Programme ANR VERSO

Projet VIPEER

Ingénierie du trafic vidéo en intradomaine basée sur les paradigmes du Pair à Pair

Décision n° 2009 VERSO 014 01 à 06 du 22 décembre 2009
T0 administratif = 15 Novembre 2009
T0 technique = 1er Janvier 2010

Livrable 2.1

Deployment of a stand-alone measurement testbed

Auteurs:
S.Vaton, C.Lohr, M.K.Sbai (Telecom Bretagne), Y.Hadjadj-Aoul, K.Singh (INRIA), S.Moteau (France Telecom)
Compilé par:
S.Vaton (Telecom Bretagne)

Décembre 2010

Telecom Bretagne; Eurecom; INRIA; France Telecom; NDS; ENVIVIO

Abstract

This is the first deliverable of the Working Group 2 of the VIPEER project. WP2 focuses on the definition of a measurement infrastructure and of several measurement primitives. The main measurement primitives are (i) the classification of flows per application or category of applications (ii) the monitoring of the Quality of Experience, that is to say the Quality of Service as it is perceived by the end users of the CDN and (iii) the monitoring of the network level QoS (delay, losses, connectivity) by active probing methods. The scope of the measurement infrastructure is to provide the CDN with some measurement information that is useful for optimizing the delivery of the contents from the point of view of the end users (QoE) and from the point of view of the entity which operates the CDN, in the case of VIPEER a telecommunications operator.

Keywords: network monitoring, QoE, QoS, traffic classification, testbed

Contents

1 Introduction

1.1 Network monitoring

Network traffic monitoring is a broad topic with many different applications among which one can cite:

- the detection of security threats with Intrusion Detection Systems
- traffic modeling from various applications (VoIP, audio or video streaming, P2P, chat, web browsing, etc...) for the evaluation of the performance offered to these applications, and the impact that they have on the QoS of other applications
- estimation of network Quality of Service (QoS) parameters such as the connectivity, the available bandwidth, the delays, the jitter...,
- real-time detection of applications in order (i) to determine the mixture of traffic in terms of applications (ii) to identify one particular application (iii) i to differentiate the treatment of flows in the routers
- estimation of the QoS as it is perceived by the end user (Quality of Experience, QoE) for different multimedia applications (video, voice...) taking into account both multimedia parameters (codecs, bit rate, motion level, etc...) and network parameters (losses, delays, etc...).

Task 2 of the VIPEER project is dedicated to the design of a "measurement layer". In the framework of VIPEER the studies on network monitoring will focus mainly on two points, on-the-fly service recognition and the monitoring of end-user perceived QoS (Quality of Experience, QoE). It will also be necessary to monitor network level QoS parameters (connectivity, delay, bandwith, etc...) focusing mainly on one-way metrics since these metrics are the most difficult to obtain.

1.2 Positioning of Task 2 with respect to the other Tasks

The positioning of Task 2 with respect to the other Tasks as it was defined in the Description of Work of the VIPEER project is described below.

1.2.1 Task2/Task 1

Task 1 is in charge of the technical coordination of the solutions to be developed. The objective of Task 1 is to define the architecture of the overall system for delivering video content with a certain level of QoE. A rough vision of the global architecture of the system which is defined in Task 1 and shall be demonstrated in Task 5 is that:

- storage capacities are disseminated in the network, either in customer premises or relatively high in the network (DSLAM, routers, etc...)
- algorithms run on top of storage capacities to orchestrate the content delivery, possibly in cooperation with a single server or with a CDN equipped with powerful servers
- the algorithms running on top of the storage capabilities are fed with (i) information on the network state (available bandwith, level of congestion, routing, etc...) (ii) and with some user-oriented information

The information concerning the end user can be :

- the quality that is experienced by the end-user (QoE) taking into account not only network conditions but also media-related parameters such as the codec, the level of motion, etc...
- some insight about the activity of the end-user; as one of the possible locations for the video content are within the customer premises (for example on the Set-Top-Box with a publicly accessible subbox operated by a virtual system) it is important to be able to track the actvity of the end-user so that the operation of the contents delivery system do not degrade the quality offered to the other applications running in the customer's premise

