. Introduction

A decade ago, Internet traffic was largely dominated by P2P file sharing, while 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: in the previous Chapter we have seen that HTTP and HTTPS are the leading application layer protocols —with increasing traffic share— as popular services run on top of them. The hosting scheme of Web services, especially for what concerns the *big players* (i.e., organizations and/or services responsible for a large share of the overall Internet traffic volume and/or number of users), has also greatly evolved: 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. 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 the dynamics behind such an approach 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.

The limitations of a simplistic IP-based traffic classification scheme (cfr. Section 3.5) have already given a glimpse of this complex scenario. In this Chapter we attempt to further unravel the complexity behind the addressing dynamics of the top Internet services running on HTTP. By using traffic traces passively collected at both cellular and fixed-line operational networks of major European ISPs, we study the associations between services, hosting organizations, and IPs assigned to the servers providing the contents. 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 end-users. The deep traffic characterization presented in this Chapter is, however, not limited to the study of passive network traces alone. As we will show, we augmented the information collected at our passive VPs with geo-IP datasets, such as MaxMind [26], which help to better understand the hosting infrastructure from both topological and geographical perspectives.

The extracted knowledge is not only useful for understanding the common issues associated to complex services, but is also vital to conceive a diagnosis solution for detecting and providing insights on impairments and quality degradations, as we shall see in next Chapters.

The remainder of this Chapter tackles the characterization of Web traffic from two different perspectives: Section 4.3 offers an overview of the major players dominating the Internet traffic, both from the hosting and service point of view. Sections 4.4, 4.5, and 4.6 focuses instead on specific services - namely YouTube, Facebook, and WhatsApp, chosen for their popularity and hosting infrastructure peculiarities.

4.2. Related Work and Contributions

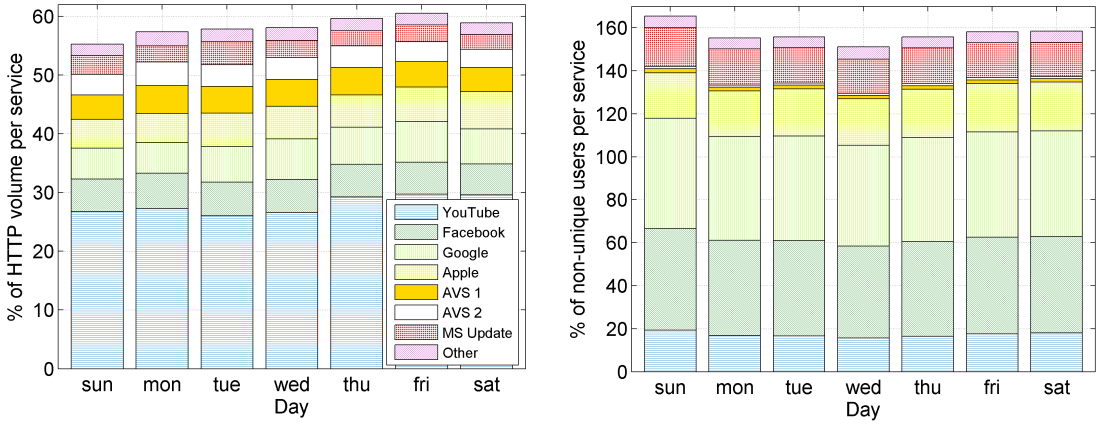
The study and characterization of the Internet traffic hosted and delivered by the top content providers has gained important momentum in the last few years [56, 54, 57, 58, 59]. In [56], authors show that most of today's inter-domain traffic flows directly between large content providers, CDNs, and 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 [54], 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 [57, 58, 59]. In particular, [58] focuses on user-content latency analysis at the Google CDN, [59] provides a comprehensive study of the Akamai CDN architecture, and [57] characterizes the performance of both Akamai and Limelight in terms of server availability and delay (note that this paper has been withdrawn by request of Microsoft due to some inaccuracies flagged by Akami and Limelight, but we cite it due to some interesting ideas). Deeper performance analysis of major CDNs are also provided in [60] (analysis of Yahoo's content delivery) and [61] (behavior of a very popular and open CDN – CoralCDN).

An interesting direction exploited some years ago in terms of CDN characterization and analysis is the one proposed by authors in [62, 63], where they proposed a systematic approach for CDN operators and ISPs to collaborate in the delivery of content to the end-customers, using the best of both sides (i.e., content location for the CDN, network performance and topology for the ISP).

There has also be a recent surge of papers analyzing the structure, performance and functioning of Google's and Microsoft's CDNs [64, 65, 66], pointing to the still lack of general understanding on such issues.

Despite the attention on the characterization of Web traffic received by the research community, literature is still lacking studies based on large-scale analysis, in particular applied



(a) Share of HTTP volume per service.

(b) Number of non-unique users per service.

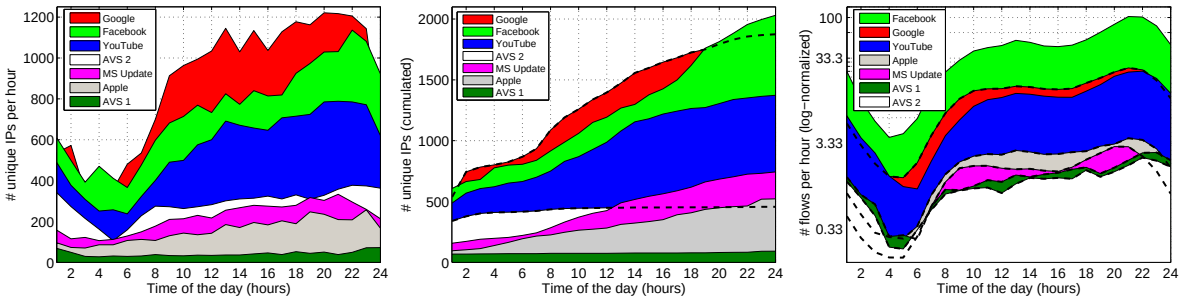
Figure 4.1.: Share of daily HTTP traffic volume and users of the top 10 services accessed in this network during the studied week. YouTube is the killer application w.r.t. volume in mobile networks, with a volume close to 30% of the overall HTTP traffic. Facebook and Google Search are the top services w.r.t. number of users, being accessed by about 50% of them.

in cellular networks. The results showed in this Chapter aim at filling this gap. Indeed, to the best of our knowledge, we are the first to (i) provide such deep large-scale characterization of Internet traffic in cellular networks, (ii) compare the analysis of a popular video streaming service (i.e., YouTube) from both the perspective of fixed-line and cellular networks, (iii) study the provisioning systems of other popular Internet services (i.e., Facebook and WhatsApp). Based on our results, some recent papers have started presenting results in some of these directions [67].

4.3. Understanding the Provisioning Systems of the Internet's Big Players

In this section we focus on the hosting and service delivery analysis. This includes a characterization of the number and temporal provisioning of the IP addresses used for each service, the placement of the hosting servers, and the identification of load balancing techniques.

The results illustrated in this Section are obtained by the analysis of a full week of HTTP(S) traffic traces collected in 2013 at the mobile broadband network of a major European ISP. The dataset characteristics are summarized in Table 4.1. The architecture for the collection, storage and analysis of such traces is similar to the one showed in the previous Chapter (cfr. Section 3.3) and it is based on the METAWIN passive monitoring system, the DBStream stream data-warehouse and HTTPTag, described in Section 3.4. The obtained dataset consists of more than half a billion of passively observed HTTP flows, aggregated in a per-hour basis. For each flow, the following meta-data are available: the contacted URL, the



(a) Unique IPs per hour. (b) Cumulative number of unique IPs. (c) Number of flows per hour.

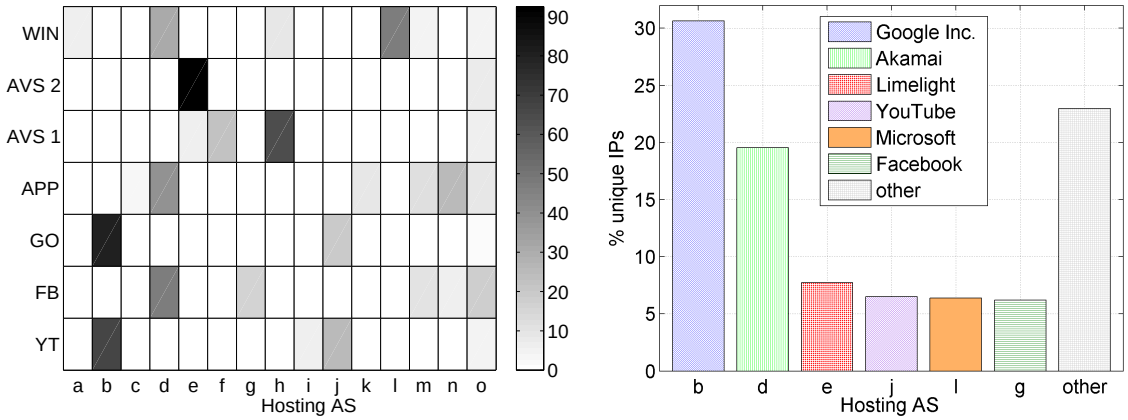
Figure 4.2.: Evolution of unique IPs and num. of flows for the top-7 services on a single day. Google Search, Facebook and YouTube dominate the IP space and account for the majority of the flows. Thanks to Akamai, Facebook is the most IP-distributed service, using more than 2000 different IPs on a single day.

contacted IP address, the total bytes exchanged with the contacted IP, and a timestamp. In addition to the Full Qualified Domain Name (FQDN), the corresponding service is deduced by using HTTPTag, as showed in the previous Chapter. As mentioned, the dataset also includes the name of the organization owning the contacted IP, extracted from the MaxMind ASes databases.

To limit the number of services to study, the analysis is performed exclusively for the top-7 services identified in Figure 4.1, which account for the majority of the HTTP traffic volume. The service ranking has been obtained by first classifying the HTTP flows by using the HTTPTag tool and then computing the per-service traffic volume as the sum of uplink and downlink traffic including the TCP header (Figure 4.1(a)) and by user count (Figure 4.1(b)). The list of services under study ordered by volume consists of: `YouTube` (YT), `Facebook` (FB), `Google` (GO), `Apple` (i.e., App Store and iTunes - APP), two well-known Adult Video Streaming services (`AVS 1` and `AVS 2`), and `Microsoft Updates` (WIN). The list also includes `Others`, which corresponds to other three video streaming services grouped together. YouTube is clearly the killer HTTP service in terms of volume, carrying almost 30% of the total HTTP volume. Facebook and Google Search account for a smaller share of the daily volume, both around 5%, but are accessed by a much higher number of users, about 50% of the total users in the network.

network type	cellular 3G
monitoring system	METAWIN at Gn interface
ticket type	HTTP tickets
length	7 days (Mon - Sun)
time	Q1 2013

Table 4.1.: Dataset used for characterization of HTTP Services Provisioning



(a) Services & hosting organizations.

(b) Share of IPs per organizations.

Figure 4.3.: 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 4.2.: Top hosting organizations and ASes in terms of number of unique IPs of the top-10 services (non-ordered list).

4.3.1. Big Players' addressing space

Let us begin by analyzing the number of unique IP addresses used to deliver each of these top-7 services on a single day. Figures 4.2(a) and 4.2(b) depict the evolution of the number of unique IPs per hour and the accumulated number of unique IPs on a single day, whereas Figure 4.2(c) plots the number of HTTP flows per hour (values are normalized for protecting privacy and business sensitive information). For 6 out of the 7 services (i.e., all except AVS 1), there is a clear correlation between usage and number of unique 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 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 we explore now how distributed are these unique IPs

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)

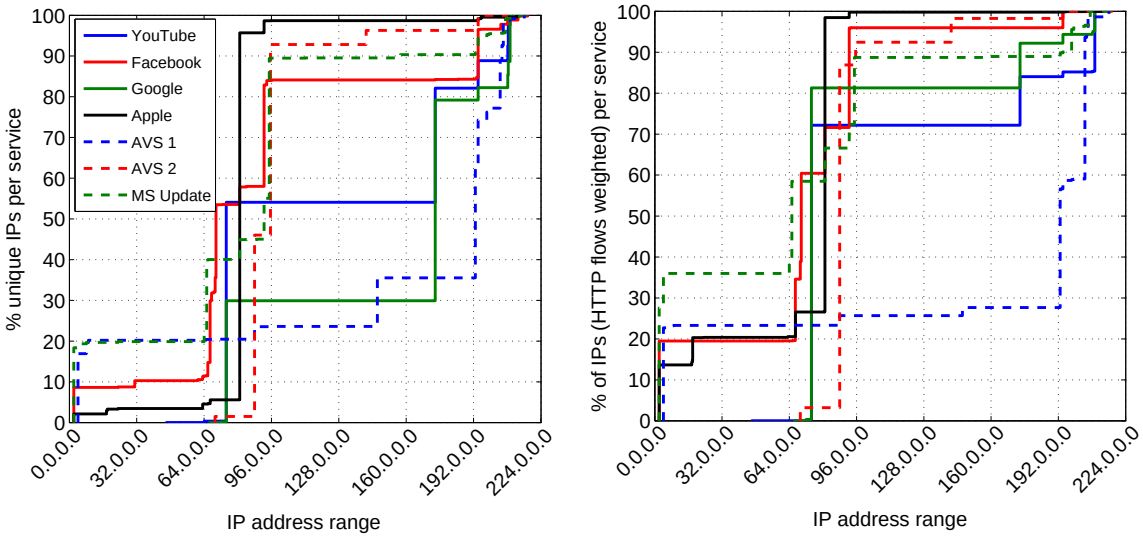
Table 4.3.: Number of IPs and blocks hosting the top-7 services. The top /24 subnetworks are defined in terms of number of HTTP flows delivered.

in terms of the different organizations owning them. Figure 4.3(a) shows the fraction of unique IPs per service hosted by the list of organizations and ASes described in table 4.2. 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 [54]. 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.

Table 4.3 provides a summary on the number of IPs and *potential* /16 and /24 subnetworks 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 of servers (e.g., more than 27.000 in 2008 according to [57]) following the *enter deep into ISPs* approach [57], by deploying content distribution servers inside ISP POPs. The idea behind such an approach is to get close 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 in terms of IP blocks, having around 450 different IPs scattered around 6 /16 IP blocks. As shown in Figure 4.3(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 [57], 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 4.3(a) is that many IPs delivering different services are usually hosted by the same organization. For example, Akamai hosts content



(a) IP range of unique IPs. (b) IP range of unique IPs (weighted by num. flows).

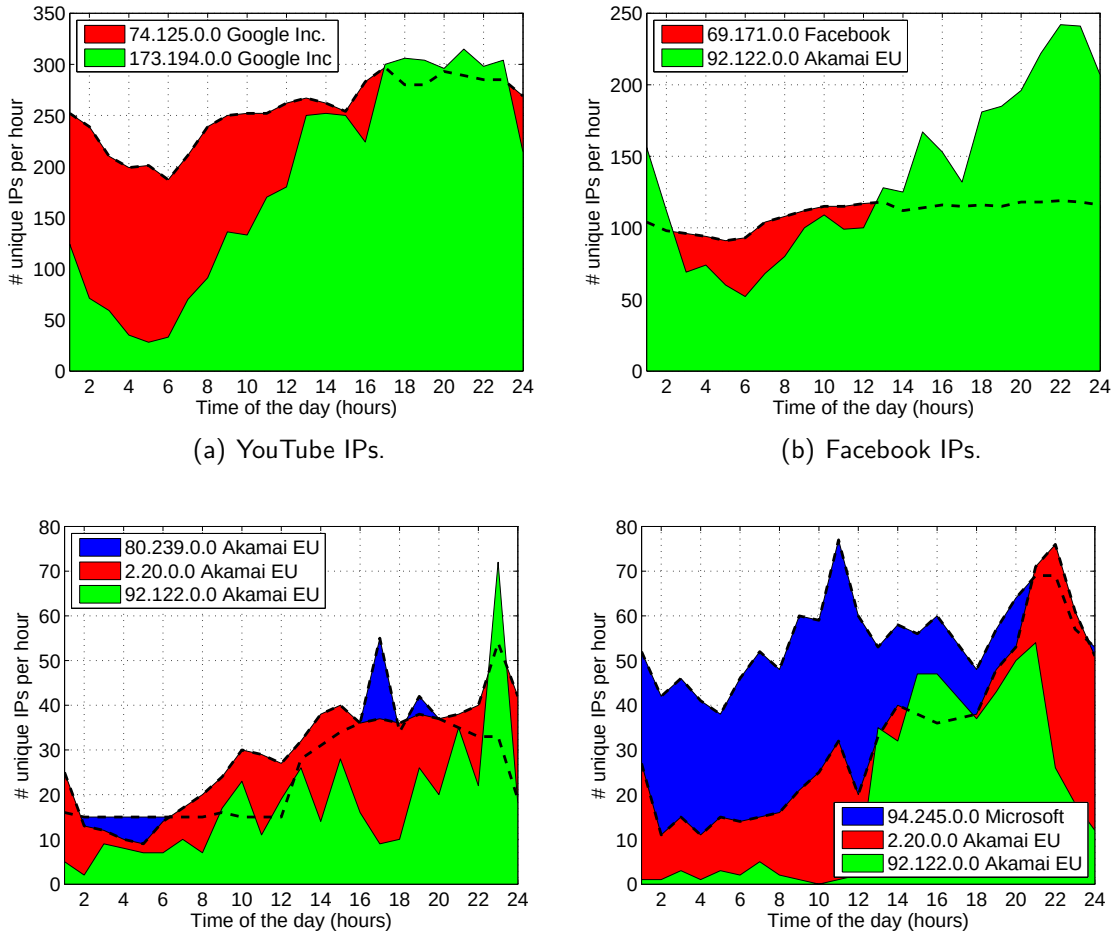
Figure 4.4.: Distribution of the IP range associated to the tagged services on a single day. AVS 1 is highly distributed in terms of different IP blocks, whereas AVS 2 is mostly served from a small number of blocks.

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 4.4 depicts the distribution of the IP address ranges associated to the different top-7 services on a single day. Figure 4.4(a) depicts the distribution of IPs without considering the actual number of HTTP flows being served by each IP, whereas Figure 4.4(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 4.3, AVS 1 has only 92 unique IPs delivering its content, which are distributed along 23 different $/16$ IP blocks. Figure 4.4(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.

4.3.2. Temporal dynamics of IP addresses

We move-on the analysis to the temporal evolution of the IPs used by some selected services and CDNs. Figure 4.5 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 4.5(a) and 4.5(b) respectively. Two /16 blocks are plotted in each case, the former



(a) YouTube IPs.

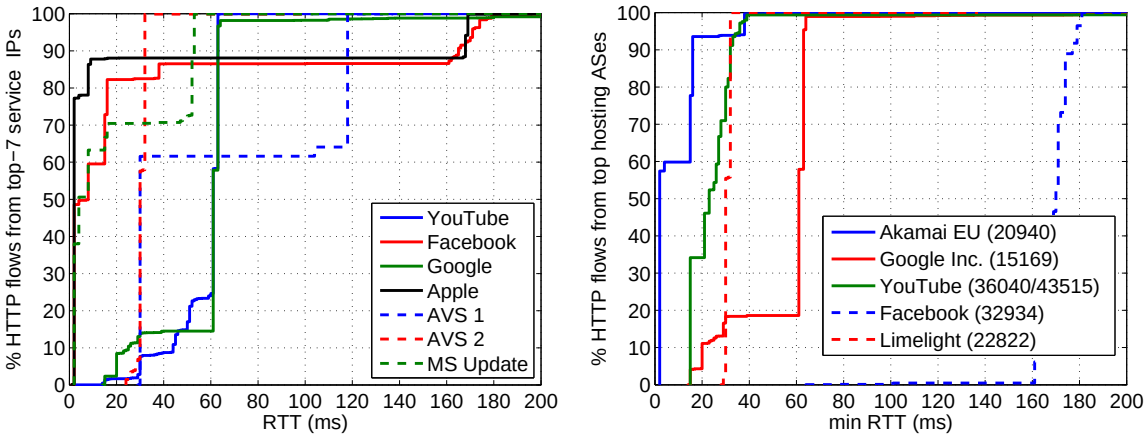
(b) Facebook IPs.

(c) Apple IPs.

(d) Windows Update IPs.

Figure 4.5.: Temporal evolution of number of hourly unique 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.

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 sever 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 4.5(c) and 4.5(d) show similar behaviors for 3 different IP



(a) min RTT per services.

(b) min RTT per hosting organization.

Figure 4.6.: 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.

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).

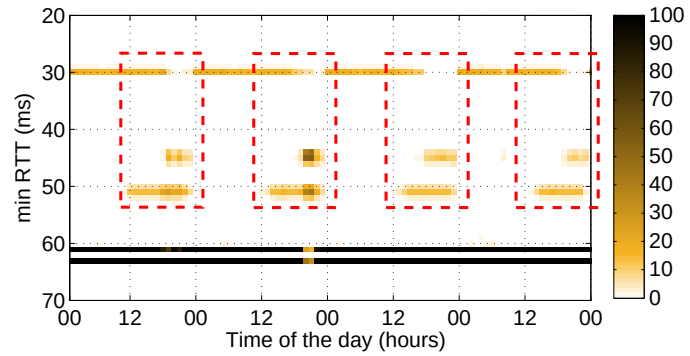
4.3.3. CDN servers location and load balancing policies

The last part of this Section is devoted to the identification of CDN servers location and load balancing policies. Similar to [54], we consider the Round Trip Time (RTT) to the hosting servers as a measure of the their distance from the Vantage Point. The RTT to any specific IP address consists both on 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 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, 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

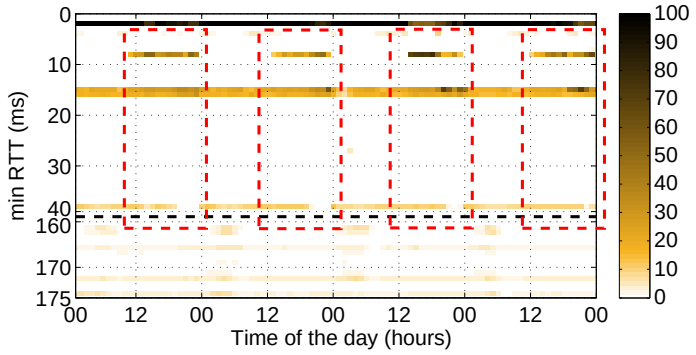
hour are periodically pinged. In particular, every unique IP is pinged with trains of 100 ICMP echo request packets every 10 minutes, resulting in a total of 6 individual values of min RTT per hour and per IP. We are very aware that obtaining such min RTT measurements by active probing is not the best 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 impacts of such shortcomings, we filter out all the inconsistent results providing different min RTT values at different hours of the day.

Figures 4.6(a) and 4.6(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 4.6(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 4.6(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 4.6(b) for the Limelight CDN. The two marked and very similar modes for Limelight min RTT in 4.6(b) reinforce the comments on the *bring ISPs to home* approach. A deeper analysis of the underlying IPs with the MaxMIND GeoIP data 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 4.2. According to Figure 4.6(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 has a major impact in the classification confusion matrix between Google Search and YouTube flows seen in figure 3.7.

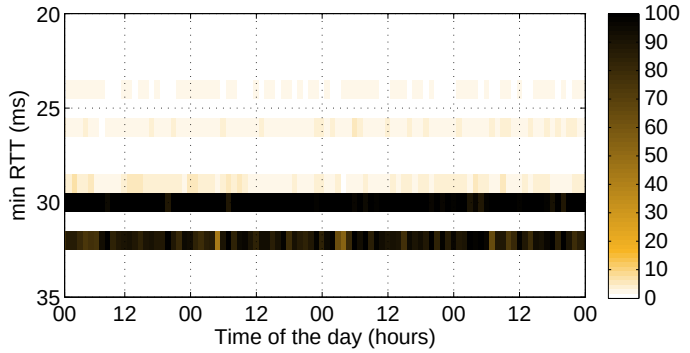
To conclude with this part of the study, we analyze now the temporal evolution of the min RTT for some selected services, aiming to show evidence on load balancing techniques employed by the Google CDN, Akamai, and Limelight. Figure 4.7 depicts the hourly evolution of the min RTT for different service flows during 4 consecutive days, from Monday to Thursday, including YouTube (mainly Google CDN), Facebook (mainly Akamai), and AVS 2 (mainly Limelight). Each column of the Figures in 4.7 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 4.7(a) plots the results for YouTube flows. The majority of the flows are delivered from the two Google locations depicted in Figure 4.6(b) at 61ms and 63ms, about 15% of the flows are served from a third location at



(a) min RTT of YouTube flows.



(b) min RTT of Facebook flows.



(c) min RTT of AVS 2 flows.

Figure 4.7.: 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.

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 4.7(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 4.7(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 no observable temporal patterns, suggesting that Limelight is not applying load balancing techniques in Europe, at least not for provisioning the corresponding service.

4.4. A Popular Video Streaming Service: YouTube

We start the second part of this Chapter by targeting YouTube. YouTube is the most popular video streaming service in the Internet, and is responsible for more than 30% of the overall traffic [56, 54]. Every minute, 100 hours of video content are uploaded, and more than one billion users visit YouTube each month[1]. This enormous popularity poses complex challenges to network operators, who need to design their systems properly to cope with the high volume of traffic and the large number of users. The challenges are bigger for mobile operators, who have to deal with an ever-increasing traffic volume with the capacity constraints of mobile networks, and in a much more competitive market. Indeed, mobile makes up to almost 40% of YouTube's global watch time, and video traffic accounts for more than 30% of the downstream peak traffic in large-scale cellular networks such as AT&T in the US [68]. Finally, the provisioning of YouTube through the massive Google CDN [58] makes the overall picture even more complicated for ISPs, as the video requests are served from different servers at different times. The highly distributed architecture and dynamic behavior of large CDNs allow achieving high availability and performance; however, content delivery policies can cause significant traffic shifts in just minutes, resulting in large fluctuations on the traffic volume carried through the ISP network paths.

These observations have motivated a large research effort on understanding how YouTube works and performs [69, 70, 71, 72], covering aspects such as content delivery mechanisms, video popularity, caching strategies, and CDN server selection policies among others. These papers focus exclusively on YouTube as observed in fixed-line networks. We now take a step further on the characterization of YouTube, additionally considering the impact of the type of network on the specific flow characteristics and provisioning behavior of the underlying servers. In particular, we perform a comparison of how YouTube is provisioned in fixed-line and mobile networks, analyzing four days of YouTube traffic traces collected in both networks. The insights of this analysis are particularly useful for shedding light on the complex mechanisms that regulate the provisioning of such a popular service and ultimately provide a valuable input for the anomaly detection and diagnosis systems that will be described later.

[1] `http://www.youtube.com/yt/press/statistics.html`

	Fixed-line dataset (FL)	**Mobile dataset (M)**
network type	fixed line (ADSL)	cellular (3G)
monitoring system	Tstat	Metawin (Gn)
ticket type	video flows	HTTP
length	$\sim$ 4 days (Mon-Thu)	$\sim$ 1 month
time	Q2 2013	Q2 2013

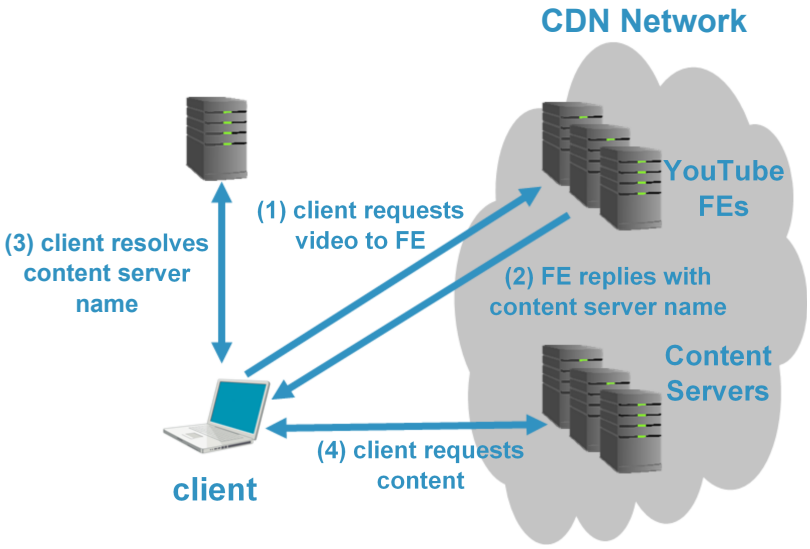
Table 4.4.: Datasets used for characterization of YouTube

In the remainder of this Section we provide an analysis of YouTube from both fixed-line and mobile vantage points. In particular, we find out that the wide-spread usage of caching in mobile networks provides high benefits in terms of delay to the contents as well as downlink throughput. In addition, we identified marked variations on the delay from the fixed-line vantage point to the YouTube servers, suggesting either a widely spread and heterogeneous server farm behind the YouTube front-ends, or the presence of a highly dynamic path-changes policy in the interconnection to the preferred YouTube servers. Finally, we estimate how the performance we have passively measured could impact end-users Quality of Experience. This is particularly important for understanding QoE anomalies, which will be one of the topics of the next Chapters.

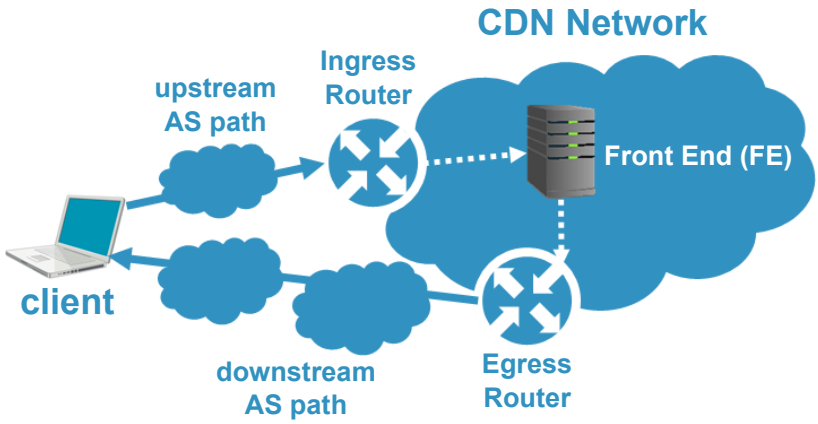
The two datasets, summarized in Table 4.4, correspond to almost 90 hours (from Monday till Thursday) of YouTube flows collected at two major European ISPs during the second quarter of 2013. In the mobile network, flows are captured at the Gn interface by the METAWIN monitoring system as already previously explained. In the fixed-line network, the monitored link aggregates 20,000 residential customers accessing to the Internet through ADSL connections and flows are captured using the Tstat passive monitoring system [7]. In both cases, the anonymized network traces are stored and analyzed in a DBStream instance and classified using HTTPTag. As done in the previous Section, the dataset is complemented with the name of the ASes hosting the content extracted from the MaxMind database.

4.4.1. Delivery Infrastructure

Google replicates YouTube content across geographically distributed data-centers worldwide, pushing content as close to end-users as possible to improve the overall performance of the video content provisioning, minimizing the effects of peering point congestion and enhancing the user experience. Google's CDN uses a complex content location and server selection strategy for optimizing client-server latency, increase QoE in general, and perform load balancing. User requests are normally redirected to the closest servers, based on Round Trip Time (RTT) measurements. For doing so, YouTube keeps a periodically updated latency map between its servers and BGP prefixes aggregating geo co-located users [73]. As depicted in Figure 4.8(a), Google uses the DNS service for redirecting requests to the preferred servers, additionally using dynamic cache selection strategies to balance the load among YouTube servers. YouTube Front End (FE) servers are those handling the original user request for a specific video, which can then redirect the user to additional YouTube servers mirroring the content. In some cases, YouTube servers located at multiple ASes of distance are selected



(a) Video retrieval workflow (extended figure, original source at [71]).



(b) Google's CDN (extended figure, original source at [73]).

Figure 4.8.: YouTube workflow for video retrieval and content location. Google's CDN uses a complex content location and server selection strategy for optimizing client-server latency, increase QoE in general, and perform load balancing. DNS is used for request re-directioning.

(see Figure 4.8(b)), resulting in higher delays and potentially impacting the performance of the video delivery in terms of download throughput.

Table 4.5 reports the number of unique server IPs serving YouTube in both networks, as well as the ASes holding the major shares of servers. To understand how these IPs are grouped, the table additionally shows the number of IPs per different network prefix. Even if the number of customers associated to the mobile network traces is much larger than in the fixed-line network, the number of unique server IPs observed in the latter is almost the double, with more than 3600 different IPs in the 90 hours, which could be probably explained by a more intensive use of video streaming services from fixed-line networks. In both cases,

Autonomous System	# **IPs**	#/**24**	#/**16**
All server IPs seen in FL	3646	97	22
15169 (Google)	2272	60	2
43515 (YouTube)	1222	12	1
36040 (YouTube)	43	2	2
All server IPs seen in M	2030	63	10
15169 (Google)	1121	38	2
43515 (YouTube)	844	15	2
LISP	35	4	3
36040 (Google)	26	5	3

Table 4.5.: Number of IPs and prefixes hosting YouTube, as observed in both Fixed-line (FL) and Mobile networks (M).

(Network) Autonomous System	% **bytes**	% **flows**
(FL) 15169 (Google)	80.8	77.3
(FL) 43515 (YouTube)	19.1	22.5
(M) LISP	69.3	66.7
(M) 15169 (Google)	30	32.7

Table 4.6.: Number of uplink and downlink bytes and flows per AS hosting YouTube in Fixed-line (FL) and Mobile (M) networks.

two Google ASes hold the majority of the IPs (i.e., AS 15169 and AS 43515), grouped in a small number of $/16$ subnets. In the mobile network we also include the observed IPs of the Local ISP (LISP), which plays a key role in the delivery of YouTube, due to the extensive usage of content caching. Indeed, it is very common in mobile networks to have forwarding caches at the edge of the network to reduce latency and speed up content delivery [68]. Even though the impact of video caching on the Radio Access Network is limited, ISPs might prefer to reduce the load on the transport network to both reduce peering costs and improve closeness to the content.

Table 4.6 shows that about 80% of the YouTube volume and number of flows are served by the AS 15169 in the fixed network, and up to 70% of the traffic is served by IPs owned by the LISP in the mobile network. This correlates pretty well with the fact that about 65% of the HTTP video content observed in the mobile network of AT&T in the US can be cached at the edge in standard forwarding proxies [68]. Still, we can not say from our analysis whether these IPs correspond to content caching performed by the LISP or also to Google servers deployed inside the ISP, which is a common approach followed by Google to improve end-user experience, known as Google Global Cache (GGC)[2]. In fact, a large share of YouTube content

[2] `https://peering.google.com/about/ggc.html`

is normally transparent to middle boxes, as videos are marked as "no-cache". A further study of this aspect could be an interesting research direction for the future.

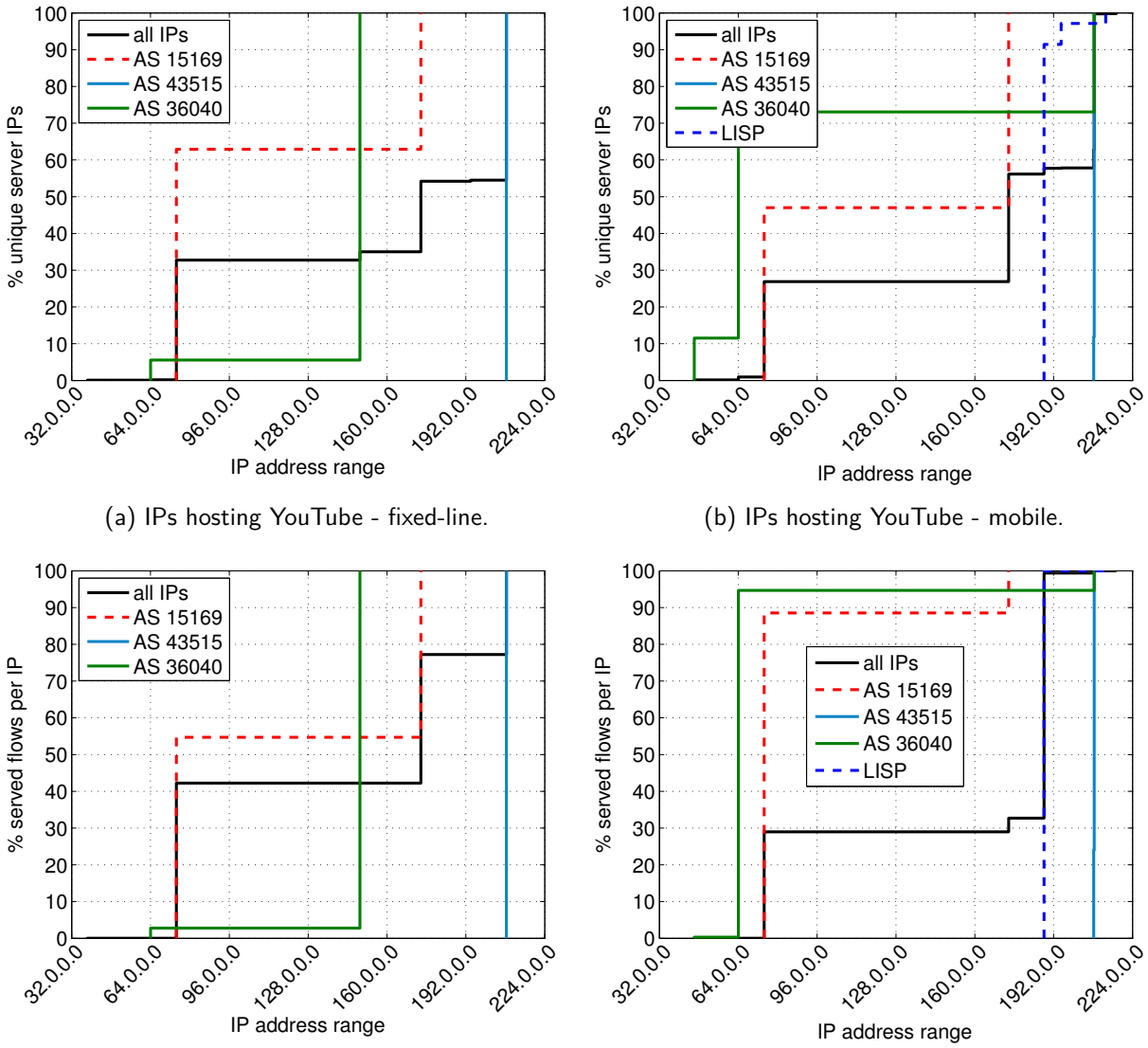
To appreciate which of the aforementioned IP blocks host the majority of the YouTube flows, Figure 4.9 depicts the distribution of the IP ranges and the flows per server IP. According to Figures 4.9(c) and 4.9(d), the majority of the YouTube flows are served by two or three well separated $/16$ blocks in the fixed-line and mobile networks respectively. Interestingly enough, only a limited fraction of YouTube traffic is served from AS 43515 in the mobile network. Figure 4.10 additionally depicts the number of flows served per IP in both networks. Separated steps on the distributions evidences the presence of preferred IPs or caches serving a big number of flows, which are most probably selected by their low latency towards the end customers.

Finally, we study the dynamics of the traffic provisioning from the aforementioned ASes. Figure 4.11 depicts (a,b) the number of active IPs and (c,d) the flow counts per hour (normalized from 0 to 100) in both networks during three consecutive days. In both networks, the active IPs from either AS 43515 or AS 15169 show an abrupt increase at specific times of the day; for example, about 200 IPs from AS 43515 become active daily at about 10:00 in the fixed-line network, whereas IPs from AS 15169 almost triple at peak hours (between 17:00 and 23:00) in the mobile network. Note that the number of active IPs from the LISP is constant during the whole period, showing their main role in the delivery of YouTube flows. In terms of flow counts, Figure 4.11(c) evidences a very spiky behavior in the flows served from AS 43515, and some of the load balancing policies followed by Google in the region of the fixed-line ISP, e.g., a drastic switch from AS 15169 to AS 43515 of the flows served at about 18:00. In the mobile network, the LISP servers handle the majority of the flows daily, and as a consequence, the dynamics of the flow counts are much smoother. This indirectly implies that the load forecasting from each of the servers is much straightforward in the mobile network, resulting in a potentially much easier traffic management at the core network.

4.4.2. How Far Away are YouTube Videos?

As done in the previous Section, we investigate now the latency and the location of the previously identified servers, considering the distance to the vantage points in terms of Round Trip Time (RTT). RTT measurements are passively performed on top of the YouTube flows in the fixed-line network. Mobile networks usually employ Performance Enhancement Proxies (PEPs) to speed-up HTTP traffic, and therefore, passive min RTT measurements on top of HTTP traffic provide incorrect results [74]. We therefore consider an active measurement approach in the mobile network, running standard pings from the vantage point to get an estimation of the min RTT to the servers. As before, we then weight the obtained min RTT values by the number of flows served by each IP to get a rough picture of where the flows are coming from.

Figure 4.12 shows the distribution of the min RTT values for the flows observed in both networks. Steps in the CDF suggest the presence of different data-centers or clusters of co-located servers. Figure 4.12(a) shows that about 65% of the flows in the fixed-line network come from servers most probably located in the same country of the ISP, as min RTT < 5



(a) IPs hosting YouTube - fixed-line.

(b) IPs hosting YouTube - mobile.

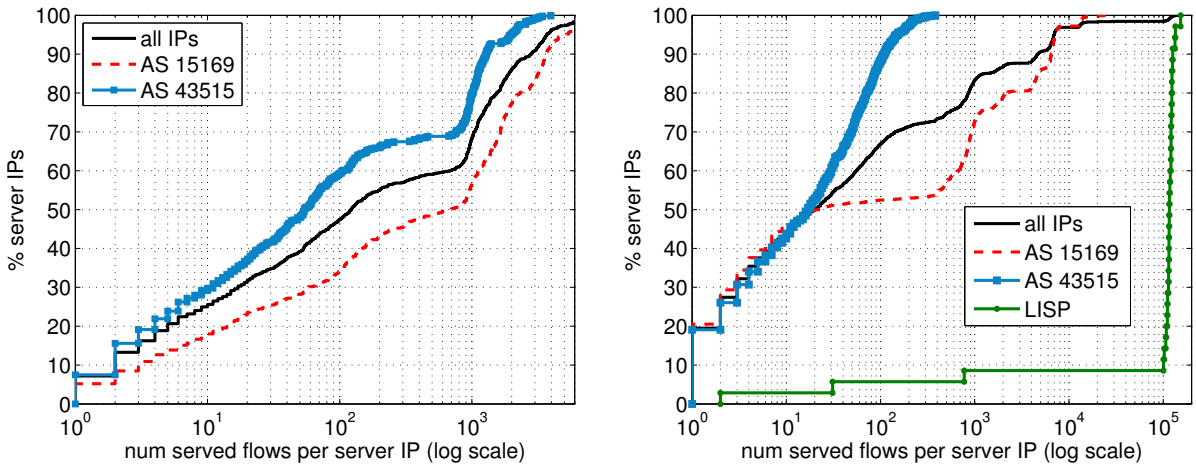
(c) Flows per server IP - fixed-line.

(d) Flows per server IP - mobile.

Figure 4.9.: IP ranges distribution and flows per server IP hosting YouTube. The majority of the YouTube flows are server by very localized IP blocks.

ms. This is coherent with the fact that Google selects the servers with lower latency to the clients. A further differentiation by AS reveals that the most used servers in AS 15169 are located much closer than the most used servers in AS 43515. As depicted in Figure 4.12(b), the lion share of the flows in the mobile network comes from the LISP servers, which are located inside the ISP (min RTT $<$ 2 ms). The rest of the flows served from AS 15169 are located at potentially two geographically different locations, one closer at around 40 ms from the vantage point, and one farther at about 70 ms.

The richness of the passive RTT measurements performed in the fixed-line network permits to further study the dynamic behavior of the servers' selection and load balancing strategies



(a) Flows per IP in fixed-line.

(b) Flows per IP in mobile.

Figure 4.10.: Flows per IP and per AS. Clear sets of IPs serve a large share of the flows, evidencing the presence of preferred caches.

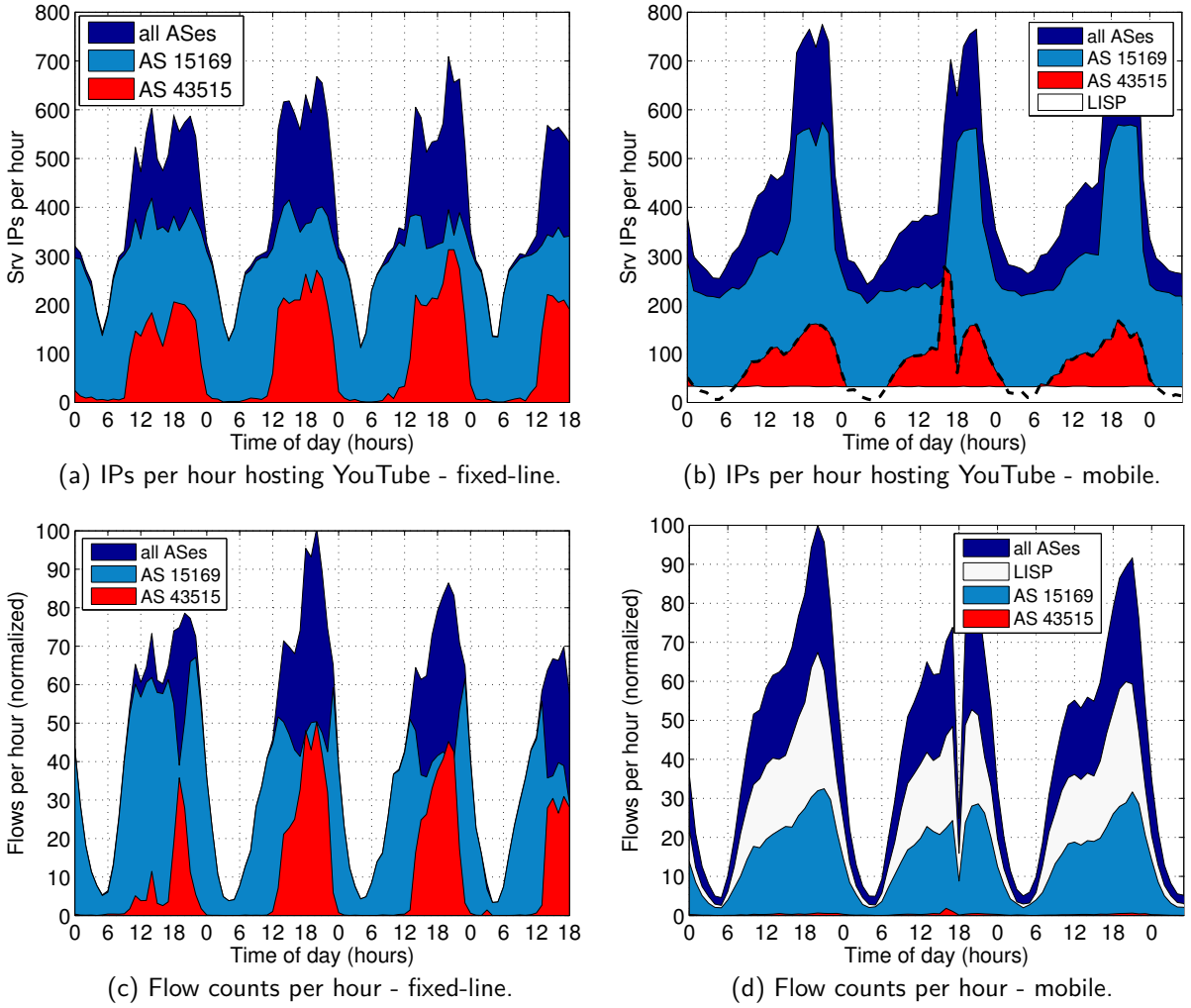
used by Google to choose the servers. Figure 4.13(a) depicts the variation of the distribution of min RTT measured on the YouTube flows for a complete day, considering contiguous time bins of 3 hours length. Correlating these results with those in Figure 4.11(c) permits to better understand the daily variations. Whereas the majority of the flows are served from very close servers until mid-day, mainly corresponding to AS 15169, servers in farther locations are additionally selected from 14:00 on, corresponding to the increase in the number of flows served from AS 43515.

Finally, Figures 4.13(b) and 4.13(c) reveal a very interesting pattern which could be potentially harmful for the performance of the video delivery, but that we were not able to diagnose. The Figures depict the min RTT values observed during a complete day for flows hosted at different IPs in two $/24$ subnets at AS 15169 and AS 43515, namely $74.125.13.0/24$ and $208.117.250.0/24$ respectively. The interesting observation is that the min RTT to the same set of IPs varies with a very structured pattern, presenting different clusters of min RTT values in both subnets. For example, min RTT values of 5, 9, and 14 ms are systematically observed for the flows served from IPs at $208.117.250.0/24$.

These marked variations could be the result of strong and very periodic congestion events, which is in fact very unlikely. We tend to believe that either a very spread and heterogeneous server farm behind the YouTube front-end servers in the corresponding IPs, or the presence of a highly dynamic path-changes policy in the interconnection to the specific YouTube servers is the origin of such a behavior.

4.4.3. YouTube Traffic and Performance

We study now the characteristics of the YouTube flows as observed from both vantage points, as well as the performance achieved in terms of downlink throughput. Figure 4.14 depicts the distribution of flow size for the different hosting ASes. Figure 4.14(a) shows that about 20%



(a) IPs per hour hosting YouTube - fixed-line.

(b) IPs per hour hosting YouTube - mobile.

(c) Flow counts per hour - fixed-line.

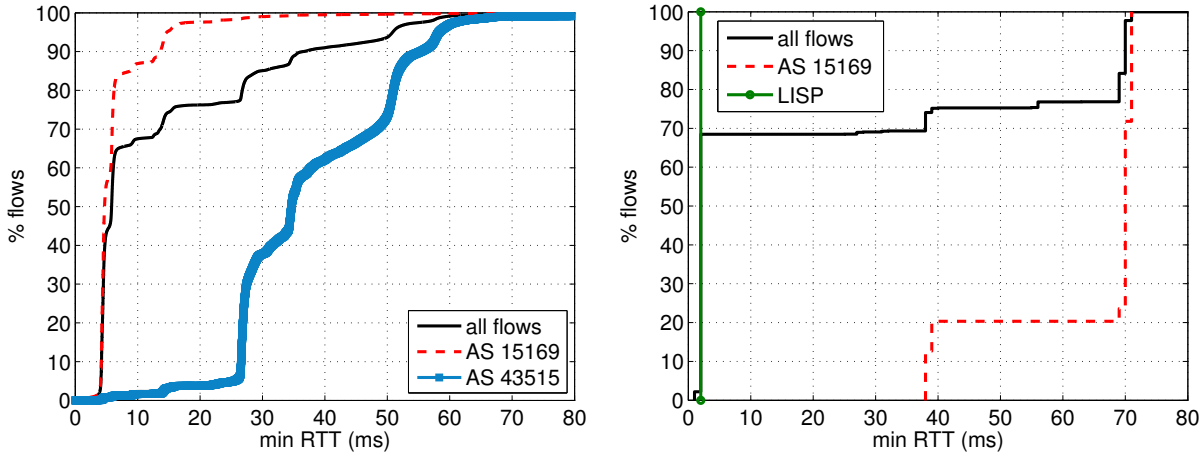
(d) Flow counts per hour - mobile.

Figure 4.11.: IPs and flows per hour during 90 hs. The glitch in the flow counts in the mobile network is caused by maintenance of the monitoring probe.

of the flows served in the fixed-line network are smaller than 1 MB, and that flows served by the AS 43515 are slightly smaller than those provided by the AS 15169 in this network.

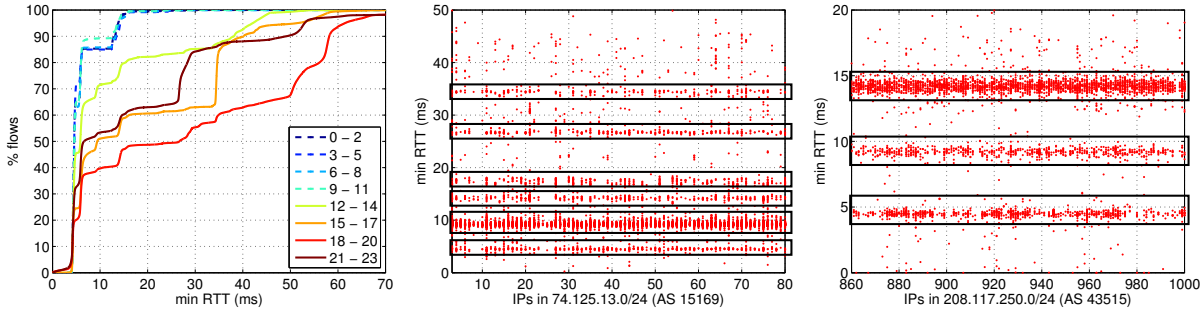
The CDF reveals a set of marked steps at specific flow sizes, for example at 1.8 MB and 2.5 MB. Our measurements and studies performed in [75] reveal that YouTube currently delivers 240p and 360p videos in chunks of exactly these sizes, explaining such steps. A similar behavior is observed for chunks of bigger sizes. About 75% of the flows are smaller than 4 MB, 90% of the flows are smaller than 10 MB, and a very small fraction of flows are elephant flows, with sizes higher than 100 MB.

The flows considered in Figure 4.14(b) for the mobile network are only those with a size bigger than 1 MB. This filtering is performed as a means to improve the estimation of the downlink throughput in our traces. Surprisingly, the flows served by the AS 43515 in the mobile network tend to be rather larger than those provided by the other ASes, and more



(a) min RTT (passive) in fixed-line. (b) min RTT (active) in mobile.

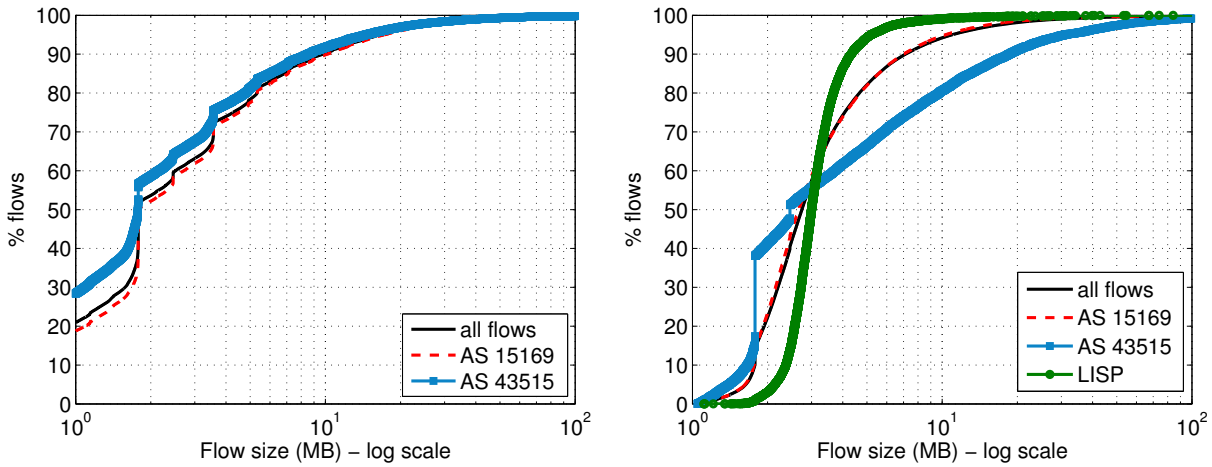
Figure 4.12.: min RTT to servers in different ASes. Latency is passively measured on top of the YouTube flows in the fixed-line network, whereas active RTT measurements are performed in the mobile network.



(a) min RTT - daily variation. (b) min RTT - AS 15169. (c) min RTT - AS 43515.

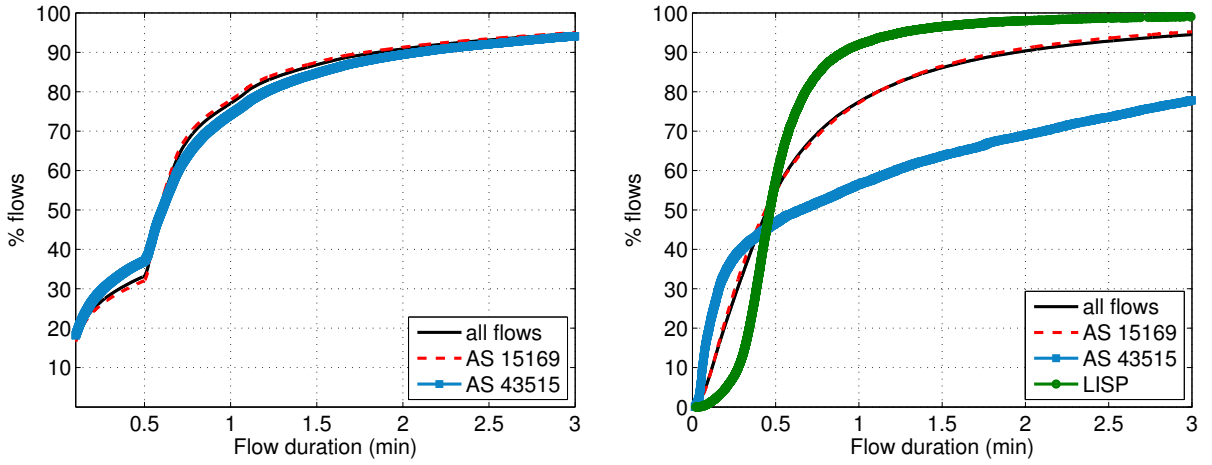
Figure 4.13.: min RTT dynamics in the fixed-line network. (a) The server selection strategies performed by Google are not only based on closest servers. (b,c) Strong variations on the min RTT to the same Google IPs suggest the presence of path changes or very heterogeneous latencies inside Google's datacenters.

than 20% of the flows served by this AS are bigger than 10 MB. The interesting observation comes when analyzing the size of the flows served by the LISP. The CDF reveals a very concentrated flow size between 2 MB and 4 MB, suggesting that the cached contents (or those served by YouTube servers inside the ISP) could potentially cover, at least in terms of flows size, 75% of the flows observed in the fixed-line network. We have not investigated the characteristics of the YouTube videos hosted by the LISP IPs and those served in the fixed-line network, which would provide further insights about the type of contents that are potentially cacheable. We plan to do so in future studies, following the approach used in [68].



(a) YouTube flow size - fixed-line. (b) YouTube flow size - mobile.

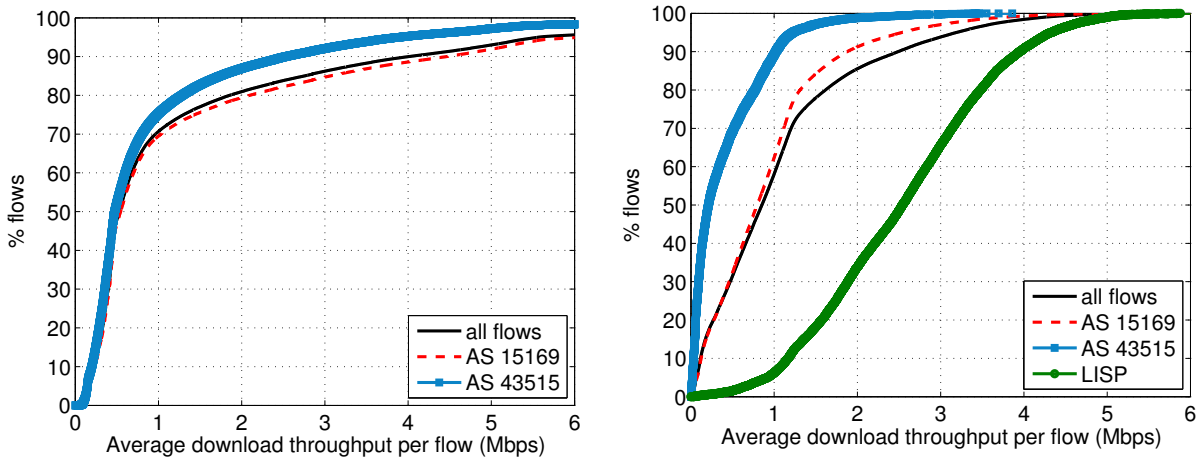
Figure 4.14.: YouTube flows sizes. The steps in the CDF at sizes 1.8 MB, 2.5 MB, 3.7 MB, etc. correspond to the fixed chunk-size used by YouTube to deliver videos of different resolutions and bitrate.



(a) YouTube flow duration - fixed-line. (b) YouTube flow duration - mobile.

Figure 4.15.: YouTube flows duration. About 85% of the flows observed in both networks are shorter than 90 seconds. A large share of flows have an average duration of about 30 seconds.

Figure 4.15 depicts the distribution of the flows duration, in minutes. The flow duration in both networks is below 3 minutes for about 95% of the total flows. The abrupt step in the CDF of the flows observed in the fixed-line network at about 30 seconds is most probably linked to the aforementioned video chunk sizes, but we were not able to verify this observation. About 85% of the flows observed in both networks are shorter than 90 seconds. Similar to the flow size, the flows served from AS 43515 are rather longer in the



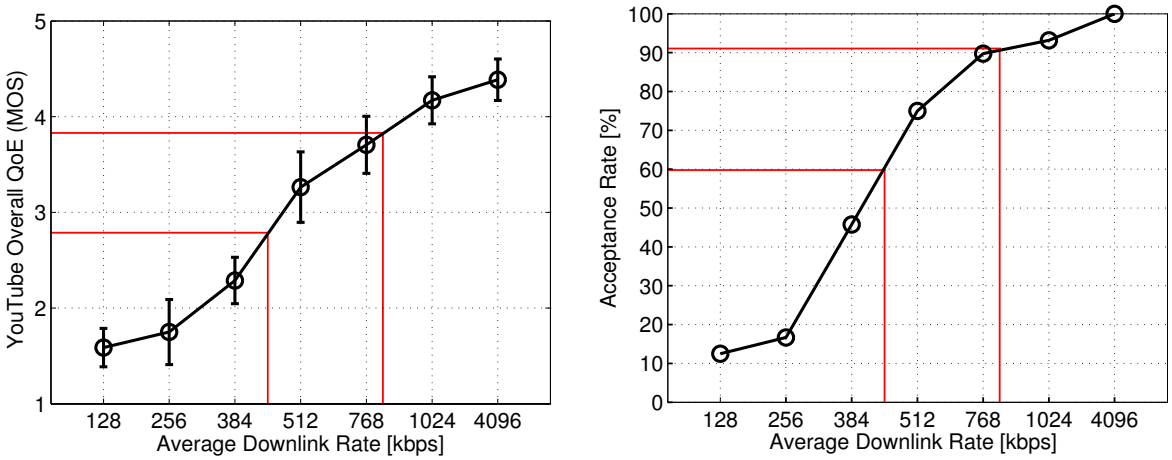
(a) YouTube throughput- fixed-line.

(b) YouTube throughput - mobile.

Figure 4.16.: Average YouTube flow downlink throughput per AS. Flows served by the LISP are the ones achieving the highest performance, evidencing the benefits of local content caching and low-latency servers.

mobile network, with more than 20% of the flows lasting more than 3 minutes. Given the small fraction of traffic served from AS 43515 in the mobile network, we can not say for sure that the behavior of the servers in this AS is different when it comes to different types of networks. Still, the important differences in the flow characteristics coming from AS 43515 in both networks might suggest some kind of network (or device) awareness on the way YouTube video is provisioned, as observed in [72]. Finally, and also correlating with previous observations, the distribution of the duration of the flows served by the LISP IPs is concentrated around 30 seconds, matching pretty well the aforementioned abrupt step in the CDF of the flow duration in fixed-line networks.

To conclude this part, Figure 4.16 reports the distribution of the average downlink throughput per flow measured in both networks, discriminating by hosting AS. The downlink throughput is the main network performance indicator that dictates the experience of a user watching YouTube videos [76]. Both Figures 4.16(a) and 4.16(b) consider only flows bigger than 1 MB, to provide more reliable and stable results (i.e., avoid spurious variations due to the TCP protocol start-up). The downlink throughputs achieved in both networks are rather similar, with more than 15% of the flows achieving a throughput above 2 Mbps. This suggests that the downlink throughput is partially governed by the specific video encoding bitrates and the flow control mechanisms of YouTube and not exclusively by the specific access technology. Still, when analyzing the performance results per AS, it is evident that the flows served by the LISP are the ones achieving the highest performance, with an average flow downlink throughput of 2.7 Mbps. This out-performance evidences the benefits of local content caching and low-latency servers for provisioning the YouTube flows.



(a) YouTube overall QoE vs. downlink rate.

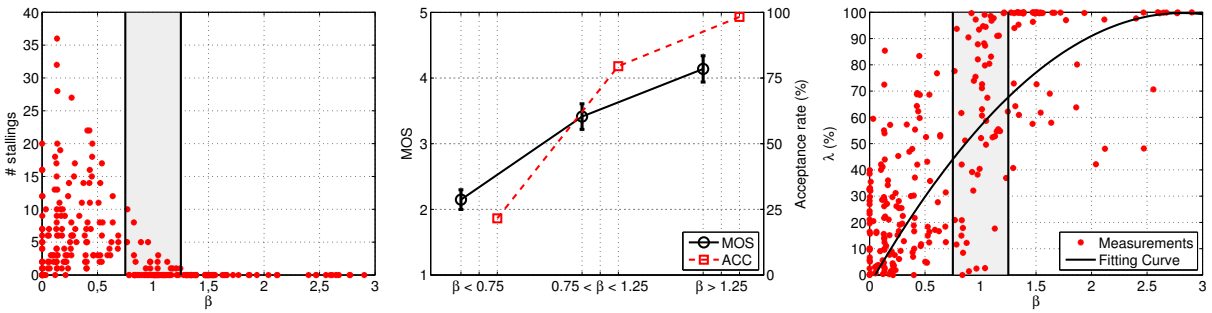
(b) YouTube acceptability vs. downlink rate.

Figure 4.17.: YouTube overall QoE and acceptability in terms of average downlink rate. The curves correspond to a best-case scenario, in which only 360p videos were considered. In a more general case with higher resolution videos (e.g., 1080p), the downlink rate has an even stronger effect on the user experience. The Figs. are taken with permission from the study performed at [78].

4.4.4. From Performance to Quality of Experience

Even if the download throughput has a direct impact on the performance of the video provisioning [77], authors in [78, 76] show that the main impairment affecting the QoE of the end-users watching HTTP video-streaming videos are playback stallings, i.e., the events when the player stops the playback. One or two stalling events are enough to heavily impact the experience of the end user. Given that the analyzed measurements report the average per flow download throughput as one of the monitoring KPIs, we rely on our previous results to better understand how download throughput relates to QoE and stallings in YouTube.

Figure 4.17 reports the overall QoE and the acceptance rate as declared by users watching YouTube videos during a field trial test conducted and reported in [78], both as a function of the average download rate. During this one-month long field trial test, about 40 users regularly reported their experience on surfing their preferred YouTube videos under changing network conditions, artificially modified through traffic shaping at the core of the network. Figure 4.17(a) shows the overall QoE as a function of the average download rate, using a 5-points MOS scale, where 1 corresponds to very bad QoE and 5 to optimal. The Figure clearly shows that the overall QoE drops from a MOS score close to 4 at 800 kbps to a MOS score below 3 at 470 kbps. A MOS score of 4 corresponds to good QoE, whereas a MOS score below 3 already represents poor quality. The same happens with the service acceptance rate, as reported in Figure 4.17(b). In the analysis, we shall consider the thresholds $T_{h_1} = 400$ kbps and $T_{h_2} = 800$ kbps as the throughput values splitting by bad, fair, and good QoE. Both curves correspond to a best-case scenario, in which only 360p videos were watched by the users. As we see next, both 360p videos and videos with higher resolutions are present



(a) # stallings vs. β. (b) MOS and acceptance vs. β. (c) User engagement (λ) vs. β.

Figure 4.18.: β = ADT/VBR as a metric reflecting user experience and engagement. Users have a much better experience and watch videos for longer time when $\beta > 1.25$. This threshold corresponds to an ADT = 700 kbps in 360p videos, which is the value recommended by video providers in case of 360p videos.

in the dataset, thus QoE degradations are potentially worse than those reported.

In addition, we introduce a simple yet effective QoE-based KPI to monitor the QoE of YouTube videos from network measurements. In [76], authors have devised a Deep Packet Inspection based approach to estimate stallings in YouTube from passive measurements at the core network. However, the used techniques can not be applied when YouTube flows are carried over HTTPS as it is currently happening, simply because it is no longer possible to access the encrypted content of the traffic. Therefore, using the same measurements of the field trial, we introduce a new approach. Intuitively, when the average download throughput (ADT) is lower than the corresponding video bit rate (VBR), the player buffer becomes gradually empty, ultimately leading to the stalling of the playback. We define β = ADT/VBR as a metric reflecting QoE. Figure 4.18 reports (a) the measured number of stallings events and (b) the QoE user feedbacks as a function of β. In particular, no stallings are observed for $\beta > 1.25$, and user experience is rather optimal (MOS > 4). As a direct application of these results, if we consider standard 360p YouTube videos, which have an average VBR = 600 kbps [72], an ADT = 750 kbps would result in a rather high user QoE, which is the value recommended by video providers in case of 360p videos. Figure 4.18(c) additionally shows how the fraction λ = VPT/VD (video played time and duration) of the video time actually viewed by the end users actually increases when β increases, specially above the $\beta = 1.25$ threshold.

To conclude the study of YouTube, we have shown that the usage of caching in mobile networks provides high benefits in terms of delay to the contents as well as downlink throughput. We have also identified a very interesting behavior on the latency to the YouTube servers in the fixed-line network.

4.5. An Online Social Network: Facebook

We continue the service-specific characterization by characterize the traffic and the delivery infrastructure of the highly popular web service Facebook. Facebook is the most popular and widely spread Online Social Network (OSN), with hundreds of millions of users worldwide sharing and accessing content on a daily basis[3]. Facebook content is mainly hosted by the well known Akamai CDN, which represents the most dynamic and widely deployed CDN today, with more than 137,000 servers in more than 85 countries across nearly 1,200 networks[4]. Facebook content is additionally hosted by the Facebook organization itself, with servers present both in the US and Europe. Finally, the intensive usage of transparent caching at the edge of ISP networks [47] and the deployment of large CDN caches inside the ISPs makes that a large fraction of the Facebook content accessed by the users is hosted at the premises of multiple network operators. Our study permits to better understand how the Facebook content is hosted and served by the aforementioned organizations. We show that the way these normally serve Facebook contents is very dynamic and complex to characterize, even revealing in some cases unexpected and interesting load balancing events.

network type	cellular 3G
monitoring system	METAWIN at Gn interface
ticket type	HTTP tickets
length	2 months (total)
time	Q2 2012 and Q2 2013

Table 4.7.: Dataset used for characterization of Facebook

The analysis of OSNs has been a very fertile domain in the last few years [79, 80, 81, 82, 83, 84]. Authors in [79] study the power-law and scale-free properties of the interconnection graphs of Flickr, YouTube, LiveJournal, and Orkut, using application-level crawled datasets. The work in [80] present a study on the privacy characteristics of Facebook. Some papers [81, 82] study the new Google+ OSN, particularly in terms of popularity of the OSN, as well as the evolution of connectivity and activity among users. Authors in [83, 84] focus on the temporal dynamics of OSNs in terms of user-interconnections and visited links, using again publicly crawled data of popular OSNs such as Facebook, Twitter, as well as a large Chinese OSN. All these papers rely on crawled web-data and do not take into account the traffic and networking aspects of OSNs. From the best of our knowledge, the study provided in this Section is the first tackling Facebook from the network perspective as seen in an operational cellular network.

The dataset used for the analysis' showed in this Section corresponds to one month of HTTP flow traces collected at the usual Vantage Point at core of a European ISP in mid 2013 (cfr. Table 4.7).

[3] `http://newsroom.fb.com/key-facts`
[4] `http://www.akamai.com/html/about/facts_figures.html`

Country	% hosted volume
Europe (generic)	46.8%
Local country	37.2%
Ireland	12.7%
Neighbor country	2.1%
United States	1.1%
Unclassified	0.1%

Table 4.8.: Top Facebook hosting countries by volume.

AS/Org.	# IPs	#/24	#/16
All	6551	891	498
Akamai	2264	132	48
Facebook AS	294	57	5
LO	26	8	6
NO1	368	26	14
NO2	374	33	9

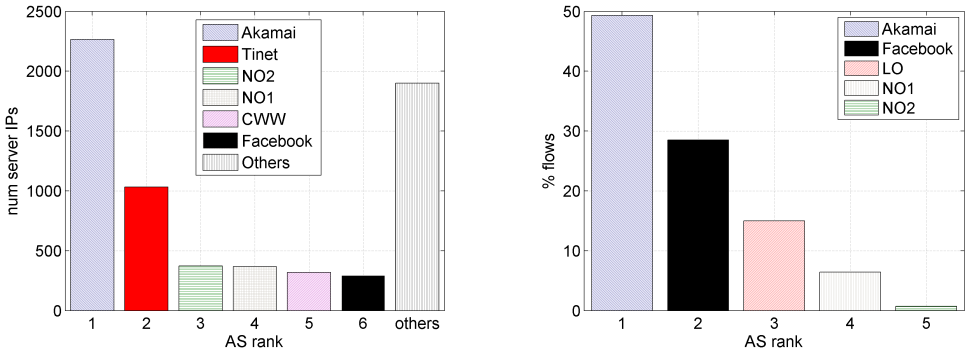
Table 4.9.: Number of IPs and blocks hosting Facebook.

4.5.1. Traffic and Content Delivery Infrastructure

As done in the case of YouTube, we start by characterizing the Facebook traffic as seen in our traces, with a special focus on its underlying hosting/delivery infrastructure. Due to the high number of daily users and the high volumes of served traffic, Facebook follows a sophisticated content delivery strategy. Indeed, we observed more than 6500 server IPs hosting Facebook contents in our traces, distributed across 20 countries and more than 260 different ASes. This confirms the wide-spread presence of several organizations hosting Facebook contents, turning the service provisioning into a very tangled scenario. Figure 4.19 shows the main organizations/ASes hosting Facebook content, both in terms of number of unique server IPs observed and share of delivered flows. Akamai is clearly the key player in terms of Facebook content hosting, delivering almost 50% of the flows in our traces, using more than 2260 different server IPs. Interesting enough is the large number of server IPs observed from two organizations which actually deliver a negligible share of the flows: the Tiscali International Network (Tinet) and Cable & Wireless Worldwide (CWW). We believe these organizations are only caching spurious Facebook contents. In the remainder of the study we focus on the top 5 organizations/ASes in terms of served flows, depicted in Figure 4.19(b): Akamai, Facebook AS, the Local Operator (LO) which hosts the vantage point, and two neighbor operators, Neighbor Operator 1 (NO1) and Neighbor Operator 2 (NO2).

4.5.2. Geographical Diversity of Facebook Hosting Servers

Table 4.8 provides an overview of the geographical diversity of the Facebook hosting infrastructure, listing the top countries where servers are located in terms of volume. The servers' location is extracted from the MaxMind GeoCity database, which is highly accurate at the country level [27]. "Europe (generic)" refers to a generic location within Europe for which MaxMind did not return a more accurate information. Almost 99% of the traffic comes from servers and data centers located in Europe, close to our vantage point, while only 1% of the traffic comes from other continents. This is due to three factors: (i) Akamai, the biggest Facebook content provider, has a very geographically distributed presence, pushing contents as close as possible to end-users [59]; (ii) operators heavily employ local content caching, and large CDNs like Akamai tend to deploy caches inside the ISPs networks, explaining the amount of traffic coming from the local country as well as neighboring countries to the van-



(a) Server IPs per AS.

(b) Share of flows hosted per AS.

Figure 4.19.: Unique server IPs used by the top hosting organizations/ASes and flow shares per hosting AS, considering the complete dataset. Akamai is clearly the key player in terms of Facebook content hosting.

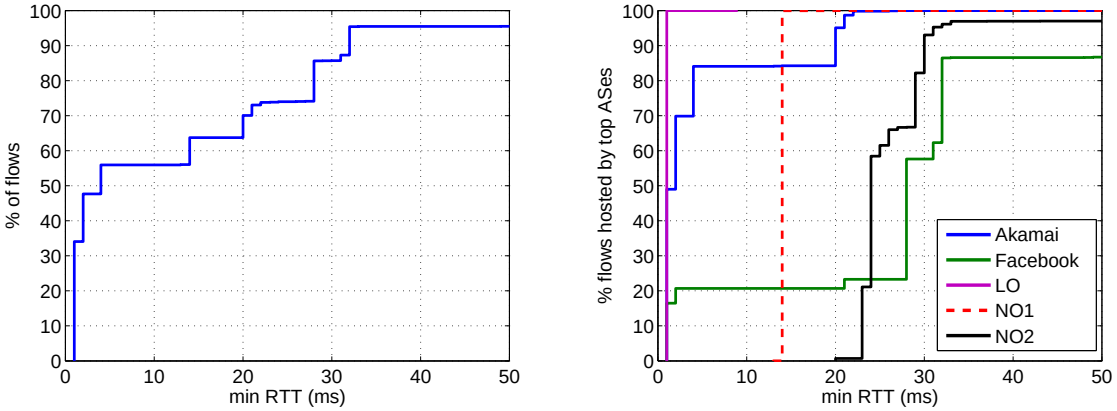
tage point; (iii) the rest of the traffic is handled directly by Facebook, which has servers split between Ireland (headquarter of Facebook International) and the US.

To complement this picture, we investigate the location of these servers from a network topology perspective, considering the distance to the vantage point in terms of Round Trip Time (RTT), as already previously done. We rely again to active RTT measurements collected through the standard ping tool to overcome the presence of Performance Enhancement Proxies (PEPs) (cfr. Section 4.4.2).

Figure 4.20 plots the cumulative distribution of the minimum RTT to (a) all the server IPs hosting Facebook, and (b) the aforementioned top orgs./ASes. Values are weighted by the number of flows hosted at each IP, to get a better picture of where the traffic is coming from. As a further confirmation of the geographical diversity, the distribution of min RTT presents some steps or "knees", suggesting the existence of different data centers and/or hosting locations. The largest majority of flows are served by close serves, located at less than 5 ms from the vantage point. As we mentionned, Akamai deploys its servers following the "enter deep into ISPs" approach [57], placing content distribution servers inside ISP POPs, which explains the short latency to the vantage point. The LO is the one with shortest delays for all the flows it serves and, along with the NO1, is the one with the least geographical diversity, with only one visible location. Three main steps appear in the CDF of the Facebook servers, which correspond to the headquarters in Ireland (min RTT about 30ms), the servers in the US (min RTT $>$ 100ms), and some servers located at only few milliseconds from the vantage point. Traceroutes to those servers revealed a direct connection to the Internet eXchange Point (IXP) of the local country, explaining the so low delays.

4.5.3. Facebook IP Address Space

Table 4.9 provides a summary on the number of unique server IPs observed in the traces, and the $/24$ and $/16$ IP blocks covered by the top orgs. hosting Facebook. Akamai and Facebook



(a) Min RTT to all server IPs.

(b) Min RTT to top hosting ASes.

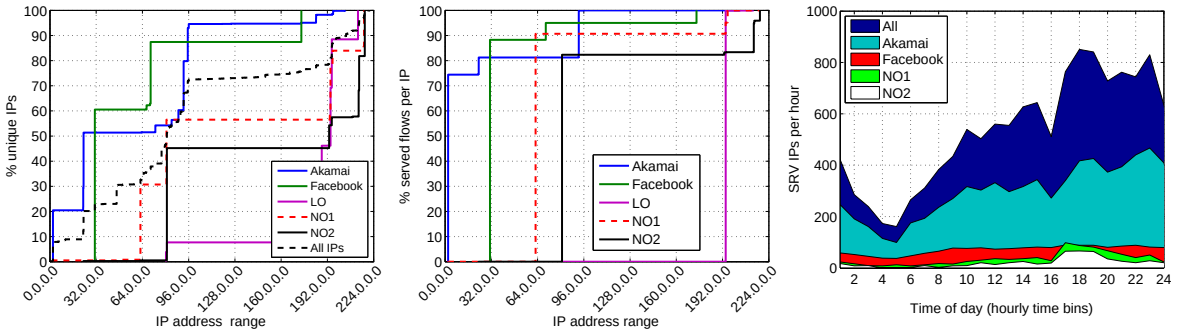
Figure 4.20.: Distribution of overall min RTT and min RTT per top hosting ASes to server IPs, weighted by the number of flows hosted.

together account for about 2560 servers scattered around almost 200 $/24$ IP blocks, revealing again their massively distributed infrastructure. However, we shall see next that only a few of them are actually hosting the majority of the flows.

Figure 4.21 depicts the distribution of the IP address ranges associated to the top organizations during the observation period, as well as the daily utilization of IPs per top organizations. Figure 4.21(a) considers the distribution of IPs itself, whereas 4.21(b) weights each of the server IPs by the number of flows delivered. Despite the high number of $/24$ IP blocks, only few of them are responsible for the largest majority of the flows per org.. In particular, 75% of Akamai flows are served by only one single address range, covering a small number of $/24$ IP blocks. The same observation is valid for Facebook AS and the two Neighbor Operators, with 89%, 91% and 82% of their flows hosted at one single range respectively. Finally, the LO serves almost all the flows from a small range of IPs. Figure 4.21(c) shows the daily usage of these IPs on a single day, considering the number of unique server IPs per hour, per org. The number of active IPs (i.e., IPs serving flows in the corresponding time slot) used by Akamai follows the daily utilization of the network, peaking at the heavy-load time range. Interestingly, the IPs exposed by Facebook AS are constantly active and seem loosely correlated with the network usage. This comes from the fact that Facebook AS servers normally handle all the Facebook dynamic contents [54], which include the user sessions keep-alive.

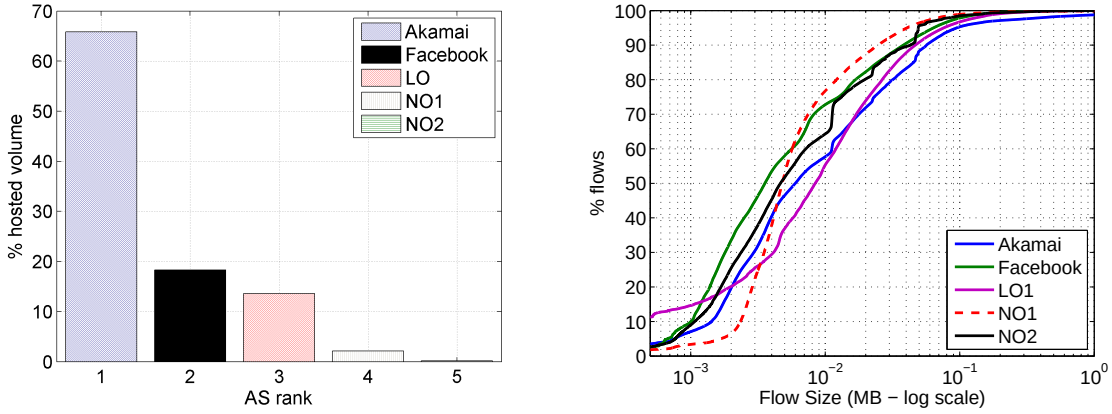
4.5.4. Facebook flow sizes

Figure 4.22 depicts the volume share of Facebook contents hosted by each org./AS, as well as the flow size distributions. Akamai hosts more than 65% of the total volume observed in our traces, followed by Facebook AS itself with about 19%. Comparing the volume shares in Figure 4.22(a) with the flow shares in Figure 4.19(b) evidences a clear distinction on the



(a) IP range of unique server IPs. (b) Fraction of served flows per unique server IP. (c) # hourly unique server IPs per AS.

Figure 4.21.: Distribution of the server IP range per AS. Akamai shows the most diverse IP range, but most of the flows hosted by Akamai come from a single subnet.

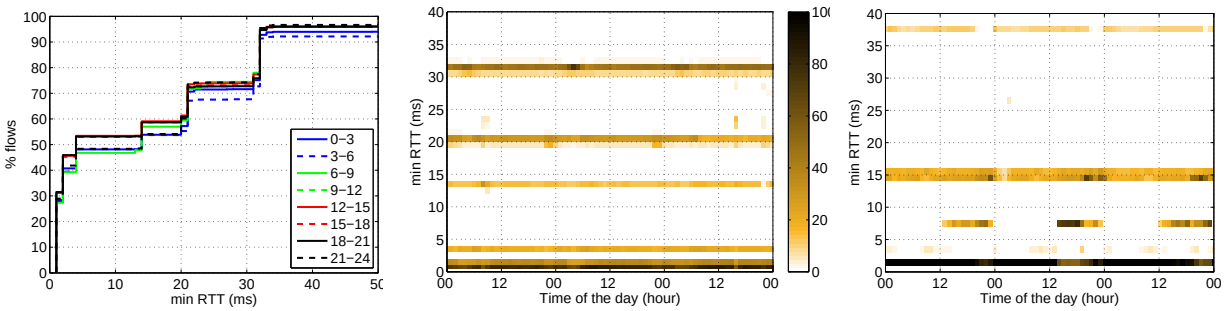


(a) Shares of hosted volume per org./AS. (b) Distribution of flow sizes.

Figure 4.22.: Hosted volume and distribution of flow sizes per organization. Akamai is clearly the leading hosting company for Facebook with about 65% of the total served volume. Akamai is also responsible for serving bigger flows (i.e., static contents and video and pictures) while Facebook AS serves smaller flows (i.e., dynamic contents).

content sizes handled by both Akamai and Facebook AS: while Akamai hosts the bigger flows, Facebook AS serves only a small share of the service content. Indeed, as previously flagged by other studies [54], Akamai serves the static contents of major web services (e.g., photos, songs, videos, etc.), whereas the Facebook AS covers almost exclusively the dynamic contents (e.g., chats, tags, session information, etc.).

To further explore this distinction, Figure 4.22(b) reports the distribution of the flow sizes served per org.. The CDF reveals that Akamai clearly serves bigger flows than Facebook AS. The remaining ASes tend to host bigger flows than Facebook AS, which is coherent with the



(a) minRTT daily dynamics (2013). (b) minRTT dynamics (2013). (c) minRTT dynamics (2012).

Figure 4.23.: Temporal variations of the min RTT to Facebook servers. The temporal patterns in 2012 show a strongly periodic load balancing cycle, focused in a small number of hosting regions. Results in 2013 suggest that Facebook content delivery is becoming more spread and load balancing cycles are less evident. In the heat maps of figures (b) and (c), the darker the color, the bigger the fraction of flows served from the corresponding min RTT value.

fact that ISPs caching is generally done for bigger objects, aiming at reduce the load on the core network.

4.5.5. Content Delivery Temporal Dynamics

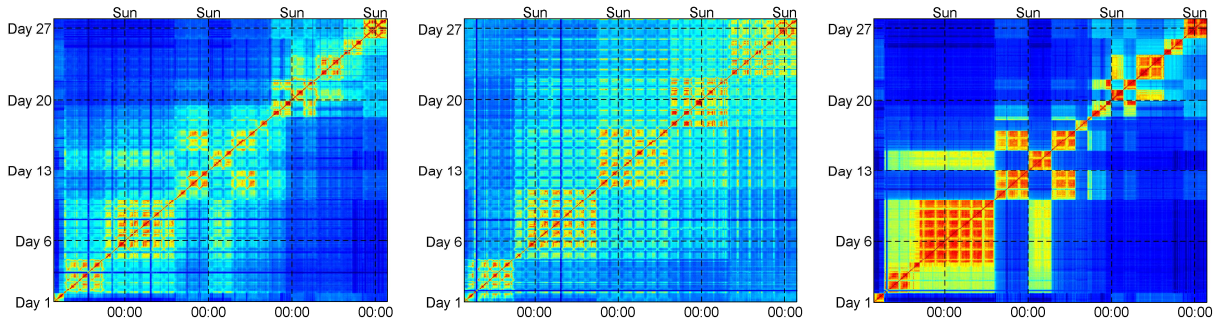
The characterization performed in previous Section only considers the static characteristics of Facebook during the complete duration of the dataset. In this Section we focus on the temporal dynamics of the Facebook content delivery. To start with, we focus on the temporal evolution of the min RTT reported in Figure 4.20. Figure 4.23(a) depicts the temporal variation of the CDF for all the flows and for a complete day, considering a single CDF every three hours period. The CDFs are rather stable during the day, but present some slight variations during the night and early morning. To get a better picture of such dynamics, Figure 4.23(b) depicts the hourly evolution of the min RTT for all the Facebook flows during 3 consecutive days, being the first day the one analyzed in Figure 4.23(a). Each column in the Figure depicts the PDF of the min RTT for all the served flows, using a heat map-like plot (i.e., the darker the color, the more concentrated the PDF in that value). The flagged variations are observed during the first day, with some slight shifts between 6am and 12am from servers at 14ms and 20ms. The heat map also reveals some periodic flow shifts between 9pm and midnight from servers at 20ms, but impacting a small fraction of flows. Figure 4.23(c) presents the same type of heat map for Facebook flows, but considering a dataset of 2012 from the same vantage point. The temporal patterns in 2012 show a much stronger periodic load balancing cycle, focused in a small number of hosting regions at 7ms, 14ms, and 37ms. Comparing the results from 2012 with those in 2013 suggests that Facebook content delivery is becoming more spread in terms of hosting locations, and load balancing cycles are becoming a-priori less marked. However, when deeply analyzing the complete dataset of 2013, conclusions are rather different.

To drill down deeply into this issue, we analyze the dynamics of the content delivery for the complete dataset, spanning 28 consecutive days. Instead of considering the variations of the min RTT, we consider now the variations on the number of flows served by the observed IPs. Changes in the distribution of the number of flows coming from the complete set of 6551 server IPs reflect variations in the way content is accessed and served from the hosting infrastructure observed in our traces. For this analysis, we consider a time granularity of one hour, and therefore compute the distribution of the number of flows provided per server IP in consecutive time slots of one hour, for the complete 28 days. This results in a time-series with a total of $24 \times 28 = 672$ consecutive distributions. To quantify how different are two distributions in the resulting time-series, we use a symmetric and normalized version of the Kullback-Leibler (KL) divergence described at [29].

To visualize the results of the comparison for the complete time span of 28 days, we use the *Temporal Similarity Plot* (TSP), a graphical tool briefly introduced in Chapter 2. The TSP in Figure 4.24 shows the distributions of all the Facebook flows across all the server IP addresses providing Facebook content, over the 28 days. Each plot is a matrix of 672×672 pixels; the color of each pixel $\{i, j\}$ shows how similar are the two distributions at times t_i and t_j: blue represents low similarity, whereas red corresponds to high similarity.

The three TSPs in Figure 4.24 represent the distribution variations for (a) all the observed IPs, (b) the Akamai IPs and (c) the Facebook AS IPs. Let us begin by the TSP for all the observed server IPs in Figure 4.24(a). The regular "tile-wise" texture within periods of 24 hours evidences the presence of daily cycles, in which similar IPs are used to serve a similar number of flows. The lighter zones in these 24 hour periods correspond to the time of the day, whereas the dark blue zones correspond to the night-time periods when the traffic load is low. The low similarity (blue areas) at night (2am-5am) is caused by the low number of served flows, which induces larger statistical fluctuations in the computed distributions. This pattern repeats almost identical for few days, forming multiple macro-blocks around the main diagonal of size ranging from 2 up to 6 days. This suggests that during these periods, the same sets of IPs are used to deliver the flows, with slight variations during the night periods, similarly to what we observed in Figure 4.23(a). However, the analysis of the entire month reveals the presence of a more complex temporal strategy in the (re)usage of the IP address space. For example, there is a reuse of (almost) the same address range between days 10-12 and days 15-16. Interestingly, we observe a sharp discontinuity on days 18-19, as from there on, all the pixels are blue (i.e., all the distributions are different from the past ones).

To get a better understanding of such behaviors, Figures 4.24(b) and 4.24(c) split the analysis for Akamai and Facebook AS IPs only. The Figures reveal a different (re)usage policy of the IPs hosting the contents. In particular, Akamai uses the same servers for 4 to 7 days (see multi-days blocks around the main diagonal). When it changes the used addresses, the shift is not complete as we can observe the macro-blocks slowly fading out over time. This suggests a rotation policy of the address space of Akamai, on a time-scale of weeks. However, we cannot prove this conjecture because of the limited duration of the analyzed dataset. On the other hand, Facebook AS does not reveal such a clear temporal allocation policy. It alternates periods of high stability (e.g. between days 4 and 10) with highly dynamic periods (e.g., from day 18 onward). It is interesting noticing that Facebook AS is the responsible for the abrupt change in the distributions observed from the 18th day



(a) All IPs). (b) Akamai. (c) Facebook AS.

Figure 4.24.: TSP of hourly flow count distributions over 28 days for all the observed IPs, Akamai IPs, and Facebook AS IPs. A blue pixel at $\{i, j\}$ means that the distributions at times t_i and t_j are very different, whereas a red pixel corresponds to high similarity.

on, in the TSP of the overall traffic.

Our deeper analysis reveals that Akamai and Facebook AS actually employ periodical rotations of the servers they used to provide the contents, alternating periodic cycles of relatively low dynamics with more abrupt changes, especially as observed for the case of Facebook AS.

4.6. An Instant Messaging System: WhatsApp

The last section of this Chapter is dedicated to WhatsApp, the leading Instant Messaging (IM) service. WhatsApp is a cross-platform mobile application which allows users worldwide to instantly exchange text messages and multimedia contents such as photos, audio and videos. It currently handles more than 64 billion messages per day, including 700 million photos and 100 million videos [85]. With half a billion of active users, it has become the fastest-growing company in history in terms of users [86]. Such an astonishing popularity does not only have a major impact on the traditional SMS/MMS business, but might also have a remarked impact on the traffic, especially due to the sharing of multimedia messages.

WhatsApp is a relatively new service, and its study has been so far quite limited. Some recent papers have partially addressed the characterization of the WhatsApp traffic [87, 88], but using very limited datasets (i.e., no more than 50 devices) and considering an energy-consumption perspective, which means that there is no literature on the characterization of WhatsApp from the network perspective. The study of this service provided for the first time in this Section follows a similar approach of the analysis of the other services showed before. It should be noted, however, that this case slightly differs from previous ones, as WhatsApp is not a Web service. For this reason and for the use of encryption, the characterization of such service requires more effort, starting from the classification and extraction of relevant flows from the dataset. This is, in fact, the perfect showcase for the need of a more sophisticated classification scheme which also considers DNS analysis to

address encrypted protocols beyond HTTP (cfr., Section 3.4.3).

In this section we provide a large-scale characterization of the WhatsApp service. By analyzing a week of cellular traffic flows collected in February 2014 at the same Vantage Point mentioned in previous sections (cfr. Table 4.10), we shed light on the WhatsApp hosting network architecture, the characteristics of the generated traffic, and the performance of media transfers, specially as perceived by the end users. As WhatsApp runs on top of encrypted connections, our measurements are complemented with a dissection of the WhatsApp protocol through hybrid measurements, enabling a subsequent passive monitoring at the large-scale. In addition, due to its large worldwide popularity, the WhatsApp dataset is augmented with geo-distributed DNS active measurements using more than 600 RIPE Atlas boxes distributed around the globe [8]. As we shall see next, this section it is not just about finding which flows belong to the WhatsApp service and analyze them. Indeed, there are many measurement challenges associated to the characterization of such a service: the data gathering, the processing and the interpretation are already very complex per se, given the number of different measurement sources and datasets.

Our main novel findings are the following: (i) Despite its worldwide popularity, WhatsApp is a fully centralized service hosted by the cloud provider SoftLayer at servers located in the US. (ii) While the application is mainly used as a text-messaging service in terms of transmitted flows (more than 93%), video-sharing accounts for about 36% of the exchanged volume in uplink and downlink, and photo-sharing/audio-messaging for about 38%. (iii) Despite achieving flow download throughputs of 1.5 Mbps on average, about 35% of the total file downloads are potentially badly perceived by users. (iv) Flow duration characteristics depend on the device OS. In particular, different platforms employ different app-level timeouts.

Besides these results, we provide in this section an overview on the worldwide WhatsApp outage reported on February the 22nd of 2014 [89], characterizing the event as observed from the analyzed dataset. The measurements are complemented with external Online Social Networks (OSNs) feeds (Twitter in this case) to verify that the outage was negatively perceived by the users, immediately at the time were the event occurred, additionally demonstrating the feasibility of using OSNs data to provide near real-time evidence of user quality impairments in large scale service outages.

4.6.1. Application Overview

WhatsApp uses encrypted communications, therefore the first step to analyze its functioning in the wild is to better understand its inner working. To this end, we rely on the manual

network type	cellular 3G
monitoring system	METAWIN at Gn interface
ticket type	DNS tickets and flow-level traces
length	7 days (Mon - Sun)
time	Q1 2014

Table 4.10.: Dataset used for characterization of WhatsApp

domain	protocol (port)	type
cX,eX,dX	XMPP (5222, 443)	chat & control
mmiXYZ,mmsXYZ	HTTPS (443)	media (photo/audio)
mmvXYZ	HTTPS (443)	media (video)

Table 4.11.: Third level domain names used by whatsapp.net and communication types.

inspection of hybrid measurements. We actively generate WhatsApp text and media flows at end devices (both Android and iOS), and passively observe them at two instrumented access gateways. We especially paid attention to the DNS traffic generated by the devices.

WhatsApp uses a customized version of the open eXtensible Messaging and Presence Protocol (XMPP) [90]. XMPP is a protocol for message oriented communications based on XML. Not surprising, our measurements revealed that WhatsApp servers are associated to the domain names whatsapp.net (for supporting the service) and whatsapp.com (for the company website). As indicated in Table 4.11, different third level domain names are used to handle different types of traffic (control, text messages, and multimedia messages). When the client application starts, it contacts a *messaging* or *chat server* {e|c|d}X.whatsapp.net listening on port 5222, where X is an integer changing for load balancing. This port is assigned by IANA to clear-text XMPP sessions. Nevertheless, the connection is SSL-encrypted. This connection is used for text messages as well as control channel, and is kept up while the application is active or in background. If the connection is dropped, a new one with the same or another messaging server is immediately re-established. In case the application client is not running, the message notification is delivered through the OS push APIs.

The application also offers the capability of multimedia contents transfer, including photos, audio and video. Transfers are managed by HTTPS *multimedia (mm) servers* listening on port 443. Those servers are associated to different domain names depending on their specific task: mmsXYZ.whatsapp.net and mmiXYZ.whatsapp.net are both used for audio and photo transfers, while mmvXYZ.whatsapp.net are exclusively reserved for videos. For each object, a dedicated TLS-encrypted connection towards a *mm server* is established. Uploads are started immediately, while downloads of large objects need to be manually triggered by the receiving user to avoid undesired traffic. These servers do not perform any transcoding. As we shall see in next, the two server classes have very different network footprints. While connections to chat servers are characterized by low data-rate and long duration (specially due to the control messages), media transfers are carried by short and heavy flows.