1.2.2 Task2/Task 3

Task 3.3. is in charge of developing a statistical engine which implements the adaptive, multidimensional and contextual side of the overall content adaptation architecture. This will take into account: (i) the static or semi-static context (type of access network, device type, content type as defined - for example - in a Service Level Agreement, popularity of the content) (ii) the dynamic or real-time contexts including the availability of ressources and possible QoS issues for delivering the content. Each statistical engine will be fed with metrics about the network state (reachability, available bandwith, delay, jitter, losses, etc...) which should be defined jointly by Task 2 and Task 3.3. The value of the metrics will be provided by Task 2 as an input to the statistical engines of Task 3.3. Value of the metrics will be used by Task 3.3. in order to adapt the distribution of the content to the dynamic context and also for reporting about achieved QoS levels.

1.2.3 Task 2/Task 5

Task 5 is responsible for the tests and demonstrations. Task 5.2. will integrate the developments and prototypes obtained in Tasks 2, 3 and 4. A label of the *Images et Réseaux* cluster has been obtained which may ease access to the Imagin'Lab platform, where real users are going to test new telecom services or products as beta-testers with a non-commercial network. It will be necessary to consider how we will migrate the measurement platform which is currently under development on the Imagin'Lab platform.

1.3 Scope and summary of this deliverable

The scope of this deliverable is to describe the monitoring testbed that will be settled down for VIPEER. We will discuss both the architecture of that testbed and the technical implementation details of its main functionalities: (i) network level QoS parameters monitoring (ii) traffic classification (iii) QoE monitoring.

Chapter 2 is dedicated to the analysis of the requirements and possible tools for the monitoring of network level QoS parameters (connectivity, bandwith, delay, etc.) Chapter 3 is dedicated to the analysis of QoE monitoring methodologies and to the description of PSQA (Pseudo-Subjective Quality Assessment), a method developed by partner IRISA. Chapter 4 will introduce the field of traffic classification and describe Nicofix, a tool developed by partner Institut Telecom. Chapter 5 describes the architecture of the monitoring testbed that will be deployed among the partners. Chapter 6 is a conclusion.

2 Network-level QoS monitoring

Task 3.3. will develop a statistical engine which will implement the adaptive contextual side of the content adaptation architecture. The context takes into account some static elements (such as access network type, device type, popularity of the content, etc. . .) and some real-time contexts including the availability of ressources and possible QoS issues. Each statistical engine will be fed with metrics about the network state (reachability, available bandwith, delay, jitter, losses, etc...). Task 2 is in charge of providing these metrics. It is first of all necessary to define more precisely the requirements of the statistical engines that is to say which are the metrics which will be fed to the statistical engines. Then it is necessary to identify which methods exist in order to monitor these quantities and which tools are publicly available.

Monitoring a dynamic and distributed network such as the Internet is a difficult task [1]. But monitoring is essential if one wants to evaluate the perception that the end users really have of the network. Monitoring is necessary in order to troubleshoot the root cause of degradations. This is a complex task since there are many different places where degradations can occur. The degradation can come from the host or from the network (packet treatment delays, queues, losses, TCP dynamics, etc.)

Monitoring can be performed at different levels: network, transport, application. It can be performed using active or passive methods, located in one network element or point-to-point. The goal can be to monitor a SLA, to detect faults, to analyse the QoS of an application.

Some simple methods such as Ping or TraceRoute exist. Ping uses ICMP packets and permits the estimation of RTT delays and packet losses. TraceRoute uses UDP packets with different TTL and analyzes the ICMP packets received from routers in order to monitor the route to destination, the delay in each router and the packet losses. But simple monitoring methods based on ICMP packets have some limitations. Delays and losses are asymetric. ISP can filter out ICMP packets or reduce their processing time so that average metrics values obtained for ICMP packets are not verified with other protocols.

The IPPM group at IETF has defined different categories of network level metrics: connectivity (RFC 2678), delay (RFC 2679), losses (RFC 2680), RTT delay (RFC 2681), losses pattern (RFC 3357), jitter (RFC 3393), etc. Different tools exist for measuring these metrics; some of them are open-source, while others are proprietary.

For example different tools exist in order to monitor bandwith. Different metrics can be defined: link bandwith, bottleneck link bandwith, available bandwith on a path, etc. Nettimer is a tool for measuring bottleneck link bandwith. End-to-end

available bandwith can be monitored by tools such as Nettimer or Spruce. Pathchar monitors the bandwith of each link over a path.

One way metrics are difficult to obtain since they require collaboration and an accurate synchronization of both end hosts. OWAMP (RFC 4656) standardizes the widespread collection of one way active measurements. Hosts with very accurate time reference are more and more widely available. The goal of OWAMP is to facilitate the widespread deployment of open OWAMP servers that would make inter-domain one way active delay measurement as common as RTT measurement using tools like Ping. At least two open source implementations of OWAMP are available from http://e2epi.internet2.edu/owamp/ and from http://www.av.it.pt/jowamp/.

Saturne is an active measurement tool developed by the RSM department of Télécom Bretagne. Saturne evaluates one-way metrics (delay and losses). It performs measurements on IPv4 and IPv6 networks. A non-intrusive version is available for IPv6 networks that evaluates directly the application flow instead of sending probes. Saturne supports the export of monitoring data to some collector.

Netflow is a functionality developed by CISCO in order to differentiate flows and route them rapidly. Netflow can be used in order to obtain flow-level statistics. Netflow has been normalized at IETF and is implemented by many vendors. But Netflow has some limitations. Flow records computation is very demanding for the routers. As a consequence a sampled version of Netflow has been developed. In this version only 1 out of N packets is processed, a typical value of N being 1000. Sampling introduces some biases in flow-level statistics. Netflow records can be used in order to estimate point-to-point delays if accurate time-sampling is available. The correlation between flow records from different hosts requires some specific data mining methods since the cardinality of the flow identifier state space is huge.

IPFIX is an IETF working group that was created from the need of standardization for IP flow information export from routers. The standard should define how IP flow information is to be formatted and transferred from an exporter to a collector. Requirements were outlined in the original RFC 3917. The working group chose Cisco Netflow Version 9 as the basis for IPFIX. Using the IPFIX protocol, an Exporter sends flow level information to a Collector in a "push" manner. IPFIX prefers SCTP as its transport layer protocol but it can also use TCP or UDP. There are few available implementations of the IPFIX protocol. Some implementations can be found from http://aircert.sourceforge.net/fixbuf/ or http://luca.ntop.org/draft-deri-ipfix-impl-00.txt

As one can see there is an important activity of standardization at the IETF concerning QoS monitoring. Different tools are available, some of them under GPL licence. These tools could be used by VIPEER partners to monitor QoS parameters. The monitoring infrastructure will be deployed among the different partners as this is explained in Chapter 5. There is still some work to be done in order to define more precisely which metrics will be required by Task 3.3 and select the tools which will provide these metrics.

3 QoE monitoring

3.1 Introduction

Providing Quality of Service (QoS) has always been an important task for Internet Service Providers. QoS covers a large set of aspects of networking systems, and represents today a global concept focusing on network-oriented ways of characterizing the behavior of infrastructures, applications and services, sharing that common quality keyword. But QoS does not address another critical component of the global system, the user. When we consider the behavior of the systems from the user perspective and with the idea of quality in mind, we can rely in what can be called "simple" ways of capturing it, consisting in using specific metrics that we know are strongly correlated with quality. Some examples are response times (i.e. delays), jitter, different measures of losses, etc. A more direct but challenging way is to try to directly quantify this quality aspect of the service provided by the user. In this context, we speak about Quality of Experience (QoE), in order to underline this focus on the user. For video applications, the main component of QoE is obviously the perceptual quality of the video sequences themselves as seen by the final user. The QoS metrics mentioned before, such as delays or losses (that is, specific metrics measuring precise aspects of them, such as the mean end-to-end delay over some interval, or the packet loss rate in equilibrium), have an impact on the perceptual or perceived quality, but the latter depends on many aspects related to network factors such as delays, or losses, and also on the characteristics of the coding scheme used, the bandwidth of the video flow, the possible protection against losses, etc. One of the challenging problems we will address in VIPEER is how to measure this global quality metric, as perceived by the user, of the service provided by the CDN.