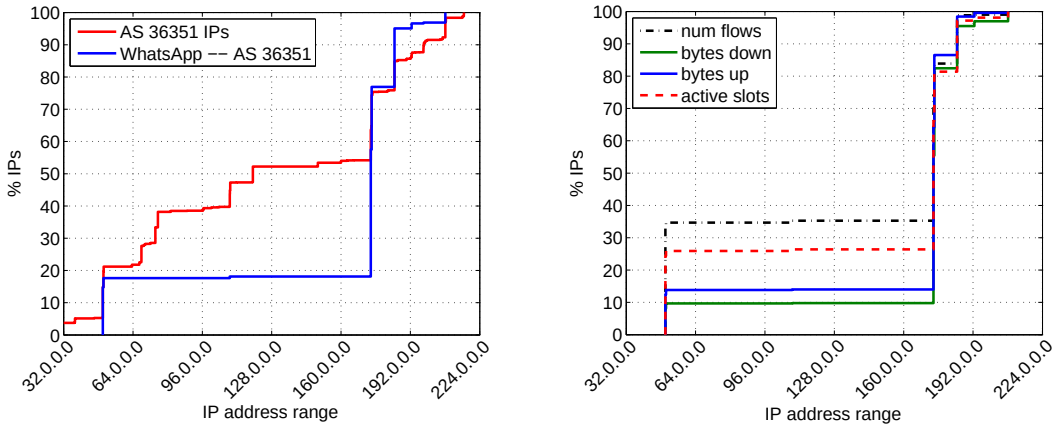
4.6.2. Hosting Infrastructure

The first part of the study focuses on discovering where the servers are located. For doing so, we rely on the analysis of a complete week of WhatsApp traffic traces, consisting of more than 150 million anonymized flows collected at the usual Vantage Point in a mobile network, from 18/02 till 25/02, by the METAWIN monitoring system and analyzed through DBStream. In the following analysis, volume and flow counts are normalized to preserve business privacy, and time-series are constructed with 10-min time slots resolution.

WhatsApp communications are encrypted, thus we need to rely on the DNS-based ap-

Service/AS	# IPs	#/24	#/16	#/8
WhatsApp	386	51	30	24
SoftLayer (AS 36351)	1,364,480	5330	106	42

Table 4.12.: Number of server IPs and prefixes used by WhatsApp.



(a) WhatsApp/SoftLayer IP ranges.

(b) Weighted IP ranges.

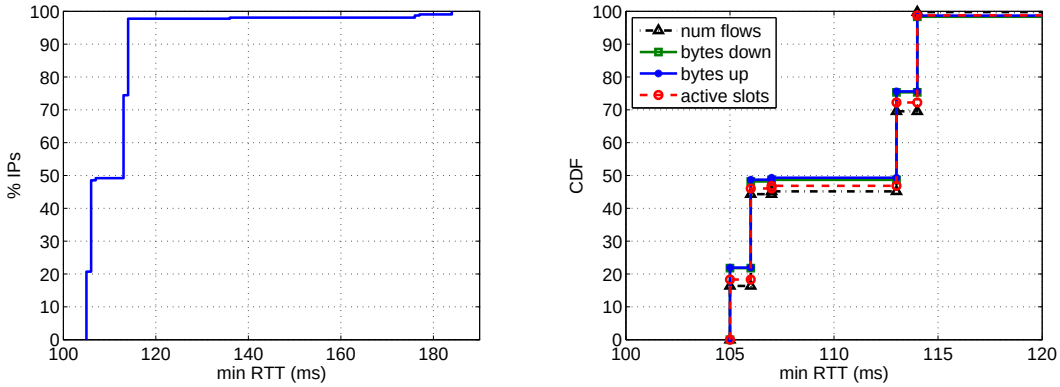
Figure 4.25.: Ranges of IPs hosting WhatsApp. The range of server IPs is highly distributed, covering 51 different /24 prefixes and 24 /8 ones, however, when weighting this distribution by flow number, the majority of the traffic corresponds to IPs falling in 3 /16 ranges. The range 50.22.225.0/24 captures a main share.

proach illustrated in Section 3.4.3 (HTTPTag2) for extracting the relevant flows from the dataset.

The complete one-week server IP mappings resulted in a total of 386 IPs identified as hosting the service, belonging to a single AS called SoftLayer (AS number 36351) [5]. To avoid biased conclusions about the set of identified IPs from a single vantage point, we performed an active measurements campaign using the RIPE Atlas measurement network [8], where we analyzed which IPs were obtained when resolving the same FQDNs from 600 different boxes around the globe during multiple days. These active measurements confirmed that the same set of IPs is always replied, regardless of the geographical location of the requester. SoftLayer is a US-based cloud infrastructure provider consisting of 13 data centers and 17 Points of Presence (PoPs) distributed worldwide. Using MaxMind geoloacalization capabilities, we observed that despite its geographical distribution, WhatsApp traffic is handled mainly by the data centers in Dallas and Houston. Given that the city-location accuracy of public GeoIP databases such as MaxMind is questionable [27], we confirmed through traceroutes and active RTT measurements that the servers are indeed located in the US.

Table 4.12 reports the different number of prefixes covered by the identified IPs in SoftLayer. Note that we consider different netmasks (e.g., /24, /16, /8) for simple counting

[5]SoftLayer: Cloud Servers, in `http://www.softlayer.com`



(a) min RTT per WhatsApp IP.

(b) Weighted min RTTs.

Figure 4.26.: min RTT distribution of WhatsApp server IPs. The distribution presents some clear steps indicating the existence of different data centers or hosting locations.

and aggregation purposes, i.e., we do not claim that the prefixes are fully covered/own by the ASes. The range of server IPs is highly distributed, covering 51 different $/24$ prefixes and 24 $/8$ ones. The table additionally shows the total number of SoftLayer IPv4 IPs. Figure 4.25(a) shows the intersection of both IP address ranges. As depicted in Figure 4.25(b), when weighting the IP ranges by volume, the majority of the traffic corresponds to IPs falling in 3 /16 ranges. However, in terms of flows and activity (measured as 10-min active slots), the range 50.22.225.0/24 captures a main share.

To complement the picture of the servers location, we again investigate the distance to the vantage point in terms of RTTs, analyzing the minimum RTT values, obtained from active ping measurements, as the passive dataset in use do not include this information. Figure 4.26 plots the distribution of the minimum RTT to (a) all the server IPs hosting WhatsApp, and (b) the same min RTT values, weighted by the previously considered 4 features (i.e., flows, active slots, and traffic volumes) to get a better understanding of the traffic sources. The distribution presents some clear steps indicating the existence of different data centers or hosting locations. The min RTT is always bigger than 100ms, confirming that WhatsApp servers are located outside Europe, where our vantage point is located. Figure 4.26(b) shows that the service is evenly handled between two different yet potentially very close locations at about 106 ms and 114 ms, which is compatible with our previous findings.

To further understand how the hosting infrastructure of WhatsApp is structured, Figure 4.27 depicts the distribution of server IPs over the same previous 4 features. The Figures additionally depict chat and multimedia servers to discriminate their roles. Regarding (a) number of flows and (b) active time slots, we clearly observe how chat servers handle the biggest share of the flows, with a highly active set of server IPs. On the contrary, multimedia servers are much less active and handle a limited share of flows. In terms of volume, the picture is completely the opposite when considering traffic volumes in (c) downlink and (d) uplink directions.

Figure 4.28 shows the dynamics of WhatsApp for 3 consecutive days, including the number of active server IPs, the fraction of flows and traffic volume shares, discriminating by chat and mm traffic. The mm category is further split into photos/audio (mmi and mms) and video

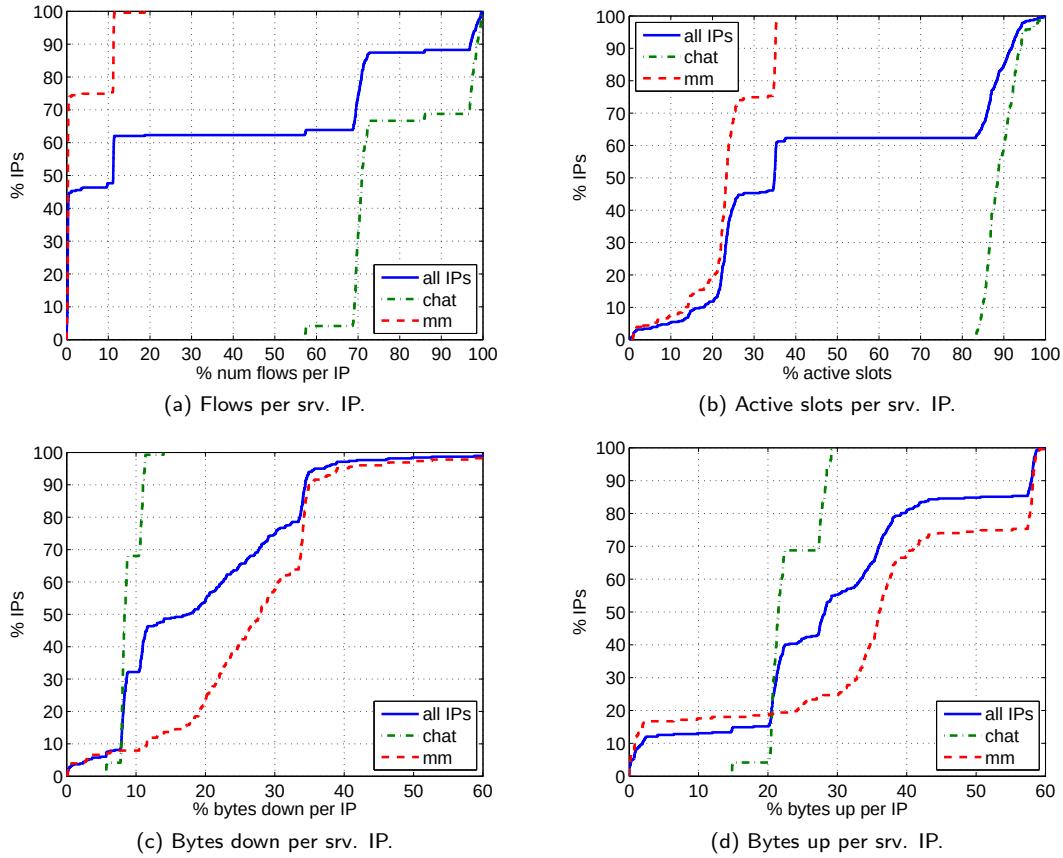
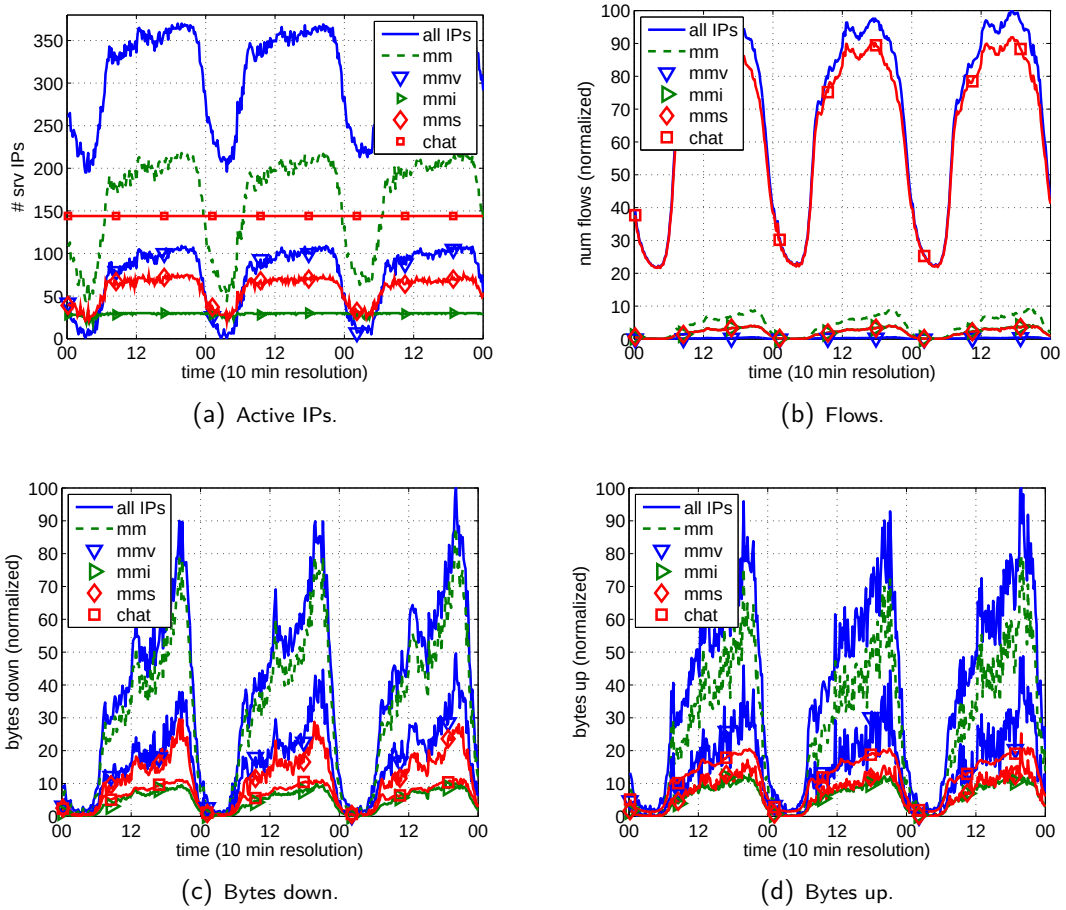


Figure 4.27.: WhatsApp server IPs in terms of volume, flows, and activity shares. As expected, multimedia and chat flows have very different characteristics.

(mmv). The time-series present a clear night/day pattern with two daily peaks at noon and 8pm. Figure 4.28(a) indicates that more than 350 IPs serve WhatsApp flows during peak hours. Note that no less than 200 IPs are active even in the lowest load hours. When analyzing the active IPs per traffic type, we see how chat servers are constantly active, as they keep the state of active devices to achieve an efficient and fast push of the messages to the device. Figure 4.28(b) shows the flow count shares, revealing how chat flows are clearly dominating. Once again we stop in the mmi and mms servers, which seem to always handle the same share of flows, suggesting that both space names are used as a mean to balance the load in terms of photos and audio messages. Finally, Figures 4.28(c) and 4.28(d) reveal that even if the mm volume is higher than the chat volume, the latter is comparable to the photos and audio messaging volume, specially in the uplink. Tab. 4.13 summarizes these shares of flows and traffic volume. The reader should note that our dataset does not include flows transmitted over WiFi, thus some of these results might be biased due to users potentially using WiFi for large file transfers. We are currently analyzing this potential bias as part of our ongoing work, and our first results confirm that our observations are still valid.

As a conclusion, our measurements confirmed that WhatsApp is a centralized and fully US-based service. This is likely to change in the near future after Facebook's WhatsApp



(a) Active IPs. (b) Flows.

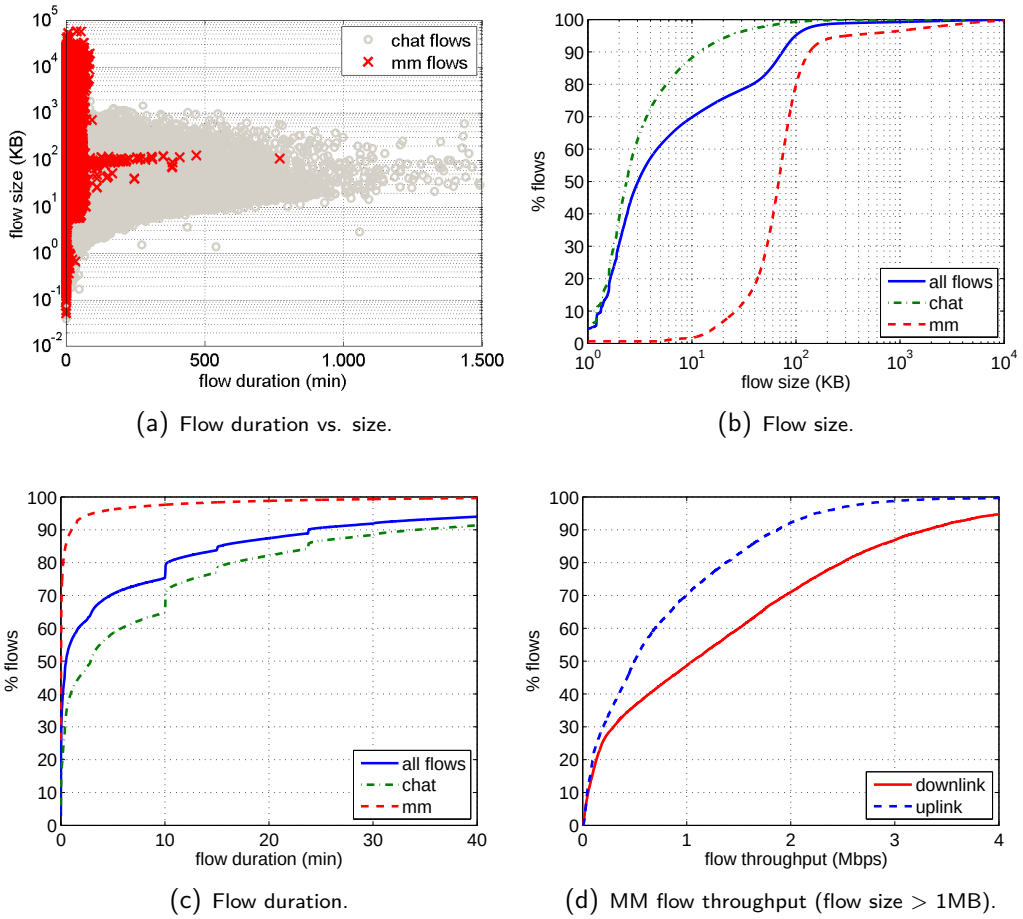
(c) Bytes down. (d) Bytes up.

Figure 4.28.: WhatsApp server IPs dynamics over 3 consecutive days. More than 350 IPs serve WhatsApp during peak hours. Chat servers are constantly active to keep the state of devices.

features	chat	mm	mmv	mmi	mms
# $\text{bytes}_{\text{down}}$	16.6%	83.0%	38.8%	12.8%	29.8%
# bytes_{up}	29.5%	70.2%	35.2%	15.0%	17.9%
# flows	93.4%	6.2%	0.3%	2.9%	2.9%
$\frac{\#\ \text{bytes}_{\text{down}}}{\#\ \text{bytes}_{\text{down+up}}}$	60.6%	76.3%	75.1%	70.0%	81.9%

Table 4.13.: Volume and flows per traffic category.

acquisition. As for now, all messages among users outside the US are routed through the core network. Being Brazil, India, Mexico and Russia the fastest growing countries in terms of users [85], such a centralized hosting infrastructure is likely to become a problematic



(a) Flow duration vs. size.

(b) Flow size.

(c) Flow duration.

(d) MM flow throughput (flow size > 1MB).

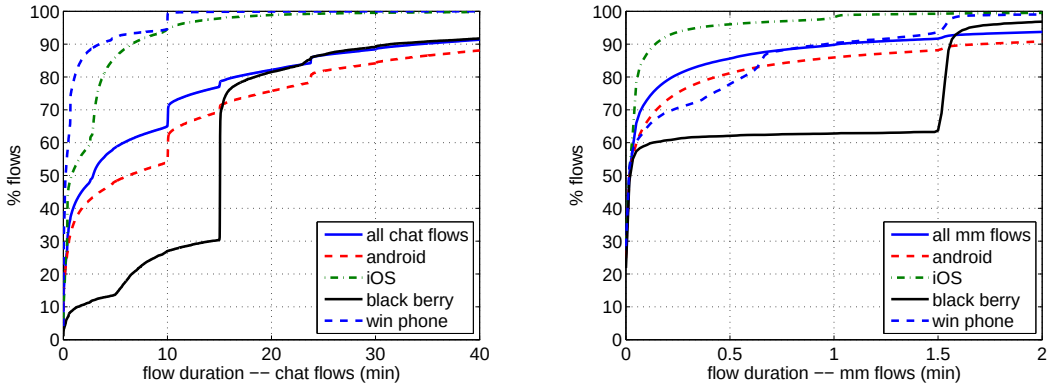
Figure 4.29.: WhatsApp flow characteristics and performance.

bottleneck. Indeed, we will see that the high latencies to US servers are a potential cause of bad QoE for users downloading multimedia files, due to an increased download time and a reduced TCP throughput.

4.6.3. Flow Characteristics

We study now the characteristics of the WhatsApp traffic in terms of size and duration. Additionally, we evaluate the performance of the service, computing the transfer throughputs as the Key Performance Indicator (KPI). Flow durations are measured with a coarse-grained resolution of one second (this is a limitation of the monitoring system, given the large amount of processed traffic), considering the time-stamps of the first and the last packet of a standard 5-tuple measured flow (note that flows are unidirectional) and adaptive flow time-outs, see [6] for additional details. Flow throughput is estimated as the ratio between the total transferred bytes and the flow duration. Note that given the one second resolution, throughput values are somehow an underestimate of the real throughput. Still, the obtained results about flow duration allows us to claim that the absolute errors are marginal.

Figure 4.29(a) shows a scatter plot reporting the flow duration vs. the flow size, discrimi-

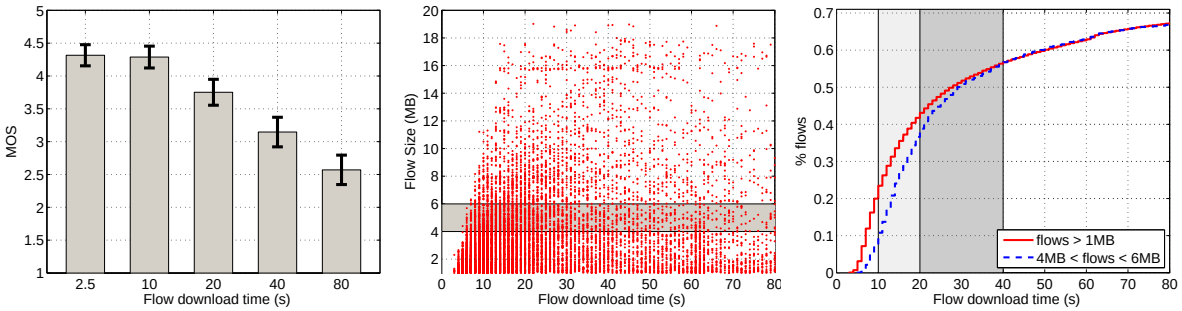


(a) chat flows. (b) mm flows.

Figure 4.30.: Flow duration per different OS. The steps in the distributions are an evidence of different time-outs imposed by the OSes.

nating by chat and mm flows. Whereas mm messages are sent over dedicated connections, resulting in short-lived flows, text messages are sent over the same connection used for control data, resulting in much longer flows. For example, some chat flows are active for as much as 62 hours. The protrusion at around 100KB is due to the fact that the client perform compression of images and most of media flows are close to that size. Fig. 4.29(b) indicates that more than 50% of the mm flows are bigger than 70 KB, with an average flow size of 225 KB. More than 90% of the chat flows are smaller than 10 KB, with an average size of 6.7 KB. In terms of duration, Figure 4.29(c) shows that more than 90% of the mm flows last less than 1 min (mean duration of 1.8 min), whereas chat flows last on average as much as 17 minutes. The flow duration CDF additionally reveals some clear steps at exactly 10, 15 and 24 minutes, suggesting the usage of an application time-out to terminate long idle connections. This behavior is actually dictated by the operating system of the device. To better understand it, we performed a device OS classification based on manual labeling of each device based on its IMEI, covering more than 90% of the observed flows. Note that the device IMEI is not contained in the WhatsApp messages, but comes from other monitoring sources in METAWIN. Figure 4.30(a) splits the analysis of the chat flow duration per device OS. The Figure clearly shows that the aforementioned time-out is mainly OS-dependent, as different platforms show different values. Three different time-outs are visible for Android devices at 10, 15 and 24 mins; iOS uses a very short time-out of 3 mins, BlackBerry devices have 15 mins. long time-outs, whereas Windows Mobile phones favor 10 mins. time-outs. On the contrary, in the case of mm flows in Figure 4.30(b), all the different OS show a similar behavior, with the exception of BlackBerry and Windows Phone, using a 90 secs. time-out. These observations might have a major impact on the performance of the Radio Access Network, due to different OS synchronization times and uneven resources reservation requests. Indeed, it has been recently shown that applications that provide continuous online presence such as WhatsApp can generate a significant burden on the signaling plane in cellular networks [88].

Considering flow throughput, Figure 4.29(d) depicts the uplink and downlink throughputs for flows bigger than 1 MB. This filtering is performed as a means to improve the throughput



(a) WhatsApp overall quality. (b) Flow size vs. download time. (c) Flow download time distribution.

Figure 4.31.: QoE in WhatsApp, considering flows bigger than 1MB. According to QoE models obtained in lab experiments, 35% of WhatsApp multimedia transmission flows seen at our VP are potentially badly received by end users.

estimations. A-priori, one might expect that the long RTTs involved in the communications to the US servers might heavily impact the achieved performance. This is confirmed for about 30% of the transmitted flows, which achieve a throughput smaller than 250 kbps. However, higher throughputs are obtained for the largest shares of flows, achieving an average per flow downlink/uplink throughput of 1.5 Mbps/800 kbps. Still, as we show next, a big share of the file downloads can actually result in a very poor quality of experience for the users.

4.6.4. Quality of Experience in WhatsApp

In the previous Section we considered the transfer throughput as the main KPI reflecting service performance. However, in order to better understand the impacts of transfer throughputs on the experience of the users, we performed a QoE-based study of WhatsApp, relying on subjective QoE tests performed in the lab, following well defined standards for realizing the tests and analyzing the results [91, 92]. In a nutshell, 50 participants (45%/55% male/female, 23 average age, 60%/40% students/employees) provided their feedback in terms of Mean Opinion Scores (MOSs), reflecting their experienced quality while using WhatsApp for transferring video and music files. The study consisted of users receiving a multimedia file of 5MB to download on their smartphones as a WhatsApp shared file. Different network conditions were emulated by connecting the phones to a network emulator, introducing different download throughput profiles via traffic shaping. At the end of each download, the user rates the overall quality in a 1-to-5 MOS scale, where 5 means excellent experience and 1 means a very bad one. Note that the file size of 5MB has a clear motivation behind: MP3 music files and short videos have a similar size. While it is clear that the 5MB flow size reflects only a fraction of the total flows (as depicted in Figure 4.29(b)), the performed study permits to have some rough ideas of what the users perceive of the service in terms of quality in this case. A deeper WhatsApp QoE-based study is part of our current work.

Figure 4.31(a) shows the QoE results for different download throughput values, translated into waiting times. Download time is in fact the most relevant feature as perceived by the user when analyzing file transfers [93], as this is directly linked to anxiety and satisfaction. The Figure shows that users tolerate transfers of up to 20s long with a good overall experi-

ence, whereas transfers lasting more than 80s are considered as very bad quality. A threshold of about 40s permits to approximately discriminate between good and bad experience. Figure 4.31(b) plots the Flow Size vs. the Flow Download Time (FDT) for the large-scale dataset, considering only flows bigger than 1MB. If we focus on the range of flows with sizes around 5MB, we see that while the majority of the flows have a FDT below 40s, there are many downloads which highly exceed this threshold. Indeed, Figure 4.31(c) shows the distribution of the FDTs, both for all the flows with size between 4MB and 6MB, as well as for all the flows bigger than 1MB. From these CDFs, one can say that almost 40% of the WhatsApp downloads with size between 4MB and 6MB have a FDT lower than 20s, resulting in good user experience. About 60% still result in an acceptable quality, and about 35% are potentially badly or very badly perceived. Finally, if we now assume that users are generally non experts and that file sizes are not taken into account into their quality expectations when downloading a video or a song through WhatsApp, we could say that similar results are observed for the complete dataset of downloaded flows bigger than 1MB. Of course this last observation is rather controversial, but still presents some notions on the experience of the end users. Concluding the analysis of WhatsApp, we see that the architectural design of WhatsApp, with servers centralized in the US, might actually have an impact on the experience of the users.

4.7. Summary

In this Chapter 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 organization and the way services are delivered to the end-users.

Our main source of information has been passive measurements captured at two different Vantage Points collected during several campaigns conducted over 2013 and 2014, imported in the DBStream data-warehouse system and classified using the techniques described in Chapter 3. Additionally, we complemented our datasets with geo-IP information provided by MaxMind and both local and geo-distributed active measurements (e.g., ping and RIPE Atlas UDMs).

We have seen that Internet traffic is largely dominated by few big players with very different approaches in deploying their provisioning systems. Among our finding, we have shown how dynamic and distributed are current major CDN players, like Google for YouTube, 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 and services, like Limelight and WhatsApp, reflecting a different architecture philosophies.

All the results presented in this chapter refer to normal operations of the considered services. With the acquired knowledge, we can now move on to the next Chapter, where we will study their anomalous behaviors.

5. Large-scale Network Anomalies

Notice of adoption from previous publications

Parts of the contents of this Chapter have been published in the following papers:

[P5] **P. Fiadino**, A. D'Alconzo, P. Casas, "Characterizing Web Services Provisioning via CDNs: The Case of Facebook", in *The 5th International Workshop on Traffic Analysis and Characterization (TRAC2014)*, 2014.

[P6] A. D'Alconzo, P. Casas, **P. Fiadino**, A. Bär, A. Finamore, "Who to Blame when YouTube is not Working? Detecting Anomalies in CDN-Provisioned Services", in *The 5th International Workshop on Traffic Analysis and Characterization (TRAC2014)*, 2014.

[P8] P. Casas, A. D'Alconzo, **P. Fiadino**, A. Bär, A. Finamore, "On the Analysis of QoE-based Performance Degradation in YouTube Traffic", in *10th International Conference on Network and Service Management (CNSM2014)*, 2014.

[P10] P. Casas, A. D'Alconzo, **P. Fiadino**, A. Bär, A. Finamore, T. Zseby, "When YouTube doesn't Work – Analysis of QoE-relevant Degradation in Google CDN Traffic", in *IEEE Transactions on Network and Service Management*, vol. 11, no. 4, 2014.

[P11] **P. Fiadino**, A. D'Alconzo, A. Bär, A. Finamore, and P. Casas, "On the Detection of Network Traffic Anomalies in Content Delivery Network Services", in *Teletraffic Congress (ITC), 2014 26th International*, 2014.

[P12] M. Schiavone, P. Romirer-Maierhofer, **P. Fiadino**, P. Casas, "Diagnosing Device-Specific Anomalies in Cellular Networks", in *ACM CoNEXT Student Workshop*, 2014.

[P13] **P. Fiadino**, M. Schiavone, P. Casas, "Vivisecting WhatsApp through Large-Scale Measurements in Mobile Networks", extended abstract in *ACM SIGCOMM*, 2014.

[P14] **P. Fiadino**, M. Schiavone, P. Casas, "Vivisecting WhatsApp in Cellular Networks: Servers, Flows, and Quality of Experience", *in Traffic Monitoring and Analysis (TMA 2015)*, 2015.

[P15] **P. Fiadino**, P. Casas, M. Schiavone, A. D'Alconzo, "Online Social Networks Anatomy: on the Analysis of Facebook and WhatsApp in Cellular Networks", *in IFIP Networking 2015*, 2015.

The author of this thesis provided major contribution to the analysis of network traces that produce the results. Arian Bär and Mirko Schiavone supported the interaction with the streaming data-warehouse system DBStream. The author has collaborated with Mirko Schiavone and Peter Romirer-Maierhofer to the investigation of DNS anomalies presented in Section 5.7. The work has been supervised by Dr. Pedro Casas, Dr. Alessandro D'Alconzo and Prof. Tanja Zseby.

5.1. Introduction

In Chapter 4 we have characterized the network footprint of major hosting providers and popular Internet services. By exploiting tools and techniques described in Chapter 2 and 3, we have focused on the complex dynamics behind large scale provisioning systems. Indeed, Internet services rely more and more on highly distributed and sophisticated Content Delivery Networks that have the main role of pushing the contents as close as possible to end users to achieve efficient load balancing and improve overall performance. In some service classes, such as video streaming or social networks with large user bases, this complex provisioning approach is more remarked, as seen, respectively, in the study cases of YouTube and Facebook.

The deep characterization previously provided allows to unravel the complexity of the traffic patterns seen in the passive traces. However, the non-disclosed CDN internal policies, which are space and time variant, impose serious challenges in the modeling of detection and diagnosis systems for unexpected anomalous events. The variability of traffic patterns, in fact, makes it difficult to isolate changes that are the cause of potential negative effects.

In this Chapter we again resort on real traffic passively captured at large operational networks to study real-world anomaly use cases. The anomalies that will be studied have been observed during the characterization of the previously seen services. In particular, we now focus on events affecting Facebook, WhatsApp, YouTube and their provisioning systems. In addition, we will describe a category of device-related anomalies – i.e., anomalies affecting specific sub-populations of devices – which are very popular in cellular networks, but also very harmful.

The study of these events is not only useful for understanding the characteristics of different anomaly types, but also assess which of the involved parties is negatively affected by those. Some anomalies impact the access operator network and cost planning, while some others impact the perceived quality by end users, as we shall see.

The characterization of these use cases has been done by manually checking a number of traffic features and demonstrates the advantages of using different metrics to produce a complete diagnosis of the event. This study completes the requirements and provide an input for finally approaching the goal of this work: the design of algorithms for automatically diagnose network anomalies, which will be the subject of the last two Chapters of this thesis.

5.2. Related Work and Contributions

As we have seen in the related work section of the previous Chapter (cfr. Section 4.2), CDNs have received great attention due to the increased volume of traffic they deliver. However, despite the large literature, only some of these studies have considered the problem of detecting and analyzing anomalies in such CDN scenarios. Worth mentioning is the work of Stoica [94], where authors present a taxonomy of quality problems in video distribution through CDNs, using a large-scale dataset of client-side measurements. Among their findings is the observation that between 30-60% of the quality problems they observed are related to the content provider, the CDN, or the client ISP. Also the work of the Google operations'

team is relevant to us [95], where authors present a framework to diagnose large latency changes in Google's CDNs' delivered traffic. In addition, the work from Mellia's research group on monitoring CDNs builds on the direction we have followed.

In a broader basis for characterizing network anomalies, one of the former papers working on the analysis of anomalies in large scale networks is the one of Lakhina et al. [96], which targeted more general types of anomalies such as network outages, flash crowds, high-rate flows, etc. Unfortunately, besides [94], there has not been much work on producing a comprehensive taxonomy on the types of anomalies observed in current Internet scale services.

A major step in the characterized of anomalies in CDN-based services has been taken by us in our publications [P5, P6, P8, P10-P15], where we present multiple case studies occurred in the popular services previously analyzed. Some very recent work builds on top of our results in this direction; for example, authors in [97] present a system to unveil sudden changes in the YouTube provisioning infrastructure.

Our work is the first to fully characterize a number of anomaly case studies occurring in cellular networks, in addition to fixed-line networks, considering different types of network traces. The reason why this topic remains largely unexplored by the research community must be sough, again, in the lack of data, an issue already mentioned in Chapter 1, and in the undisclosed nature of CDN policies, which makes it difficult to isolate anomalous patterns. The traffic monitoring and analysis research community has a limited access to real-world passive traces collected in large scale operational networks. The time-limited nature of the datasets available prevents researchers from monitoring CDN traffic on the long term to uncover and characterize anomalies.

5.3. Large-scale Changes in Service Provisioning: the case of Akamai and Facebook

In the previous chapter (cfr. Chapter 4.5), we have analyzed the highly distributed provisioning system of Facebook, the most popular social network to date. Recall that Facebook content is delivered through a sophisticated and highly distributed content delivery infrastructure. The big majority of the Facebook content is hosted by the Akamai CDN. Parts of the content are hosted under Facebook's own Autonomous System (AS), split between its headquarters in the USA and Ireland. Finally, an important share of the content is served by the ISPs, which maintain large transparent caches, and may additionally host Akamai servers inside their premises. By using the same dataset of Section 4.5, we unveil unexpected events in how Facebook traffic is served by the involved ASes.

5.3.1. Multi-caches Selection Policies

Let us first show with a simple example the intrinsic multi-caches and daily load balancing policies employed in the delivery of Facebook traffic flows. Fig. 5.1 shows the per-hour distribution (CCDF) of the RTT of the flows carrying Facebook content for a complete

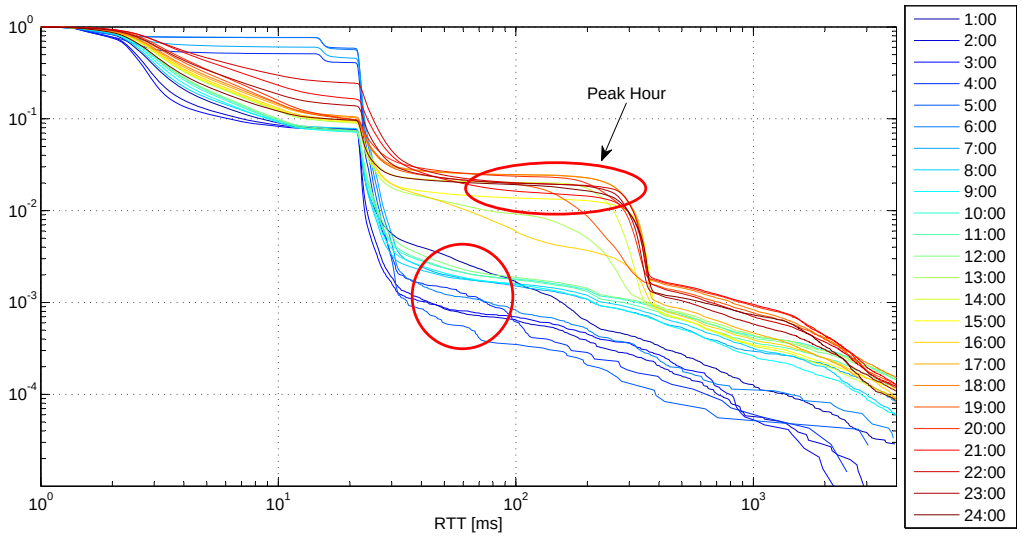


Figure 5.1.: Daily RTT CCDFs for Facebook flows. There is a clear shift on the selected servers between the first and the second half of the day.

day. For each Facebook flow, the RTT is passively computed as the delay between the SYN and the SYNACK packets during the TCP 3-way handshake. Given that the probe is at the Gn interface of a 3G mobile network, the user-side part of the RTT is excluded. The Figure reveals the typical daily patterns of the RTT distributions. The occurrence of "bumps" or knees in the distribution indicates the presence of different caches, located at different propagation distances from the vantage point. In addition, there is a clear change on the selected servers providing the content during the first and the second half of the day, revealing the existence of a time-of-day based load balancing policy.

Let us move to the study of the anomalies. To show some of the unexpected traffic changes caused by the selection of caches serving Facebook, Figure 5.2 depicts the 4-days evolution of the number of flows (obfuscated to protect operator's business sensitive details) and the corresponding number of unique server IPs delivering Facebook content, aggregated in 5-min time bins and split by hosting organization/AS. We include the top-4 organizations in terms of delivered volume, which correspond to Akamai, Facebook AS, the Local Operator (LO), and the most important Neighbor Operator (NO1). The plot also includes another Neighboring Operator we refer to as NO2, which plays a key role in this analysis. The flow share across the 5 organizations remains practically constant during the day. There is a clear daily pattern in the number of active IPs, and it is worth noting how Akamai systematically doubles the number of deployed servers during the peak hours (21:00-23:00), flagged by the dotted rectangles. As expected, Akamai and Facebook AS serve the largest share of Facebook flows. Akamai employs many more servers, and as shown in Fig. 5.3, it hosts the largest flows corresponding to the static Facebook contents, showing the role breakdown through the different organizations.

Figure 5.2 additionally shows the occurrence of four anomalies, identified as A, B, C and D, which break the normal traffic pattern. We clarify to the reader that these events are

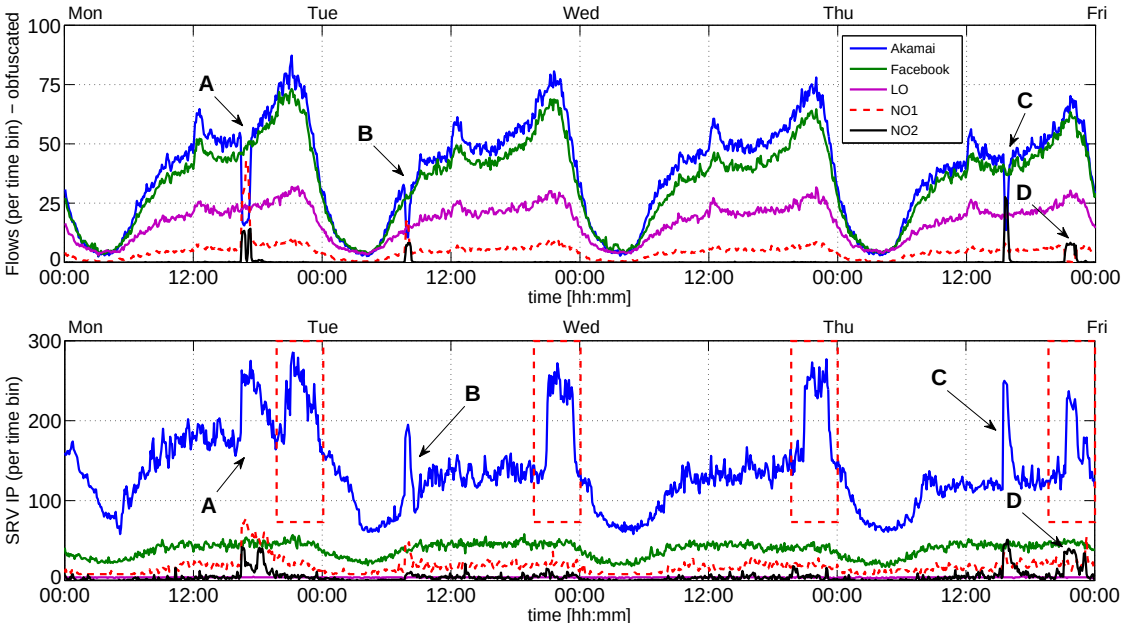


Figure 5.2.: Flow counts (up) and server IPs (down) per AS, 5-min aggregation.

assessed as "unexpected" or anomalous with respect to the behavior observed in our traces, i.e., from the perspective of the ISP hosting the vantage point. In this study we do not have enough data (e.g., from multiple vantage points) to find the root causes of such behaviors, which might be the result of more complex and planned activities by the involved ASes. Anomalies A and B have similar characteristics: even if the number of IPs steeply increases, the number of flows and traffic volume served by Akamai abruptly decreases. The number of flows served from NO1 and NO2 abruptly increase, and so does the number of active IPs in both ASes. This strongly indicates that flows served by Akamai under normal operation (i.e., the majority of the time) are now served by neighboring ISPs. Akamai actually deploys servers inside the ISPs (cfr. Section 4.3), which also explains the synchronized shift of flows. Fig. 5.3 depicts a 12 hours zoom around the events C and D. During the event C, the Akamai drop is again compensated by NO1 and NO2 in terms of volume. However, unlike NO2, there is a limited increase in the number of flows served from NO1, suggesting that the latter takes over the largest flows from Akamai. Event D differs from the previous ones since it does not involve Akamai, and it is characterized by a swap in the number of flows between NO1 and NO2.

We acknowledge that we do not know the ground truth or root causes causing the aforementioned unexpected – from the ISP perspective – cache selection events. A possible cause could be an outage in the Akamai AS or a scheduled maintenance. We did not observe any abrupt variation in the total traffic, throughput, average RTT to the active IPs, nor in the number of erroneous HTTP responses during the events A-D, suggesting that the cache selection did not impact the end-user QoE. However, we argue that these fast and signifi-

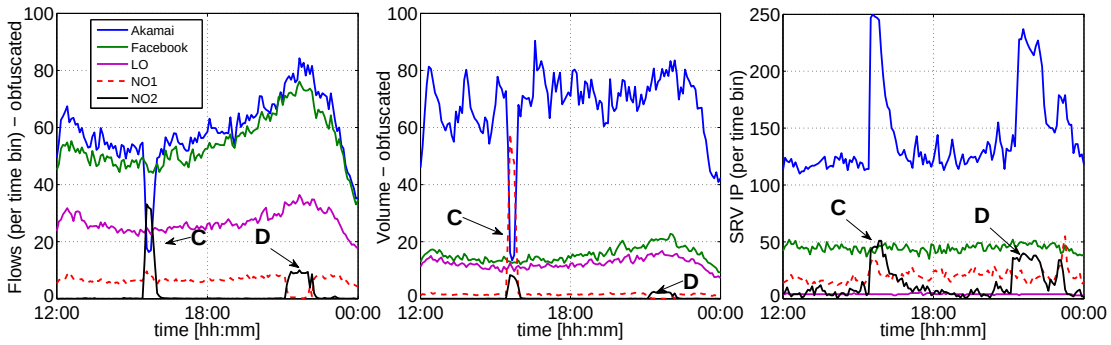


Figure 5.3.: Flow counts, volume and server IPs per AS, for 12 hours.

cant traffic shifts might be highly costly for the LO. Indeed, we verified via traceroutes that Akamai, NO1, and NO2 are neighbors to LO. As reported in the Internet AS-level topology archive[1], the relation between LO and Akamai is peer-to-peer (P2P), whereas the relation between LO and both NO1 and NO2 is customer-to-provider (C2P). In a nutshell, the P2P relation results in no transit costs for the LO for the flows served by Akamai, whereas the C2P relation might represent additional transit costs for the LO for flows coming from NO1 and NO2. For this reason, such events are worth to be automatically detected and analyzed.

5.3.2. Temporal Characteristics of Facebook and Akamai Traffic

To get further insights on the aforementioned anomalies, we investigate the temporal evolution of the probability distributions of the flow counts across IPs serving the Facebook content. The flow counts are computed for each observed server IP, considering different time-scales to enable multi-scale analysis (e.g., from 1' to 60'). The distribution of the flow counts across the server IPs is computed after each time bin. Finally, by comparing the distributions referring to different time intervals through the modified K-L divergence (6.3), we get a direct insight on how the flow load balancing is performed among the IPs of the different organizations. To visualize and quantify the degree of (dis)similarity of a large number of distributions over days and even weeks, we use an ad-hoc graphical tool proposed in [29], referred to as *Temporal Similarity Plot* (TSP). We recall from the previous Chapter that the TSP allows pointing out the presence of temporal patterns and (ir)regularities in distribution time series, by simple graphical inspection. The TSP is a symmetrical checkerboard heat-map like plot, where each point $\{i, j\}$ represents the degree of similarity between the distributions at time bins t_i and t_j.The blue palette represents low similarity values, while reddish colors correspond to high similarity values

Figure 5.4 gives an example of a TSP for the distributions of all the Facebook flows across all the server IPs providing Facebook content, over the complete span of the dataset, on a time-scale of 1 hour.Note the regular "tile-wise" texture within a period of 24 hours, due to the daily cycle. The lighter zones correspond to the day-time periods, whereas the dark blue zones correspond to the night-time periods when the traffic load is low. The

[1]Internet AS-level Topology Archive, `http://irl.cs.ucla.edu/topology/`.

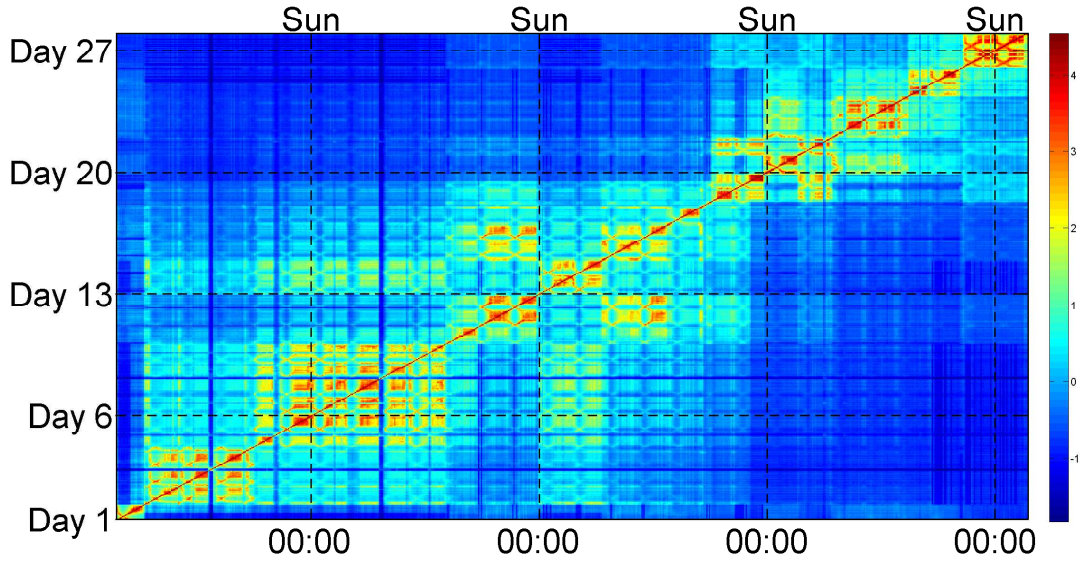
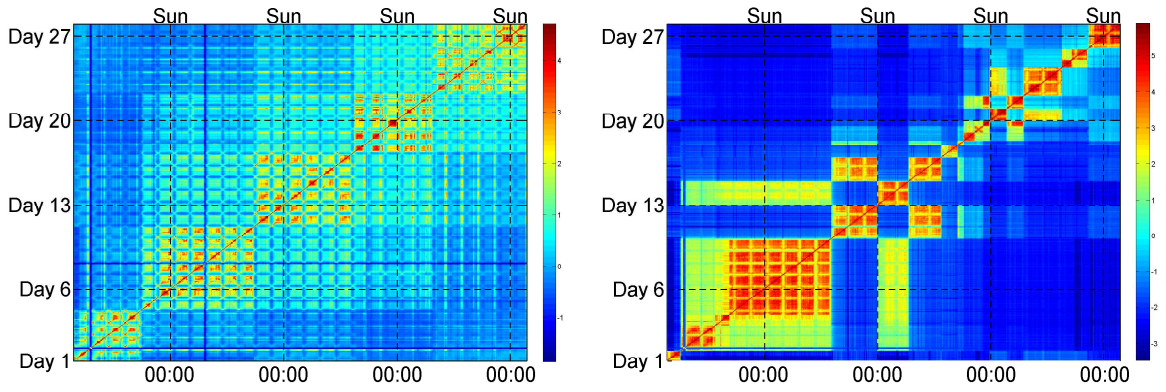


Figure 5.4.: TSP of flow count distributions at 1h time scale, over 28 days.

low similarity at night (02:00-05:00) is caused by the low number of flows, inducing larger statistical fluctuations. This pattern repeats almost identical for a few days, forming multi-days macro-blocks around the main diagonal, of size ranging from 2 up to 6 days. Besides the basic tile-texture, the analysis of the entire observation period reveals the presence of a more complex temporal strategy in the (re)usage of the IP address space. Indeed, it discloses a re-usage of (almost) the same address range between days 4-10 and 14-15, and between days 11-13 and 16-17. Finally, we observe a sharp discontinuity on days 19-20.