3.2 State of the art on monitoring solutions for video

Some commonly used objective measures for video are explained below.

3.2.1 Peak Signal to Noise ratio

The most common and simple objective video quality assessment is the Peak Signal–to–Noise Ratio (PSNR).We can define the Mean Square Error (MSE) between

the original video sequence o and the distorted sequence d as:

$$MSE = \frac{1}{K.M.N} \sum_{k=1}^{K} \sum_{m=1}^{M} \sum_{n=1}^{N} [o_k(m,n) - d_k(m,n)]^2$$

where each video sequence has K frames of $M \times N$ pixels each, and $o_k(m,n)$ and $d_k(m,n)$ are the luminance pixels in position (m,n) in the kth frame of each sequence. The PSNR is the logarithmic ratio between the maximum value of a signal and the background noise (MSE). If the maximal luminance value in the frame is L (when the pixels are represented using 8 bits per sample, $L = 255$) then:

$$PSNR = 10 \log_{10} \frac{L^2}{MSE}$$

An extension to color video sequences has been proposed, by considering also the chrominance. The first advantage of PSNR is that it is easy to compute. However, it is not appropriate in our QoE context since it may not correlate well with perceptual quality measures.

3.2.2 Video Quality Metrics (VQM)

Video Quality Metric (VQM) [13] is developed by the "Institute for Telecommunication Sciences" (ITS). First it extracts some features from both the original and the distorted sequences. The features are the objective measures that characterize perceptual changes of the sequence by analyzing spatial, temporal, and chrominance information. Then, a set of quality parameters are computed comparing the original and distorted features. Using these parameters, a classification assigns a global quality measure. The classification is a linear combination calculated using functions that model human visual masking. These impairments are then statistically pooled to obtain a single quality measure for the total sequence. Thus, VQM makes a comparison between the original and distorted sequences based only on a set of features extracted independently from each video. It is useful when it is not possible to have the original and received sequences at the same time, for instance in a network. But it still needs some information about both sequences including the original.

VQM is accepted as an objective video quality standard by ANSI, and some studies shows a good correlation with subjective tests for low bitrate encodings while it does not perform well for encodings with high bitrates.

3.2.3 Moving Picture Quality Metric (MPQM) and Color Moving Picture Quality Metric (CMPQM)

Moving Picture Quality Metric (MPQM) [12] and its color extension, Color Moving Picture Quality Metric (CMPQM) [12, 11], were developed by researchers working at the "École Polytechnique Fédérale de Lausanne" (EPFL). They are the most used objective metrics based on the Human Vision System (HVS) model.

Stimuli of the same amplitude are perceived different when they are included in flat spatial areas or in areas including edges. In general, stimuli with different spatial

and temporal frequencies are not perceived in the same way by the human eye. The HVS models these and other aspects of human visual perception, and it is included on the MPQM metric (and on the CMPQM metric) to improve its performance and robustness. In particular, MPQM incorporates the most important human perception phenomenon: contrast sensitivity and masking. These factors account for the fact that a minimal threshold is needed to detect a signal change, and the threshold depends on the contrast of the foreground/background relation.

MPQM considers a set of distorted perceptual components obtained from the original sequence and their difference with the distorted one. Each perceptual component is computed using signal processing filters, and they measure in some way the perceptual differences between the original sequence and the distorted sequence. Each component has sensitivity in the perceptual quality, considering its weight, a global distortion E is computed (as an important improvement of the MSE in the PSNR metric). Finally a Masked PSNR is defined:

$$MPSNR = 10 \log_{10} \frac{L^2}{E^2}$$

CMPQM is an extension to MPQM that also use the chrominance values. At first step it transforms the original and the distorted sequences to the linear opponent-color space (B/W, R/G, B/Y). Then, the computation follows in a very similar way th for the original MPQM. The authors of these proposals show that both metrics, MPQM and CMPQM, correlate well with subjective assessment in particular scenarios, especially for high bit rate codifications. Nevertheless, in more general situation, the correlation is more variable.