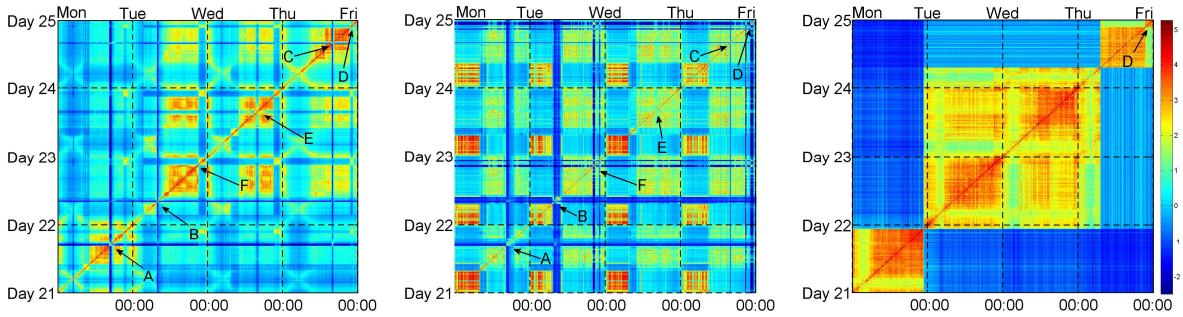
To better understand these behaviors, we separately plot the two main sources of Facebook flows, namely Akamai and the Facebook AS. Comparing Figures 5.5(a) and 5.5(b) against Figure 5.4 shows a very different allocation policy used by the two organizations. Akamai uses the same IPs for 4 to 7 days (see multi-day blocks around the main diagonal). When it changes the IPs the shift is not complete, as we can observe the macro-blocks slowly fading out over time. This suggests a rotation policy of the address space of Akamai on a time-scale longer than a month. However, we cannot prove this conjecture because of the limited duration of the analyzed dataset. Facebook AS does not reveal such a clear temporal allocation policy. It alternates periods of high stability (e.g. between days 4-10) with highly dynamic periods (e.g., from day 19 onward). Note that Facebook AS is responsible for the IP reuse between days 4-10 and 14-15, and between days 11-13 and 16-17, and for the abrupt change on days 19-20, both already identified in Figure 5.4. Finally, NO1 always uses two distinct address sets during the night and the day periods, as depicted in Figure 5.6(b).

We can use the TSPs to identify, by graphical inspection, the aforementioned anomalies in the traffic distributions. Indeed, a transient anomalous event appears in the TSP as a full blue cross centered on the main diagonal, at the time of the event. Figure 5.6 shows the TSPs of the flow counts distributions between days 21 and 24 at a 5 minutes time-scale (i.e., the same period and aggregation depicted in Figure 5.2), for Akamai, NO1, and Facebook AS respectively. The events A, B, and C are clearly visible in the TSPs of Akamai



(a) Akamai (b) Facebook AS

Figure 5.5.: TSP of flow counts distributions at 1h time-scale.



(a) Akamai (b) NO1 (c) Facebook AS

Figure 5.6.: TSP of flow counts distributions at 5' time-scale.

and NO1, and are totally absent from the Facebook AS TSP. These events are also clearly visible in the TSP of NO2 (not reported for space limitations), and are in total accordance with the analysis for the flow counts time-series in Figs. 5.2 and 5.3. Regarding the event D, it is observable in all the TSPs, even though it is completely invisible in the time-series of flow counts and volume of Facebook AS in Figure 5.3. Furthermore, Figs. 5.6(b) and 5.6(a) pinpoint the presence of two more anomalous events in the Akamai and NO1 traffic, namely the events E and F, that are completely invisible in the flow and volume plots. This additionally justifies the usage of probability distribution based approaches for detecting such abnormal events.

5.4. OSN Service Outages: two Close Facebook Events

We devote this section to the analysis of two outages in the Facebook service which occurred at one month of distance one from the other, in September and October 2013 respectively.

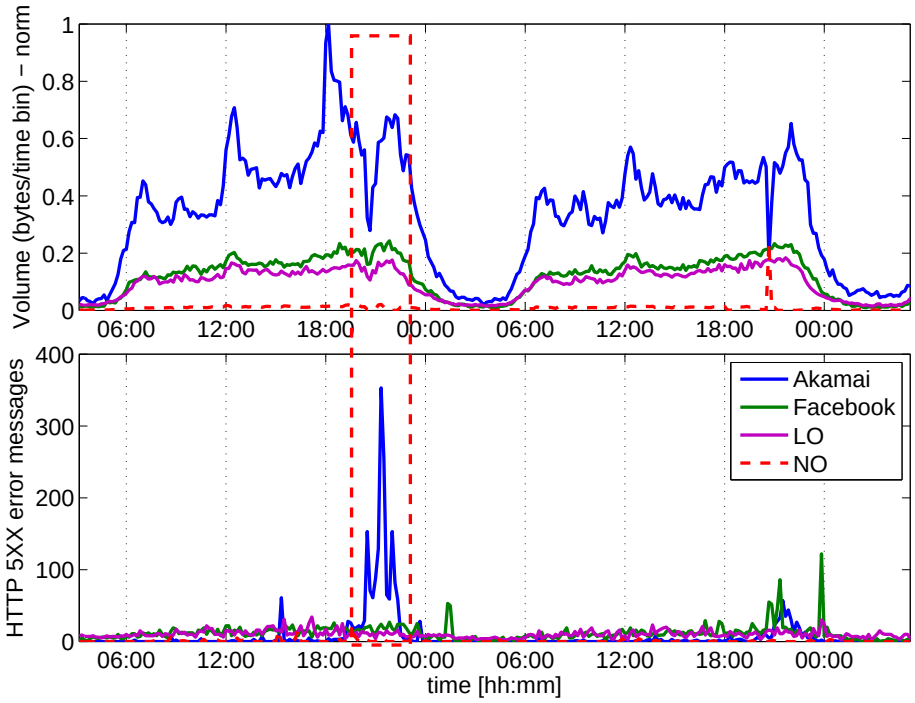


Figure 5.7.: Detection of Facebook outages in Septmeber 2013. (up) Facebook downlink traffic volume per AS and (down) HTTP server error message (e.g. 5XX) counts.

Such outages are not directly linked to the cache selection policies employed by the CDNs serving the content, as they do not involve any shift of traffic among ASes hosting the service. Still, they are related to them, as they may occur at different ASes independently. However, differently from the events previously described, in these two cases the change in the traffic patterns resulted in a partial service unavailability for end-users, as we shall see next.

5.4.1. First Facebook Outage: September

Figure 5.7 depicts the first interesting anomalous event in the Facebook traffic served by Akamai, which we claim corresponds to a large outage in Akamai servers during a time frame of about 2 hours in September 2013. The total volume served by Akamai, Facebook AS and LO abruptly drops during this outage, being Akamai the organization showing the highest change. Different from the events previously analyzed in Figures 5.2 and 5.3, no other organization takes over the dropped traffic, suggesting the occurrence of an outage.

To further understand the root causes of the abrupt drop, Figure 5.7 additionally plots the time series of the count of HTTP server error messages (i.e., 5XX HTTP answers) corresponding to the Facebook HTTP flows served by the aforementioned ASes. The high increase in the counts for Akamai is impressive, meaning that during the volume drop, the HTTP web traffic hosted by Akamai was not available for many users. The increase of the 5XX messages continues for about half an hour after the apparent recovery, flagging some

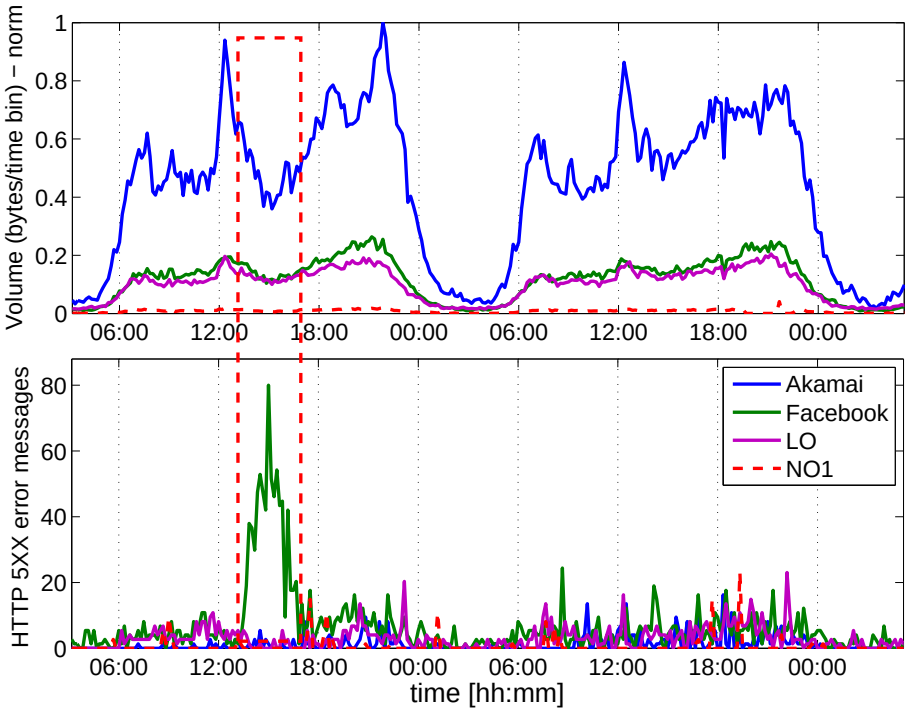


Figure 5.8.: Detection of Facebook outages in October 2013. (up) Facebook downlink traffic volume per AS and (down) HTTP server error message (e.g. 5XX) counts.

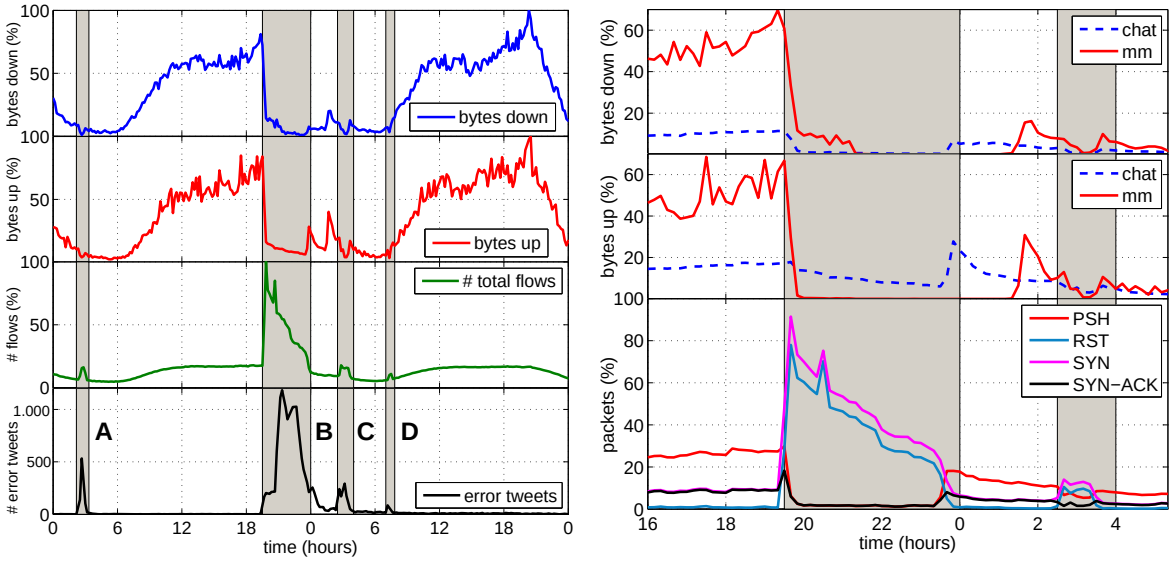
transient effects which might be linked to the re-start of some servers.

Interestingly, there are no noticeable variations in the counts for the other ASes, suggesting that the outage is only part of the Akamai CDN and is not related to the Facebook service itself. As we said before, we do not have any ground truth flagging this outage in the Akamai CDN. However, we also detected an outage of very similar characteristics about one month later, for which we have the ground truth of its occurrence, disclosed in the international press[2].

5.4.2. Second Facebook Outage: October

Figure 5.8 depicts this new outage occurring in October 2013. The drop in the served volume is not as marked as before, and in this case, the increase in the HTTP error message counts occurs for the servers under Facebook AS and not Akamai. However, the characteristics are very similar: a drop in the overall served volume with no other organization taking over, as well as a marked increase in the HTTP error messages counts. According to the press release, this Facebook outage was caused by maintenance issues. As a final statement on the importance of rapidly detecting and diagnosing these types of events we cite directly the press release, which claims that the flagged outage impacted millions of Facebook users on

[2]http://www.theguardian.com/technology/2013/oct/21/facebook-problems-status-updates



(a) Aggregated traffic and OSN feedback.

(b) A zoom-in of the TCP flag counts.

Figure 5.9.: The WhatsApp worldwide outage. During the event, there is a clear drop in traffic volume both downlink and uplink, while the flow counter increases. This happens because end terminals repeatedly try to re-contact the service increasing the number of TCP SYN packets.

more than 3.000 domains. Interestingly for ISPs, the experts behind the press release advise to check the status of large services like Facebook before actually starting a troubleshooting phase on their internal systems.

5.5. IM Service Black-out: the case of WhatsApp

In this section we focus on the analysis of the major WhatsApp worldwide outage reported since its beginning as observed in our traces (cfr. Section 4.6).

The outage occurred in February the 22nd of 2014, and had a strong attention in the media worldwide. The event is not only clearly visible in our passive traces, but can also be correlated with the near real-time user reactions on social networks. Through the online downdetector application [3] we accessed the counts of tweeter feeds containing the keyword "whatsapp", coupled with keywords reflecting service impairments such as "outage", "is down", etc.. We refer to these tweets as *error tweets*.

5.5.1. Black-out at a glimpse

Figure 5.9(a) depicts the time series of the share of bytes exchanged with the WhatsApp servers, the share of flows, as well as the number of error tweets during two consecutive days encompassing the outage. The traffic drastically dropped on the 22nd at around 19:00

[3] Downdetector.com, `http://downdetector.com/`.

CEST (event B), and slowly started recovering after midnight, with some transient anomalous behaviors in the following hours (events C and D). Traffic volumes in both directions did not drop completely to zero but some non-negligible fraction of the traffic was still being exchanged, suggesting an overloading problem of the hosting infrastructure. In terms of number of flows, there is a clear ramp-up on the flow counts. This apparently counterintuitive aspect will be clarified later. Very interestingly, there is an evident correlation between the events B, C and D and the number of WhatsApp-related error tweets. The users reacted on the social network immediately after the beginning of the outage, with the viral effect reaching its highest point after one hour. There is an additional outage event marked as A, which is clearly observable in the error tweet counts and has exactly the same signature of events B, C and D, i.e., a slight drop in the traffic volume and an increase in the flows count, this time characterized by a smaller intensity due to the low utilization of the service during night. As a take away of this social data analysis, one can use such information as an additional ground truth for near real-time detection of QoE-relevant anomalies in popular services such as WhatsApp.

5.5.2. TCP Flags Counters

To better drill-down the anomaly, Figure 5.9(b) depicts a 12-hour zoom-in of the traffic volume trends, split by chat and multimedia traffic (cfr. WhatsApp service characterization in Section 4.6), along with the counters of TCP flags. The bytes down counters show that the residual downlink traffic exchanged during the first part of the anomaly is due to previously queued mm transfers. In fact, while chat servers stopped working, media servers are still up and running at the beginning of the outage. We recall that connections to chat servers are also used for application control, hence they provide links to media contents. If such links have been delivered before the chat outage, the users might still be able to retrieve media objects. The chat traffic in the uplink direction does not drop to zero but slowly fades out, which actually corresponds to control flows trying to re-establish the lost connections. In particular, the TCP flags counters reveal an steeped increase of SYN packets, indicating that devices were repeatedly trying to reconnect after the servers abruptly flashed the connections (RST flags). This suggests that the servers were still reachable, thus the failure occurred at the application layer. The SYN and RST counters decrease gradually, revealing a back-off mechanism of the client application. These connection attempts explain the high increase in the flow counts during events A–D, as well as the persistence of uplink traffic to chat servers. This behavior affected the whole WhatsApp addressing space.

5.6. Quality of Experience Degradation: the case of Youtube

In Section 5.3 we have studied an example of large-scale changes in the provisioning system of an Internet service, with consequent shifts of large amount of traffic from one source to another. In that specific study case, we do not have any evidence that said change impacted

the functionality of the service as perceived by final users. In this section, by describing a similar event, we show how CDN cache selection policies may also have a strong impact on the service quality as experienced by the end users. This is not only a main issue for the end-users, but also for the ISP providing the Internet access to the contents, as customers will in most cases directly blame the ISP for the bad QoE, even if the origin of the problems is located outside its boundaries.

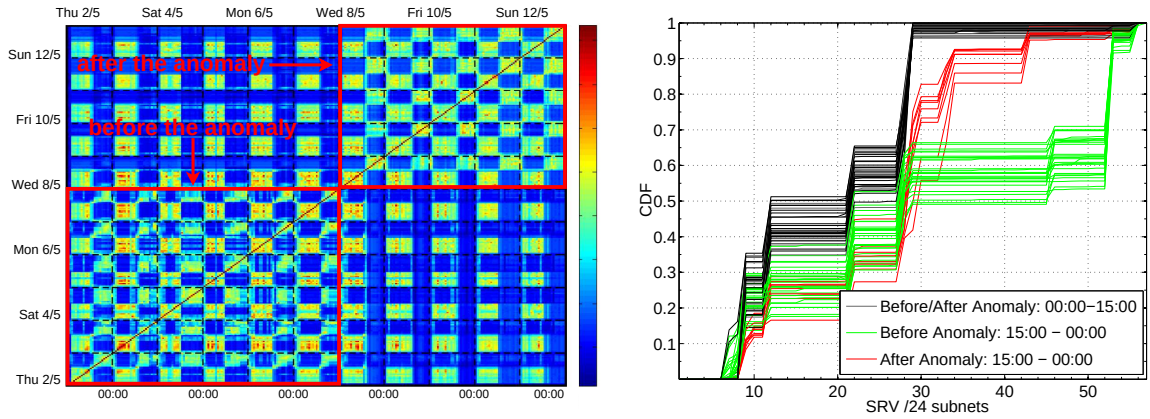
The anomalous event characterized in this section correspond to a real case in which an unexpected cache selection and load balancing policy employed by Google results in an important drop on the average download throughput for the end-users watching YouTube videos. Indeed, conversations with the ISP confirmed that the effect was indeed negatively perceived by the customers, which triggered a complete Root Cause Analysis (RCA) procedure to identify the origins of the problem. As the issue was caused by an unexpected caches selection done by Google, the ISP internal RCA did not identify any problems inside its boundaries. This anomaly is a clear example of how standard diagnostic procedures followed by operators should always be complemented with a verification of the status of the services being accessed by the users, which in many cases are the root of the problems.

The dataset is the same used in Section 4.4 and corresponds to one month of HTTP video streaming flows collected at the fixed-line network of a major European ISP, from April the 15th till May the 14th, 2013.

As reported by the ISP operations team, the anomaly occurs on Wednesday the 8th of May. Fig. 5.10(a) shows the TSP of the video volume served by the different IPs in the dataset, aggregated in $/24$ subnetworks, and using a time-scale of 1 hour. Similar to the Akamai case, we can appreciate a marked daily periodicity behavior in the TSP. Specifically, there are two subnet sets periodically re-used in the first and second half of the day. The TSP clearly reveals that a different subnet set is used during the second half of the day from the 8th of May on, revealing a different cache selection policy. This change is also visible in the CDFs of the per subnet volume depicted in Fig. 5.10(b). Indeed, we can see that the same set of subnets is used between 00:00 and 15:00 before and after the anomaly, whereas the set used between 15:00 and 00:00 changes after the 8th, when the anomaly occurs.

5.6.1. Evidences of QoE Degradation

Despite this detected change in the cache selection policy employed by Google, such a modification does not justify by itself the QoE degradation reported by the ISP. To further investigate this issue, we plot in Figure 5.11 the time series of three different performance indicators related to the YouTube download performance and to the end-user QoE. Figure 5.11(a) depicts the median across all YouTube flows of the download flow throughput during the complete week. There is a normal reduction of the throughput on Monday and Tuesday at peak-load time, between 20:00 and 23:00 UTC. However, from Wednesday on, this drop is much larger, and drops way below the bad QoE threshold $T_{h_1} = 400$ kbps, flagging a potential QoE impact to the users. Figure 5.11(b) plots the entropy of the QoE classes built from thresholds $T_{h_1} = 400$ kbps and $T_{h_2} = 800$ kbps, consisting of bad QoE for flows with average download throughput below T_{h_1}, fair QoE for flows with average download throughput between T_{h_1} and T_{h_2}, and good QoE for flows with average download throughput



(a) TSP of volume per CDN /24 subnet.

(b) Volume per CDN /24 subnet.

Figure 5.10.: YouTube traffic volume distributions per CDN /24 subnets. There is a clear change in the hosting settings, highlighted both by the TSP and the CDF.

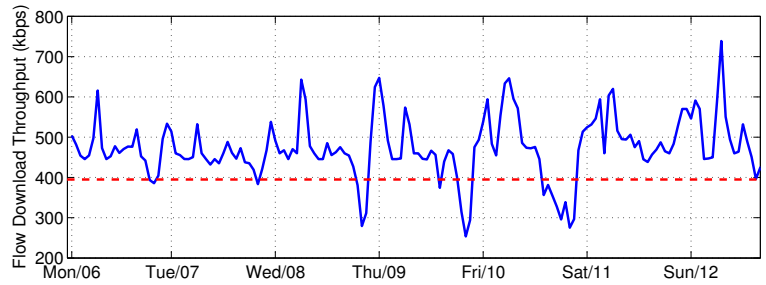
above T_{h_2}. Recall that these thresholds correspond to the QoE mappings presented in Figure 4.17, which only cover 360p videos. Still, as previously depicted in Figure 4.14(a), the largest majority of the videos observed in the dataset corresponds to 360p videos and higher bitrate videos, thus T_{h_1} and T_{h_2} are somehow conservative thresholds, and QoE impairments might be even higher under the proposed QoE classes. The drop in the throughput combined with the marked drop in the time series of the QoE classes entropy actually reveals that a major share of the YouTube videos are falling into the bad QoE class. Finally, Figure 5.11(c) actually confirms that these drops are heavily affecting the user experience, as the time series of the KPI β falls well into the video stallings region, depicted in Figure 4.18 in Section 4.4).

The anomaly can also be statistically detected as a large deviation on the distribution of relevant features, for example, in the distribution of the average download flow throughput. Figure 5.12 depicts the distribution of the average video flows download rate at peak hours, both before and during the anomaly. There is a clear reduction on the video flows download throughput during the anomaly, which results in the aforementioned QoE-relevant impairments. This is actually a powerful detection approach: the idea of using statistical distribution for the detection will be further studied in the next Chapter as a basis for automatic detection of anomalies. In the remainder of this Section, we continue manually investigating other traffic features relying on our domain knowledge.

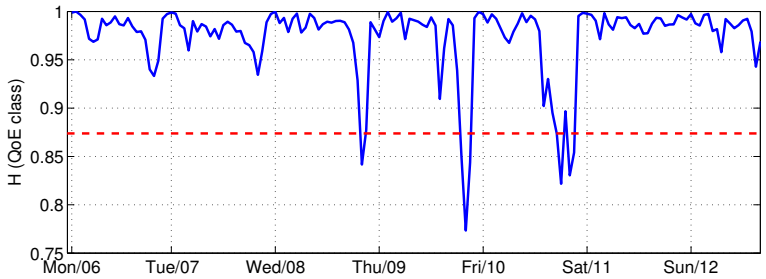
5.6.2. Investigating the Anomaly

As we said before, in this case study we exclude potential problems at the end devices or home networks, as we are targeting a large-scale anomaly, impacting a large share of the monitored customers. In addition, we recall that the ISP internal RCA did not identify any problems inside its boundaries, so we also exclude the ISP network from the analysis. Therefore, we shall only focus on the YouTube servers and on the download paths performance.

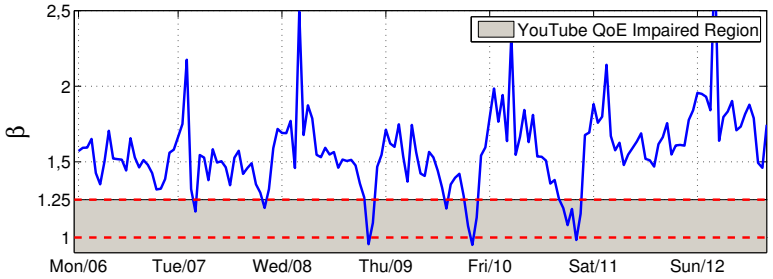
Figure 5.13 depicts the time series of the per hour users and bytes downloaded normalized counts during the analyzed week. While there is a drop in the number of bytes downloaded



(a) Median of the flow download throughput per hour for all YouTube flows.



(b) Entropy of QoE classes per hour for all YouTube flows.



(c) Median of β per hour for all YouTube flows.

Figure 5.11.: Detecting the QoE-relevant anomaly. There is a clear drop in the download flow throughput from Wednesday till Friday at peak-load hours, between 20:00 and 23:00 UTC. The combined drop in the entropy of the QoE classes and in the KPI β reveal a significant QoE degradation.

from Wednesday afternoon on, there are no significant variations on the number of users during the working week (i.e., Monday till Friday), so we can be sure that the throughput and QoE strong variations observed in Figure 5.11 are not tied to statistical variations of the sample size. Using the QoE-related results depicted in Figure 4.18(c), we can assume that the drop in the bytes downloaded suggests that the bad QoE affected the users engagement with the video playing, resulting in users dropping the watched videos when multiple stallings occur (i.e., when $\beta < 1.25$).

Let us drill down on the YouTube server selection strategy and the servers providing the videos. Figure 5.14(a) depicts the number of server IPs providing YouTube flows per hour. The first interesting observation is that the server selection policy used in the first 4 days of

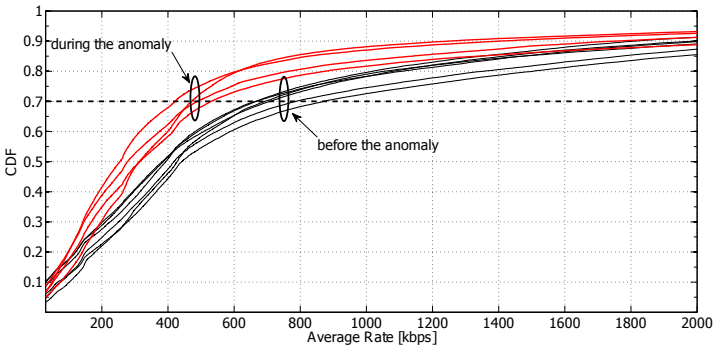
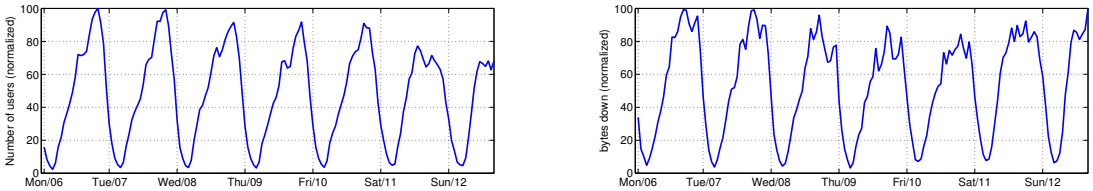


Figure 5.12.: Throughput distributions before and during the anomaly at the peak hour (9pm). The number of flows with lower throughput after the change in the hosting settings.



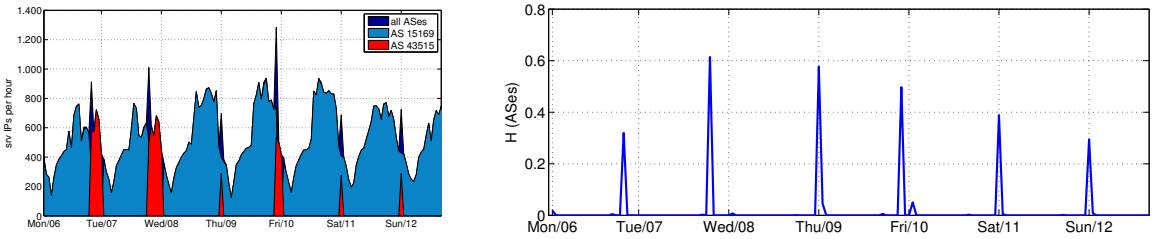
(a) Users count (normalized). (b) Bytes down count (normalized).

Figure 5.13.: Users and bytes down during the week of the anomaly. There are no significant changes during the specific times of the flagged anomaly.

the dataset (April the 15th till the 18th) and during the first 2 days of the week under study (May the 6th and the 7th) is markedly different, specially in terms of servers selected from AS 43515, confirming the large distribution change previously observed in Figure 5.10. As depicted in Figure 5.14(b), where the entropy time-series of the AS distribution corresponding to the monitored server IPs is presented, there is a sharp shift in the distribution of hosting ASes around peak-load hours. This shift corresponds to server IPs selected from AS 43515 rather than from AS 15169. In addition, there is an important reduction on the number of servers selected from AS 43515 on the days of the anomaly. This suggests that a different server selection policy is set up exactly on the same days when the anomalies occur.

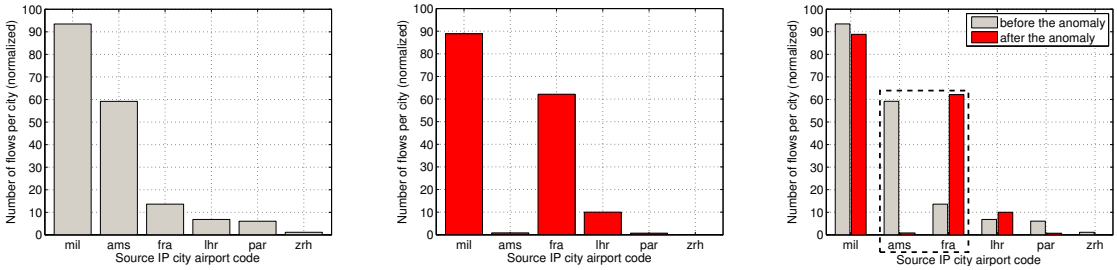
5.6.3. Geo-location Diagnosis Approach

We now take a step further in characterizing this CDN server selection policy, by taking a server geo-localization approach. The DNS-based re-directioning used by YouTube imposes a specific structure on the video identifiers requested to the content servers, which additionally include the name of the city where the server hosting the requested content is located. This city name is formated as an airport code, better known as IATA code (e.g., FRA for Frankfurt, AMS for Amsterdam, etc.). YouTube obfuscates this information, but it can be retrieved by



(a)IPs per hour hosting YouTube. (b) Entropy of the AS numbers time-series.

Figure 5.14.: IPs hosting YouTube during the week of the anomaly.



(a) Normal daily flows count per city. (b) Anomalous daily flows count per city. (c) Shift on the daily flows per top-6 cities.

Figure 5.15.: Geo-localization of the detected anomaly. There is a major shift in the daily number of YouTube flows coming from servers in Amsterdam to Frankfurt, suggesting that the problem is linked either to servers in Frankfurt, or to the new server-to-customer network paths.

reverse engineering (a description on how to do it is out of scope). Using this information, we can study the geographical location of the new servers selected from the 8th on.

Figure 5.15 reports the daily number of flows (normalized) served from the top cities hosting the YouTube content in our traces on (a) a day before the 8th of May and (b) a day after the 8th of May. The top cities hosting the YouTube videos in this case study are Milano and Amsterdam, followed by Frankfurt and other EU cities. The comparison presented in Figure 5.15(c) shows that the newly selected servers are mainly located in Frankfurt and London, and that almost all the flows served from Amsterdam are shifted to these cities in the new cache-selection policy. Figure 5.16 complements this geo-localization view on the traffic by reporting the daily distribution of the YouTube flows per city and per $/24$ subnetwork and AS. The shift is done from a single $/24$ subnetwork in AS 43515 to more than five $/24$ subnetworks in AS 15169. Very interestingly, the servers located in Amsterdam are almost no longer used after the shit on the 8th.

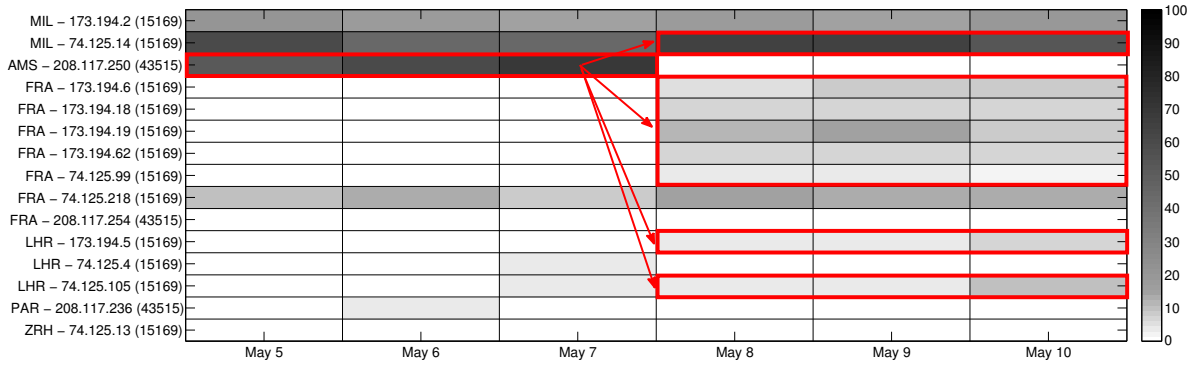


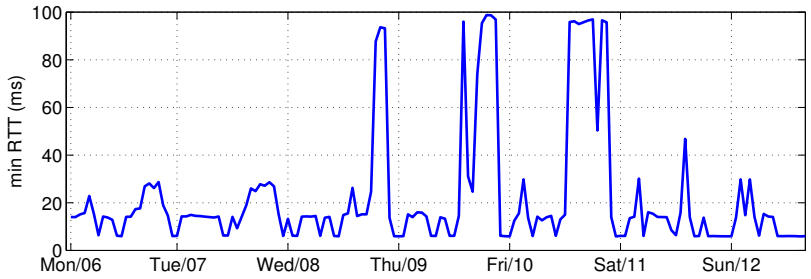
Figure 5.16.: Daily distribution of the YouTube flows per city and $/24$ subnetwork. Each column adds to 100%, and the darker the color, the higher the fraction of flows hosted. Starting on May the 8th, the lion share of the YouTube flows, normally served from Amsterdam, are shifted to Frankfurt and London.

5.6.4. Assessing Path-related Issues

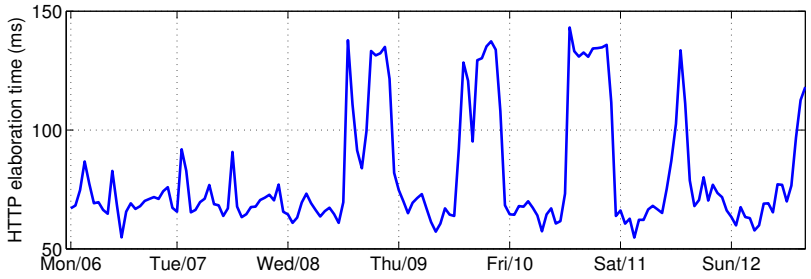
Given this change in the server selection policy, we try to find out if the problem arises from the newly selected servers, or if the problem is located in the path connecting these servers to the users. Figure 5.17 studies the latency from users to servers during the complete week. Figure 5.17(a) depicts the median of the min RTT per hour as measured on top of all the YouTube flows. The marked increase in the RTT evidences that the servers selected during the anomaly are much farther than those used before the anomaly. This increase impacts directly on the HTTP elaboration time (i.e., time between HTTP request and reply), as depicted in Figure 5.17(b). To understand if these latency increases are additionally caused by path congestion, Figure 5.17(c) plots the time series of the difference between the min RTT and the average RTT values; in a nutshell, in case of strong path congestion, the average RTT shall increase (queuing delay), whereas the min RTT normally keeps constant, as it is directly mapped to the geo-propagation delay. The differences before and during the anomalies do not present significant changes, suggesting that the paths between servers and clients are not suffering from congestion. This is also confirmed by the analysis of the packet retransmissions, which do not present significant variations. Indeed, by applying the techniques presented in [98], we were not able to identify the presence of a capacity bottleneck on the downstream paths.

5.6.5. Assessing Server-related Issues

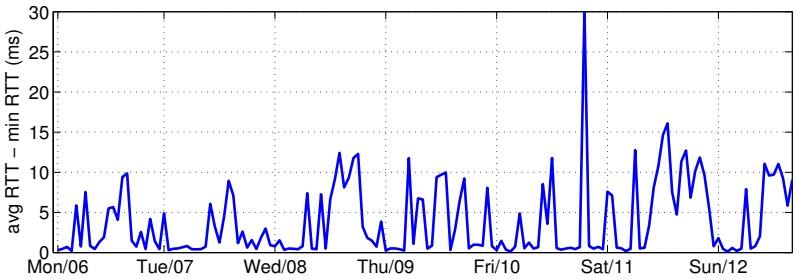
The last part of the diagnosis focuses on the YouTube servers. Figure 5.18 depicts the daily average download throughout of YouTube flows per city and per $/24$ subnetwork, using the geo-localization information described before. The color of each geo-temporal slot reflects the QoE of the users accessing the corresponding servers, based on the thresholds defined in Section 4.4.4 (green = good QoE, yellow = average QoE, red = bad QoE). As expected, the



(a) Median of min RTT per hour for all YouTube flows.



(b) Median of HTTP elaboration time per hour for all YouTube flows.



(c) Median of avg RTT - min RTT per hour for all YouTube flows.

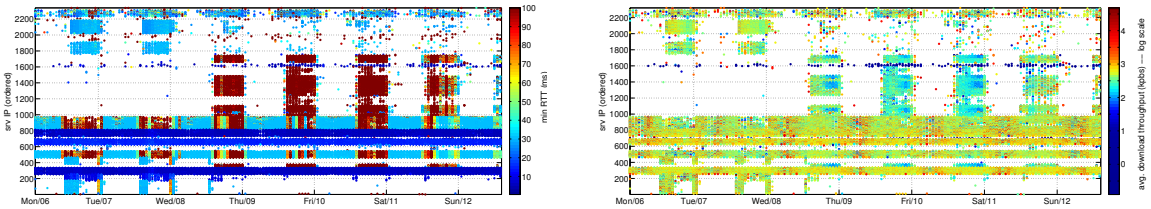
Figure 5.17.: The servers selected during the anomaly are much farther than those used before. While there is a marked increase in the server elaboration time, the avg. queuing delay (difference between avg. and min. RTT) remains bounded during the anomaly, so we discard the hypothesis of path congestion.

shift depicted in Figure 5.16 from Amsterdam to Frankfurt is accompanied by a very strong degradation on the QoE of the users.

Figure 5.19 depicts the average (a) min RTT and (b) download flow throughput per server IP in a heatmap like plot. Each row in the plots corresponds to a single server IP. The previously flagged min RTT increase is clearly visible for the new set of IPs which become active from 15:00 to 00:00 from Wednesday on. For those server IPs, Figure 5.19(b) shows the important throughput drop during peak-load hours. Note however that large min RTT values do not necessary result in lower throughputs, as many of the servers used before and

	May 5	May 6	May 7	May 8	May 9	May 10
MIL – 173.194.2 (15169)	1330	1275	1250	1300	1200	1350
MIL – 74.125.14 (15169)	1250	1250	1160	1300	1220	1140
AMS – 208.117.250 (43515)	680	655	650	900	0	0
FRA – 173.194.6 (15169)	0	0	0	380	330	320
FRA – 173.194.18 (15169)	0	0	0	370	300	230
FRA – 173.194.19 (15169)	0	0	0	260	200	165
FRA – 173.194.62 (15169)	0	0	0	325	250	275
FRA – 74.125.99 (15169)	0	0	0	190	190	135
FRA – 74.125.218 (15169)	1360	1340	1240	1260	1300	1280
FRA – 208.117.254 (43515)	670	850	0	610	1150	1380
LHR – 173.194.5 (15169)	1820	1660	0	940	1030	1170
LHR – 74.125.4 (15169)	1850	1040	1360	0	0	0
LHR – 74.125.105 (15169)	1550	1190	940	990	1060	1130
PAR – 208.117.236 (43515)	165	540	500	0	0	0
ZRH – 74.125.13 (15169)	1075	2265	1370	0	0	0

Figure 5.18.: Daily average download throughout of YouTube flows per city and $/24$ subnetwork. The flows shifted to Frankfurt on the 8th of May are provisioned with a very low throughput. Colors reflect the QoE of the users (green = good, yellow = average, red = bad), based on the thresholds defined in Section 4.4.4.



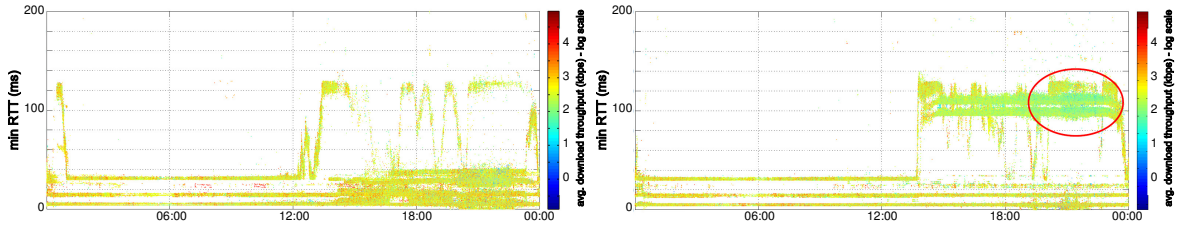
(a) Average min RTT per server IP.

(b) Average download flow throughput per server IP.

Figure 5.19.: There is a new set of server IPs providing YouTube videos from Wednesday on from farther locations. As visible in (b), the average download flow throughput for each of these new server IPs is much lower than the one obtained from other servers.

during the anomaly are far located but provide high throughputs. Figure 5.20 further studies this drop, comparing the relation between min RTT and average download flow throughput before and during the anomaly. The increase of the min RTT is not the root cause of the anomaly. However, there is a clear cluster of low throughput flows coming from far servers during the peak-load hours.

The conclusion we draw from the diagnosis analysis is that the origin of the anomaly is the cache selection policy applied by Google from Wednesday on, and more specifically, that the additionally selected servers between 15:00 and 00:00 were not correctly dimensioned to handle the traffic load during peak hours, between 20:00 and 23:00. This shows that the dynamics of Google's server selection policies might result in poor end-user experience, on the one hand by choosing servers which might not be able to handle the load at specific times, or even by selecting servers without considering the underlying end-to-end path performance.



(a) minRTT and avg. download rate per flow – before the anomaly.

(b) minRTT and avg. download rate per flow – during the anomaly.

Figure 5.20.: The increase of the min RTT is not the root cause of the anomaly, as there are no major issues previous to the anomaly. However, there is a clear cluster of servers offering low throughput during the peak-load hours on an anomalous day.

5.7. Unveiling Device-specific Anomalies through DNS Analysis

In this Section we focus on *device-specific* anomalies observed in cellular networks[4]. This class of anomalous events has gained a lot of interest by ISPs because of it high frequency and the detrimental impact it has on their network.

These anomalies refer to the unreachability of remote Internet services that support some functionality of a specific class of devices, such as the push notifications for a mobile operating system for smartphones. Given the importance of such functionalities, affected end terminals continuously try to re-contact the remote service highly impacting the normal cellular traffic patterns. This could result in an extremely armful overloading due to synchronized communication patterns, severely impacting overall network performance. This makes rapidly detect and diagnosing device-specific anomalies crucial for cellular ISPs.

From our operational experience, application-specific anomalies are particularly visible in the DNS traffic. Modern Internet applications, in fact, heavily rely on complex load balancing mechanism based on the diversification of DNS answers to different clusters of users. The Time to Live (TTL) of those answers is usually short, in the order of seconds. Hence, every time a user tries to access the remote service, it is likely to generate a new DNS query, inducing changes in the normal DNS usage patterns. Indeed, abrupt changes in the DNS requests count can be considered as a symptom of such anomalies.

We passively monitor DNS queries and answers in a Nation-wide cellular network by using the usual approach and vantage point previously described. Observing DNS traffic also allows to gain a number of meta-information that can be exploited in the diagnosing of the anomalies. These meta-data include the anonymized ID of the end host device, the requested remote service, the manufacturer, OS, as well as the local network settings (i.e., APN, RAT, DNS server IP).

The diagnosis process of these anomalies consists in observing the aforementioned traffic

[4]This study has been originally published in [P12]

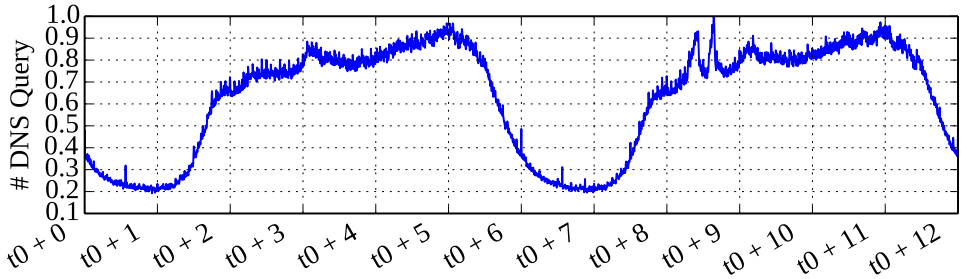


Figure 5.21.: DNS requests count over two days. Two spikes in the morning of the second day suggest the presence of the anomaly. Original source at [P12].

features and checking which one of them undergo a substantial *pattern change*. By correlating these *diagnostic information*, one could build up a sort of signature for the anomaly and isolate the sub-population of affected devices and the responsible remote service, offering a full characterization of the event. We are now introducing the concept of correlating *diagnostic signals*. For the moment, we focus on the manual correlation of changes in such signals and we postpone the formalization and the automation of this to the next Chapter.

5.7.1. Anomaly Characteristics

We present now the study we have conducted on an instance of this anomaly class which has been observed at our cellular VP. Figure 5.21 shows the time series of the total DNS requests count observed in the network for two consecutive days. Two significant and anomalous spikes are observed on the second day, which are easily spotted by the abrupt-change detection algorithm.

Figure 5.22 provides a closer look into the anomaly, comparing the time series of the total DNS requests count and the entropy of 4 selected features: FQDN, manufacturer, APN, and ID. Similarly to the previous Section, the entropy measures the *uniformity* of the distribution of DNS counters across the variables of feature itself. When the time series of the entropy presents some glitches (i.e., spikes or notches), this gives an indication of a change in the normal traffic patters that could be attributed to the presence of an anomaly. We will describe and evaluate entropy-based anomaly detectors in the next Chapter (cfr. Section 6.5). The 4 features are extracted for each DNS request-response transaction (to preserve user privacy, any user related data are removed on-the-fly). The other available diagnostic signals are omitted for brevity, as they show a behavior similar to the reported next. We notice that some of the observed diagnostic signals are correlated in a minor way to the anomaly. This is the case for ID, TAC, RAT, and DNS *rcode*, therefore we can exclude the cases in which the anomaly is caused by few users, a specific RAT, etc.. On the contrary, dimensions such as FQDN and manufacturer present a very high correlation with the spikes in the DNS count, suggesting that the issue might be due to specific devices (manufacturer) querying for certain services (FQDN). Features such as APN and server IP show partial correlation to the anomaly, thus need to be further cross-checked.

The next step of the diagnosis is to drill down each of the dimensions that are highly correlated with the anomaly. This can be achieved, e.g., by comparing the heavy hitters

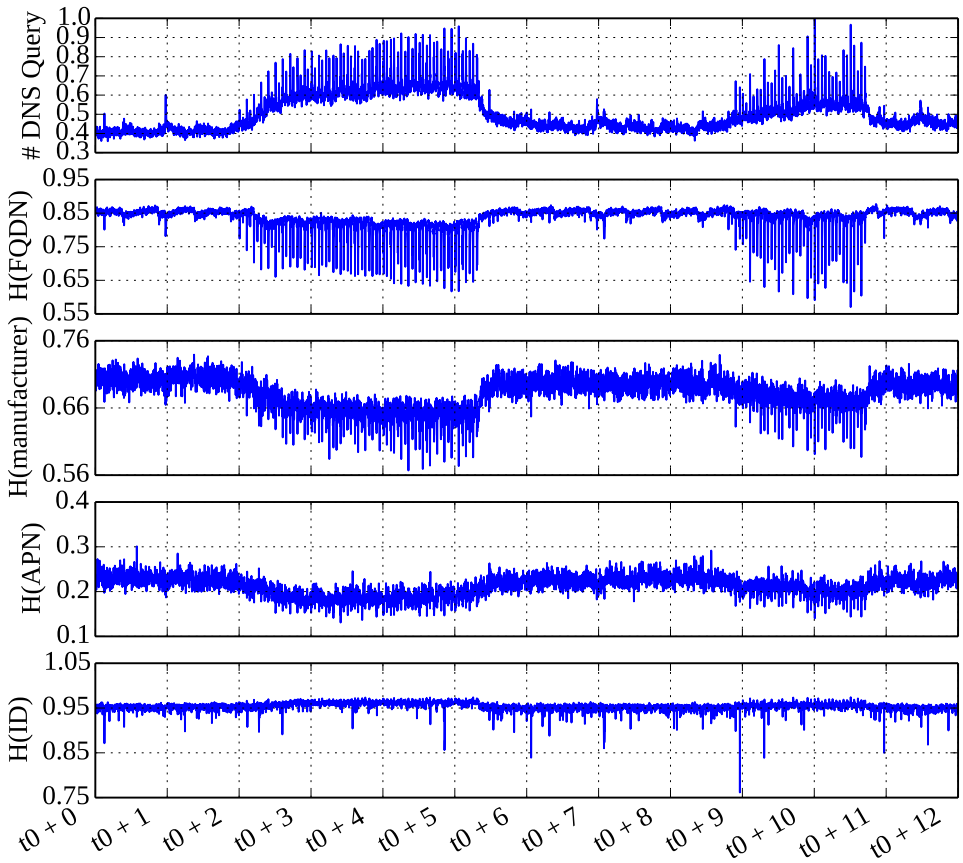


Figure 5.22.: Entropy of selected features. The timeseries of some entropies are altered during the anomaly. Original source at [P12].

before and during the anomaly. Figure 5.23 reports the specific case for the FQDN. The plot shows the time-series of the most requested FQDNs during the anomaly. We observe that, while some of the top FQDNs associated to well-known services present a stable behavior (`well_known_1/2`), the FQDNs `anomalous_cdn_1/2` and `anomalous_direct_1` show a significant increase. The first two refer to content of a specific popular OTT service delivered via a major Content Delivery Network (CDN), whereas the third one points directly to the specific OTT service, showing that the problem is actually related to this service.