3.2.4 Normalization Fidelity Metric (NVFM)

Normalization Video Fidelity Metric (NVFM) [12], also developed by EPFL, is based on a visibility prediction followed by a normalization stage. As MPQM, the prediction is made in the pixel domain, using space and time linear transformations, but is applied to the original sequence and the distorted sequence (instead to its difference as in MPQM). The perceptual components obtained in the prediction stage are normalized based on the ratio between the excitatory and inhibitory of a inter–channel masking that consider the sensitivity weight. Finally, the measure metric is computed as the squared vector sum of the difference of the normalized responses.

3.2.5 Other works in Objective Video Quality Measures and Comparison

Perceptual Evaluation of Video Quality (PEVQ) compares the distorted video sequence with the reference video sequence and provides an estimate of Mean Opinion Score (MOS) (see section 3.2).

Structural Similarity Index (SSIM) [14, 15] is a structural distortion based technique. All the previously described methods are error based. Instead, HVS is not oriented towards extracting structural information from the viewing field. Therefore, a measurement of structural distortion should be a good approximation of perceived

Parameter	PSNR	VQM	MPQMS	CMPQM	NVFM	SSIM
Use of Original Sequence	Yes	Yes	Yes	Yes	Yes	Yes
Chrominance Consideration	No	Yes	No	Yes	Yes	Yes
Mathematical Complexity	Simple	Very Complex	Complex	Complex	Complex	Complex
Correlation with Subjective Methods	Poor	Good	Varying	Varying	Varying	Unknown

Table 3.1: Objective Metric Comparison

image distortion. Only studied by its authors, it is, yet, not clear the correlation of this approach with subjective tests.

Noise Quality Measure (NQM) [3] models the error source with a linear frequency distortion and additive noise injection; the two sources are considered independent. A distortion measure (DM) is used for the effect of the frequency distortion, and a noise quality measure (NQM) is used for the effect of the additive noise. A global perceptual quality based on the two measures (NQM and DM) is not defined.

Table 1 summarizes the considered objective metrics. Observe that all objective video quality metrics use the original video sequence (and the distorted video sequence). Therefore, it is not possible to use them in a real–time test environment, because the received and the original video are needed at the same time in the same place. Also in some applications the complex computations involved are a limitation to their use. But the most important disadvantage of these metrics is that they often provide assessments that do not correlate well with human perception. As IRT says: "Despite efforts to develop objective measuring methods, the results repeatedly fail to reflect quality as perceived by the human eye, which is uninterested in purely logical approaches. Subjective tests are therefore recommended for checking the quality of digital videos" [5].

3.3 PSQA method

In VIPEER we will use a metric without reference called PSQA as Pseudo-Subjective Quality Assessment, proposed in [9]. PSQA is a general approach for measuring perceived quality, based on a learning process allowing to capture the way humans look at the sequences, from the quality viewpoint. We explore the behavior of a panel of humans with which subjective testing sessions are performed, with the goal of establishing a mapping between QoS-oriented metrics and QoE. This mapping is built using a specific learning tool called Random Neural Network. PSQA is a general methodology, which means that it can work under quite different conditions and has been tested on several types of networks and applications, not only for video

but, for instance, also for voice, and not only for one-way flows but also in the case of interactive communications. On the other side of the coin is the fact that PSQA is not universal, and that given a specific type of application and/or network, a measuring module must be built, which may need to solve new research problems. This is one of the goals of this work package in VIPEER. Thus, a new PSQA based module is needed to be designed after considering the associated parameters and after conducting subjective tests.

The idea is to have several distorted samples evaluated subjectively by a panel of human observers. Then the results of this evaluation are used to train a RNN in order to capture the relation between the parameters that cause the distortion and the perceived quality. In general, the distorted sequences used in the test phase are generated for a given context, and therefore a new PSQA module must be generated for every new context. The following text discusses PSQA with the context of UDP or TCP based applications.