The mapping of the TAC codes to the manufacturer of the devices requesting the FQDNs related to the anomaly also reveal a specific smartphone type involved in the anomaly. In particular, the specific anomalous service runs on all these devices, but not on the other smartphone types. W.r.t. the dimensions presenting partial correlation, we found that all the different APNs are affected by the anomaly but in a different manner, suggesting that different APNs are configured for different customers. Indeed, different APNs are normally linked to different default DNS servers.

As a main conclusion, the proposed approach is helpful in highly reducing the time spent by the network operator in the diagnosis of unexpected traffic behaviors. In particular, this service outage resulted in an abrupt increase in the number of connection attempts from a large number of devices, and its fast diagnosis was paramount to understand the nature of

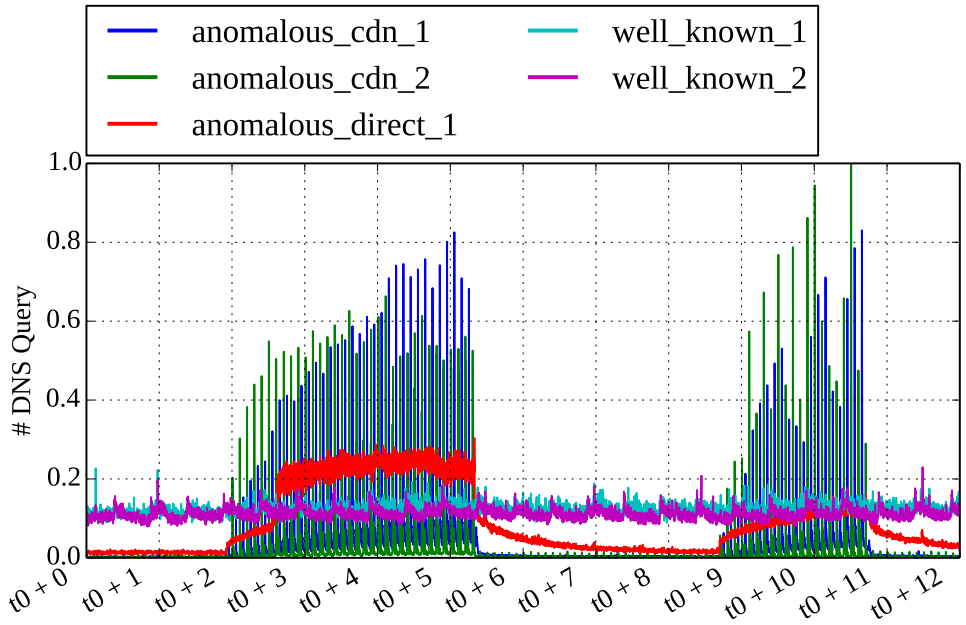


Figure 5.23.: DNS requests per FQDN class. The FQDNs `anomalous_cdn_1/2` (in blue) and `anomalous_direct_1` (in red) show a significant increase. Original source at [P12].

such an anomaly.

5.8. Summary

From the previous Chapters, we learned that most of modern Internet scale services rely on complex Content Delivery Networks (CDNs), which push contents as close as possible to the end-users to improve their Quality of Experience (QoE) and to pursue their own optimization goals. Adopting space and time variant traffic delivery policies, CDNs serve users' requests from multiple servers/caches at different physical locations and different times. CDNs traffic distribution policies can have a relevant impact on the traffic routed through the Internet Service Provider (ISP), as well as unexpected negative effects on the end-user QoE.

In this Chapter we have studied a number of anomaly use-cases affecting the actors involved in Internet services, i.e., the content and service providers, ISPs and the end-users. Promptly detecting and diagnosis such anomalies is still an open issue: the dynamics of traffic patterns make it difficult to distinguish changes that are potentially linked to anomalous behavior from the *physiological* ones. Nevertheless it is of vital importance for ISPs to do so in order to increase its visibility on the overall operation of the network, as well as to promptly answer possible customer complaints. The study was carried out by manually investigating the anomalies correlating a number of traffic features. By using the insights collected so far, in the following Chapter we will start describing a structured framework for the detection and diagnosis of network anomalies.

6. Advanced Anomaly Detection Techniques

Notice of adoption from previous publications

Parts of the contents of this Chapter have been published in the following papers:

[P6] A. D'Alconzo, P. Casas, **P. Fiadino**, A. Bär, A. Finamore, "Who to Blame when YouTube is not Working? Detecting Anomalies in CDN-Provisioned Services", in *The 5th International Workshop on Traffic Analysis and Characterization (TRAC2014)*, 2014.

[P11] **P. Fiadino**, A. D'Alconzo, A. Bär, A. Finamore, and P. Casas, "On the Detection of Network Traffic Anomalies in Content Delivery Network Services", in *Teletraffic Congress (ITC), 2014 26th International*, 2014.

[P18] **P. Fiadino**, A. D'Alconzo, M. Schiavone, P. Casas, "Challenging Entropy-based Anomaly Detection and Diagnosis in Cellular Networks", in *ACM SIGCOMM 2015 Poster/Demo session*, 2015.

[P19] **P. Fiadino**, M. Schiavone, A. D'Alconzo, P. Casas, "Towards Automatic Detection and Diagnosis of Internet Service Anomalies via DNS Traffic Analysis", in *International Wireless Communications & Mobile Computing Conference - TRAC (IWCMC 2015)*, 2015.

[P20] **P. Fiadino**, A. D'Alconzo, M. Schiavone, P. Casas, "RCATool – A Framework for Detecting and Diagnosing Anomalies in Cellular Networks", in *27th International Teletraffic Conference (ITC27)*, 2015.

The statistical anomaly detection algorithm published in the above mentioned papers and described in this Chapter is based on the work done in [29]. The algorithm has been improved and implemented by the author of this thesis as part of a larger framework (cfr. next Chapter). The author has a major role in the design and implementation of the extended version of the algorithm, in the generation of semi-synthetic datasets, in the evaluation and in the comparison with the entropy-based detection technique. The author has collaborated with Mirko Schiavone and Peter Romirer-Maierhofer to a preliminary study on the entropy-based detection system applied on a device-specific anomaly study case. The work has been supervised by Dr. Pedro Casas, Dr. Alessandro D'Alconzo and Prof. Tanja Zseby.

6.1. Introduction

Despite the long literature and assorted list of proposed systems for detecting anomalies in large-scale operational networks, Internet Service Providers (ISPs) are still looking for an ultimate solution which might effectively detect and diagnose the ever-growing number of network traffic anomalies they face in their daily business. Indeed, we have seen that the sophisticated provisioning systems used by Internet services induce continuous changes that

impact all the involved actors (i.e., service providers, access operators and final customers) making Anomaly Detection (AD) a moving target.

Given this complexity, the Anomaly Detection's challenge is twofold: on the one hand it is difficult to efficiently detected changes, on the other hand it is paramount to sort out which of those are the effect of anomalous behavior. In the remainder of this thesis we try to fill this gap by presenting and evaluating a framework for detecting and diagnosing network anomalies by exploiting the know-how acquired during the deep study of traffic characteristics previously presented, both in normal operation mode (cfr. Chapter 4) and during anomalies (cfr. Chapter 5).

Detection is achieved by extracting and analyzing *symptomatic* traffic features, and by flagging a warning as soon as one or more of them show a significant change. The investigation of the root causes for such deviations is done by looking at changes in separate *diagnostic* traffic features, which convey information directly linked to the potential origins of the detected anomalies. For the purpose of detecting significant changes in both the symptomatic and the diagnostic features, we resort to the analysis of their full statistical distribution.

Features are further processed to define what we shall refer to as analysis *signals*. A signal describe the statistical characteristics over time of the corresponding feature, and allow for abstraction and generalization of the framework's input definition. For example, a relevant feature used in our framework is the number of DNS requests per observed FQDN per time bin; in this case, a signal associated to this feature could be defined as the mean number of requests, the total number, the full empirical distribution, the entropy, etc. Two signals derived from the same feature might yield completely different detection results: for example, an anomaly could be easily spotted when analyzing the entropy of a certain feature, but not through its mean value. The separation between feature and signal allows to decouple the meaning of an input from the information it exposes for detecting anomalies.

This Chapter will focus on the study of advanced detection systems, based on the lesson learned from the manual analysis of the anomaly case studies presented in Chapter 5. We target, in particular, those anomalies that are intrinsically more difficult to characterize, such as the device-specific class presented in Section 5.7 and the QoE degradations in video streaming services presented in Section 5.6. The next Chapter will be dedicated to the final part of the framework that relies on Machine Learning techniques to conclude the diagnosis process.

The proposed solution is evaluated using real and synthetic data from operational, nation-wide ISPs (both fixed-line and cellular), the latter generated from traffic statistics to resemble the real mobile network traffic. Furthermore, we compare the achieved performance against well-known entropy based analysis revealing the superiority of the proposed technique in a number of prototypical cases.

6.2. Related Work and Contributions

There has been considerable amount of research about anomaly detection in network traffic. Chandola et al. provide a comprehensive survey on standard anomaly detection techniques

[99], many of which have been applied in networking. For example, a large set of works applies concepts and techniques imported from fields like Neural Networks, Self-Organizing Maps [100], Genetic Algorithms [101], Fuzzy Logic [102], Data Mining [103], Machine Learning [104]. Focusing on statistical-based methods, most work rely on the analysis of scalar time-series, typically of total volume. They adopt various techniques like Discrete Wavelet Transform [105], Holt-Winter [106], CUSUM [107] and others.

A millstone in the field of anomaly detection in large scale networks was set by the work of Lakhina et al. [108, 109], where authors introduced the application of the well known Principal Components Analysis technique to the detection of network anomalies in traffic matrices. Since then, many papers followed on a similar direction [110, 111, 112, 113].

In general, it is commonly accepted that information-theoretic concepts, and in particular entropy measures, are well-suited for identifying the statistical properties relevant for the purpose of anomaly detection [114, 115]. However, such schemes fail in detecting events that do not cause appreciable changes in the total traffic volume. This is particularly critical when the underlying per-user volume is heavy-tailed, since the physiological fluctuations caused by few heavy-hitters may mask the anomaly.

In this thesis we endorse a distribution-based approach presented in [29] that is intrinsically more powerful, as it looks at the entire distribution, rather than only at the total sum or summarizing metrics like the entropy. The cost is of course a larger amount of data to be processed, and higher complexity of the monitoring platform. Only a few other authors have started to investigate anomaly detection on distributions (e.g., [116, 117]), in order to take into account the intrinsic data variability while detecting anomalous deviations.

Notice that the focus of our work is not on conceiving (yet) another detection algorithm. Our contribution is rather investigating the effectiveness of the distribution-based detection algorithm proposed in [29], that we have opportunely modified so as to be able to cope with dynamic traffic patterns and provide all the information needed for the diagnosis. The improved version of the algorithm become the core building block of the novel detection and diagnosis system that we propose in this thesis.

6.3. Detection and Diagnosis Framework Overview

We now introduce a generic framework to detect and diagnose large-scale network traffic anomalies based on the analysis of passively captured network traces. The design of such framework is derived by the lessons learned in the study of the anomalies described in Chapter 5. We take a step forward by formalizing the detection and diagnosis methodologies previously introduced in order to overcome the limitations of the manual inspection of anomalous events.

Figure 6.1 sketches an overview of the proposed framework. From traffic tickets we derive two sets of signals denoted as *symptomatic* signals and *diagnostic* signals. All signals are checked for significant changes from their reference of "normality". However, the symptomatic signals are designed such that their changes directly relate to the presence of abnormal and potentially harmful events. On the other hand, changes in the diagnostic signals per-se do not have a negative connotation, but rather ease and guide the interpretation of the

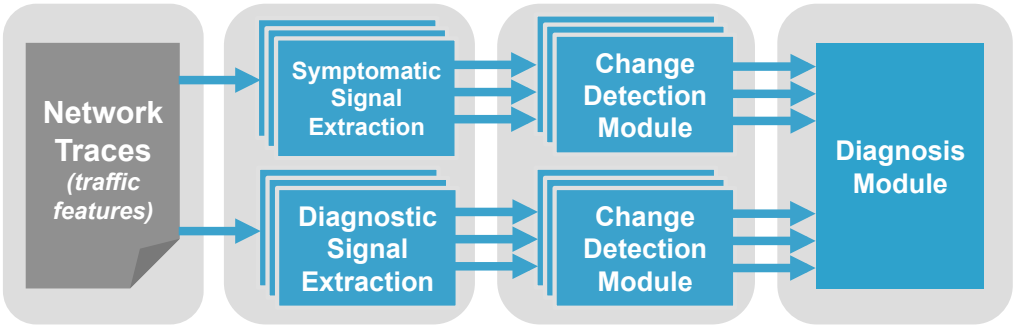


Figure 6.1.: Overview of the diagnosis framework.

anomalous event. In the diagnosis step, deviations of the symptomatic signals are correlated with the subset of simultaneously changing diagnostic signals to provide a comprehensive characterization of the event.

We assume that a passive network monitoring system tracks the traffic at a Vantage Point (VP) in a production network, i.e., some aggregation link in a fixed-line network or a core-network link of a mobile network, and returns information of interest in the form of traffic tickets. Such tickets summarize not only info related to the specific transaction, but also meta-data that provide further information about the end host device and the network settings. For example, in the case of DNS traffic originated in a mobile network, tickets shall report not only details about the DNS queries and the corresponding answers, but also the meta data listed in Table 6.1, such as the anonymized ID of the end host device, the manufacturer, OS, as well as the local network settings (i.e., APN, RAT, DNS server IP). Flow-level tickets would include the 4-tuple consisting of source/destination address and port and additional meta information related to the flow, such as the average throughput and the round-trip-times (RTTs).

Although in this work we focus on the analysis of DNS and video-flows traffic, the proposed framework is generic enough to operate with different network data sources, such as other application layer information, which might contribute to further refine the anomaly diagnosis.

The output of the change detection feds the final *Diagnosis* block of the framework.This

Field Name	**Description**
Device ID	Anonymized device identifier
Manufacturer	Device manufacturer
OS	Device operating system
APN	Access Point Name
RAT	Radio Access Type
DNS Server	IP address of the DNS resolver
FQDN	Fully Qualified Domain Name of remote service
Error Flag	Status of the DNS transaction

Table 6.1.: DNS ticket information (meta-data).

Notation	Description
f	generic traffic feature
τ	time bin length defining the aggregation scale
$i \in \{1, \ldots, n(t)\}$	i-th element of f of the $n(t)$ observed in t
$f_i^\tau(t)$	generic counter of f at the t-th time bin of length τ
$\mathcal{F}^\tau(t) = \{f_i^\tau(t)\}$	set of counters aggregated at τ time scale
$X_f^\tau(t)$	empirical distribution of the feature f aggregated at τ time scale

Table 6.2.: Summary of the signal notation.

module is capable of automatically classifying anomalies by relying on supervised classification techniques. In addition, it finds temporal correlations of signals' changes and is ultimately able to generate event fingerprints enriched with diagnostic information in correspondence to one or more anomaly symptoms.

In the remainder of this Chapter we describe the first two fundamental stages of the proposed framework, namely the definition and extraction of symptomatic and diagnostic signals and the design of suitable detection schemes for abrupt changes. In particular, we will focus on the comparison of two different detection techniques: an entropy-based (cfr. Section 6.5) and a distribution-based approach (cfr. Section 6.6) .The description of the final diagnosis module will be object of study of the next Chapter.

6.4. Signals Extraction

Let us now formalize the definition of features and signals as considered in the proposed framework, as well as describe the applied change detection technique.

For the generic feature f derived from the DNS tickets, we indicate by $f_i^\tau(t)$ the generic counter observed at the t-th time bin of length τ. For instance, if f represents the number of DNS requests for a FQDN every τ minutes, $i \in \{1, \ldots, n(t)\}$ is the i-th requested FQDN, while $f_i^\tau(t)$ counts the number of DNS requests for the i-th FQDN (out of the $n(t)$), over the t-th time bin. The i-th counter can also be associated to other fields, such as the OS version, the Error Flag value, the number of DNS queries generated by the i-th device, etc. The length of τ defines the timescale of the data aggregation, which in turn defines the timescale of the observable anomalous events. The set of counters $\mathcal{F}^\tau(t) = \{f_i^\tau(t)\}$ can be used to derive the empirical distribution of the feature f, denoted by $X_f^\tau(t)$. This notation is summarized in Table 6.2.

By properly grouping features, we can obtain aggregated statistics for different "views" on the data. Considering the example above, the FQDN counters can be further grouped to obtain, e.g., 2-nd Level Domain (2LD) or 3rd Level Domain (3LD) counters and statistics. We can also use the counters for computing the overall number of DNS requests as $N(t) = \sum_i f_i^\tau(t)$. As the following analysis can be done independently of the specific selected time scale, we omit the superscript τ from now on.

6.5. Entropy-based Anomaly Detection

A particularly popular approach for detecting anomalies in network traffic is the one represented by entropy-based analysis [109]. Although entropy-based approaches have been successfully applied for anomaly detection in the past [114, 115, 109], in fixed-line networks and using only network- and transport-layer features such as IPs and ports, their application in the aforementioned context has severe limitations, given the characteristics of current traffic anomalies. In the following, in fact, we will thoroughly test this methodology and show that it fails in coping with anomalies characterized by lower intensities.

The entropy of a feature captures the dispersion of the corresponding probability distribution in a single number, thus it is highly appealing for the analysis. However, such a compression necessarily looses relevant information about the higher distribution moments of the analyzed feature, limiting the detectability of some anomalies. Given the empirical distribution $X_f^\tau(t)$ of a certain feature f, we can compute the normalized entropy as:

$$H(X) = -\frac{1}{\log(|\Omega|)} \sum_{\omega \in \Omega} x(\omega) \log x(\omega), \tag{6.1}$$

where Ω and $|\Omega|$ are a discrete probability space and its cardinality, respectively, and $x(\omega)$ is the probability of element ω [1]. The entropy of a feature f is a well-suited synthetic index for describing an entire distribution, and in particular, useful for detecting important changes.

Entropy-based detectors work by flagging abrupt changes in the time series of the empirical entropy of certain traffic features, related to the specific anomaly. We consider the well-known, yet effective, Exponential Weighted Moving Average (EWMA) algorithm, where past observations are weighted such that the older ones count less in the determination of the expected current value. For the observed value $y(t)$ at the time bin t, the value predicted by EWMA is calculated as:

$$\tilde{y}(t) = \lambda y(t) + (1 - \lambda)\tilde{y}(t - 1), \tag{6.2}$$

where λ controls the filter *memory*, that is the weight of the past samples in computing the moving average: the higher λ the higher the weight of the newer samples. Then, the Upper Control Limit (U_{CL}) and the Lower Control Limit (L_{CL}) are defined as $U_{CL}(t) = (1 + \sigma)\tilde{y}(t - 1)$ and $L_{CL}(t) = (1 - \sigma)\tilde{y}(t - 1)$, respectively, where σ is a slack factor that controls the width of the acceptance region for normality. Finally, the detection algorithm flags an anomaly if $y(t) \notin [L_{CL}(t), U_{CL}(t)]$. Note that by opportunely tuning λ and σ, the EWMA algorithm becomes able to accommodate for typical daily and weekly patterns of real network traffic.

6.6. Distribution-based Anomaly Detection

The second anomaly detection scheme relies on the temporal analysis of the entire probability distributions.

[1] Note that, in case of empirical values such as traffic features, $x(\omega)$ is actually the fraction of ω at time t.

By that it is particularly suited to cope with anomalies that involve multiple services and/or affect multiple devices at the same time. The considered non-parametric anomaly detection algorithm computes the degree of similarity between the current distribution $X_f^\tau(t)$ to a set of (anomaly-free) distributions in a dynamic "observation window" $W(t)$, which describe the "normal" behavior. The heuristic used for the construction of the reference set follows a progressive refinement approach that takes into account the structural characteristics of traffic such as time of day variations, presence of pseudo-cyclic weekly patterns, and long term variations. The comparison between the current distribution $X_f(t)$ and the associated distributions reference set involves the computation of two compound metrics based on a distribution divergence metric L. The first metric, called *internal dispersion* (or *upper bound*), is a synthetic indicator defining the maximum distribution deviation that can be accounted to normal statistical fluctuations, therefore it defines acceptance region for the anomaly detection test. The second one, called *external dispersion* (or *average distance*), is a synthetic indicator extracted from the set of divergences between the current distribution $X_f(t)$ and those in the reference. The detection test checks if the average distance exceeds the upper bound. As for the distance metric between two distributions p and q, defined over a common discrete probability space Ω, we rely on a symmetrized and normalized version of the *Kullback-Leibler* divergence (ENKLd) defined as:

$$L(p,q) = \frac{1}{2}\left(\frac{D(p||q)}{H_p} + \frac{D(q||p)}{H_q}\right), \tag{6.3}$$

where $D(p||q)$ is the KL-divergence, defined as

$$D(p||q) = \sum_{\omega\in\Omega} p(\omega)\log\left(\frac{p(\omega)}{q(\omega)}\right). \tag{6.4}$$

Analogously $D(q||p)$ is the KL-divergence between q and p, and H_p and H_q are the entropy of p and q, respectively.

Notice that using a distribution-based approach is intrinsically more powerful, as it considers the entire distribution of different traffic features, rather than only specific moments of the distributions (e.g., mean-based, variance-based, or percentile-based change detection). More specifically, a distribution divergence metric – such as ENKLd – measures the point-to-point difference of p and q, before calculating its entropy. Thus, the obtained divergence strictly depends on both the distributions under exam and relies on a much more fine-grained information. On the contrary, the entropy is just capturing an intrinsic characteristic of the single distributions.

To better clarify this, let us consider the case of two entirely different distributions with the same entropy. The first detection approach would fail in flagging their difference. On the other hand, the ENKLd relies on the single points of the distributions to calculate their distance. Of course, this comes with an additional computational cost because, from a procedural point of view, it needs to keep the full empirical distributions over time – depending on the reference window width – to do the comparison.

That said, it is particularly suited for detecting macroscopic traffic anomalies, that is events

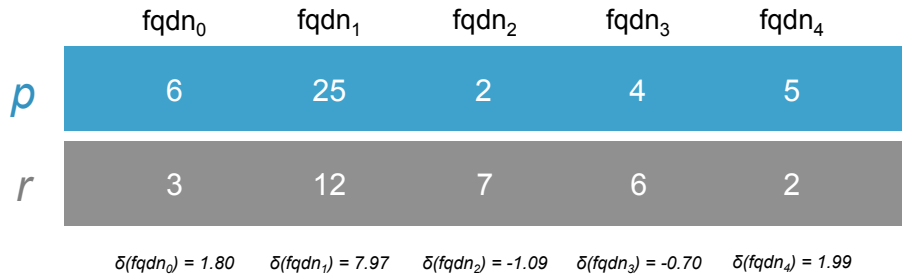


Figure 6.2.: A simplistic example drift values computation in two distributions: p (current) and r (reference). The algorithm will report $fqdn_1$ as the element that contributed most to the distribution change.

that involve multiple services and/or affect multiple devices at the same time.

We refer the interested reader to [29] for further details. In the following, we focus on our improvements applied on the algorithm, namely the reporting of changing elements (cfr. Section 6.6.1) and the self adapting reference window (cfr. Section 6.6.2).

6.6.1. Reporting changing elements

When a change is flagged by the detection algorithm, it also returns the list of the elements $\omega \in \Omega$ that have contributed most to the deviation of the current distribution $X_f(t)$ from the reference of normality. The procedure for identifying the top changing elements easily follows from the eq. (6.4). Let p be the current distribution, and r the *reference* distribution computed as by averaging the distributions in the reference set. We define $\delta(\omega)$, the *drift value* of ω, as follows:

$$\delta(\omega) = p(\omega) log \frac{p(\omega)}{r(\omega)}, \tag{6.5}$$

which intuitively represents the contribution of the element ω to the overall distribution change. At every iteration the changing elements are sorted by value. Note that in general $p(\omega) \neq r(\omega)$ due to statistical fluctuations, hence $\delta(\omega) \neq 0$ $a.e.$ in Ω. In order to get a compact representation of the change, the algorithm reports only the top elements accounting for $s\%$ of the overall change. From our experience, $s = 50\%$ provides best results, that is the algorithm returns only the few elements really responsible for the distribution change, and discharge those resulting from random statistical fluctuations. Notice that reporting the list of the most significant variables is a key feature for the diagnosis process. Indeed, it allows having fine-grained information on the root causes of the distribution change, in addition to the mere change notification.

Consider, as a simplifying example, the distributions p (current) and r (reference) of DNS query counters across FQDNs in the common space Ω, depicted in Figure 6.2. The Figure shows the DNS query counters for each $fqdn_i \in \Omega$ in the two distributions, along with their drift value $\delta(fqdn_i)$. The algorithm will raise an alarm as p is anomalous and will only report $fqdn_1$ as the element responsible for most ($> 50\%$) of the distribution change.

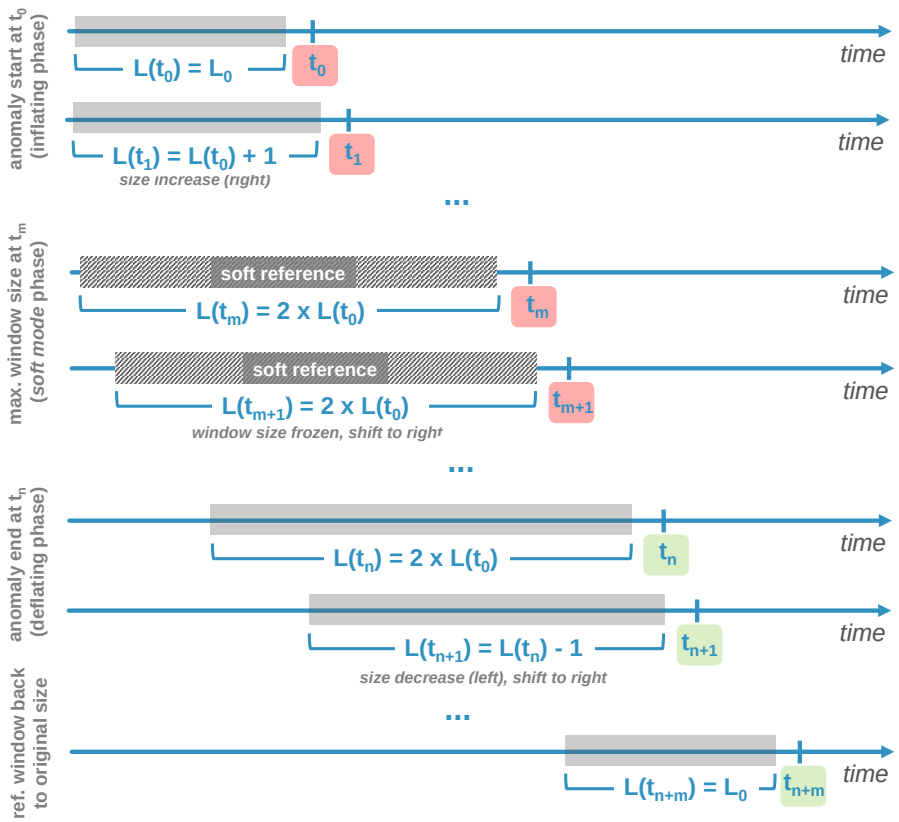


Figure 6.3.: Self-adapting Reference Window algorithm for long-lasting anomalies. Anomalous timebins are marked in red, normal timebins are marked in green. When the anomaly starts, the reference window increases its size till a maximum value (e.g., two times initial size). Then it enters a *soft mode* (i.e., anomalous timebins in the reference window are allowed in the reference set). When in soft mode, the window size is frozen and shift to the right. In this phase, all the distributions in the reference window are considered for the reference set. When the anomaly ends, the reference window gradually decreases down to the original size.

6.6.2. Self-adaptation to long-lasting changes

The algorithm originally proposed in [29] has been designed with the purpose of detecting large-scale security or performance anomalies. Such anomalies are expected to be transitory and have a limited duration. Consequently, the mechanism for updating the observation window is designed such that $W(t)$ is *frozen* till the anomaly is over. However, we now apply the same algorithm to the detection of changes in the diagnostic signals, which may exhibit long-lasting anomalies corresponding to *working point* changes. For example, let us consider a popular service that updates its naming scheme, heavily impacting the distribution of counters per FQDNs. In this case $W(t)$ remains locked (i.e., it is not shifted forward in time till the anomaly is over). In this case the detection algorithm would keep flagging warnings indefinitely, making it unusable.

To overcome this limitation, we have modified the observation window updating mechanism and the reference-set identification algorithm as detailed in the following. Let indicate by m and $L_0(t) = m\tau$ the number of valid distributions and the length of the observation window $W(t)$, respectively, when it contains no anomalous distributions. When an anomaly is detected, the length of observation window is increased by one to $L_1(t) = (m+1)\tau$, whereas the number of valid distributions remains m. As soon as a sequence of m anomalies is detected, the observation window length becomes $L_m(t) = 2m\tau$, and the algorithm enters the *soft test* state where it let anomalous distributions enter the reference-set. If the anomalous distributions are consistent enough (i.e., they are statistically similar), and the anomaly lasts longer than m, then it is likely they are selected as new reference for normality, and the test turns negative. From this moment onward, every time the test is negative $W(t)$ is reduced such that $L(t+1) = L(t) - 2\tau$, till it gets back to the initial size L_0. At that point the algorithm exits the *soft test* state. Notice that in this way the algorithm accommodates to the new working point after a transitory phase of $m\tau$ during which the algorithm keeps flagging anomalies.

To clarify the improved reference mechanism, Figure 6.3 depicts an example of execution of the self-adapting reference window algorithm. In the example, the anomaly starts at t_0, when the original reference window size is $L(t_0)$. After the beginning of the anomaly, the window size gradually increases (*inflating phase*) till reaching the maximum allowed value, e.g. $L(t_m) = 2 * L(t_0)$. The detection algorithm then enters in *soft mode* till the anomaly is over. In this phase the window size is not increased, but it is only shifted to right. After the end of the anomaly at t_n, the reference window gradually decreases till reaching the original size. Note that only long-lasting anomalies (with duration greater than $m-1$) force the detection algorithm to enter in soft mode.

6.7. Anomaly Modeling and Data Generation

We evaluated the proposed framework for longer than six months in 2014 with DNS traffic from the operational cellular network of a nationwide European operator, and for one month with YouTube flows from a European fixed-line ISP. The extensive experimentation allowed us to collect results in a number of paradigmatic case-studies exposing features and limitations of the framework. Still, the number of traffic anomalies observed in the corresponding period was relatively low, limiting as such our performance analysis exclusively to those few real cases. In principle, one could resort to test traces obtained in a controlled environment (laboratory) or by simulations, but these approaches would miss the complexity and heterogeneity of the real traffic.

To bypass this hurdle, we adopted a methodology based on semi-synthetic data, derived from real traffic traces as suggested in [118]. Such an approach does not only allow to extensively analyze the performance of the framework with a large number of synthetic, yet statistically relevant anomalies, but also permits to protect the operator's business sensitive information, as neither real data traces nor real anomalies are exposed.

To illustrate the procedure, next we explain both how to generate semi-synthetic background DNS traffic, as well as for modeling the DNS-related anomalies for replicating them.

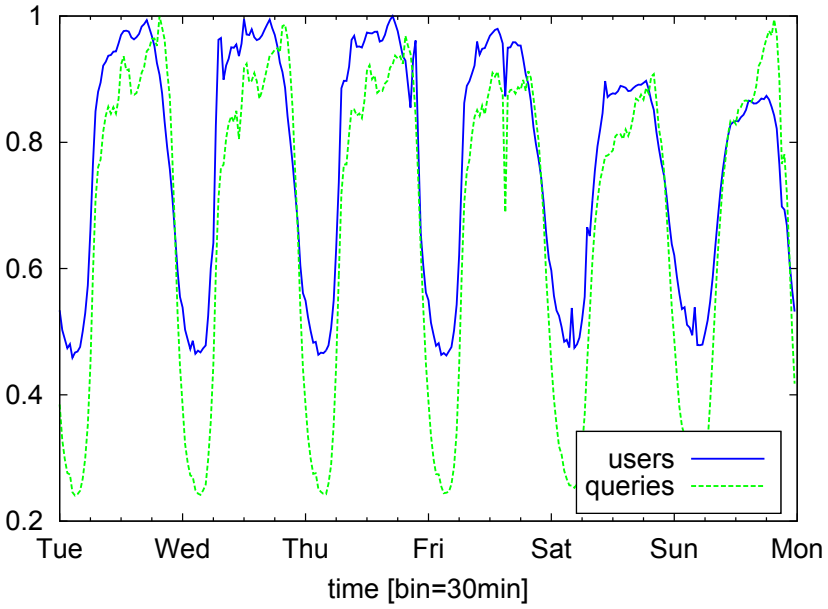


Figure 6.4.: Daily trend of the number of active users and total DNS query count in the semi-synthetic dataset.

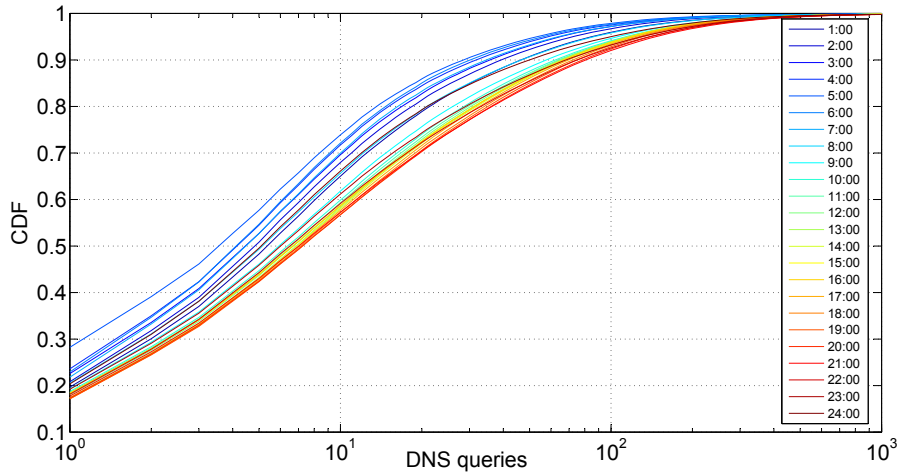
As a template for these anomalies, we rely on the use-cases introduced in Section 5.7, namely the *device-specific* anomalies. This is justified both by the complexity of these anomalies and by the relevance they have for ISPs.

The procedure for video flows is analogous and it is omitted for the sake of brevity.

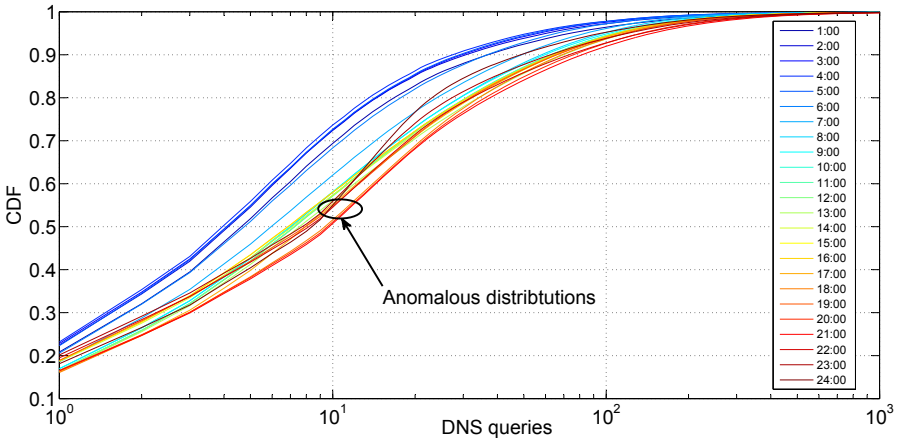
6.7.1. Construction of semi-synthetic background traffic

The procedure for constructing the semi-synthetic dataset is conceived with the objective of maintaining as much as possible the structural characteristics of the real, normal operation (i.e., anomaly-free) traffic, while eliminating possible (unknown) anomalies present in real traces. Exploring real traces, we observed that the traffic yields some fundamental temporal characteristics. In particular, the traffic is non-stationary due to time-of-day variations. This effect is not limited to the number of feature counters, but rather applies to their entire distributions. Distribution variations depend on the change of the applications and terminals mix, which in turn induce modifications in the traffic patterns. Furthermore, we found that, besides a strong 24-hours seasonality, the traffic exhibits a weekly pseudo-cycle with marked differences between working days and weekends/festivities [119]. Finally, traffic remains pretty similar at the same time of day across days of the same type.

The first step of the construction procedure consists of manually labeling and removing possible anomalous events. However, as the complete ground truth is unknown in real traffic, we cannot completely rely on individual labeling of alarms. Therefore, we have to accept that minor anomalies may go undetected if their effect is comparable with purely random fluctuations. Then, the dataset is transformed to eliminate possible residual (unknown) anomalies



(a) Normal day



(b) First day of the E_2 anomaly.

Figure 6.5.: Hourly trend of the distribution of number of devices across query count over one day of the semi-synthetic dataset.

present in the real traffic, while preserving the above mentioned structural characteristics. The transformation procedure is described as follows.

Let us consider a real dataset spanning a measurement period of a few weeks, for a total of m consecutive one-day intervals (e.g., $m = 28$ in our case). Each one-day period starts and ends at 4:00 am local time: this is the time-of-day where the number of active devices reaches its minimum (considering a single time-zone). Denote by m_W and m_F the number of working and festivity (W- and F-) days, respectively, in the real dataset (e.g., $m_F = 8$ and $m_W = 20$), and by K the total number of 1-min timebins ($K = 28 \cdot 24 \cdot 60 = 40320$). For each device i consider the vector $\mathbf{d}_i \equiv \{c_i^{\tau_0}(k),\, k = 1, 2, \ldots, K\}$ at the minimum timescale ($\tau_0 = 1$ minute) across the whole real trace duration, where each element $c_i^{\tau_0}(k)$ is the list of the DNS tickets related to device i at time k. For those timebins where device i is inactive, the corresponding element in $\mathbf{d}_i$ is empty. We now divide this vector into m blocks, each one corresponding to a single one-day interval. Each block is classified as W- or F-block based on the calendar day. At this point we apply a random *scrambling* within the W class:

each W-block element of $\mathbf{d}_i$ is randomly relocated at the same time position selected among all W-days. The same scrambling is applied independently to the F-blocks. In this way we obtain a new vector $\widetilde{\mathbf{d}}_i$ where the position of the blocks has been scrambled, separately for W- and F-blocks, but the time location and the F/W intervals have been maintained. Finally, from the set of scrambled vectors $\widetilde{\mathbf{d}}_i$ we can derive a new set of distributions for each timebin k and timescale τ, for all the considered traffic features.

The dataset obtained in this way retains certain characteristics of the real dataset, while others, such as minor statistical fluctuations, are eliminated. The most important change is that the random scrambling of the individual components $\mathbf{d}_i \rightarrow \widetilde{\mathbf{d}}_i$ results in the *homogenization* of the individual daily profiles — separately for W- and F-days. This eliminates any minor residual local anomaly that survived the manual labeling by spreading it out across all one-day intervals of the same F/W type. In other words, all W-days in the new dataset share the same (synthetic) aggregate daily profile. Same applies to F-days. Note however that the synthetic dataset retains the most important characteristics of the real process. In the first place, it keeps the time-of-day variations of the number of active devices (see Figure 6.4). However, the total number of queries at time k changes as permuted devices issue (in general) different amount of DNS queries. Secondly, the semi-synthetic dataset maintains the differentiation between the two classes of W- and F- days, although it eliminates any differentiation *within* each class (e.g., between Saturday and Sunday). Thirdly, it keeps the differentiation between distributions for different time-of-day. This is clear from Figure 6.5(a), which shows the hourly Cumulative Distribution Functions (CDFs) of the number of devices across query count during one day of the semi-synthetic dataset. The result of the procedure is an anomaly-free DNS dataset *structurally similar* to the real trace.

6.7.2. Modeling and generation of synthetic anomalies

During six months of experimentation we encountered a few recurring large-scale DNS traffic anomalies. Investigating these events we found some common traits and we conceived a procedure for reproducing them along with their most relevant characteristics. In particular, we identified two exemplary event types (E_1 and E_2 from now).

In both the cases, we model an outage of an Internet service for a specific sub-population of devices, which react by repeatedly and constantly issuing DNS queries to resolve the requested service throughout the anomaly. Involved devices are identified by fixing a specific OS (with its different versions). Moreover, we aim at modeling the correlation between the selected sub-population and the unreachable service. Therefore, we separately rank the 2LDs of the FQDNs for anomalous and background traffic, and select the most popular 2LD of the former that is not in the latter. As a simple example of such types of anomalies, we have observed events in which Apple devices running a specific version of iOS lost their persistent connectivity to certain servers providing the Apple push-notification service (which is the core of the remote notifications used in virtually every iOS App), resulting in a surge of DNS requests to locate new servers, and the resulting "scanning" of the complete IP address space of Apple push-notification service. Such an event was perceived by the ISP as an internal sort of DDoS attack, as a large population of their own customer devices starting "bombarding" the network, starving resources at the access in some specific regions.

Type	E_1	E_2
Start time t_1	9:00	13:00
Duration d	1h	2 days
Involved devices D	10%	5%
Back-off time	5 sec	180 sec
Manufacturer	single popular	multiple
OS	single (with sub-ver)	single (with sub-ver)
Error flag	+5% timeout	—
FQDN	top-2LD for involved devices	top-2LD for involved devices

Table 6.3.: Characteristics of the anomalous DNS traffic for types E_1/E_2.

Event E_1 This type models the case of a short lived (i.e., hours) high intensity anomaly (e.g., 10% of devices repeating a request every few seconds), where all the involved devices are produced by a single manufacturer and run the same OS. In this case, the number of involved terminals and the overall number of additional queries is such to overload the local DNS servers. The latter effect is modeled by increasing the number of `time-out` codes in the Error Flag field.

Event E_2 This type models a long lasting (i.e., days) low-intensity anomaly (e.g., 5% of devices repeating requests every few minutes). Differently from the previous case, the involved terminals are produced by multiple manufacturers, even if they share the same OS. Given the low-intensity, we did not introduce a modification in the distribution of the Error Flag. Figure 6.5(b) shows the changes in the distribution of number of devices across query counts introduced by this event (cfr. Figure 6.5(a)). Note that although E_2 type anomalies are of relatively low intensity, their identification is important as, in our experience, they may lead to problems on the signaling plane, such as resources starvation at the radio access.

Tab. 6.3 summarizes the characteristics of the two event types and the actual values used for generating the anomalous ticket dataset in the experiments discussed below (cfr. Sec. 6.8).

To illustrate the anomaly generation procedure, we consider an event of type E_1 of duration $d = 1h$, starting at $t_1 = 9:00$. Starting from t_1 at each time-bin, $D = 10\%$ of all the active terminals are randomly extracted from the semi-synthetic background traffic, such that the OS is the selected one and the manufacturer is always the same. For each involved terminal, we generate one additional DNS ticket every 5 seconds, which are then added to the semi-synthetic dataset. The FQDN in these tickets is randomly chosen among the domains in the 2LD identified as explained above. Finally, the Error Flag is changed to `time-out` in 5% of the overall DNS tickets, so as to model the resolver overload. The last step consists of mangling both the anomalous and the background traffic.

The procedure for generating type E_2 is analogous, but differs in the selection of the anomalous terminals (same OS, but not necessarily same manufacturer). The Error Flag is unaffected in this case.

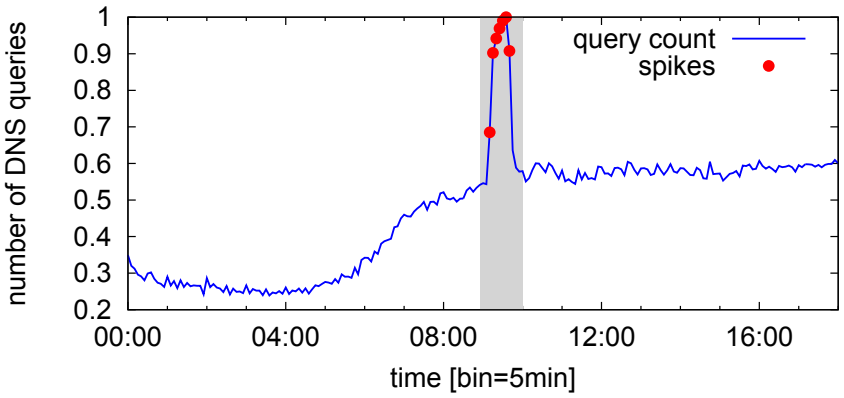


Figure 6.6.: EWMA change point detector applied to symptomatic signal (query count) in event type E_1. The event is highlighted in the gray area. The red dots marked as *spikes* correspond to the alarms flagged by the detector.

6.8. Evaluation with Synthetic DNS Anomalies

In this Section we present the results on the performance evaluation of the framework in terms of detection and diagnosis capabilities, for the case of the aforementioned synthetic anomalies of type E_1 and E_2. For each type of anomaly, we report the results obtained by each of the two proposed detection approaches. Reported results refer to optimal parameter settings in terms of detection capabilities and number of false positives.

We consider as symptomatic signal the distribution of number of devices across query counts, i.e., the amount of the devices issuing a given number of requests within each time bin. In fact, perturbations in this distribution indicate that a device sub-population deviates from the usual DNS traffic patterns, thus pointing to potential anomalies. The diagnostic signals are instead the distributions of query count across the variables of the features previously listed in Table 6.1, excluding the device ID.

6.8.1. Analysis of Event type E_1

As described in Sec. 6.7.2, the first event is characterized by a short duration (1 hour) and a high intensity, as it involves a large population of devices (10%) of the same (popular) manufacturer and running one specific OS. The evaluation is performed at a $\tau = 5$ minutes time scale.

Approach 1. Figure 6.6 shows the time-series of the DNS query count, used as symptomatic signal. The gray area highlights the event time span, from 9am to 10am. The increase on the number of DNS queries is clearly visible, resulting in about the double of queries as observed in normal operation conditions at that time of the day. The red points in the Figure indicate the deviations flagged by the EWMA algorithm. The entropy trend of the diagnostic features is depicted in Figure 6.7. Given the high intensity of the event, marked variations are visible in all the diagnostic signals. In fact, the fraction of DNS queries generated by devices with a specific manufacturer and OS changes during the event, hence

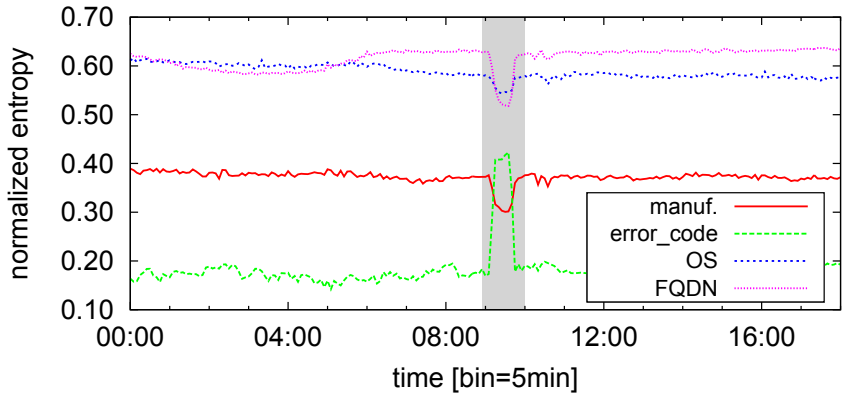


Figure 6.7.: Normalized entropy of diagnostic features in event type E_1. All signals are clearly altered (spikes and notches) during the event.

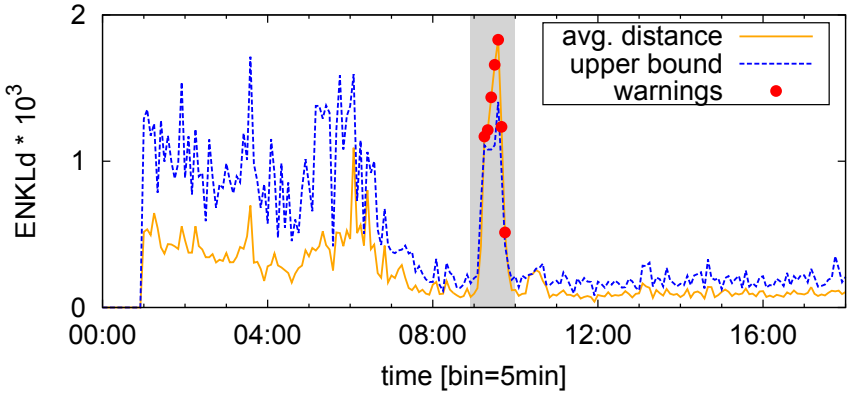
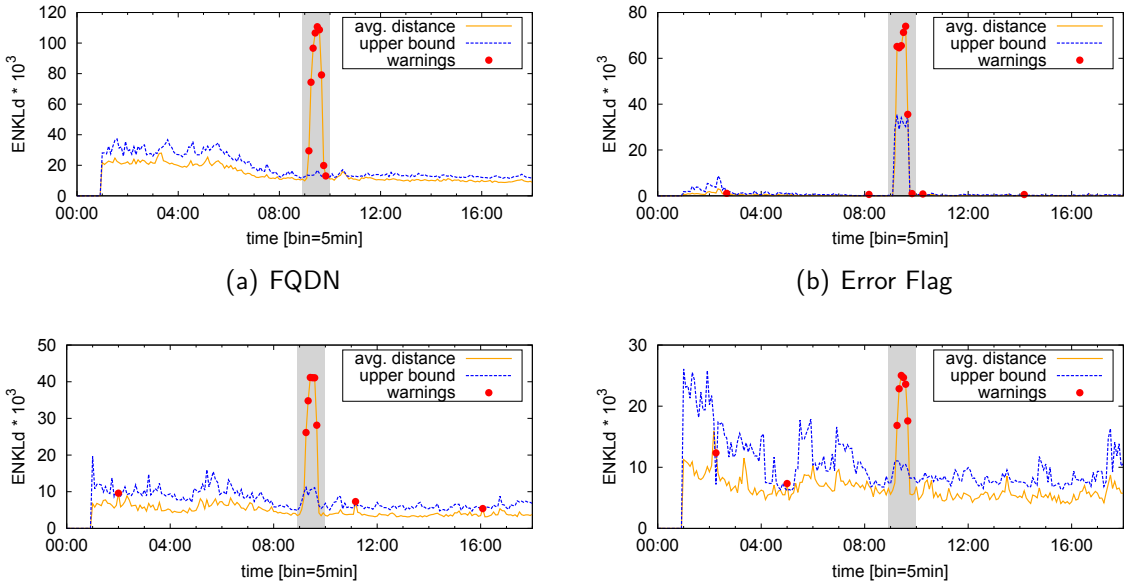


Figure 6.8.: Output of the distribution-based detector for the symptomatic signal (number of devices across query count), in E_1 type event.

the entropy of the respective dimensions exhibit a sharp decrease. Similarly, the FQDN entropy signal decreases, since the affected devices repeatedly try to contact a specific service. On the contrary, the Error Flag diagnostic signal shows a significant increase. In fact, the increased share of `time-outed` queries perturbs the distribution else concentrated around the `successful` value, with a consequent spike in the entropy value. The notches and spikes in the entropy, as well as in the query count, are easily detected by the EWMA algorithm.

Approach 2. Figure 6.8 shows the output of the distribution-based detector: the yellow curve represents the average distance between the distribution of the number of users per query count and the distributions in the reference set, while the blue dashed curve is the upper bound for acceptability. As in the previous case the duration of the event E_1 is highlighted in gray. The timebins where the average distance is above the upper bound are marked with red points. The Figure shows that the distribution deviations are correctly detected. The same applies for the diagnostic signals as depicted in Figure 6.9, showing marked changes in the FQDN, Error Flag, Manufacturer, and OS distributions during the event.



(a) FQDN (b) Error Flag

(c) Manufacturer (d) OS

Figure 6.9.: Output of the distribution-based detector for the diagnostic signals in E_1 type event. All the signals exhibit distribution changes during the event.

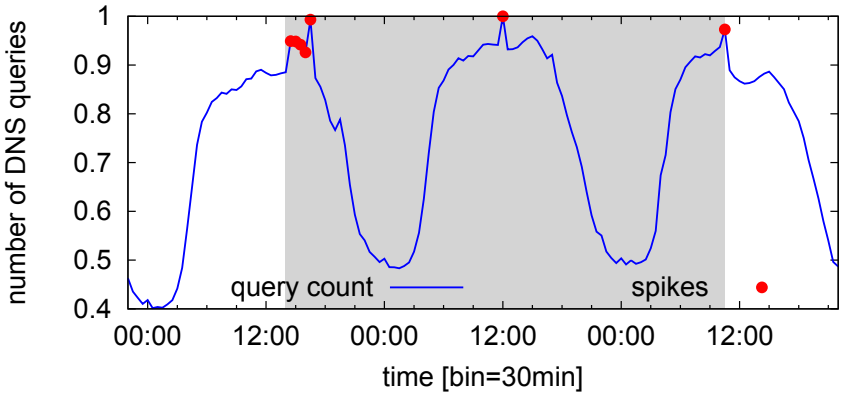


Figure 6.10.: EWMA change point detector applied to symptomatic signal (query count) in event type E_2. The event is highlighted in the gray area. The red dots marked as *spikes* correspond to the alarms flagged by the detector.

Summarizing the performance of both detectors for analyzing anomalies of type E_1, both approaches allow to accurately detect the changes on the symptomatic signal, as well as on the diagnostic signals. The changes on all signals are simultaneously detected, providing a reliable input to the diagnosis step.

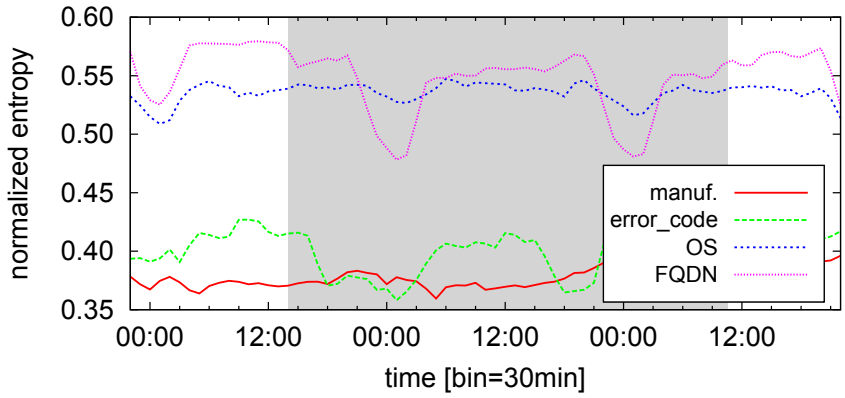


Figure 6.11.: Normalized entropy of diagnostic features in E_2 type event. No clear evidence of the underlying event, with the exceptions of two notches in the FQDN signal.

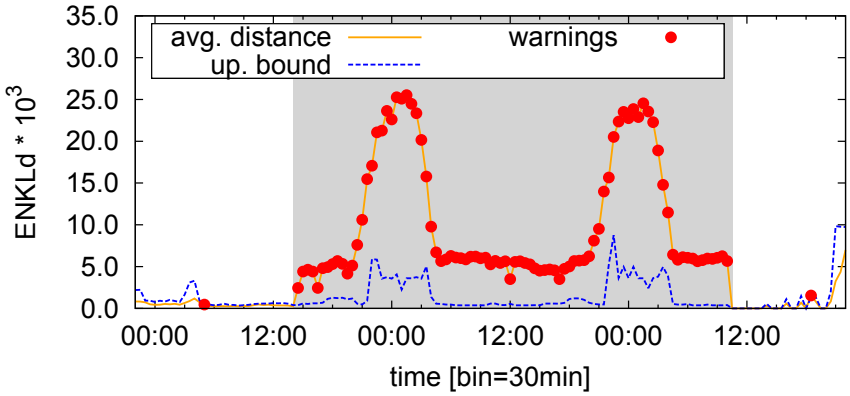
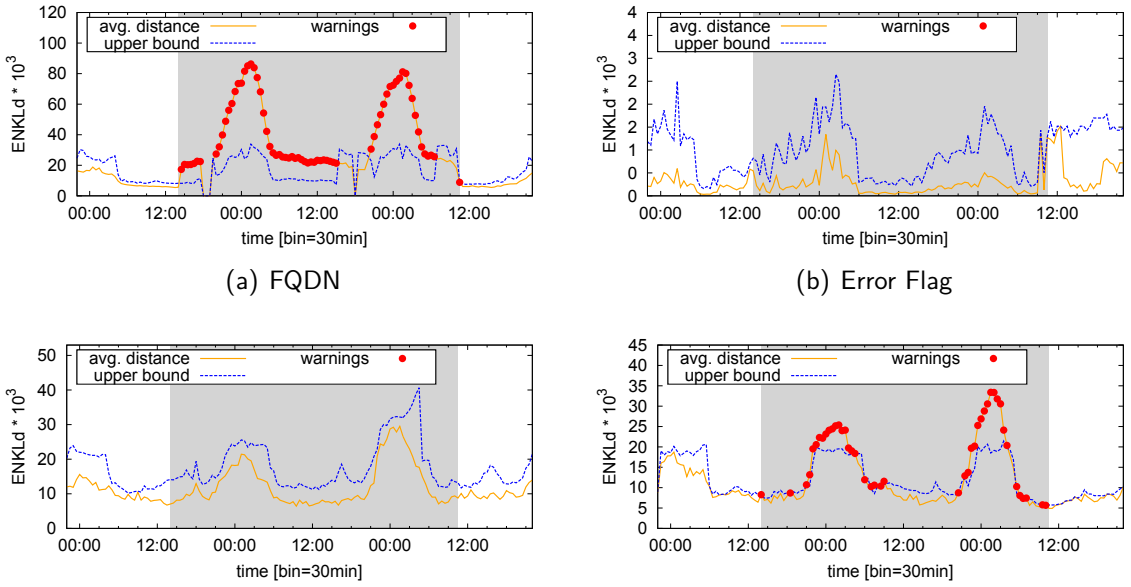


Figure 6.12.: Output of the distribution-bases detection on the symptomatic signal (number of users across query count), in E_2 type event.

6.8.2. Analysis of Event type E_2

Differently from E_1, the type E_2 anomaly involves a smaller population of devices (5%) running an OS pre-installed by a number of different manufacturers. Similarly to the previous case, the affected terminals continuously try to re-contact the servers hosting an unreachable service. Because of the low intensity and the longer duration of the event, the analysis is performed at a $\tau = 30$ minutes time scale.

Approach 1. Figure 6.10 depicts the time-series of the query count during a period of 3 days, which includes the anomalous event, starting at 1pm of the first day and lasting till 11am of the third day. The counter shows a slight increase during the anomaly, but the EWMA detection algorithm only flags changes at the beginning, and is not able to track the anomaly during its complete time span. Missing the detection on the symptomatic signal is especially serious as it compromises the whole diagnosis process. Figure 6.11 plots the trend of the diagnostic signals. Only the FQDN entropy exhibits evident changes during the



(a) FQDN (b) Error Flag

(c) Manufacturer (d) OS

Figure 6.13.: Output of the distribution-based analysis for the diagnostic signals in E_2 type event. Distributions of FQDN and OS exhibit changes during the event, while manufacturer and Error Flag are unaffected.

night-time, when the increased number of requests for the affected service stems out from the background night traffic. For the rest of the diagnostic signals, it is hard to claim that the small, low-speed observed changes could be detected, specially as they look very similar to the patterns observed during normal operation. Indeed, the EWMA algorithm fails to track the full dynamics of the event. Regarding the OS signal, the changes in the distribution induced by E_2 are not sufficient to alter the entropy signal. We recall that E_2 does not affect the Error Flag signal by design.

Approach 2. Contrarily to the previous case, the distribution-based approach detects low-intensity anomalies involving multiple devices. Figure 6.12 plots the output of distribution-based algorithm for the symptomatic signal. The average distance (yellow curve) flags changes in the distributions of the number of users across query count, which correspond to deviations in the CDFs shown in Figure 6.5(b). Therefore, the approach is able to capture and detect the entire dynamics of the event. The distance between the two curves is more marked during the night hours, when the number of DNS queries related to the anomaly are statistically more relevant, cfr. Figure 6.5(b). The output of the distribution-based detector applied to the diagnostic signals is shown in Figure 6.13. The FQDN signal output, depicted in Figure 6.13(a), is correlated with the symptomatic signal: the plot reports a sequence of drifts from the reference set highlighting the whole span of the event. As in E_2 there is no anomalous behavior on the Error Flag distribution, a correct functioning of the detector would result in no alarms for this signal, which is exactly depicted in Figure 6.13(b). Figure 6.13(c) shows that also the manufacturer dimension is not involved, while in Figure 6.13(d) there

are evidences of the OS-related nature of the anomaly.

In conclusion, the experiments show that lower intensity anomalies are not correctly captured by the Approach 1, as the entropy is a too coarse metric, failing to reveal the effects of this type of anomalies.

6.8.3. Comparing Detection Strategies

For a better comparison of the two detection strategies, we have also investigated the behavior of algorithms for different parameters settings.

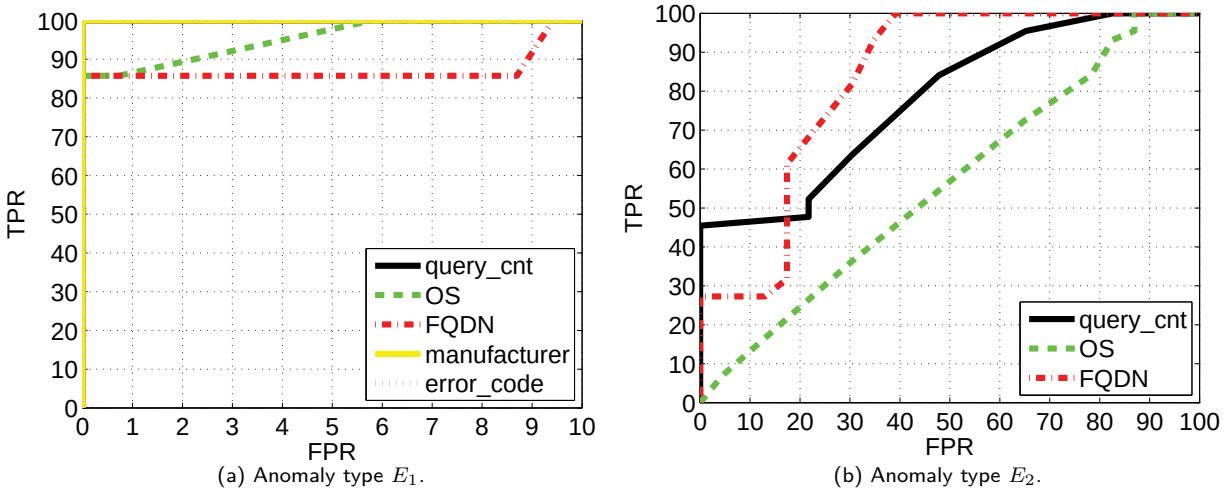
Figure 6.14 depicts the ROC curves obtained in the detection of the two events. The curves reflect the True Positive and False Positive Rates (TPR and FPR) obtained when changing the detection thresholds of both approaches. Each anomalous sample corresponds to a 5 minutes time bin, during the entire span of the anomaly (about 1hr for E_1, and about 2 days for E_2). The symptomatic signal is in both cases the DNS query count per device, using either its entropy or the full distribution. Given the characteristics of E_1, there are four relevant diagnostic signals which show an abrupt change at the time of the anomaly: the manufacturer and OS (same type of devices are impacted), the FQDN (points to the requested, unavailable service) and the error code (the local DNS servers get overloaded and time-outs increase). By contrasting Figures 6.14(a) and 6.15(a), it is evident that both approaches are capable of detecting the abrupt changes induced by this anomaly, resulting in almost perfect detection for the impacted signals, assuming an optimal tuning of the two algorithms.

In the case of E_2 type anomaly, by design, the only impacted diagnostic signals are OS and the FQDN. However, Figure 6.14(b) shows that the entropy-based approach completely fails to flag and characterize the anomaly, as the FPR becomes too high for being applied in practice, both for the symptomatic and the diagnostic signals. On the contrary, Figure 6.15(b) shows that the performance of the distribution-based approach is superior, reinforcing the evidence of its supremacy against entropy-based analysis.

Figure 6.15(a) depicts the Receiver Operating Characteristic (ROC) curves for both the symptomatic and diagnostic signals for the event of type E_1. The Figure shows that for almost all the signals the algorithm attains perfect detection performance tolerating at most 3% of false positives, whereas for detecting 90% of Manufacturer's distribution changes 6% of false positives should be expected. For the event E_2, Figure 6.15(b) shows perfect results for the symptomatic signal (i.e., query count), whereas for the diagnostic signal 90% TP is reached with 10% FP. The slightly lower precision is due to the smaller population involved in the anomaly, which in turn induces smaller distribution changes in the diagnostic signals. Therefore, we have to allow few more false positives, caused by normal fluctuations in the traffic, in order to correctly flag all the anomalous time bins.

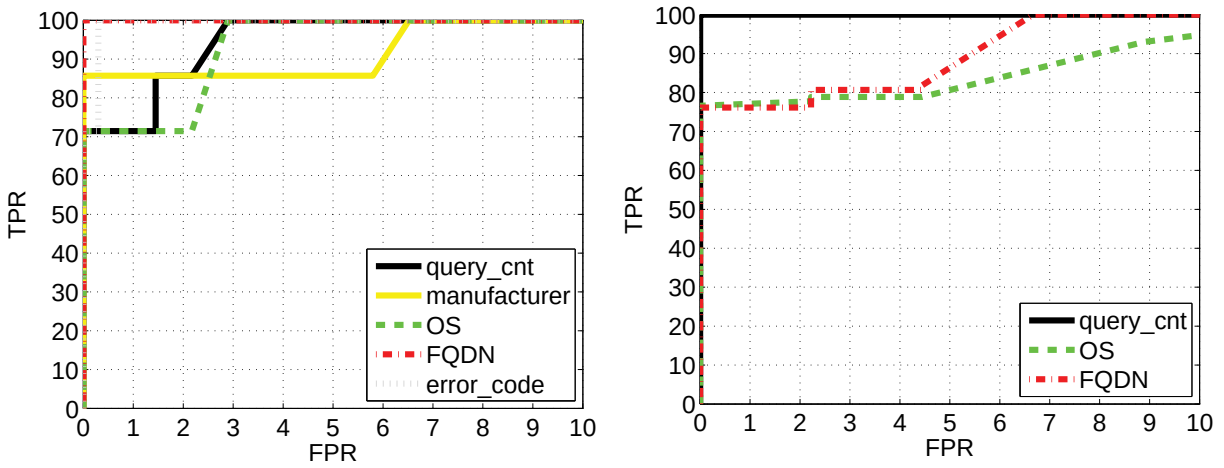
6.8.4. Test on different intensities

To further compare the two detectors, we have generated variants of E_1 and E_2 changing the fraction of the device population involved in the anomaly between 0.1% and 20%. As an example we report in Figure 6.16 results of such an investigation for the diagnostic feature



(a) Anomaly type E_1. (b) Anomaly type E_2.

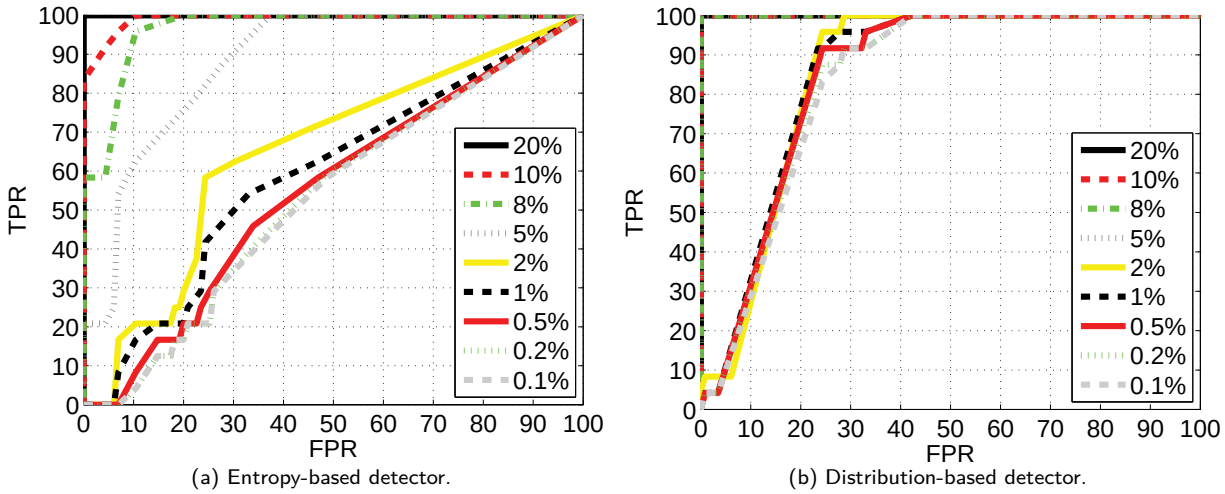
Figure 6.14.: ROC curves for the detection of changes in the corresponding symptomatic and diagnostic signals. Entropy-based detection performs properly with anomaly E_1, but completely fails with E_2.



(a) Anomaly type E_1. (b) Anomaly type E_2.

Figure 6.15.: ROC curves for the detection of abrupt changes in the corresponding symptomatic and diagnostic signals. The distribution-based detector performs well on both events.

OS, which showed to be the most difficult to be revealed, among the others. In particular, Figure 6.16(a) shows that the performance of the entropy-based detector falls below acceptability already when the fraction of the user population involved in the anomaly goes below 5%. On the contrary, Figure 6.16(b) shows that the performance of the distribution-based detector are quite good and are practically independent from the number of devices involved in the anomaly. Similar results have been obtained on other signals, and are omitted for brevity.



(a) Entropy-based detector. (b) Distribution-based detector.

Figure 6.16.: ROC curves for the detection of changes in the symptomatic signal OS, for different percentage of the devices population involved in the anomalies.

Resuming, evaluations show that lower intensity anomalies tend to not be correctly flagged by the entropy-based detector, as in this case the entropy results into a too coarse metric for revealing the effects of this type of anomalies, irrespective of the algorithm used for flagging abrupt changes. This limitation calls for the adoption of the distribution-based approach that from our experiments appears to be better suited for working properly in a number of different scenarios. This approach has the only drawback of a much higher computational cost, both in terms of CPU and memory. From our operational experience, however, this algorithm is still feasible in networks of millions of users by using reasonable hardware coupled with the online processing capabilities of the stream data warehouse we use, DBStream.

Field Name	**Description**
client IP	Anonymized device identifier
server IP	remote YouTube server IP address
avg download rate	average flow down-link throughput
elaboration time	delay between client request and server reply
external RTT	RTT measured between VP and remote server
internal RTT	RTT measured between VP and end device
beta	ratio between video bit-rate and throughput

Table 6.4.: Tstat flow-level ticket information.

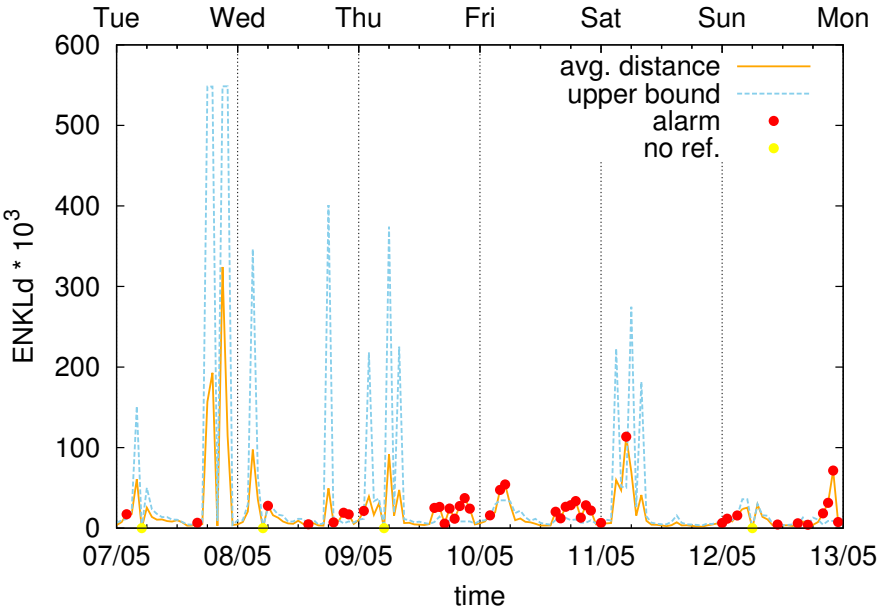


Figure 6.17.: Output of the distribution-based analysis for the symptomatic signal distribution of Average download rate, used as trigger for the diagnosis procedure during the YouTube anomaly.

6.9. A Real-World Scenario

After the evaluation of the two detection strategies illustrated in the previous Sections using synthetic traffic, we now apply the distribution-based approach on a number of traffic features in a real-world anomaly. The use case we refer is one of those characterized in Chapter 5: the degradation of the perceived QoE for a large user population accessing YouTube video contents.

This anomaly has been already extensively discussed and characterized in Section 5.6, based on expert knowledge and ad-hoc analysis. Here we show that the entire procedure can be automatized, to a large extent, adopting the proposed distribution-based detection and diagnosis procedure.

We recall that the origin of the analyzed anomaly is the cache selection policy applied by Google from Wednesday on, and more specifically, the servers selected between 15:00 and 00:00 that were not correctly dimensioned to handle the traffic load during peak hours, between 20:00 and 23:00, leading to users' Quality of Experience (QoE) degradation. In Section 5.6 we have shown that it is possible to detect such an anomaly analyzing the time series of the distribution of the Average video download rate, along with the median of the β parameter—a QoE based Key Performance Indicator (KPI) defined as the ratio between the average download rate and the video bit rate—which allows to estimate the presence of stallings in the video playback.

Figure 6.17 plots the output of the distribution-based detector for the average download rate, and flags the presence of changes during the peak hours from Wednesday to Friday

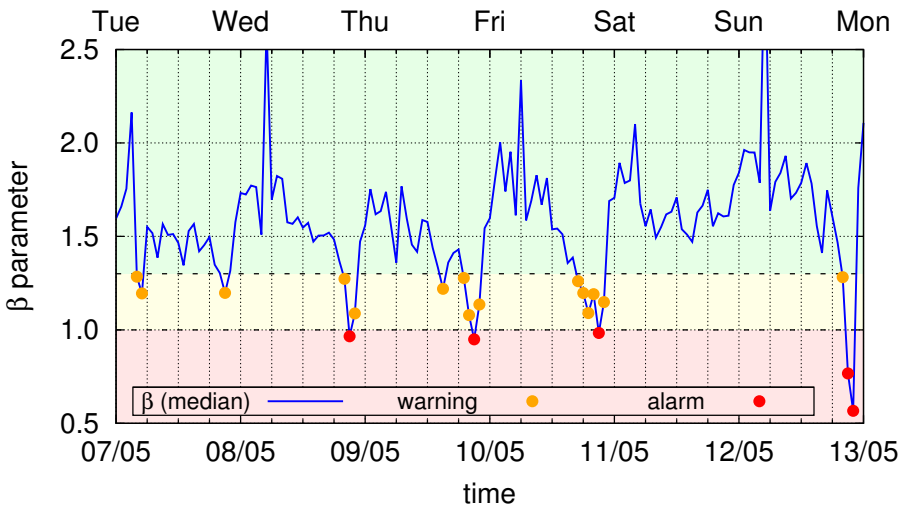
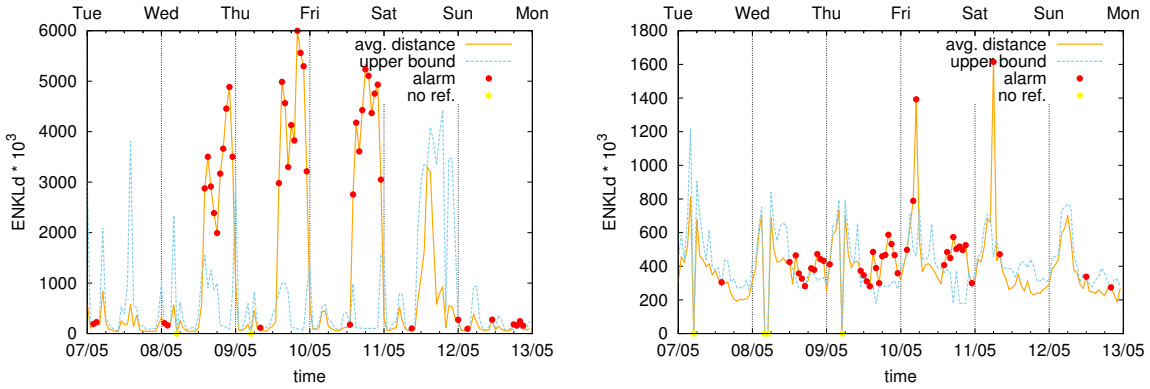


Figure 6.18.: Median of β per hour for all YouTube flows. The acceptance thresholds are highlighted with different colors.

and on Sunday. Figure 6.18 plots the trend of the median β parameter in the period and two thresholds for $\beta = 1$ and $\beta = 1.25$, which in turn identify three regions for the video QoE, i.e., bad, poor and fair represented with red, orange and green, respectively. These thresholds are derived from the QoE mappings previously presented, and correspond to 400 and 800 kbps, respectively, in case of 360p average bit rate videos. The Figure reports a reduction of the throughput on Tuesday at peak-load time, between 20:00 and 23:00 UTC. However, from Wednesday on, this drop gets below the bad QoE threshold. The drop in the throughput coupled with the marked drop in the time series of β allows to reveal the presence of a change that is heavily affecting the user experience. Therefore, we use them as symptomatic signals.

The list of features we are using for the diagnosis process is summarized in Table 6.4. Given that the diagnosis part focuses on the YouTube servers, as diagnostic signals we have considered the distribution of flows per server IP, and the elaboration time (i.e., the time elapsed from the video request and first returned video segment). Furthermore, we have considered the minimum internal and external RTT, which are representative of the network distance from the vantage point to the end device and from the vantage point to servers, respectively. Results reported in Figure 6.19(a) show that a different set of Google servers was selected to serve the YouTube traffic in the afternoon from Wednesday onward. Also, Figure 6.19(c) and Figure 6.19(d) show that the new servers where further located from our vantage point, and that there was no relevant ISP internal routing change in the same period. However, the selection of the new servers negatively impacted the elaboration time (see Figure 6.19(b)), to the point that the perceived service QoE fell below the acceptability threshold for a considerable share of the user population (cfr. Figure 6.17).

Final diagnosis. To conclude, the final diagnosis of the event is that a new cache selection policy applied by Google from Wednesday on provokes an anomaly, i.e., a decrease of average downlink throughput with consequent QoE degradation. The presence of the



(a) Server IPs

(b) Elaboration Time

(c) External RTT

(d) Internal RTT

Figure 6.19.: Output of the distribution-based analysis for the diagnostic signals in the YouTube anomaly. The anomaly is caused by a shift in the distribution of flows across server IPs. It changes in the distribution of elaboration times and internal/external RTT complement the picture on the event and support the diagnosis process.

new policy is confirmed by two diagnostic signals (distribution of flows per server IP and per external RTT bins). The new servers deployed in the afternoon, from 15:00 to 00:00 were not correctly dimensioned to handle the traffic load during peak hours, between 20:00 and 23:00, as indicated by the change in the elaboration time distribution.

Notice that by combining the detector output for the symptomatic and diagnostic signals, we have automatically drawn the same conclusions as already obtained manually in Section 5.6. Next Chapter is devoted to the procedure used to fully automatize this diagnosis process.

6.10. Summary

In this Chapter we proposed the design and evaluation of some building blocks for an automatic detection and diagnosis framework. In particular, we focused on the module responsible to detect changes in the monitored traffic features. Our results unveiled the limitations of using simple change point detection algorithms in the case of low intensity anomalies that involve relatively small sub-populations of users. From our experience, this type of anomalies are frequent in operational mobile networks, and still far from being innocuous (cfr. problems on the signaling plane). To overcome this limitation, we have presented a more complex change detection scheme that relies on the entire probability distribution of the monitored signals rather than the entropy values. Using this detection approach, the system is able to cope with anomalies that involve multiple services and/or affect multiple devices at the same time.

Given the general lack of large-scale ground-truth datasets to test the performance of systems like ours, we developed an approach to generate semi-synthetic data, derived from real traffic traces. For our tests, we have generated synthetic datasets starting from real DNS traces. However the approach could be easily exploited in different kind of measurements, generating in this way different symptomatic and diagnostic signals. Even if the traffic generator was done for the sake of evaluating our detection algorithms, we believe that is a nice side-contribution of this thesis, as it would help the owners of real data to make such datasets available for the research community without disclosing any privacy or business sensitive information.

Finally we have tested our detection approach on one of the real-work use cases presented in the previous Chapter. The results we obtained were very encouraging and demonstrate the effectiveness of our techniques confirming the same diagnosis we previously obtained by manual inspection.

In the next Chapter we complete the picture of the proposed framework by describing its final component, i.e., the *diagnosis module*, responsible for correlating the changes, classifying the anomalies and automatically reporting the findings.

7. Towards Automatic Diagnosis of Anomalies

Notice of adoption from previous publications

Parts of the contents of this Chapter have been published in the following papers:

[P10] P. Casas, A. D'Alconzo, **P. Fiadino**, A. Bär, A. Finamore, T. Zseby, "When YouTube doesn't Work – Analysis of QoE-relevant Degradation in Google CDN Traffic", in *IEEE Transactions on Network and Service Management*, vol. 11, no. 4, 2014.

[P19] **P. Fiadino**, M. Schiavone, A. D'Alconzo, P. Casas, "Towards Automatic Detection and Diagnosis of Internet Service Anomalies via DNS Traffic Analysis", in *International Wireless Communications & Mobile Computing Conference - TRAC (IWCMC 2015)*, 2015.

[P20] **P. Fiadino**, A. D'Alconzo, M. Schiavone, P. Casas, "RCATool – A Framework for Detecting and Diagnosing Anomalies in Cellular Networks", in *27th International Teletraffic Conference (ITC27)*, 2015.

The author of this thesis had a major role in designing the automatic diagnosis procedure and run the experiments that led to the results described in the Chapter. The work has been supervised by Dr. Pedro Casas, Dr. Alessandro D'Alconzo and Prof. Tanja Zseby.

Acknowledgments

I acknowledge Sarah Wassermann (University of Liège, Belgium) for her recent work on DisNETPerf (cfr. Section 7.8.2), which was used for a preliminary investigation of reactive monitoring approaches and iterative diagnosis. Her work, together with other results, is currently under review:

[P21] P. Casas, **P. Fiadino**, S. Wassermann, S. Traverso, A. D'Alconzo, E. Tego, F. Matera, M. Mellia, "Unveiling Network and Service Performance Degradation in the Wild with mPlane", in *IEEE Communications Magazine, Network Testing Series* (under review).

7.1. Introduction

In this Chapter we continue the description of the sequential diagnosis framework previously introduced. We have already studied the components responsible for the signal extraction and change detection. We recall that the detection is achieved by extracting and analyzing *symptomatic* traffic features, and by flagging a warning as soon as one or more of them show a significant change. We now move to the final part of the framework, responsible for the

actual diagnosis of the detected anomalies. So far, we have considered the signals, and their flagged changes, independently. Our goal is now gathering all the relevant information to automatically provide a holistic view of the anomaly, including the correlation of the involved traffic features and an indication of the anomaly type.

As we shall see, the automation of the diagnosis process is achieved by exploiting Machine Learning (ML) based techniques, applying supervised approaches to automatically and rapidly classify the detected anomalies. We will test and evaluate a number of different algorithms and assess the benefits of applying feature selection. The framework ultimately collects all information in order to issue plain-text reports containing the most significant and correlated symptomatic and diagnostic details. Such a report could highly ease the interpretation of the potential problem by the operator. All the detected changes are managed in a state-full fashion, tracking the evolution of anomaly alarms across time. This allows a better understanding of the problem, as gives a notion of when an anomaly has started and how it evolves, till its resolution.

In the last part of the Chapter, we also provide initial guidelines for an evolution of our diagnosis framework. In particular, we explore the advantages and challenges associated with a *reactive* and distributed monitoring approach, which paves the way to a more advanced iterative diagnosis process. We include preliminary results tackling the analysis of Internet-paths performance through distributed active measurements, complementing the passive analysis approaches presented throughout this work. These results represent a valuable input for future research, as demonstrated by the work being developed by Sarah Wassermann in the context of her master studies.

7.2. Related Work

The problem of automatically diagnosing network anomalies is pretty well known in the Traffic Monitoring and Analysis (TMA) community, but still represents a fertile research domain, as the goal is far from being achieved.

The diagnosis of network and traffic anomalies in operational, large-scale networks dates back to the initial work of Lakhina et al. [108] in 2004 (some previous papers worked in limited or simulated datasets, mainly for the purpose of network security, which is out of our scope), where authors proposed the well-known PCA-based approach for detecting, locating and quantifying the volume of network wide anomalies in a traffic matrix. This work does not really classify the types of anomalies detected, but is probably one of the first in introducing the term of network anomaly diagnosis in modern networks. A further step was taken years later in [120], where authors study the dynamics of routing data to characterize network anomalies impacting network performance. Silveira et al. introduced URCA in [121], an approach to classify network anomalies based on manually built signatures and hierarchical clustering. In [122], authors proposed a generic technique that uses frequent item set mining to automatically extract and summarize the traffic flows causing network security anomalies.

More recently, Kanuparthy et al. proposed in [12, 123] a monitoring system based on distributed active measurements which allows operators to input domain knowledge to enhance diagnosis functionality for the purpose of troubleshooting performance issues. In [124],

Yan et al. introduced a system for end-to-end anomaly detection and localization in CDN networks, presenting results on its functioning on an operational CDN. Our work builds close to a work of the same authors [125], where authors present a generic Root Cause Analysis systems to capture and explain the sources of network problems. We follow a similar approach in our proposal, also distinguishing between symptomatic and diagnostic events for the sake of understanding anomalies, but recasting the distinction in terms of features and corresponding signals.

Finally, regarding the Machine Learning (ML) based approach for automatically classifying anomalies, the field of automatic traffic analysis and classification trough ML techniques has been extensively studied during the last decade. We recall from the related work section of Chapter 3 a detailed survey of ML techniques applied to automatic traffic classification available at [38]. In particular, a standard non-exhaustive list of supervised ML-based approaches includes the use of Bayesian classifiers [39], linear discriminant analysis and k-nearest-neighbors [40], decision trees and feature selection techniques [41], and support vector machines [42]. Unsupervised and semi-supervised learning techniques have also been used before for traffic analysis and classification, including the use of k-means, DBSCAN, and AutoClass clustering [126], and a combination of k-means and maximum-likelihood clusters labeling [127].

7.3. Dataset Description

Following the same data generation approach we have used so far (cf. Section 6.7), we construct a fully labeled dataset in which we add 16 network and service anomalies from three different anomaly classes, all of them affecting mobile devices with multiple intensities in terms of size of the affected population. In particular, the first two classes of anomalies correspond to the same types of anomalies reflected by types E_1 (type 1) and E_2 (type 2) in Section 6.7, but considering different durations: type 1 anomalies last for 2 hours, and type 2 anomalies last for 1 day. In both cases and as we did before, the size of the impacted number of devices varies between 20% and 0.5%, resulting in a total of $2 \times 7 = 14$ anomalies. We additionally introduce a third class of anomalies (type 3) which models a scenario in which all the customers of certain virtual operators (reflected by specific APNs) are affected by service outages, responding with a surge in the number of DNS queries. We take two different intensities for this anomaly class, considering a population of 12% and 3% respectively and a duration of 1 hour in both cases. Similarly to the other types, the characteristics of these type 3 anomalies have been chosen to mimic real-world events that we have observed during our measurements, including the fractions of involved population, which reflect the size of real virtual-operator customer bases.

The generated dataset finally consists of one full month of synthetically generated cellular-network DNS data, corresponding to real measurements performed in October 2013, and reported with a time granularity of 5 minutes. The dataset contains normal operation traffic, with the 16 aforementioned anomalies added on top of it.

Table 7.1 describes the set of 48 features and signals which are reported for every 5 minutes time bin of the synthetically generated dataset (i.e., the columns of the dataset, describing

each row). These are used as input for the ML-based classifier, which finally provides a classification label for every 5 minutes time bin: 0 in case of normal operation, and 1, 2 or 3, depending on the specific anomaly type. The set of features corresponds to the meta-data obtained from the DNS transactions, as explained in Section 6.7. For each feature we compute multiple percentiles as signals, additionally including the average value and the entropy. Finally, we also use as input to the ML-based classifier the output obtained from both anomaly detectors previously introduced, as these provide paramount information to understand the nature of the flagged changes.

Using such a broad set of features might be a priori against the intuition of the reader, specially because we have clearly identified in Section 6.7 the best features describing the statistical properties of the targeted anomalies. However, it is generally not possible to know in advance which is the best set of features to use in the practice. Therefore, we consider a more conservative analytical approach, taking as input a broad set of features, using later on feature selection and specific ML-based approaches to pinpoint the best of them for the corresponding targets. Note also that the descriptors used in Section 6.7 refer to the full probability distributions of the corresponding features, thus we include as signals a sampling of such distributions, represented by the five considered percentiles, the average values, and the entropy. Finally, note that we also include as inputs the output of the EWMA detector on the entropy of each of the main considered features, despite the low correlation to the anomaly types these offer, as shown in the evaluations on Section 6.8. The purpose is still to verify that the information provided by the EWMA detector is less relevant than the one provided by the distribution-based detector. In fact, the feature selection process automatically accounts for such main difference in the information provided by both detectors, as shown in Section 7.6.

7.4. Compared ML Approaches and Criteria

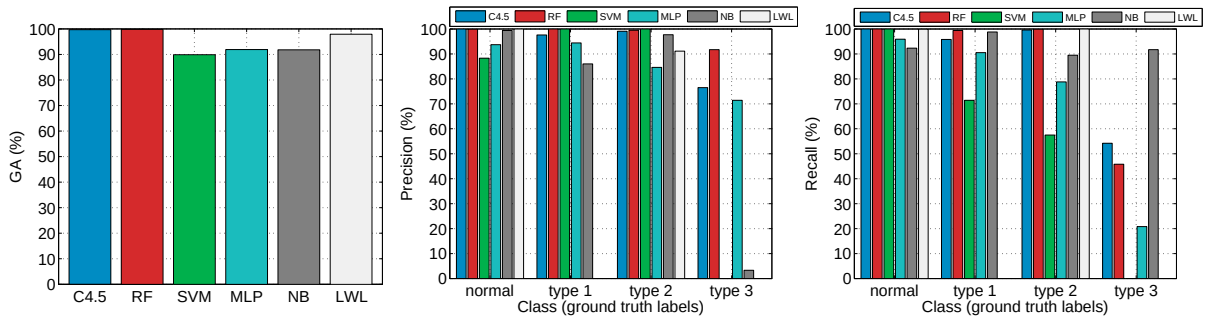
The literature offers multiple types of ML-based classifiers, covering a very wide range of approaches and techniques [38]. Many of the approaches offer "black-box" solutions, for which it becomes very challenging to understand the reasons of a particular classification result, and in particular to understand the input features leading to such a result. Decision trees are therefore a very appealing option when thinking on easing the tasks of a network operator, as they are very easy to interpret, and directly provide filtering rules, which are the basis of a network operator's job. Decision trees are one of the most powerful and simple data mining methods for decision-making, and they additionally permit to construct comprehensive signatures for the detected anomalies, using a graph structure.

Following the results of [43], we decided to build a classifier based on standard C4.5 decision tree, and compare its performance to that obtained through five standard supervised-learning-based approaches previously used in the literature: Multi-Layer Perceptron (MLP), Artificial Neural Networks, Naive Bayes (NB), Random Forest (RF), Support Vector Machines (SVM), and Locally-Weighted-based Learning (LWL). The output of the classifier is a label reflecting either normal operation (label 0) or flagging one of the specific anomaly types (labels 1, 2 or 3). We use the well-known Weka Machine-Learning software tool [30] to calibrate the six

Feature	Signal	Description
serial time	–	start of the 5' time-bin
DNS_query	querycnt	total num of DNS requests
	querycnt_ewma	output of EWMA det. (-1,0,1)
	querycnt_adtool	output of dist-based det. (≥ 0)
APN	apn_h	$H(\text{APN})$
	apn_avg	$\overline{\text{APN}}$
	apn_p99	99th-percentile
	apn_p75	75th-percentile
	apn_p50	50th-percentile
	apn_p25	25th-percentile
	apn_p05	5th-percentile
	apn_ewma	output of EWMA det. (-1,0,1)
	apn_adtool	output of dist-based det. (≥ 0)
Error_flag	error_code_h	$H(\text{Error\_flag})$
	error_code_avg	$\overline{\text{Error\_flag}}$
	error_code_p99	99th-percentile
	error_code_p75	75th-percentile
	error_code_p50	50th-percentile
	error_code_p25	25th-percentile
	error_code_p05	5th-percentile
	error_code_ewma	output of EWMA det. (-1,0,1)
	error_code_adtool	output of dist-based det. (≥ 0)
Manufacturer	manufacturer_h	$H(\text{Manufacturer})$
	manufacturer_avg	$\overline{\text{Manufacturer}}$
	manufacturer_p99	99th-percentile
	manufacturer_p75	75th-percentile
	manufacturer_p50	50th-percentile
	manufacturer_p25	25th-percentile
	manufacturer_p05	5th-percentile
	manufacturer_ewma	output of EWMA det. (-1,0,1)
	manufacturer_adtool	output of dist-based det. (≥ 0)
OS	os_h	$H(\text{OS})$
	os_avg	$\overline{\text{OS}}$
	os_p99	99th-percentile
	os_p75	75th-percentile
	os_p50	50th-percentile
	os_p25	25th-percentile
	os_p05	5th-percentile
	os_ewma	output of EWMA det. (-1,0,1)
	os_adtool	output of dist-based det. (≥ 0)
FQDN	req_fqdn_h	$H(\text{FQDN})$
	req_fqdn_avg	$\overline{\text{FQDN}}$
	req_fqdn_p99	99th-percentile
	req_fqdn_p75	75th-percentile
	req_fqdn_p50	50th-percentile
	req_fqdn_p25	25th-percentile
	req_fqdn_p05	5th-percentile
	req_fqdn_ewma	output of EWMA det. (-1,0,1)
	req_fqdn_adtool	output of dist-based det. (≥ 0)
label	–	ground truth label (0 - normal, type 1-2-3)

Table 7.1.: Features and signals used as input for the Machine Learning-based anomaly classifier (note that the serial time and the label are not considered as inputs). The signals include the entropy of the corresponding feature, multiple percentile values and the output of the two detection algorithms (i.e., EWMA applied on the entropy time-series and distribution-based detector).

learning-based algorithms and to perform the evaluations. Even though we do not explain the particular details of each of these classifiers, we refer the reader to Appendix B for a brief description on these algorithms, including their parametrization. For a more complete survey of ML techniques applied to network traffic classification, we address the interested reader to [38] and to the Weka documentation [30].



(a) Global Accuracy. (b) Precision. (c) Recall.

Figure 7.1.: Machine Learning based approaches for anomaly classification. Classification Accuracy, Precision, and Recall for normal operation instances and different anomaly-types' events. The performance of C4.5 trees is almost perfect for normal traffic and anomalies of type 1 and 2, but quality significantly drops for the anomaly type 3.