3.3.1 UDP

UDP provides unreliable service and video applications using UDP can experience packet losses that can degrade QoE. Moreover, in the case of no-reference QoE measurement, there is no explicit information available at the receiver. Thus, it can be difficult, in some cases, to measure parameters such as packet losses. However, when RTP is used with UDP then the sequence number (SN) field in RTP can be used to detect the missing packets.

In the context of UDP/RTP video streams, the PSQA function in [10], looks at the pattern of video data losses in the stream. Losses at different points in the video stream can have different impact on QoE. The video stream has many GOPs (Group of Pictures) that in turn have different types of frames: I, P and B. I frames are the reference frames, P frames are the encoded frames that are predicted from previous frames and B frames are bi-directional predicted frames. The loss of a frame has a quality impact on all the following frames that are dependent on it for decompression. This error propagation goes till another I frame is received. Hence, the position of the lost frame in a given GOP is one important parameter that affects QoE. We call it loss rank (R).

Apart from network parameters, the video characteristics are also important. For example, when there is a lot of motion in the video, even a small amount of packet loss leads to a lot of pixelisation and thus low QoE. However, in the case where there is low motion level, then some amount of packet loss is tolerable and may lead to unnoticeable pixelisation. In addition to network parameters, the PSQA function in [10] uses motion activity and quantisation parameter (QP) (parameter that controls the amount of lossy compression) to estimate QoE.

Note that the other parameters, like resolution and video bitrate, are either constant in our studies or, like delay and jitter that cause packets to miss their deadline, are converted into packet loss.

3.3.2 TCP

TCP provides reliable service and thus video applications using TCP will not see packet losses at the application layer because TCP will re-transmit the lost packets. However, if due to the delay, caused by the network or by TCP retransmission, the video packets do not arrive at the receiver before their playout deadline then playout will be interrupted and this will lead to degraded QoE.

The project VIPEER will use TCP to transmit video over dCDN. Thus, a PSQA function needs to be built for this context. The dCDN will divide the video data in chunks. In addition, the same video content will have different chunks corresponding to different video quality levels. A chunk will be the basic data unit and separate chunks will be transmitted independently by TCP. In case of congestion, the video application can switch to lower video quality and start downloading the corresponding chunk.

In case a chunk doesn't arrive before the playout deadline then it will lead to a playout interruption. Switching to different video qualities will impact QoE. Moreover, QoE will also depend on the video quality level of the chunk being downloaded.

A PSQA function needs to be built that will map the level of video quality, quality level switching and playout interruption frequency to QoE. The function will not consider video start-up delay as the target is to estimate the runtime QoE. However, the start-up delay can be used as a separate performance metric along with the runtime QoE.

3.4 Comparison with some commercial products

3.4.1 Commercial products for QoS/QoE monitoring

QoSMET is a passive measurement tool developed by VTT (Finland) in order to measure one-way QoS performance from the applications' point of view. QoSMeT is at its best when measuring the performace offered to real-time applications. QoSMeT only requires IP support and typically runs in the same device as the monitored application. What separates QoSMET from other tools is its ability to measure performance offered in one direction.

Acterna is a company offering communications test and management solutions. Some examples of its products are Quality Management System for end-to-end QoS management and PVA-1000 VoIP Network Analysis Suite, which provides analysis of VoIP calls including jitter and packet loss but lacks one way delay.

Telchemy is a company providing tools to monitor and manage the performance of real-time services such as VoIP, IPTV, IP videoconferencing, HD telepresence. Telchemy's products provide real-time visibility into service quality, accurate estimates of user experience, QoS, IPTV QoE, VoIP QoE (MOS scores and R factors), and detailed analysis of the root causes of quality degradation. With their products it is possible to monitor the service quality and get an estimate of the user perception of Quality of Service and get help to realize the reasons for quality degradation. Telchemy's Vqmon/HD tool is designed for integration into IP set-top boxes, ONUs, and residential gateways, non-intrusively monitoring video stream quality

and providing QoE /QoS feedback to service providers. VQmon/HD measures service quality in terms of both Video Transmission Quality (VSTQ) and VQS/MOS scores.Video service quality reports are automatically sent back to a central collection and mediator server using either RTCP XR or SIP, giving service providers real-time feedback on service quality and detailed troubleshooting information for individual video sessions.