To evaluate and compare the performance and virtues of the classification models, we consider three standard metrics: Global Accuracy GA, Recall and Precision. GA indicates the percentage of correctly classified instances (time-bins) among the total number of instances. Recall R_i is the number of instances from class $i = 0, \ldots, 3$ correctly classified (TP_i), divided by the number of instances in class i (n_i). Precision P_i is the percentage of instances correctly classified as belonging to class i among all the instances classified as belonging to class i, including true and false positives (FP_i). Recall and precision are two widely used performance metrics in classification. Precision permits to measure the fidelity of the classification model regarding each particular class, whereas recall measures the per-class accuracy.

$$\mathrm{R}_i = \frac{TP_i}{n_i}, \quad \mathrm{P}_i = \frac{TP_i}{TP_i + FP_i}, \quad \mathrm{GA} = \frac{\sum_{i=1}^{M} TP_i}{n} \tag{7.1}$$

7.5. Evaluation and Discussion

Figure 7.1 reports the performance of the six compared classifiers in the classification of all the 5-minutes time-bins. All the evaluations presented use 10-fold cross-validation, which means that we train and test the models for 10 different training/testing combination sets, to avoid biased results. For the sake of a fair comparison, parameters are set manually for all the models, performing an extensive trial-and-error testing phase to obtain the best results (the adopted parameters are reported in Appendix B). Figure 7.1(a) depicts the global accuracy obtained by the six approaches. All the models provide very high accuracy, above 90%. The C4.5 decision tree model achieves the same performance as the RF, but the latter uses 20 parallel C4.5 decision trees instead of a single one. SVM, MLP and NB achieve slightly worse performance in terms of accuracy, which is a-priori surprising, as at least SVMs and MLPs

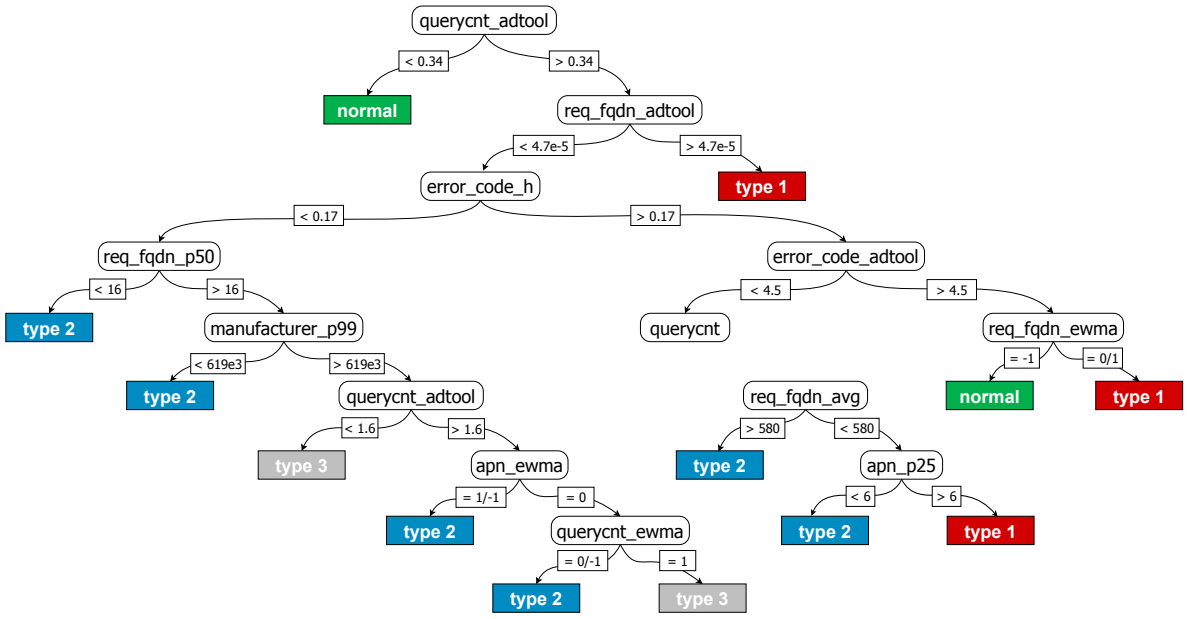


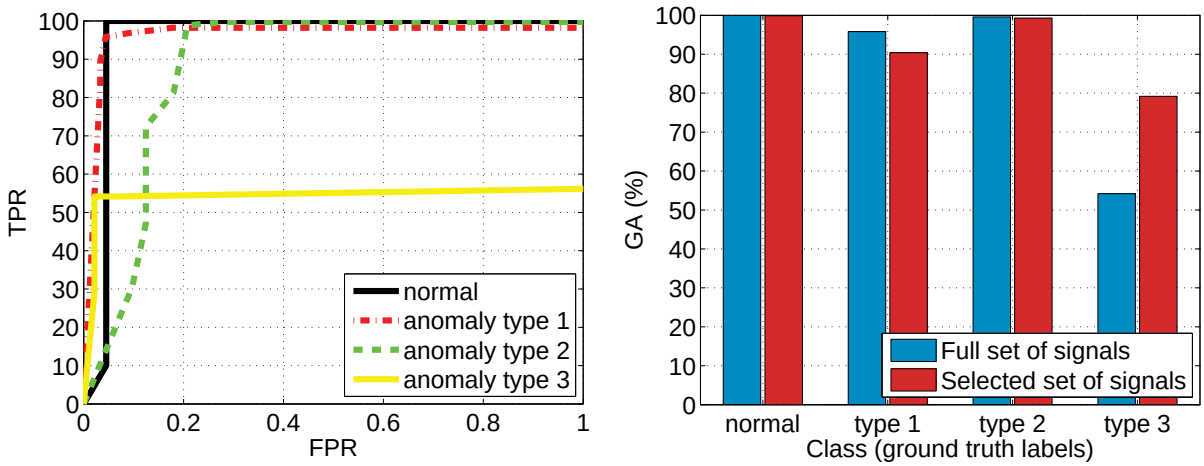
Figure 7.2.: Pruned C4.5 decision tree model for anomaly diagnosis. C4.5 achieves very high global accuracy, as well as very high precision and recall for normal traffic and type 1, type 2 anomalies. However, this tree is not capable of properly tracking type 3 anomalies. This issue can be solved by performing pre-filtering on the input features, by feature selection techniques.

have proved to be very good classifiers in previous work.

Regarding precision and recall depicted in Figures 7.1(b) and 7.1(c), we can observe that all the approaches systematically fail to properly track the type 3 anomalies. C4.5 achieves high precision and recall for normal and type 1, type 2 anomalies, but also fails to properly isolate type 3 events, resulting in a very low recall. While the problem of unbalanced classes is for sure an issue partially masking these results, the particularities of type 3 anomalies require additional efforts to properly track them. Indeed, as also shown in Figure 7.3(a), while the per-class ROC curves obtained for the first 3 classes (0, 1, and 2) by C4.5 are almost perfect (TPR = 100% for a FPR below 1%), the ROC curve for the type 3 events shows poor results. As we see next, we can greatly improved the performance of C4.5 for classification of type 3 events by performing pre-features filtering.

Finally, Figure 7.2 depicts the obtained C4.5 model. A big advantage of C4.5 is the much smaller effort required for the tuning, with respect to other more complex algorithms. After the experimentation, in fact, we relied on the standard parametrization to obtain optimal settings. Among the tested algorithms, it is the best one to support the automation of the diagnosis process, easing the reproducibility of the results in different networks and types of measurements.

As we claimed before, decision tree models also provide great insights about the process leading to a specific classification result (leaf). Using the model, a network operator can identify those features indicating a specific type of anomaly, and better infer on their nature. Features at the higher levels of the tree tend to have more distinguishing power and account



(a) Per-class ROC curves for the C4.5 tree.

(b) Improving accuracy by feature selection.

Figure 7.3.: Performance of the C4.5 anomaly classifier, and classification enhancement through feature selection.

for more population size than lower level features. In this model, the root node is the output of the distribution-based anomaly detector on the symptomatic signal, showing the paramount role and information provided by such feature and, again, its supremacy against the entropy-based EWMA detector.

Note that paths from the root node to leaves representing different anomaly types can be directly expressed as logical rules, which can be ultimately integrated into any kind of rule-based monitoring system.

7.6. Improving C4.5 Performance by Feature Selection

Using an extensive list of traffic features is not always the best strategy, as it may negatively impact classification results. Using more features increments the dimensionality of the feature space, normally introducing undesirable effects such as sparsity. At the same time, using irrelevant or redundant features may diminish performance in the practice. We show next that by carefully addressing the pre-filtering of input features by standard feature selection techniques, we can partially solve the classification problem of the aforementioned C4.5 tree, related to type 3 anomalies.

There are different search strategies and evaluation criteria to construct a sub-set of traffic features. Regarding search strategies, the idea is to test different sub-sets of features, studying local changes in the particular evaluation criterion when adding or removing features. The evaluation criterion permits to test the goodness of a particular sub-set.

In this section we apply a widely used evaluation criterion to construct a reduced sub-set of features: correlation-based evaluation. This approach basically selects sub-sets of features that are poorly correlated among each other, but highly correlated to the classes of traffic. As search strategy, we use Best-First (BF) search; BF is similar to a standard greedy exploration, but it has the ability to do backtracking, i.e., it basically keeps the previously

Figure 7.4.: State machine of change detection output. f_r is the fraction of change notification in the shift register, th is the state transition threshold. Each symptomatic and diagnostic signal has its dedicated state machine.

evaluated sub-sets so as to avoid local maximum/minimum results when there is no local improvement.

We now evaluate the impact of feature selection on the performance of the C4.5 decision tree model. By running the proposed technique, we end up with a greatly reduced set of features, going from the initial 48 features to only 4. The resulting set is composed of the following features: querycnt_adtool, apn_avg, req_fqdn_p25, and req_fqdn_adtool. Note that those features related to the EWMA detector are suppressed by the feature selection process, as expected. Interestingly, selected features have a high correlation to the type 3 anomaly characteristics, which are directly linked to APN and DNS query counts. To conclude, Figure 7.3(b) shows the per-class accuracy obtained by the C4.5 model for both input features' sets (i.e., the full set of features, and the pruned, 4 features' set). While the performance obtained in the classification of type 1 anomalies is slightly worse when performing feature selection, there is a great improvement in the detection performance for the type 3 anomalies, partially compensating the initial problems of the C4.5 model as depicted in Figure 7.2.

7.7. Signal Change Correlation and Reporting

In this Section, we provide guidelines for the design of the final stage of the framework (cfr. Figure 6.1) ultimately responsible for producing the diagnosis report on anomalies. Specifically, we cover two aspects: (i) the definition of a state machine to handle change detection outputs in a state-full manner, (ii) the time-correlation of the signal changes, (iii) the definition of the event report, containing the diagnostic information, the anomaly type and a description of the temporal dynamics of the anomaly.

7.7.1. Change Detection: from State-less to State-full

The instances of the change detection module notify the occurrence of significant modifications detected on the corresponding (symptomatic or diagnostic) signal, along with the elements which contribute the most to the change (cfr. Figure 6.1). The notification, along with the label assigned by the ML classifier, is done independently at each iteration in a state-less fashion. However, the detectors output may flip from *anomalous* to *normal* during the same event, depending on the algorithm sensitivity and on the anomaly intensity.

The first task of the diagnosis module is to consolidate the changes referring to the same event. This is done, independently for each signal, by means of a finite state machine. The finite state machine, depicted in Figure 7.4, consists of three states, namely `Normal`,

`Warning` and `Anomaly`. The state transitions depend on the number of change notifications in a shift register containing the last n_s outputs of the detector. We indicate by f_r the fraction of change notifications in the register. The initial state (`Normal`) corresponds to $f_r = 0$. As soon as the first change is detected (i.e., the event starts), the signal state switches to `Warning` ($f_r > 0$). The signal remains in the same state, till a sufficient number of changes has been detected. State transitions depend on a threshold th: when $f_r > t_h$, the signal enters the `Anomaly` state. When the detector stops flagging changes, the fraction f_r starts decreasing, till it goes back to `Warning` and, eventually, `Normal`.

7.7.2. Correlation of Signals and Generation of Reports

The main purpose of the diagnosis module is to temporarily correlate symptomatic and diagnostic signals: in a nutshell, by locating those diagnostic signals which show a change at the same time of the detected anomaly, one gets a more targeted and specific indication of which features might be altered by the anomaly. This objective is achieved by means of *reports*, which describe single *events*.

We define as *event* a period of time corresponding to anomalous traffic patterns with a stable signature. Its life-cycle is controlled by symptomatic signals. An event is considered started as soon as one or more symptomatic signals switch from `Normal` to `Warning` state. Conversely, it is considered as closed (i.e., the anomaly is over) if the symptomatic signals are back to `Normal` state, or if there is a change in the signature (i.e., an *event change*); in the latter case, a new event with the new signature is initialized. A change in the signature could be caused by either a change in the involved signals (i.e., a new diagnostic signal enters `Warning` state, or a signal that was in `Warning`/`Anomaly` goes back to `Normal`), or in the list of changing elements of one of the diagnostic signals (e.g., a new FQDN appears, disappears, or changes sign).

Recall from Section 6.3, that symptomatic signals are used as evidence of anomalous behavior, while diagnostic signals provide additional information for the diagnosis and the event. The report is enriched with the label assigned by the ML-based module and the diagnostic information provided by the involved signals, including the lists of the most relevant changing elements (cfr. Sec. 6.4). The changing elements $\delta(\omega)$ have a sign, depending on whether their share in the distribution is increasing or decreasing. An example of an event report is reported below:

```
label          anomaly type #1

symptom        query_per_user  [10,+][1,-]
diagnostic     manufacturer    [Pineapple,+]
diagnostic     OS              [youOS_v2.1,+]
diagnostic     FQDN            [youcloud.com,+]

start_warn     2015-02-14 12:00:00
start_alarm    2015-02-14 13:00:00
end_alarm      2015-02-14 17:30:00
end_warn       2015-02-14 18:00:00
```

The report could be read as: an anomaly of *type 1* has been detected: a change in the symptomatic signal distribution of *query per user* has started at `start_warn` due to an increase in the number of terminals issuing 10 queries, and a decrease of terminals issuing 1 query, in every 30min *time-bin*. The event report indicates that the devices produced by `Pineapple`, equipped with version `2.1` of the operative system `youOS` increased their shares of queries in the respective distributions. Also, the event report indicates that the number of queries for the FQDN `youcloud.com` has increased. This shall be the typical signature generated for an anomaly of type E_1 (cfr. Sec. 6.7) lasting six hours, where involved terminals retry to access `youcloud.com` every 5 minutes (10 requests every 30 minutes). Notice that the event report describes the event throughout its entire duration, starting with the first time-bin in which the symptomatic signal firstly enters in `Warning` state and terminating with the last time-bin before it switches back to `Normal` state.

7.8. Iterative Diagnosis – Drilling-down into Anomalies

One more thing...

Steven Paul Jobs

Summarizing the diagnosing process described so far, we adopt a scheme based on sequential actions: (i) we start by passively monitoring the network, (ii) we extract a statistical representation of two sets of traffic features, which we called symptomatic and diagnostic signals, (iii) we continuously monitor them to detect the presence of significant deviations from their normal patterns, (iv) we use ML techniques to classify the detected anomaly, and finally (v) we report a summary describing the anomalous event, composed of all the information to support the troubleshooting. In short, we defined a *Sequential Diagnosis Paradigm*, as depicted in Figure 7.5. As we have seen in the evaluations, this approach is successful in a number of scenarios. Indeed, as in the case of the DNS anomalies, an ISP could employ this detection and diagnosis scheme to automatically tackle these frequent and disrupting anomalies. However, there is still room for improvements. In the remaining of this Section, we propose a possible research direction, based on the lesson learned during these studies and some interesting preliminary results.

Before investigating a possible evolution of our framework, we shall first analyze its limitations. To this end, let us consider again the anomaly that impacted the perceived QoE of YouTube users in a fixed-line operation network. In Section 5.6 we described the anomaly by manually analyzing all the involved traffic features, while in Section 6.9 we tested our distribution-based detection algorithm to automatize the process. The results obtained in the two cases were compatible. We shortly recall that the anomaly was caused by a temporary change of the set of YouTube servers assigned to the users passively observed in our VP located in a fixed-line network. For the whole duration of the anomaly, we observed a decrease of the average downlink throughput during peak-times (i.e., late afternoon/evening), resulting in a degradation of other QoE-based KPIs. By observing the other traffic features, we came to two conclusions: (i) the problem was not in the ISP boundaries (i.e., at the access network) and (ii) the root cause was a possible under-dimensioning of the newly selected

Figure 7.5.: Sequential Diagnosis Paradigm based on a single and passive Vantage Point scheme.

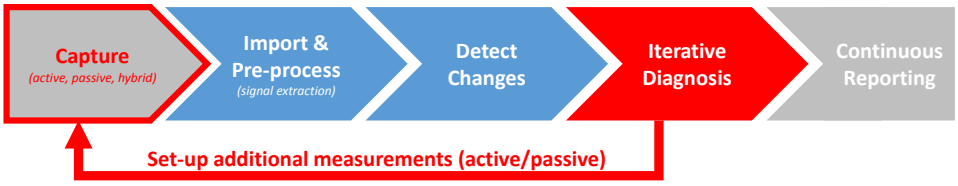


Figure 7.6.: Iterative Diagnosis Paradigm based on a control-loop feedback mechanism that allows to set-up new active or passive measurements or access existing measurements from other VPs.

servers. Note that, while the first conclusion could be certainly demonstrated by excluding the presence of internal congestion, the latter is a speculation. Although the increase of the average server elaboration time is an evidence of under-dimensioning of the servers, we cannot exclude other external root causes by only relying on a single VP located at the edge.

Diagnosing problems at the access network is somehow easier for the ISP, as this network is in its domain (even if in the general case it can still be a very challenging task for ISPs, as we have seen). However, diagnosing the problems outside its boundaries is a much more complex task. In our specific example, the possible *external* root causes can be multiple: the Google CDN server selection strategies might be choosing wrong servers, the YouTube servers might be overloaded, path changes with much higher RTT from servers to the customers might have occurred, paths might be congested, etc. Therefore, we cannot state *where* the problem is located and *who* has the responsibility to intervene, just by passively monitoring a single VP located in the access network. The complexity of modern large-scale services and the involvement of many stake-holders (the service it-self, the content provider, the transit ASes, the access network provider, and the final users) complicate the picture.

A possible solution consists in embracing a reactive monitoring approach, originally envisioned in [24]. Given that, in general, a network anomaly could potentially involve more than one stakeholder, the idea is to include additional measurements to get further insights on the possible root causes. This can be done by both accessing existing additional on-line or historical measurements, and/or setting up new ad-hoc passive and active probes to collect missing diagnostic details. In other words, the *Sequential Diagnosis Paradigm*, previously seen in Figure 7.5, can be evolved into an *Iterative Diagnosis Paradigm*, summarized in Figure 7.6. The difference between the two lies in the addition of a control feedback loop that allows to *iteratively* drill down into the diagnosis of the anomalies. Note that, the mPlane architecture (cfr. Chapter 2.1) offers the instruments to tackle this diagnosis scheme.

7.8.1. The Diagnosis Graph

We now try to define the steps of the iterative diagnosis scheme in practice by using the YouTube anomaly as reference case study. Such an iterative analysis is done by applying a set of diagnosis rules to verify the occurrence (or not) of specific signatures explaining the detected symptoms. These rules are initially defined by an expert operator, based on his domain knowledge and operational experience. Given our specific QoE-related issue to diagnose, each of the rules checks for a predefined signature characterizing the root causes.

To define the set of knowledge based rules to diagnose a problem, the first step is to identify which are the possible root causes of such problems, and where could the origins be located. The large number of possible root causes coupled with the generally much lower number of VPs providing information about the symptoms makes the enumeration of the root causes and their location a complex task. The approach we propose as example is a coarse one, in which we drill down a previously characterized YouTube anomaly to find out the main part of the end-to-end service delivery responsible for it (e.g., device, access ISP, Internet, CDN, content provider), rather than the specific network element (e.g., interconnection router, link failure, routing table, etc.). In this specific example, the origins of the QoE-relevant degradation could be potentially located at:

(i) end terminals: potential issues in the end-terminal are multiple, from software to hardware issues, as well as connectivity and signal strength among others. However, as we said before, this case study considers QoE impacts in a large number of users, and thus individual buggy terminal events are out of the scope of the diagnosis analysis. Only problems simultaneously affecting a large number of terminals are potentially considered; for example, issues related to software updates affecting a whole category of devices (i.e., iOS smartphones, Windows 8 OS, etc.).

(ii) home network: similar to previous observations for end terminal issues, the home network could be a potential issue only in case of problems affecting for example a whole category of home gateway devices. However, in this specific case, firmware updates are much less frequent than OS and software updates, and therefore we exclude the home network from the analysis.

(iii) access network: diagnosing issues at the access network heavily depends on the type of access network considered (cellular, WiFi, FTTH, ADSLx). Download throughput problems at the access can be caused by multiple issues, from congestion events to equipment outages and misconfiguration.

(iv) core network of the ISP: problems at the ISP providing the Internet access to the users are generally the most common ones. These are various, including intra-AS routing, router outages and equipment failures, misconfiguration, etc. The usage of virtualization and software-defined technologies (both at the access and core networks) adds additional sources of potential performance issues.

(v) Internet: depending on the location of the YouTube content and on the cache selection policies used by Google to answer users' requests, the YouTube flows might have to traverse multiple ASes from the YouTube servers the access ISP. As we said before, YouTube would normally assign user requests to the closest servers. Still, due to its load balancing policies, YouTube might assign users to other servers farther located, resulting in multi-AS paths from

servers to customers. As a consequence, problems related to inter-AS routing, congestion at intermediate ASes, and multi-AS paths performance degradation are potential root causes for YouTube QoE degradation.

(vi) CDN and the servers: the final part of the end-to-end service diagnosis corresponds to the servers hosting and providing the YouTube videos. Software or hardware problems of the hosting servers, overloading situations of wrongly dimensioned servers, internal problems of the hosting data-center, etc. are possible root causes to additionally diagnose.

Once we have enumerated the list of elements to diagnose, we can define a set of rules or *check-list* which shall be iteratively verified to detect the occurrence of events revealing the aforementioned problems. Table 7.2 enumerates a non-exhaustive list of the domain-knowledge based rules for diagnosing the QoE-drop event detected in YouTube.

These diagnosis rules can be structured as a *diagnosis graph*, which is used for guiding the diagnosis and drill-down of the YouTube QoE-anomaly. Figure 7.7 depicts an exemplifying decision graph, integrating some of the previous diagnosis rules. The branches of a decision graph can be either conditionally or systematically followed. In our case, the analysis is conditional, starting from the end terminals till reaching the CDN servers. Note that the verification of remote server problems is left at the end, given that an access operator has more limited visibility and knowledge on the CDN structure. This scheme should be reversed in case the diagnosis process is employed by the content operator.

The decision graph is structured in five different blocks: the *(1) QoE-relevant Anomaly Detection* block consists of the anomaly detection approaches (both entropy-based and distribution-based), coupled with the QoE-based monitoring for understanding whether the detected changes are causing QoE-relevant degradations or not. To avoid triggering the complete diagnosis process on false alarms caused by statistical variations of the monitored features, this block additionally adds a verification of the consistency of the detected anomaly. For example, important deviations in the empirical distribution of the β KPI can be caused by a sudden and important drop/increase in the number of YouTube flows, or by an abrupt modification in the number of users watching YouTube. Therefore, the verification step firstly checks for the presence of events related to major statistical variations in the number of YouTube flows and the number of users watching YouTube. The consistency step additionally defines an hysteresis-based approach for triggering the diagnosis, in which a number of consecutive anomaly alarms have to be flagged before launching the drilling down process. The *(2) End-device Diagnosis* block focuses on the specific analysis of the type of end device associated to the anomalous YouTube flows. The *(3) ISP Diagnosis* block consists of the diagnosis of the access ISP. The *(4) Internet paths Diagnosis* block focuses on the diagnosis of the end-to-end inter-AS paths, including both routing and path congestion analysis. Finally, the *(5) CDN servers Diagnosis* block allows to identify server-related performance issues from end-to-end measurements, assuming that access to in-CDN measurements is not available. Note that these five blocks do not fully cover the aforementioned set of domain-knowledge based rules. Still, the description serves as an example on how to build a diagnosis graph.

Where?	Potential Root Cause and/or Location	Check-list Items – Diagnosis Rules
Terminals and Home Networks	**Device**	For all the involved user devices corresponding to the affected flows, check the occurrence of end-device issues.
	Device OS	For all the involved user devices corresponding to the affected flows, check the heavy hitters of OS type, and the entropy of the OS class.
	Set Top Box	For all the involved boxes corresponding to the affected flows, check the heavy hitters of box-type, and the entropy of the box-type class.
Access Network	**Access Overloading**	Check the occurrence of access-overloading events during the last days, for the corresponding access networks or logical aggregation points (e.g., users in the same aggregation network, or attached to the same DSLAM, etc.). Compare to similar events for other users accessing the same servers through a different access network.
	Access Configuration	Check the occurrence of reconfiguration events related to the corresponding access networks.
	Equipment Failure	Check the occurrence of outage events reported by the KPIs monitored by the ISP at the corresponding access networks.
Core Network	**Intra-AS Routing**	For all the involved user devices corresponding to the affected flows, check the occurrence of end-device issues.
	Link Congestion	Check co-occurrence of link congestion events.
	Equipment Failure	Check the occurrence of outage events reported by the KPIs monitored by the ISP on its internal equipment, including routing/switching/forwarding equipments.
Internet	**Inter-AS Routing**	Check end-to-end path change events in the corresponding temporal span of the detected anomaly.
	Path Congestion	Check flagged events related to abrupt increases in packet retransmissions per server, or in the end-to-end queuing delay, for all the flows provisioned by the corresponding servers.
	Intermediate AS Issues	Check performance degradation events in the intermediate ASes, particularly including latency and congestion in the different end-to-end ASes path segments.
CDN Servers	**Server Reachability**	Check if geo-distributed reachability measurements to the identified servers result in non-reachability problems.
	Server Soft or Hard Failure	Check occurrence of server hardware outages and/or software-related events at each single identified server IP during the time span of the detected anomaly.
	Server Overloading	Check occurrence of overloading events at each single identified server IP during the time span of the detected anomaly.

Table 7.2.: Set of diagnosis rules/items to check for diagnosing performance issues in CDN services such as YouTube.

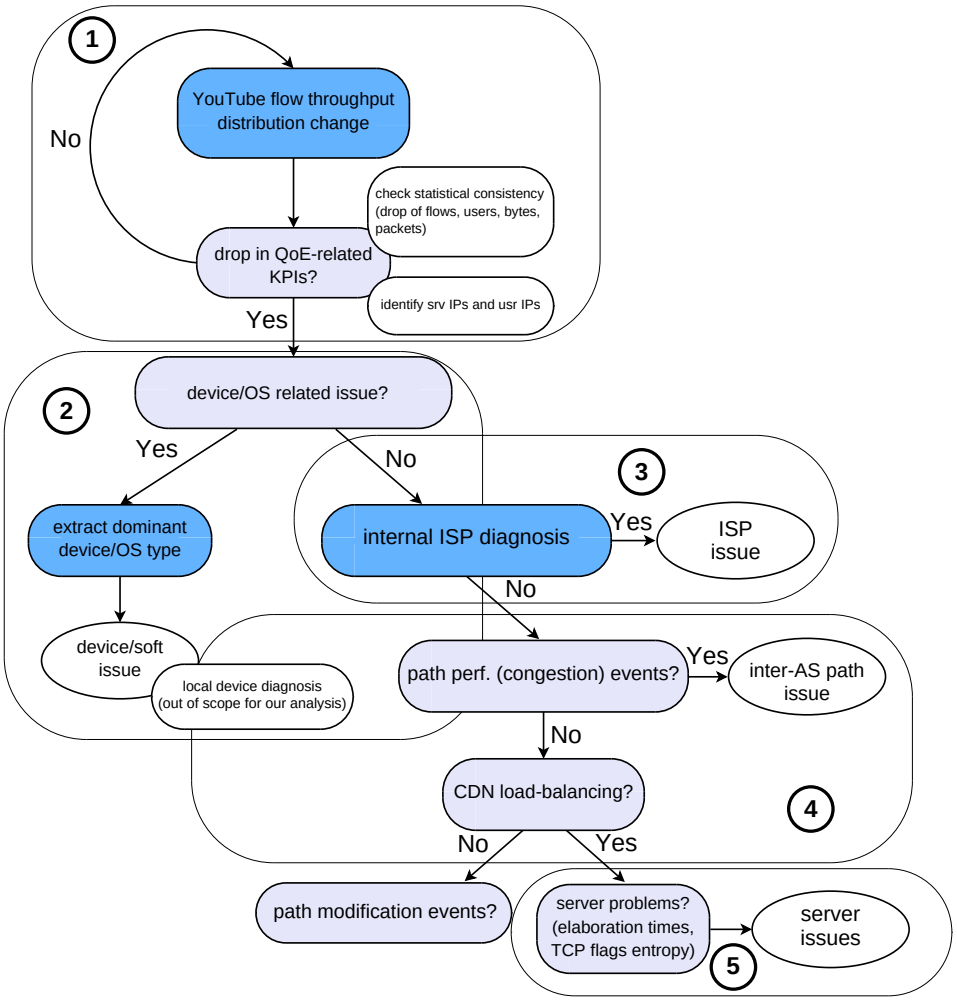


Figure 7.7.: Diagnosis graph associated to the detection and troubleshooting support of large-scale QoE-relevant anomalies in YouTube.

7.8.2. Measuring Path Performance – An Open Challenge

In the YouTube use case, as said, the evidences gathered at the VP in the access network point at an external root cause. We have seen that the increase in the server elaboration time suggests a sub-dimensioning of the servers, which are not able to cope with the traffic at peak hours. Nonetheless, we cannot exclude the presence of heavy network congestion in the paths from the newly selected Google servers to the customers.

The iterative diagnosis process could discover the real root cause by instantiating targeted active measurements, such as traceroute, to assess the presence of path congestion when a degradation in some QoE KPIs occurs. However, traceroute allows to measure the uplink path, i.e., from the VP to the remote Google servers, while our use-case calls for opposite measurements on the downlink path. Given that Google servers lie outside the boundary and control of the ISP, this is not straight forward. An optimal solution consists in a federated measurement infrastructure, which assumes the collaboration of all parties involved in the end-to-end path. As this is a too ambitious solution, the only way to perform such reverse

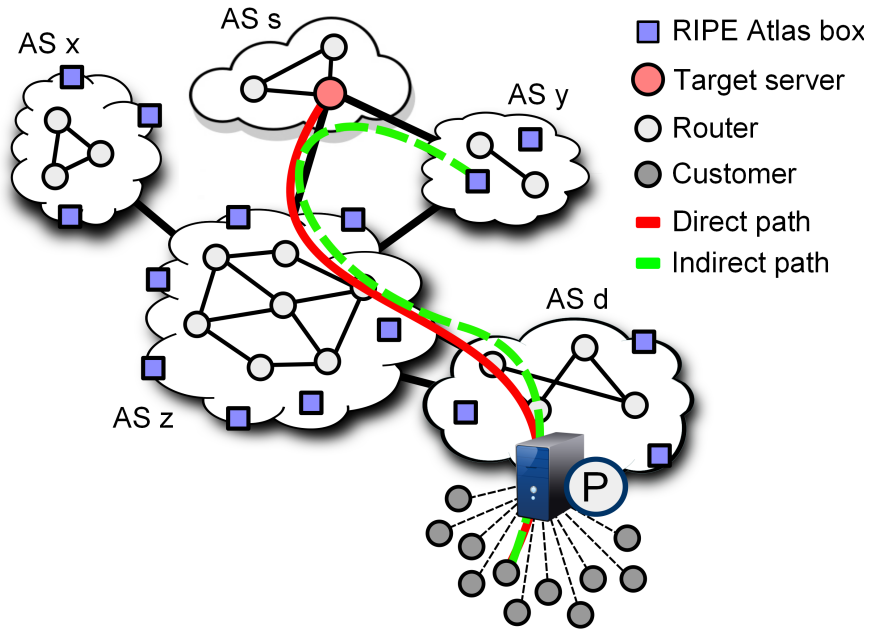


Figure 7.8.: DisNETPerf overview.

traceroute is either by using specialized tools, such as the one presented in [128], or to perform the measurements from controlled servers which are as close as possible to the original ones. The problem of executing reverse traceroute measurements has been addressed in the past [128]; however, the proposed approach heavily relies on IP spoofing and IP Record Route Option, both being not necessarily allowed in every ISP and causing potential security concerns. Therefore, we take into consideration exactly the latter option: we rely on servers close to the original ones.

To this extent, we have collaborated in the design of a specialized tool called DisNETPerf, developed in the context of the mPlane project. DisNETPerf, a Distributed Internet Paths Performance Analyzer, serves the purpose of monitoring any Internet path using the RIPE Atlas framework [8] and standard traceroute measurements. This work has been done by Sarah Wassermann under Dr. Pedro Casas' and my partial supervision, and makes use of the Atlas Toolbox [25], my library for interacting with the Atlas framework.

The first step of DisNETPerf consists of selecting a monitoring point or probe located as close as possible to a target server, i.e., a *phantom server*, to later on perform traceroute measurements towards specific destinations. In a nutshell, given a certain source content server (e.g., a YouTube server) with address IP_s, and a destination customer with IP address IP_d, DisNETPerf locates the closest phantom server (a RIPE Atlas probe) to IP_s, namely IP_c. The idea is that, if the phantom server IP_c and the content server IP_s are *close* enough, running traceroute measurements from IP_c –which is under control– to IP_d allows to get a good estimation on the status of the path from the actual server IP_s to the customer IP_d. The performance indicators include RTT per hop, end-to-end RTT, losses, etc.

As for the distance metric employed to find the closest IP_c, it takes into account both topology and delay information, in this order. It first tries to find Atlas probes in the same AS of IP_d. If it does not find any probe in the same AS, it seeks for other candidates in the

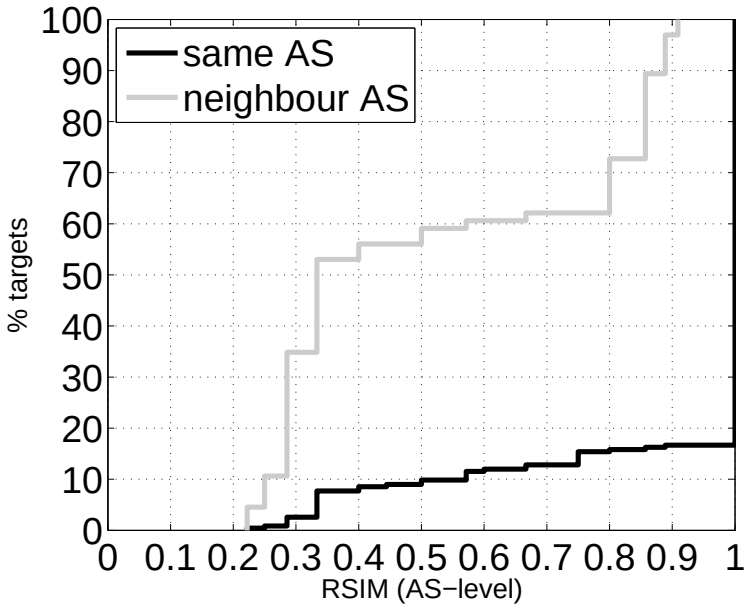


Figure 7.9.: DisNETPerf - probe selection evaluation, based on RSIM. Figure by Sarah Wassermann, available at [P21].

neighboring ASes[1]. When a first list of candidates is ready, it instruments the found probes to run ping measurements toward IP_s in order to find the one with the minimum RTT, i.e., the one with smaller propagation delay.

We say that the probe selected by DisNETPerf (i.e., IP_c) is a good probe w.r.t. IP_s and IP_d if the network path from IP_c to IP_d is highly similar to the path from IP_s to IP_d. Similar to [129], we define path similarity as the fraction of common links among both paths. Formally, we use the Route Similarity Index (RSIM), defined as:

$$RSIM(IP_c, IP_s, IP_d) = \frac{2 \times C_{links}(IP_c, IP_s, IP_d)}{T_{links}(IP_c, IP_s, IP_d)} \tag{7.2}$$

where C_{links} refers to the number of links shared in common by both paths, and T_{links} to the total number of links. A high RSIM indicates a high similarity between the considered paths. Note that links can be defined at multiple granularities; in particular, we consider links at the AS level. IP2AS mapping is done through the database provided by Maxmind [26].

Figure 7.9 presents evaluation results showing the applicability of DisNETPerf in terms of AS path similarity. The goal is to assess whether the probe selection approaches select probes with the highest similarity to the one we want to actually monitor. We use RIPE Atlas probes as both source and destination (i.e., IP_s and IP_d) so as to compute the real path (i.e., the ground-truth) between servers and customers. We randomly select 300 RIPE Atlas source probes IP_{s_i}, and consider a single fixed destination probe IP_d. For each source IP_{s_i} we run DisNETPerf to locate the closest probe IP_{c_i}, obtain both the ground truth path $IP_{s_i} \rightarrow IP_d$ and the DisNETPerf path $IP_{c_i} \rightarrow IP_d$, and compute the RSIM index

[1]Using CAIDA's AS-relationships database, `http://data.caida.org/datasets/as-relationships/`

$RSIM(IP_{c_i}, IP_{s_i}, IP_d)$.

We compute RSIM at the AS level and plot the resulting CDF. Results are reported for two different groups, the first one in which IP_{c_i} and IP_{s_i} are located in the same AS (black lines), and the second one in which IP_{c_i} is located in a neighbor AS (gray lines). There is a significant difference between groups, and the case of same AS co-location results in near optimal results. Nevertheless, we observe that about 40% of the tests yield a RSIM index $\geq$ 0.5. Note that the most relevant segment of the path to monitor for troubleshooting purposes is the one closer to the customer (where problems generally occur), thus a RSIM of 0.5 reflects a high performance of DisNETPerf for the envisioned purposes. Finally, the probes selected by DisNETPerf generally correspond to paths with the highest similarity to the ground-truth ones: in more than 80% of the performed tests, $RSIM(IP_{c_i}, IP_{s_i}, IP_d)$ results in the highest RSIM index among all the selected candidates.

This first evaluation demonstrates the applicability of DisNETPerf for end-to-end path measurements, proving to be a good candidate for supporting the iterative diagnosis of performance degradation, such as in the case of the Youtube anomaly. These preliminary results are even more encouraging considering that the RIPE Atlas framework is rapidly expanding, hence there will be higher chances to find a good phantom server to mimic any given IP with more accuracy, as the number of available Atlas probes increases. Note also that this approach is not strictly tied to RIPE Atlas, but can be used with any other distributed measurement framework such as CAIDA's Ark [10] or PlanetLab [9].

7.8.3. Outlook for a Flexible Monitoring Approach

To sum up the discussion on the iterative diagnosis, we now draw some guide-lines for a smarter and more flexible monitoring approach in the direction of the reactive measurement scheme. To this extent, we introduce three design principles that should be addressed in the future for the design of better diagnosis and troubleshooting tools in complex networks.

(i) Data collection policies The quality of the diagnosis strictly depends on the amount and representativity of the information available to explain the anomalies' root causes and effects. In our sequential diagnosis scheme we monitor a number of traffic features and we correlate changes in order to build anomaly reports. Ideally, the more traffic features covering all the potential sources of problems, the more aspects of anomalies can be unveiled. Due to feasibility issues, however, we should consider a more flexible approach that allows to regulate different levels of granularities depending on the current needs. In our framework, for example, we would allow a fine-grained and continuous collection of the symptomatic signals and a more flexible monitoring of the diagnostic signals. Note that the data collection policies do not only refer to the passive monitoring of additional traffic features, but also to the set-up of on-demand and time-limited active measurements, like, for instance, the end-to-end performance assessment through tools similar to DisNETPerf. This approach solves the scalability problems, also reducing the monitoring overhead on the network. In addition, it allows setting up specific measurements tailored for the detected anomaly, dynamically regulating the degree of details needed for the troubleshooting process.

(ii) Self-adapting Time Granularity The self-adapting approach in the data collection policies should also be valid for the selected time-granularity. During normal operations, in fact, coarse-grained feature collection (e.g., longer time scales) is enough for producing an overview of the network's status, but, for the diagnosis of anomalies, a more fine-grained view is in general required. To achieve this, we can design a sliding window mechanism to accommodate detailed measurements (e.g., 1 minute time scale) for short-term analysis and keeping coarse-grained statistics (e.g., 1 hour time scale) for long-term historical comparisons. This mechanism would ensure a timely change detection on the key symptomatic signals. Note that this sliding window mechanism could be efficiently provided by a stream data-warehouse such as the one used in this work.

(iii) Distributed Vantage Points In order to troubleshoot anomalies, it would be desirable to include multiple vantage points in the analysis. As we have seen, the single VP approach could pose limitations to the diagnosis of certain anomalies. This is especially critic for modern CDNs that dynamically balance delivery resources to different user clusters: having more points of view on large scale service provisioning systems would allow to assess the effects of the detected anomalies on a geographical and topological basis, highly increasing the overall network visibility. This could be achieved by introducing a federated measuring framework like the one suggested by the mPlane project (cfr. Chapter 2).

7.9. Summary

In this Chapter we have completed the description of a sequential framework for automatic detection and diagnosis of network anomalies. In particular, we focused on the last component, responsible of the classification, correlation and reporting. By relying on Machine Learning techniques, we have showed how to classify the detected anomalies in an automatic fashion. In particular, we investigated a number of supervised classification techniques and the effects of feature selection on classification performance. The evaluation has been done running several experiments on a synthetic dataset. We have also presented a module responsible for the time-correlation of the detected changes in symptomatic and diagnostic signals and reporting of event. The operational value of such an automatic reports is paramount, as it could potentially result in a dramatic reduction of the time spent by network operators in diagnosing unexpected events.

Lastly, we have collected the lessons learned in this thesis in order to propose an evolution of our sequential framework. Despite the promising evaluation, in fact, some types of anomalies could remain not fully diagnosed if we do not envisage a reactive and iterative diagnosis scheme. In particular, ad-hoc active measurements could help in completing the diagnosis of anomalies caused by external factors. As seen, this is not an easy task, as it assumes the access to distributed measurements outside the ISP boundaries. To this extent, we have presented some preliminary results on a distributed measurement framework designed to overcome these limitations. We believe that these results, together with the tools and methodologies provided by this thesis, provide a solid ground for future research aimed at further automatizing the troubleshooting of network anomalies.

8. Conclusions

> We can only see a short distance ahead, but we can see plenty there that needs to be done.
>
> *Alan Turing*

In this thesis I have described a process aimed at better understanding some complex, and still not fully explored, Internet dynamics. The methodology and the obtained results can support operators in understanding the footprint of Internet scale services on their networks and can dramatically reduce the time spent in diagnosing unexpected events. The structured procedure starts from the differentiation of Web services, followed by a deep characterization phase, which includes both normal operation scenarios as well as anomalous events, and finally terminates with a description of a systematic approach for the detection and diagnosis of anomalies. It is not by chance that the logical sequence of chapters, and corresponding contents, reflects the chronological order in which I have worked on the different topics.

I started with the first essential problem: the Web traffic classification. In fact, before understanding which are the most popular services and what are their network footprints, it was firstly needed to distinguish them in the passive traces. In this part, I designed a classification approach based on hostname pattern-matching, which can be enriched with DNS information in case the hostname is not available, as it happens in encrypted protocols (e.g., HTTPS).

In the second step I established the foundations of this work: a deep characterization of network traces. I focused on the key players in today's Internet, offering two orthogonal perspectives. The first consisted in unveiling the main hosting organizations and highlighting their different provisioning approaches, while the second focused on three popular Internet applications, namely Facebook, YouTube, and WhatsApp. In this part, among our main findings, I have shown how dynamic and distributed are current major CDN players like Google and Akamai, providing not only a large number of servers or IP addresses at highly distributed datacenters, but also making use of load balancing techniques to shift HTTP flows among their preferred hosting locations. I have also shown evidences on a more static approach followed by other CDNs, like Limelight, reflecting a different philosophy for CDN architectures. The same distinction is present for the three examined services: while Facebook and YouTube are characterized by very sophisticated traffic dynamics involving a number of geographically spread data centers, WhatsApp employs a more static and simple approach, mainly relying on two Softlayer data centers for all users world-wide. Despite the existence of previous studies focusing on different aspects (such as the application's energy consumption, usage patterns and single node monitoring), from the best of our knowledge, we have been

the first in characterizing Facebook and Whatsapp from the network perspective through large-scale monitoring and the first to compare the caching strategies of YouTube in two different access technologies. Besides the interesting results and the novel findings, I believe that the main contribution of this part is the structured approach in characterizing Internet services. To do this, I relied on standard network measurement techniques, as well as new ways of observing complex provisioning systems from original perspective. Among those, I mention the comparison of traces from different network types, the inference of Quality of Experience from flow characteristics, the study of users' perception of the anomaly by analyzing social media feeds, the graphical representation of the evolution of load balancing policies using heat maps and temporal similarity plots.

After collecting a sufficient knowledge base on the inner functioning of large scale provisioning systems, I shifted the attention to unexpected behaviors. In the course of the characterization phase, I have collected a number of study cases of important changes that occurred in the hosting infrastructures of popular applications, resulting in outages, malfunctioning, or undesired circumstances for network operators. In particular, I have shown that the caching selection policies employed by major CDNs might have a significant impact on both the ISP carrying the traffic and the end-customers. By considering traffic from different network types and services, I showed that these events are not bound to a particular location or type of network. The characterization of such anomalies was done by manually inspecting different traffic features in order to fully describe their causes and impacts. As I showed, some anomalies are far from being easy to diagnose, as their impact is often not clearly evident, mostly when it implies a degradation of the users' Quality of Experience.

The last section of this thesis was dedicated to the detection and diagnosis of network anomalies. I relied on the lessons learned in the characterization phase and I proposed a systematization of the detection of such anomalies. I presented a framework for automatic diagnosis of large scale Internet anomalies based on the analysis of passively captured network data. Its key idea is to apply a change detection algorithm to a set of meaningful signals extracted from network measurements, and then correlate those signals which show similar abnormal behavior on a similar time-span. To this extent, I have thoroughly tested two different detection schemes both on real network traces and semi-synthetic datasets. The presented results unveiled the limitations of using state-of-the-art simple change point detection algorithms in the case of low intensity anomalies that involve relatively small subpopulations of users. From our experience, this type of anomalies are frequent in operational mobile networks, and still potentially very harmful (cfr. problems on the signaling plane). To overcome this limitation, I have presented a more complex change detection scheme that relies on the entire probability distribution of the monitored signals rather than the entropy values. Using this detection approach, the system is able to cope with anomalies that involve multiple services and/or affect multiple devices at the same time.

Given the general lack of large-scale ground-truth datasets to test the performance of systems like ours, I designed an approach to generate semi-synthetic data, derived from real traffic traces. On the one hand, this approach allowed me to properly evaluate the detection capabilities of the algorithms, on the other hand, it could provide useful guidelines in the synthetic traffic generation for network operators, that would be able in this way to disclose datasets to the research community without risking to reveal privacy or business sensitive

details.

The detection of network anomalies, however, is only the first step in their understanding and troubleshooting. In fact, the main challenge lies in the automatic diagnosis, which could be achieved by offering detailed anomaly reports to operators. Such reports would help to speed up the study of anomalies, currently done relying on manual, time-consuming and error-prone procedures. Despite the large literature on anomaly detection in large-scale networks, I still miss a proper automatic approach for diagnosis. To fill this gap, I concluded this thesis showing how to classify the detected anomalies in an automatic fashion exploiting Machine Learning techniques. In particular, I investigated a number of supervised classification algorithms and the effects of feature selection on classification performance. The diagnosis procedure is finally concluded with the automatic correlation of the involved traffic features and the reporting of all the collected diagnostic information, including the label produced by the classification.

Outlook on Future Research

To conclude, we have seen that the study of highly decentralized provisioning systems is not an easy task. It requires an analytical approach and deep domain knowledge. Given the complexity of modern Internet scale services, unveiling performance issues, outages, and malfunctioning in general is paramount, but still a major challenge for operators. In this work, I have addressed some research questions (cfr. "Research Questions and Contributions" in Section 1.4). Still, there are some open issues and potential improvements that are worth considering for future research. In the following, I provide a short summary of four potentially open topics.

Improving Classification of Encrypted Contents The first contribution of this thesis is the traffic classification approach used for distinguishing Web services in network traces. This was functional for the characterization of top services. At the beginning of the study, I relied on hostname pattern matching using the `GET` field of HTTP tickets. However, in the course of the long characterization phase, I needed to update the classification technique in order to be able to tackle encrypted protocols, such as HTTPS and other SSL/TLS based applications, that were heavily increasing their share every day. I then introduced an upgrade that relies on DNS to compensate the lack of the requested hostname. While this approach has proved to be very effective for our goals and currently permits to classify a good share of the overall flows, it has at least two limitations. The first is the use of DNS caching; this prevents browsers to issue new DNS queries and consequently make the association between user's flow and remote service impossible. Luckily, this is a quite rare case as CDNs rely on short DNS TTLs to quickly redirect users to different front-end servers, limiting the use of browser caching. The second limitation comes from the increasingly cryptic naming schemes employed by CDNs. At the moment, I try to reverse engineer this naming scheme in order to build suitable regular expressions, but the validity of such regex could cease at any time. For example, Akamai uses specific sets of FQDNs for Facebook of the type `fbstatic-*.akamaihd.net`, where the 3LD sub-string `fbstatic` suggests that the

corresponding service is Facebook. This reverse engineering is clearly time consuming. For the moment, I still do not have a way-around.

Continuing the Characterization Efforts The characterization of traffic is the fundamental pillar for unveiling obscure Internet's dynamics. The knowledge base created is then used for traffic engineering, network planning, anomaly detection and other key tasks. Most of the time spent in this work has been actually dedicated to this, focusing in particular to some "big players" case studies. As I have mentioned, our main contribution is the structured methodology used to uncover the main characteristics of large scale provisioning systems. The results themselves are definitely interesting and give important insights to network operators to understand what it is currently happening in their network. However, they come with an expiration date: tomorrow new applications and hosting schemes will arise changing the Internet's landscape. Therefore, it is important to continue characterizing the traffic and update the way we understand Internet patterns. As Internet evolves, our domain knowledge should evolve with it.

Anomaly Detection and Concurrent Events As I have remarked in the course of this thesis, my aim was not conceiving "yet another anomaly detection algorithm", as a lot of effort has been already put in this topic by the research community. For our detection and diagnosis framework I endorsed a powerful existing algorithm, that I improved in order to cope with dynamic traffic patterns and the frequent change of CDN settings. A crucial point, however, has not been considered: the presence of concurrent anomalies. The large literature on anomaly detection also lacks studies on the presence of multiple anomalies at the same time. However, given the complexity of network patterns, which increases the chances of failures and unexpected events, I strongly believe that this is an important open issue that should be addressed next.

Reactive Monitoring and Iterative Diagnosis The diagnosis framework I have presented provides a good contribution in advancing the state of the art in the field of anomaly detection. However, our sequential diagnosis system could be still improved. In the last part of this thesis (cfr. Section 7.8), I have drawn the guidelines for possible future research for continuing this work. In particular, I envision a distributed approach for network measurements that allows a more sophisticated and flexible iterative diagnosis scheme. Such a flexible system would be able to automatically re-adapt it-self by regulating the type and granularity of data collection and possibly access further existing datasets or set up new ad-hoc measurements. I believe that, among the others, this is the main next step in order to achieve true network visibility on a large scale.

List of Publications

[P1] **P. Fiadino**, A. Bär, P. Casas, "HTTPTag: A Flexible On-line HTTP Classification System for Operational 3G Networks", in *IEEE INFOCOM Poster/Demo Session*, 2013.

[P2] P. Casas, **P. Fiadino**, "Mini-IPC: A Minimalist Approach for HTTP Traffic Classification using IP Addresses", in *Wireless Communications and Mobile Computing Conference (IWCMC)*, 2013

[P3] P. Casas, **P. Fiadino**, A. Bär, "IP Mining: Extracting Knowledge from the Dynamics of the Internet Addressing Space", in *The 25th International Teletraffic Congress (ITC2013)*, 2013

[P4] P. Casas, **P. Fiadino**, A. Bär, "Understanding HTTP Traffic and CDN Behavior from the Eyes of a Mobile ISP", in *Passive and Active Measurements Conference / Poster session (PAM2014)*, 2014

[P5] **P. Fiadino**, A. D'Alconzo, P. Casas, "Characterizing Web Services Provisioning via CDNs: The Case of Facebook", in *The 5th International Workshop on Traffic Analysis and Characterization (TRAC2014)*, 2014.

[P6] A. D'Alconzo, P. Casas, **P. Fiadino**, A. Bär, A. Finamore, "Who to Blame when YouTube is not Working? Detecting Anomalies in CDN-Provisioned Services", in *The 5th International Workshop on Traffic Analysis and Characterization (TRAC2014)*, 2014.

[P7] P. Casas, **P. Fiadino**, A. Bär, A. D'Alconzo, A. Finamore, M. Mellia, "YouTube All Around: Characterizing YouTube from Mobile and Fixed-line Network Vantage Points", *The European Conference on Networks and Communications (EuCNC2014)*, Bologna, Italy, 2014.

[P8] P. Casas, A. D'Alconzo, **P. Fiadino**, A. Bär, A. Finamore, "On the Analysis of QoE-based Performance Degradation in YouTube Traffic", in *10th International Conference on Network and Service Management (CNSM2014)*, 2014.

[P9] P. Casas, **P. Fiadino**, A. Sackl, A. D'Alconzo, "YouTube in the Move: Understanding the Performance of YouTube in Cellular Networks", in *Wireless Days 2014 Conference (WD2014)*, Rio de Janeiro, Brazil, 2014.

[P10] P. Casas, A. D'Alconzo, **P. Fiadino**, A. Bär, A. Finamore, T. Zseby, "When YouTube doesn't Work – Analysis of QoE-relevant Degradation in Google CDN Traffic", in *IEEE Transactions on Network and Service Management*, vol. 11, no. 4, 2014.

[P11] **P. Fiadino**, A. D'Alconzo, A. Bär, A. Finamore, and P. Casas, "On the Detection of Network Traffic Anomalies in Content Delivery Network Services", in *Teletraffic Congress (ITC), 2014 26th International*, 2014.

[P12] M. Schiavone, P. Romirer-Maierhofer, **P. Fiadino**, P. Casas, "Diagnosing Device-Specific Anomalies in Cellular Networks", in *ACM CoNEXT Student Workshop*, 2014.

[P13] **P. Fiadino**, M. Schiavone, P. Casas, "Vivisecting WhatsApp through Large-Scale Measurements in Mobile Networks", extended abstract in *ACM SIGCOMM*, 2014.

[P14] **P. Fiadino**, M. Schiavone, P. Casas, "Vivisecting WhatsApp in Cellular Networks: Servers, Flows, and Quality of Experience", *in Traffic Monitoring and Analysis (TMA 2015)*, 2015.

[P15] **P. Fiadino**, P. Casas, M. Schiavone, A. D'Alconzo, "Online Social Networks Anatomy: on the Analysis of Facebook and WhatsApp in Cellular Networks", *in IFIP Networking 2015*, 2015.

[P16] P. Casas, **P. Fiadino**, M. Schiavone, "QoMOSN - On the Analysis of Traffic and Quality of Experience in Mobile Online Social Networks", in *European Conference on Networks and Communications (EuCNC 2015)*, 2015.

[P17] P. Casas, M. Varela, **P. Fiadino**, M. Schiavone, H. Rivas, R. Schatz, "On the Analysis of QoE in Cellular Networks: from Subjective Tests to Large-scale Traffic Measurements", *in International Wireless Communications & Mobile Computing Conference - TRAC (IWCMC 2015)*, 2015.

[P18] **P. Fiadino**, A. D'Alconzo, M. Schiavone, P. Casas, "Challenging Entropy-based Anomaly Detection and Diagnosis in Cellular Networks", in *ACM SIGCOMM 2015 Poster/Demo session*, 2015.

[P19] **P. Fiadino**, M. Schiavone, A. D'Alconzo, P. Casas, "Towards Automatic Detection and Diagnosis of Internet Service Anomalies via DNS Traffic Analysis", in *International Wireless Communications & Mobile Computing Conference - TRAC (IWCMC 2015)*, 2015.

[P20] **P. Fiadino**, A. D'Alconzo, M. Schiavone, P. Casas, "RCATool – A Framework for Detecting and Diagnosing Anomalies in Cellular Networks", in *27th International Teletraffic Conference (ITC27)*, 2015.

[P21] P. Casas, **P. Fiadino**, S. Wassermann, S. Traverso, A. D'Alconzo, E. Tego, F. Matera, M. Mellia, "Unveiling Network and Service Performance Degradation in the Wild with mPlane", in *IEEE Communications Magazine, Network Testing Series*, (under review), 2015.

[P22] **P. Fiadino**, A. D'Alconzo, P. Casas, Y. Zseby, M. Schiavone, M. Mellia, "Automatic Anomaly Detection and Diagnosis in Operational Networks: the Quest for the Holy

Grail", submitted to *IEEE Journal on Selected Areas in Communications, Special Issue on Measuring and Troubleshooting the Internet*, 2015.

[P23] **P. Fiadino**, P. Casas, A. D'Alconzo, A. Bär, "Grasping Popular Applications in Cellular Networks with DBStream, a Big Data Analytics Platform", submitted to *IEEE Transactions on Network and Service Management, Special Issue on Big Data Analytics for Management*, 2015.

[P24] A. Bär, P. Casas, A. D'Alconzo, **P. Fiadino**, L. Golab, M. Mellia, E. Schikuta, "DBStream: A Holistic Approach to Large-Scale Network Traffic Monitoring and Analysis", submitted to *Computer Networks, Special Issue on Machine Learning, Data Mining and Big Data Frameworks for Network Monitoring and Troubleshooting*, 2015.

[P25] P. Casas, **P. Fiadino**, M. Varela, M. Schiavone, H. Rivas, R. Schatz, "MobiQoE – Taming Quality of Experience in Mobile Devices using Large-scale Traffic Measurements", submitted to *International Journal of Autonomous and Adaptive Communications Systems*, 2015.

Bibliography

[1] International Telecommunication Union (ITU). `http://www.itu.int/en/ITU-D/Statistics/Pages/stat/default.aspx`, 2015. [Online; accessed 01-September-2015].

[2] Bates, Tony and Smith, Philip and Huston, Geoff. CIDR Report. `http://www.cidr-report.org/as2.0/`, 2015. [Online; accessed 01-September-2015].

[3] Cisco Visual Networking Index: Forecast and Methodology, 2014âĂŞ2019. Technical report, Cisco Systems, Inc., 2015. [Online; accessed 27-May-2015].

[4] MicroMarket Monitor. North America Content Delivery Network Market Research Report. `http://www.micromarketmonitor.com/market/north-america-cdn-delivery-1764715362.html`, 2013.

[5] Wireshark – Home Page. `https://www.wireshark.org/`, 2015. [Online; accessed 01-September-2015].

[6] F. Ricciato, P. Svoboda, J. Motz, W. Fleischer, M. Sedlak, M. Karner, R. Pilz, P. Romirer-Maierhofer, E. Hasenleithner, W. JÃďger, P. KrÃijger, F. Vacirca, and M. Rupp. Traffic Monitoring and Analysis in 3G Networks: Lessons Learned from the METAWIN Project. *Elektrotechnik und Informationstechnik*, 123(7-8):288–296, 2006.

[7] A. Finamore, M. Mellia, M. Meo, M.M. Munafo, and D. Rossi. Experiences of Internet Traffic Monitoring with Tstat. *Network, IEEE*, 25(3):8–14, May 2011.

[8] The RIPE Atlas measurement network. `https://atlas.ripe.net/`, 2015. [Online; accessed 15-August-2015].

[9] PlanetLab, an open platform for developing, deploying, and accessing planetary-scale services. `https://www.planet-lab.org/`, 2015. [Online; accessed 15-August-2015].

[10] Archipelago (ark) Measurement Infrastructure. `http://www.caida.org/projects/ark/`, 2015. [Online; accessed 15-August-2015].

[11] Andreas Hanemann, Jeff W. Boote, Eric L. Boyd, Jérôme Durand, Loukik Kudarimoti, Roman Lapacz, D. Martin Swany, Szymon Trocha, and Jason Zurawski. Perfsonar: A service oriented architecture for multi-domain network monitoring. In *Proceedings of the Third International Conference on Service-Oriented Computing*, ICSOC'05, pages 241–254, Berlin, Heidelberg, 2005. Springer-Verlag.

[12] P. Kanuparthy, D. Lee, W. Matthews, C. Dovrolis, and S. Zarifzadeh. Pythia: detection, localization, and diagnosis of performance problems. *Communications Magazine, IEEE*, 51(11):55–62, November 2013.

[13] mPlane – Project Home Page. `http://www.ict-mplane.eu/`, 2015. [Online; accessed 01-September-2015].

[14] Mobile Carriers Must Avoid the 'Dumb-Pipe' Syndrome. `http://adage.com/article/guest-columnists/carriers-uniquely-positioned-targeted-mobile/298210/`, 2015. [Online; accessed 01-September-2015].

[15] mPlane – Reference Implementation. `https://www.ict-mplane.eu/public/mplane-reference-implementation-ri`, 2015. [Online; accessed 01-September-2015].

[16] Joseph M. Hellerstein, Vern Paxson, Larry L. Peterson, Timothy Roscoe, Scott Shenker, and David Wetherall. The network oracle. *IEEE Data Eng. Bull.*, 28(1):3–10, 2005.

[17] David D. Clark, Craig Partridge, J. Christopher Ramming, and John T. Wroclawski. A knowledge plane for the internet. In *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '03, pages 3–10, New York, NY, USA, 2003. ACM.

[18] University of Oregon Route Views Project. `http://www.routeviews.org/`. [Online; accessed 01-September-2015].

[19] Thomas Bourgeau, Jordan Augé, and Timur Friedman. Tophat: Supporting experiments through measurement infrastructure federation. In Thomas Magedanz, Anastasius Gavras, NguyenHuu Thanh, and JeffryS. Chase, editors, *Testbeds and Research Infrastructures. Development of Networks and Communities*, volume 46 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 542–557. Springer Berlin Heidelberg, 2011.

[20] M-Lab home page. `http://www.measurementlab.net/`. [Online; accessed 01-September-2015].

[21] Tstat – Home Page. `http://tstat.polito.it/`, 2015. [Online; accessed 01-September-2015].

[22] A. Bar, A. Finamore, P. Casas, L. Golab, and M. Mellia. Large-scale network traffic monitoring with DBStream, a system for rolling big data analysis. In *Big Data (Big Data), 2014 IEEE International Conference on*, pages 165–170, Oct 2014.

[23] mPlane – List of Use Cases. `https://www.ict-mplane.eu/public/use-cases`, 2015. [Online; accessed 01-September-2015].

[24] Mark Allman and Vern Paxson. A Reactive Measurement Framework. In Mark Claypool and Steve Uhlig, editors, *Passive and Active Network Measurement*, volume 4979 of *Lecture Notes in Computer Science*, pages 92–101. Springer Berlin Heidelberg, 2008.

[25] Pierdomenico Fiadino. Atlas Toolbox – Github repository. `https://github.com/pierdom/atlas-toolbox`, 2015. [Online; accessed 01-September-2015].

[26] MaxMIND GeoIP Databases. `http://www.maxmind.com`, 2015. [Online; accessed 15-September-2015].

[27] Ingmar Poese, Steve Uhlig, Mohamed Ali Kaafar, Benoit Donnet, and Bamba Gueye. IP Geolocation Databases: Unreliable? *SIGCOMM Comput. Commun. Rev.*, 41(2):53–56, April 2011.

[28] DBStream – Home Page. `https://github.com/arbaer/dbstream`, 2015. [Online; accessed 01-September-2015].

[29] Alessandro D'Alconzo, Angelo Coluccia, and Peter Romirer-Maierhofer. Distribution-based Anomaly Detection in 3G Mobile Networks: from Theory to Practice. *International Journal of Network Management*, 20(5):245–269, 2010.

[30] Weka: Data Mining Software in Java. `http://www.cs.waikato.ac.nz/ml/weka/`, 2015. [Online; accessed 15-August-2015].

[31] Gregor Maier, Anja Feldmann, Vern Paxson, and Mark Allman. On Dominant Characteristics of Residential Broadband Internet Traffic. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '09, pages 90–102, New York, NY, USA, 2009. ACM.

[32] A. Gerber and R. Doverspike. Traffic Types and Growth in Backbone Networks. In *Optical Fiber Communication Conference and Exposition (OFC/NFOEC), 2011 and the National Fiber Optic Engineers Conference*, pages 1–3, March 2011.

[33] Silvio Valenti, Dario Rossi, Alberto Dainotti, Antonio Pescapè, Alessandro Finamore, and Marco Mellia. Data traffic monitoring and analysis. chapter Reviewing Traffic Classification, pages 123–147. Springer-Verlag, Berlin, Heidelberg, 2013.

[34] A. Dainotti, A. Pescape, and K.C. Claffy. Issues and future directions in traffic classification. *Network, IEEE*, 26(1):35–40, January 2012.

[35] L. Deri, M. Martinelli, T. Bujlow, and A. Cardigliano. ndpi: Open-source high-speed deep packet inspection. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2014 International*, pages 617–622, Aug 2014.

[36] S. Alcock and R. Nelson. Measuring the accuracy of open-source payload-based traffic classifiers using popular internet applications. In *Local Computer Networks Workshops (LCN Workshops), 2013 IEEE 38th Conference on*, pages 956–963, Oct 2013.

[37] Tomasz Bujlow, Valentín Carela-Español, and Pere Barlet-Ros. Independent comparison of popular {DPI} tools for traffic classification. *Computer Networks*, 76:75 – 89, 2015.

[38] T. T.T. Nguyen and G. Armitage. A Survey of Techniques for Internet Traffic Classification Using Machine Learning. *Commun. Surveys Tuts.*, 10(4):56–76, October 2008.

[39] Andrew W. Moore and Denis Zuev. Internet Traffic Classification Using Bayesian Analysis Techniques. In *Proceedings of the 2005 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '05, pages 50–60, New York, NY, USA, 2005. ACM.

[40] Matthew Roughan, Subhabrata Sen, Oliver Spatscheck, and Nick 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*, IMC '04, pages 135–148, New York, NY, USA, 2004. ACM.

[41] Nigel Williams, Sebastian Zander, and Grenville Armitage. A Preliminary Performance Comparison of Five Machine Learning Algorithms for Practical IP Traffic Flow Classification. *SIGCOMM Comput. Commun. Rev.*, 36(5):5–16, October 2006.

[42] Silvio Valenti, Dario Rossi, Michela Meo, Marco Mellia, and Paola Bermolen. Accurate, Fine-Grained Classification of P2P-TV Applications by Simply Counting Packets. In Maria Papadopouli, Philippe Owezarski, and Aiko Pras, editors, *Traffic Monitoring and Analysis*, volume 5537 of *Lecture Notes in Computer Science*, pages 84–92. Springer Berlin Heidelberg, 2009.

[43] Pedro Casas, Johan Mazel, and Philippe Owezarski. MINETRAC: Mining Flows for Unsupervised Analysis & Semi-supervised Classification. In *Teletraffic Congress (ITC), 2011 23rd International*, pages 87–94, Sept 2011.

[44] Laurent Bernaille, Renata Teixeira, Ismael Akodkenou, Augustin Soule, and Kave Salamatian. Traffic classification on the fly. *SIGCOMM Comput. Commun. Rev.*, 36(2):23–26, April 2006.

[45] Valentín Carela-Español, Pere Barlet-Ros, Albert Cabellos-Aparicio, and Josep Solé-Pareta. Analysis of the impact of sampling on netflow traffic classification. *Comput. Netw.*, 55(5):1083–1099, April 2011.

[46] Sunghwan Ihm and Vivek S. Pai. Towards understanding modern web traffic. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11, pages 295–312, New York, NY, USA, 2011. ACM.

[47] Jeffrey Erman, Alexandre Gerber, Mohammad T. Hajiaghayi, Dan Pei, and Oliver Spatscheck. Network-aware Forward Caching. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 291–300, New York, NY, USA, 2009. ACM.

[48] Jeffrey Erman, Alexandre Gerber, and Subhabrata Sen. HTTP in the Home: It is Not Just About PCs. *SIGCOMM Comput. Commun. Rev.*, 41(1):90–95, January 2011.

[49] Fabian Schneider, Bernhard Ager, Gregor Maier, Anja Feldmann, and Steve Uhlig. Pitfalls in HTTP Traffic Measurements and Analysis. In *Proceedings of the 13th International Conference on Passive and Active Measurement*, PAM'12, pages 242–251, Berlin, Heidelberg, 2012. Springer-Verlag.

[50] Wei Li, Marco Canini, Andrew W. Moore, and Raffaele Bolla. Efficient Application Identification and the Temporal and Spatial Stability of Classification Schema. *Comput. Netw.*, 53(6):790–809, April 2009.

[51] Ignacio N. Bermudez, Marco Mellia, Maurizio M. Munafo, Ram Keralapura, and Antonio Nucci. DNS to the Rescue: Discerning Content and Services in a Tangled Web. In *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, IMC '12, pages 413–426, New York, NY, USA, 2012. ACM.

[52] Tatsuya Mori, Takeru Inoue, Akihiro Shimoda, Kazumichi Sato, Keisuke Ishibashi, and Shigeki Goto. Sfmap: Inferring services over encrypted web flows using dynamical domain name graphs. In Moritz Steiner, Pere Barlet-Ros, and Olivier Bonaventure, editors, *Traffic Monitoring and Analysis*, volume 9053 of *Lecture Notes in Computer Science*, pages 126–139. Springer International Publishing, 2015.

[53] Arian Bär, Antonio Barbuzzi, Piertro Michiardi, and Fabio Ricciato. Two Parallel Approaches to Network Data Analysis. In *Proc. LADIS 2011: The 5th Workshop on Large Scale Distributed Systems and Middleware, Seattle*, September 2011.

[54] Vinicius Gehlen, Alessandro Finamore, Marco Mellia, and MaurizioM. MunafÃš. Uncovering the Big Players of the Web. In Antonio PescapÃĺ, Luca Salgarelli, and Xenofontas Dimitropoulos, editors, *Traffic Monitoring and Analysis*, volume 7189 of *Lecture Notes in Computer Science*, pages 15–28. Springer Berlin Heidelberg, 2012.

[55] David Naylor, Alessandro Finamore, Ilias Leontiadis, Yan Grunenberger, Marco Mellia, Maurizio Munafò, Konstantina Papagiannaki, and Peter Steenkiste. The Cost of the "S" in HTTPS. In *Proceedings of the 10th ACM International on Conference on Emerging Networking Experiments and Technologies*, CoNEXT '14, pages 133–140, New York, NY, USA, 2014. ACM.

[56] Craig Labovitz, Scott Iekel-Johnson, Danny McPherson, Jon Oberheide, and Farnam Jahanian. Internet Inter-domain Traffic. *SIGCOMM Comput. Commun. Rev.*, 41(4):–, August 2010.

[57] Cheng Huang, Angela Wang, Jin Li, and Keith W. Ross. Measuring and Evaluating Large-scale CDNs Paper Withdrawn at Mirosoft's Request. In *Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement*, IMC '08, pages 15–29, New York, NY, USA, 2008. ACM.

[58] Rupa Krishnan, Harsha V. Madhyastha, Sridhar Srinivasan, Sushant Jain, Arvind Krishnamurthy, Thomas Anderson, and Jie Gao. Moving Beyond End-to-end Path Information to Optimize CDN Performance. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '09, pages 190–201, New York, NY, USA, 2009. ACM.

[59] Erik Nygren, Ramesh K. Sitaraman, and Jennifer Sun. The Akamai Network: A Platform for High-performance Internet Applications. *SIGOPS Oper. Syst. Rev.*, 44(3):2–19, August 2010.

[60] Mohammad Al-Fares, Khaled Elmeleegy, Benjamin Reed, and Igor Gashinsky. Overclocking the yahoo!: Cdn for faster web page loads. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11, pages 569–584, New York, NY, USA, 2011. ACM.

[61] Patrick Wendell and Michael J. Freedman. Going viral: Flash crowds in an open cdn. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11, pages 549–558, New York, NY, USA, 2011. ACM.

[62] Benjamin Frank, Ingmar Poese, Yin Lin, Georgios Smaragdakis, Anja Feldmann, Bruce Maggs, Jannis Rake, Steve Uhlig, and Rick Weber. Pushing cdn-isp collaboration to the limit. *SIGCOMM Comput. Commun. Rev.*, 43(3):34–44, July 2013.

[63] Ingmar Poese, Benjamin Frank, Bernhard Ager, Georgios Smaragdakis, and Anja Feldmann. Improving content delivery using provider-aided distance information. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, IMC '10, pages 22–34, New York, NY, USA, 2010. ACM.

[64] M. Calder, A. Flavel, E. Katz-Bassett, R. Mahajan, and J. Padhye. Analyzing the Performance of an Anycast CDN. In *Proceedings of the 2015 ACM Conference on Internet Measurement Conference*, IMC '15. ACM, 2015.

[65] Xun Fan, Ethan Katz-Bassett, and John Heidemann. Assessing affinity between users and cdn sites. In Moritz Steiner, Pere Barlet-Ros, and Olivier Bonaventure, editors, *Traffic Monitoring and Analysis*, volume 9053 of *Lecture Notes in Computer Science*, pages 95–110. Springer International Publishing, 2015.

[66] Matt Calder, Xun Fan, Zi Hu, Ethan Katz-Bassett, John Heidemann, and Ramesh Govindan. Mapping the expansion of google's serving infrastructure. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, pages 313–326, New York, NY, USA, 2013. ACM.

[67] Weijian Sun, Xiaowei Qin, Shuang Tang, and Guo Wei. A QoE anomaly detection and diagnosis framework for cellular network operators. In *Computer Communications Workshops (INFOCOM WKSHPS), 2015 IEEE Conference on*, pages 450–455, April 2015.

[68] Jeffrey Erman, Alexandre Gerber, K. K. Ramadrishnan, Subhabrata Sen, and Oliver Spatscheck. Over the Top Video: The Gorilla in Cellular Networks. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11, pages 127–136, New York, NY, USA, 2011. ACM.

[69] Phillipa Gill, Martin Arlitt, Zongpeng Li, and Anirban Mahanti. Youtube Traffic Characterization: A View from the Edge. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, IMC '07, pages 15–28, New York, NY, USA, 2007. ACM.

[70] Michael Zink, Kyoungwon Suh, Yu Gu, and Jim Kurose. Characteristics of YouTube Network Traffic at a Campus Network - Measurements, Models, and Implications. *Comput. Netw.*, 53(4):501–514, March 2009.

[71] R. Torres, A. Finamore, Jin Ryong Kim, M. Mellia, M.M. Munafo, and Sanjay Rao. Dissecting Video Server Selection Strategies in the YouTube CDN. In *Distributed Computing Systems (ICDCS), 2011 31st International Conference on*, pages 248–257, June 2011.

[72] Alessandro Finamore, Marco Mellia, Maurizio M. Munafò, Ruben Torres, and Sanjay G. Rao. YouTube Everywhere: Impact of Device and Infrastructure Synergies on User Experience. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11, pages 345–360, New York, NY, USA, 2011. ACM.

[73] Yaping Zhu, B. Helsley, J. Rexford, A. Siganporia, and S. Srinivasan. LatLong: Diagnosing Wide-Area Latency Changes for CDNs. *Network and Service Management, IEEE Transactions on*, 9(3):333–345, September 2012.

[74] A. Botta and A. Pescape. Monitoring and Measuring Wireless Network Performance in the Presence of Middleboxes. In *Wireless On-Demand Network Systems and Services (WONS), 2011 Eighth International Conference on*, pages 146–149, Jan 2011.

[75] Raimund Schatz, Tobias Hossfeld, Lucjan Janowski, and Sebastian Egger. From packets to people: quality of experience as a new measurement challenge. In Ernst Biersack, Christian Callegari, and Maja Matijasevic, editors, *DataTraffic Monitoring and Analysis*, chapter From Packets to People: Quality of Experience As a New Measurement Challenge, pages 219–263. Springer-Verlag, Berlin, Heidelberg, 2013.

[76] Pedro Casas, Michael Seufert, and Raimund Schatz. YOUQMON: A System for Online Monitoring of YouTube QoE in Operational 3G Networks. *SIGMETRICS Perform. Eval. Rev.*, 41(2):44–46, August 2013.

[77] Antonio Liotta, Vlado Menkovski, Georgios Exarchakos, and Antonio Cuadra Sánchez. Quality of Experience Models for Multimedia Streaming. *Int. J. Mob. Comput. Multimed. Commun.*, 2(4):1–20, October 2010.

[78] Pedro Casas, Andreas Sackl, Sebastian Egger, and Raymund Schatz. YouTube & Facebook Quality of Experience in mobile broadband networks. In *Globecom Workshops (GC Wkshps), 2012 IEEE*, pages 1269–1274, Dec 2012.

[79] Alan Mislove, Massimiliano Marcon, Krishna P. Gummadi, Peter Druschel, and Bobby Bhattacharjee. Measurement and Analysis of Online Social Networks. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, IMC '07, pages 29–42, New York, NY, USA, 2007. ACM.

[80] Yabing Liu, Krishna P. Gummadi, Balachander Krishnamurthy, and Alan Mislove. Analyzing Facebook Privacy Settings: User Expectations vs. Reality. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11, pages 61–70, New York, NY, USA, 2011. ACM.

[81] Gabriel Magno, Giovanni Comarela, Diego Saez-Trumper, Meeyoung Cha, and Virgilio Almeida. New Kid on the Block: Exploring the Google+ Social Graph. In *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, IMC '12, pages 159–170, New York, NY, USA, 2012. ACM.

[82] Roberto Gonzalez, Ruben Cuevas, Reza Rejaie, and Ángel Cuevas. Google+ or Google-?: Examining the Popularity of the new OSN. *CoRR*, abs/1205.5662, 2012.

[83] Xiaohan Zhao, Alessandra Sala, Christo Wilson, Xiao Wang, Sabrina Gaito, Haitao Zheng, and Ben Y. Zhao. Multi-scale Dynamics in a Massive Online Social Network. *CoRR*, abs/1205.4013, 2012.

[84] Miltiadis Allamanis, Salvatore Scellato, and Cecilia Mascolo. Evolution of a Location-based Online Social Network: Analysis and Models. In *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, IMC '12, pages 145–158, New York, NY, USA, 2012. ACM.

[85] WhatsApp Blog. `http://blog.whatsapp.com/`, 2014. [Online; accessed 07-September-2014].

[86] H. Blodget. Everyone Who Thinks Facebook Is Stupid To Buy WhatsApp For $19 Billion Should Think Again. `http://www.businessinsider.com/why-facebook-buying-whatsapp-2014-2`, 2014. [Online; accessed 15-August-2014].

[87] Ekhiotz Jon Vergara, Simon Andersson, and Simin Nadjm-Tehrani. When Mice Consume Like Elephants: Instant Messaging Applications. In *Proceedings of the 5th International Conference on Future Energy Systems*, e-Energy '14, pages 97–107, New York, NY, USA, 2014. ACM.

[88] Andrius Aucinas, Narseo Vallina-Rodriguez, Yan Grunenberger, Vijay Erramilli, Konstantina Papagiannaki, Jon Crowcroft, and David Wetherall. Staying Online While

Mobile: The Hidden Costs. In *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*, CoNEXT '13, pages 315–320, New York, NY, USA, 2013. ACM.

[89] WhatsApp Status in Twitter. `https://twitter.com/wa_status`, 2015. [Online; accessed 13-August-2014].

[90] P. Saint-Andre. Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence (RFC-6121). Technical report, 2011.

[91] International Telecommunication Union. Methods for Subjective Determination of Transmission Quality (ITU-T Rec. P.800). Technical report, 1996.

[92] Estimating End-to-End Performance in IP Networks for Data Applications (ITU-T Rec. G.1030). Technical report, International Telecommunication Union, 1996.

[93] Pedro Casas, Hans Roland Fischer, Stefan Suette, and Raymund Schatz. A First Look at Quality of Experience in Personal Cloud Storage Services. In *Communications Workshops (ICC), 2013 IEEE International Conference on*, pages 733–737, June 2013.

[94] Junchen Jiang, Vyas Sekar, Ion Stoica, and Hui Zhang. Shedding light on the structure of internet video quality problems in the wild. In *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*, CoNEXT '13, pages 357–368, New York, NY, USA, 2013. ACM.

[95] Yaping Zhu, B. Helsley, J. Rexford, A. Siganporia, and S. Srinivasan. Latlong: Diagnosing wide-area latency changes for cdns. *Network and Service Management, IEEE Transactions on*, 9(3):333–345, September 2012.

[96] Anukool Lakhina, Mark Crovella, and Christiphe Diot. Characterization of network-wide anomalies in traffic flows. In *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement*, IMC '04, pages 201–206, New York, NY, USA, 2004. ACM.

[97] D. Giordano, S. Traverso, L. Grimaudo, M. Mellia, E. Baralis, A. Tongaonkar, and S. Saha. Youlighter: An unsupervised methodology to unveil youtube cdn changes. In *Teletraffic Congress (ITC 27), 2015 27th International*, pages 19–27, Sept 2015.

[98] Mirko Schiavone, Peter Romirer-Maierhofer, Fabio Ricciato, and Andrea Baiocchi. Towards Bottleneck Identification in Cellular Networks via Passive TCP Monitoring. In Song Guo, Jaime Lloret, Pietro Manzoni, and Stefan Ruehrup, editors, *Ad-hoc, Mobile, and Wireless Networks*, volume 8487 of *Lecture Notes in Computer Science*, pages 72–85. Springer International Publishing, 2014.

[99] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3):15:1–15:58, July 2009.

[100] A. Mitrokotsa and C. Douligeris. Detecting Denial of Service Attacks Using Emergent Self-organizing Maps. In *Signal Processing and Information Technology, 2005. Proceedings of the Fifth IEEE International Symposium on*, pages 375–380, Dec 2005.

[101] M. Ostaszewski, F. Seredynski, and P. Bouvry. A Nonself Space Approach to Network Anomaly Detection. In *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International*, pages 8 pp.–, April 2006.

[102] W. Chimphlee, A.H. Abdullah, M. Noor Md Sap, S. Chimphlee, and S. Srinoy. Integrating Genetic Algorithms and Fuzzy c-Means for Anomaly Detection. In *INDICON, 2005 Annual IEEE*, pages 575–579, Dec 2005.

[103] G. Prashanth, V. Prashanth, P. Jayashree, and N. Srinivasan. Using Random Forests for Network-based Anomaly detection at Active routers. In *Signal Processing, Communications and Networking, 2008. ICSCN '08. International Conference on*, pages 93–96, Jan 2008.

[104] Yang Li and Li Guo. An Efficient Network Anomaly Detection Scheme Based on TCM-KNN Algorithm and Data Reduction Mechanism. In *Information Assurance and Security Workshop, 2007. IAW '07. IEEE SMC*, pages 221–227, June 2007.

[105] M Raimondo and Nader Tajvidi. A Peaks over Threshold Model for Change-point Detection by Wavelets. *Statistica Sinica*, 14(2):395–412, 2004.

[106] Jake D. Brutlag. Aberrant Behavior Detection in Time Series for Network Monitoring. In *Proceedings of the 14th USENIX Conference on System Administration*, LISA '00, pages 139–146, Berkeley, CA, USA, 2000. USENIX Association.

[107] P.P.C. Lee, T. Bu, and T. Woo. On the Detection of Signaling DoS Attacks on 3G Wireless Networks. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 1289–1297, May 2007.

[108] Anukool Lakhina, Mark Crovella, and Christophe Diot. Diagnosing network-wide traffic anomalies. In *Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '04, pages 219–230, New York, NY, USA, 2004. ACM.

[109] Anukool Lakhina, Mark Crovella, and Christophe Diot. Mining Anomalies Using Traffic Feature Distributions. *SIGCOMM Comput. Commun. Rev.*, 35(4):217–228, August 2005.

[110] Augustin Soule, Kavé Salamatian, and Nina Taft. Combining filtering and statistical methods for anomaly detection. In *Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement*, IMC '05, pages 31–31, Berkeley, CA, USA, 2005. USENIX Association.

[111] Xin Li, Fang Bian, Mark Crovella, Christophe Diot, Ramesh Govindan, Gianluca Iannaccone, and Anukool Lakhina. Detection and identification of network anomalies using sketch subspaces. In *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, IMC '06, pages 147–152, New York, NY, USA, 2006. ACM.

[112] Daniela Brauckhoff, Xenofontas Dimitropoulos, Arno Wagner, and Kavè Salamatian. Anomaly extraction in backbone networks using association rules. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '09, pages 28–34, New York, NY, USA, 2009. ACM.

[113] Fernando Silveira, Christophe Diot, Nina Taft, and Ramesh Govindan. Astute: Detecting a different class of traffic anomalies. In *Proceedings of the ACM SIGCOMM 2010 Conference*, SIGCOMM '10, pages 267–278, New York, NY, USA, 2010. ACM.

[114] George Nychis, Vyas Sekar, David G. Andersen, Hyong Kim, and Hui Zhang. An Empirical Evaluation of Entropy-based Traffic Anomaly Detection. In *Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement*, IMC '08, pages 151–156, New York, NY, USA, 2008. ACM.

[115] Bernhard Tellenbach, Martin Burkhart, Didier Sornette, and Thomas Maillart. Beyond Shannon: Characterizing Internet Traffic with Generalized Entropy Metrics. In SueB. Moon, Renata Teixeira, and Steve Uhlig, editors, *Passive and Active Network Measurement*, volume 5448 of *Lecture Notes in Computer Science*, pages 239–248. Springer Berlin Heidelberg, 2009.

[116] Yu Gu, Andrew McCallum, and Don Towsley. Detecting Anomalies in Network Traffic Using Maximum Entropy Estimation. In *Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement*, IMC '05, pages 32–32, Berkeley, CA, USA, 2005. USENIX Association.

[117] Tamraparni Dasu, Shankar Krishnan, Suresh Venkatasubramanian, and Ke Yi. An Information-theoretic Approach to Detecting Changes in Multi-dimensional Data Dtreams. In *In Proc. Symp. on the Interface of Statistics, Computing Science, and Applications*, 2006.

[118] Haakon Ringberg, Matthew Roughan, and Jennifer Rexford. The Need for Simulation in Evaluating Anomaly Detectors. *SIGCOMM Comput. Commun. Rev.*, 38(1):55–59, January 2008.

[119] Peter Romirer-Maierhofer, Mirko Schiavone, and Alessandro DâĂŹAlconzo. Device-Specific Traffic Characterization for Root Cause Analysis in Cellular Networks. In Moritz Steiner, Pere Barlet-Ros, and Olivier Bonaventure, editors, *Traffic Monitoring and Analysis*, volume 9053 of *Lecture Notes in Computer Science*, pages 64–78. Springer International Publishing, 2015.

[120] Yiyi Huang, Nick Feamster, Anukool Lakhina, and Jim (Jun) Xu. Diagnosing network disruptions with network-wide analysis. *SIGMETRICS Perform. Eval. Rev.*, 35(1):61–72, June 2007.

[121] Fernando Silveira and Christophe Diot. Urca: Pulling out anomalies by their root causes. In *Proceedings of the 29th Conference on Information Communications*, INFOCOM'10, pages 722–730, Piscataway, NJ, USA, 2010. IEEE Press.

[122] Ignasi Paredes-Oliva, Xenofontas Dimitropoulos, Maurizio Molina, Pere Barlet-Ros, and Daniela Brauckhoff. Automating Root-cause Analysis of Network Anomalies Using Frequent Itemset Mining. *SIGCOMM Comput. Commun. Rev.*, 40(4):467–468, August 2010.

[123] Partha Kanuparthy and Constantine Dovrolis. Pythia: Diagnosing Performance Problems in Wide Area Providers. In *Proceedings of the 2014 USENIX Conference on USENIX Annual Technical Conference*, USENIX ATC'14, pages 371–382, Berkeley, CA, USA, 2014. USENIX Association.

[124] He Yan, A. Flavel, Zihui Ge, A. Gerber, D. Massey, C. Papadopoulos, H. Shah, and J. Yates. Argus: End-to-end service anomaly detection and localization from an ISP's point of view. In *INFOCOM, 2012 Proceedings IEEE*, pages 2756–2760, March 2012.

[125] He Yan, Lee Breslau, Zihui Ge, Dan Massey, Dan Pei, and Jennifer Yates. G-RCA: A Generic Root Cause Analysis Platform for Service Quality Management in Large IP Networks. In *Proceedings of the 6th International COnference*, Co-NEXT '10, pages 5:1–5:12, New York, NY, USA, 2010. ACM.

[126] Jeffrey Erman, Martin Arlitt, and Anirban Mahanti. Traffic Classification Using Clustering Algorithms. In *Proceedings of the 2006 SIGCOMM Workshop on Mining Network Data*, MineNet '06, pages 281–286, New York, NY, USA, 2006. ACM.

[127] Jeffrey Erman, Anirban Mahanti, Martin Arlitt, Ira Cohen, and Carey Williamson. Semi-supervised Network Traffic Classification. *SIGMETRICS Perform. Eval. Rev.*, 35(1):369–370, June 2007.

[128] Ethan Katz-Bassett, Harsha V. Madhyastha, Vijay Kumar Adhikari, Colin Scott, Justine Sherry, Peter Van Wesep, Thomas Anderson, and Arvind Krishnamurthy. Reverse Traceroute. In *Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation*, NSDI'10, pages 15–15, Berkeley, CA, USA, 2010. USENIX Association.

[129] N. Hu and P. Steenkiste. Quantifying Internet end-to-end route Similarity. In *Passive and Active Network Measurement*, Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2006.

A. Implementation Details of ADTool

ADTool is a Perl implementation of the statistical distribution-based AD algorithm presented in Section 6.6, developed in the context of the European FP7 project mPlane. It runs on top of the streaming data-warehose system DBStream and hence it requires suitable DBStream jobs to compute traffic feature distributions with the required time-granularity. It is designed to run online, i.e. it processes the distributions of features as soon as they are available in the DBStream views.

ADTool is available on the mPlane website at this URL:

```
https://www.ict-mplane.eu/public/adtool
```

Architecture

ADTool runs iteratively on the output of DBStream jobs. At every iteration, the program tries to retrieve the distribution corresponding to the last timebin available and compares it with the distributions in the reference set (i.e. all the distributions corresponding to timebins in a reference window of predefined length).

ADTool is composed of the following modules:

Configs.pm This module provides an interface between the XML configuration file and the rest of the software. The parsing of the XML file is done by the XML::Simple Perl standard module.

DataSrc.pm This module provides an interface between the PostgreSQL database used by DBStream and the rest of the software. It allows to connect to the database, query for the last available data to compute and write back the output. Both the read and the write interactions with the database are done by the standard Perl DBI module via SQL queries and inserts.

ENKLd.pm This module provides the computation of the normalized Kullback-Leibler divergence between two distribution of values. The two distributions are passed to this module as array references and do not need to be normalized in advance.

RefSet.pm This module defines the package RefSet for managing Reference Sets (collection of past distributions). After being instantiated, a "raw" RefSet object contains all the distributions in the specified reference window. The module provides functions to discard not statistically-relevant distributions (eg. not enough samples). The output code can be either 2 (if distribution to be tested is too small) or 3 (if the reference set does not contain enough samples).

ADTest.pm This module implements the test logic of the AD algorithm. It requires the distribution to be tested, the reference set and other algorithm parameters (i.e. alpha, gamma). The output code can be either 0 (if normal) or 1 (if anomalous).

DBStream Jobs

In order to run ADTool, it is firstly necessary to set-up a suitable DBStream job to compute counters of the feature for each variable and time bin. The output view of the job should have the following columns:

- `serial time`
- `variable name`
- `feature name`

Note that, a single view can be used to collect multiple feature if the `variable name` and the time resolution is compatible.

Configuration file

The configuration of the software is done via an XML file. The available options are:

- [database] host
- [database] port
- [database] username
- [database] password
- [database] features table name (output of DBStream job)
- [database] drift table name (output of ADTool)
- [analysis] start timestamp
- [analysis] end timestamp (0 means run forever)
- [analysis] name of variable upon whom the job has computed the distribution

- [analysis] feature name
- [refset] width (in days)
- [refset] guard period (in hours)
- [refset] min refset size (minimum number of distributions in refset)
- [refset] min distr size (minimum number of samples in distribution)
- [refset] m (number of top ranked distributions in refset)
- [ADtest] alpha (algorithm's sensitivity)

A sample configuration file is showed in the following listing:

```
<ADTool_config>

  <!-- description of this instance of ADTool                    -->
  <Description>
              Distribution of traffic volume
                          across YouTube servers
  </Description>

  <!-- setting up database credentials and table names -->
  <Database host="1.1.1.1" port="5440" dbname="dbstream" user="dbstream">
    <features_table>adtool_youtube_server_volume_distrib</features_table>
    <flags_table>adtool_youtube_server_volume_flags</flags_table>
    <drift_table>adtool_youtube_server_volume_drift</drift_table>
  </Database>

  <!-- setting up analysis info (variable names, timestamps, etc.) -->
  <Analysis>
    <start>1396648800</start>       <!-- analysis start timestamp  -->
    <end>0</end>                    <!-- 0 means run online        -->
    <granularity>300</granularity>  <!-- time granularity (in sec) -->

    <variable>server_ip</variable>  <!-- distrib. element names    -->
    <feature>volume_down</feature>  <!-- traffic feature name      -->
  </Analysis>

  <!-- setting reference set -->
  <RefSet>
    <width>7</width>            <!-- ref. set time window (in days)-->
    <guard>2</guard>            <!-- guard period in hours         -->
    <min_distr_size>100</min_distr_size>  <!-- min distr size      -->
    <min_refset_size>80</min_refset_size> <!-- min refset size     -->
    <slack_var>0.1</slack_var>  <!-- for comparing size of timebins-->
    <m>20</m>                   <!-- usually 1/4 min refset size   -->
  </RefSet>

  <!-- setting up distribution-based algorithm                    -->
  <ADTest>
    <alpha>0.05</alpha>          <!-- algorith sensitivity         -->
  </ADTest>

</ADTool_config>
```

Workflow

The logic is defined in the main executable `adtool.pl`. The arguments for running the program are:

- `--config <CONFIG FILE>`
- `--log <LOG FILE>`

The execution workflow is described in Figure A.1.

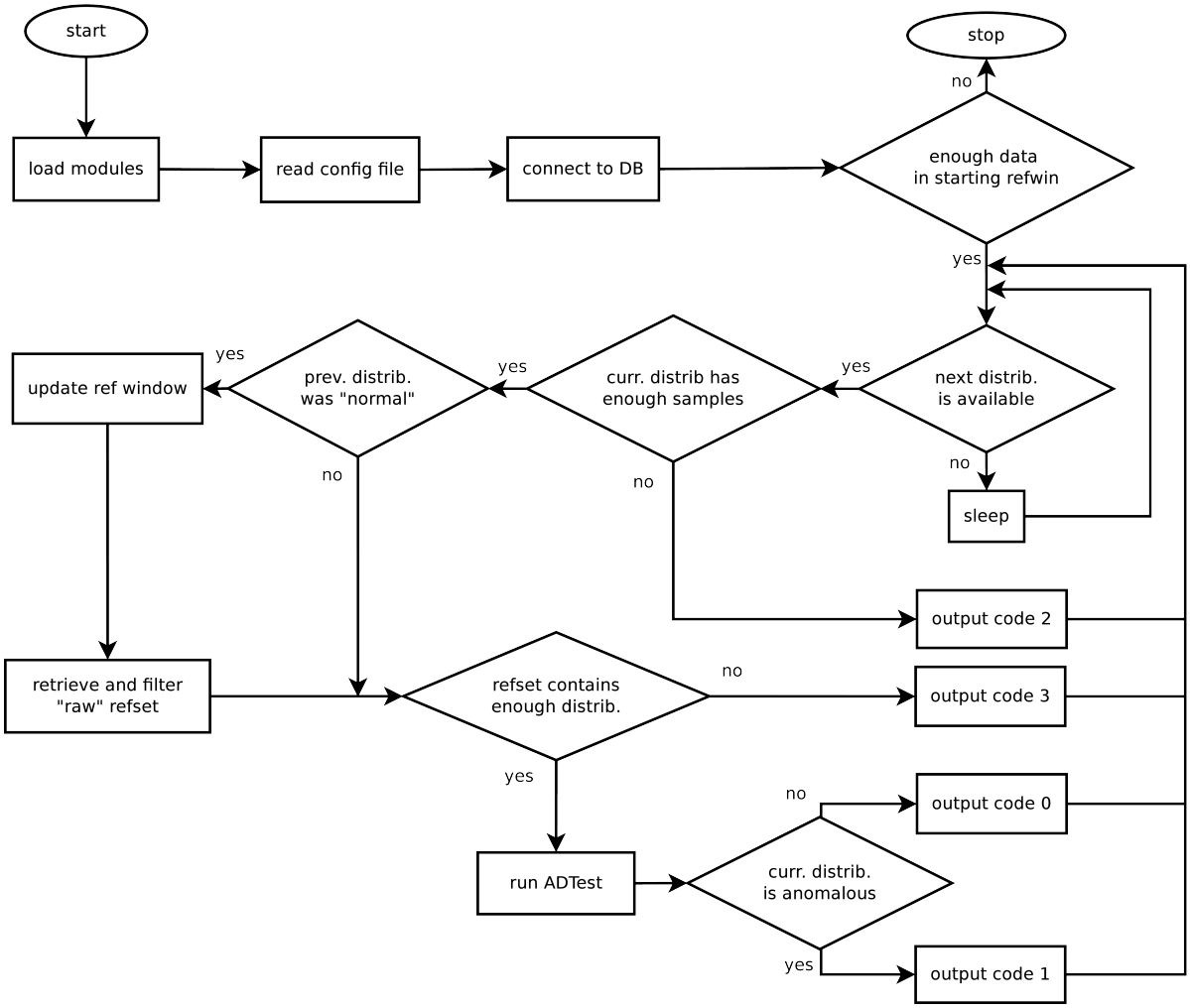


Figure A.1.: Flowchart of the execution of ADTool.

Output

At every completed iteration, the output is reported on STDOUT as well as on the database's flag table specified in the configuration. For each iteration running on a time-bin, the row

inserted in the flag table is composed by the following column:

- beginning timestamp of the timebin
- feature name (e.g., `volume-down`)
- output code (0,1,2,3)
- score
- γ (current distance)
- ϕ_α (upper bound)

The available output codes are:

- 0: distribution is "normal"
- 1: distribution is anomalous
- 2: distribution does not contain enough samples
- 3: refset does not contain enough distributions.

B. Algorithms for Anomaly Classification and Parameters

In this appendix we provide a brief description of the algorithms used for the supervised classification of anomalies described in Chapter 7. We additionally list, for each classifier, the parameters used for the experiments with Weka.

Decision Tree (DT) is a classification technique based on a tree graph, where inner nodes correspond to a condition on an attribute and leaves are the outcome (i.e., the class). It is a very popular classification algorithm due to its simplicity (it can easily be converted in a rule-based classification system) and readability (it can be graphically represented). The training follows a top-down greedy algorithm that works iteratively splitting the nodes, using either the Gini Index or the Information Gain. For the results presented in this thesis we employed the popular C4.5 implementation, which uses the latter.

binary split	true
pruning factor	0.25
minimum number of instances per leaf	1
reduced error pruning	false
subtree raising	false
reduced error pruning	false

Table B.1.: C4.5 Decision Tree settings

Random Forest (RF) is an ensemble technique based on multiple instances of decision trees, each one based on a different part of the training set. This instances are called *bootstrapped samples*. The final outcome is decided with major voting.

maximum tree depth	0 (i.e., unlimited)
number of attributes for random selection	0
number of trees	100
seed for random number generation	0

Table B.2.: Random Forest settings

Support Vector Machines (SVM) are non-probabilistic binary linear classifiers. It is considered one of the most powerful supervised classification algorithm. It works by representing each item (vector) in a multidimensional space and trying to find a linear separation (i.e., an hyperplane) for the classes. In some cases, however, a linear separation of the space is not possible, hence it uses the so-called *kernel tricks*, which increase the dimensionality of the space in order to allow a better fit.

cost parameter	1.0
tolerance term. criterion (eps)	0.001
gamma	0
kernel type	radial basis function
normalize	false
probability estimate	false

Table B.3.: Support Vector Machines (SVM) settings

Naïve Bayes (NB) is a very simple classifier based on Bayesian statistics. Despite its simplicity, it is widely used as it is very efficient in a number of scenarios, especially in high-dimensional datasets. It works by assuming that each feature is independent, which is not true in most cases, hence the adjective *naive*. This assumption allows for an easy calculation of class-conditional probabilities using maximum likelihood.

use a kernel estimator	false
use supervised discretization	false

Table B.4.: Naive Bayes (NB) settings

Locally-Weighted-based Learning (LWL) is another Bayes classifier. It overcomes the limitations of NB, i.e., the assumption of feature independence, by learning local models. Being a *lazy* algorithm, it constructs a new Bayes model using a weighted set of training instances at classification time.

num. of neighbors for weighting function (KNN)	-1 (i.e., all)
use supervised discretization	false
nearest neighbor search algorithm	LinearNN
weighting kernel	0

Table B.5.: Locally-Weighted-based Learning (LWL) settings

Multi-Layer Perceptron (MLP) is an artificial neural network composed of multiple layers of neurons (i.e., processing units). The layers are fully connected in a feed-forward scheme. Each neuron employs an activation function that maps the weighted inputs to the output that is passed to the following layer. The weights, originally set to random values, are iteratively adjusted during the training phase.

decay (learning rate decrease)	false
hidden layers	a
learning rate	0.3
momentum	0.2
nominal to binary filters	irrelevant
normalize attributes	true
normalize numeric class	false
allow reset with lower learning rate	true
seed for random number generator	0
number epochs for training	600
percentage size of validation set	0
validation threshold for termination	20

Table B.6.: Multi-Layer Perceptron (MLP) settings