Psytechnics is a traditional concurrent of Telchemy. It provides solutions for IP voice, video conferencing and Telepresence performance and service management. The monitoring solution makes it possible to troubleshoot service performance based on users actual call experience. Psytechnics' Experience Manager is a comprehensive solution delivering real-time, objective, call analysis measuring users Quality of Experience (QoE) as well as network Quality of Service (QoS) for every call, detecting and diagnosing factors that impact service quality including; noise, echo, delay and distortion for voice and distortion, blocking and freezing for video. Psytechnics for designed its MOS score specifically for real-time measurement (per-user, per-session basis).

The V-FACTOR QoE Platform by **Symmetricom**, which includes the Q-400 probe, is a complete quality of experience (QoE) solution that enables rapid identification of symptoms affecting IP-based video, voice and data services in real time, from the core network to the customer premises equipment (CPE). In addition, the audio and video are monitored precisely as the user is experiencing it. V-FACTOR offers a differentiated solution that monitors the performance of any streaming media including HDTV/SDTV, IPTV, VOD, SDV and VoIP, providing cost-effective and scalable end-to-end performance visibility and perceptual video quality assurance for service providers and cable operators worldwide.

3.4.2 Comparison of the PSQA method with other solutions

Numerous solutions, including the ones discussed above, have been introduced to measure the quality of the supported services. However, some of the commercial solutions presented in the previous section consider only the QoS measurement. In fact, **QoSMET** addresses only the monitoring of QoS performance metrics at the IP layer which allows the application to be independent from the other layers. This presents, however, many accuracy limitations. The solution proposed in **Acterna** improves this last by addressing some physical layer metrics such as the real-time analysis of the air interface. However, this solution considers only QoS parameters without considering their impact on the supported applications.

In general, as shown in Table 3.1, the QoE measurement solutions can be categorised based on the following criteria:

- Type of reference used: whether they require full (original video), reduced or no reference.
- Mathemetical complexity
- QoE prediction accuracy: whether it is close to the subjective quality scores given by the real users.

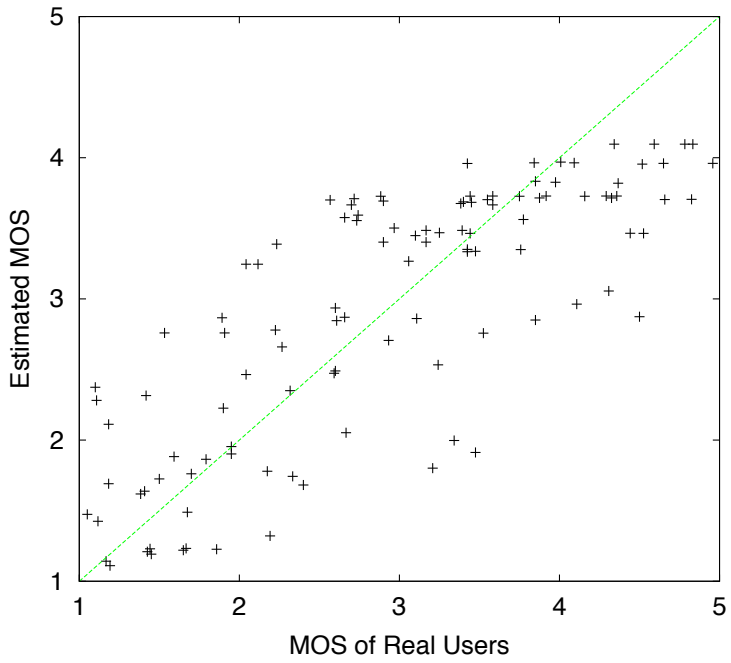

Figure 3.1: Real scores vs Estimated QoE scores with ITU G.1070 opinion model

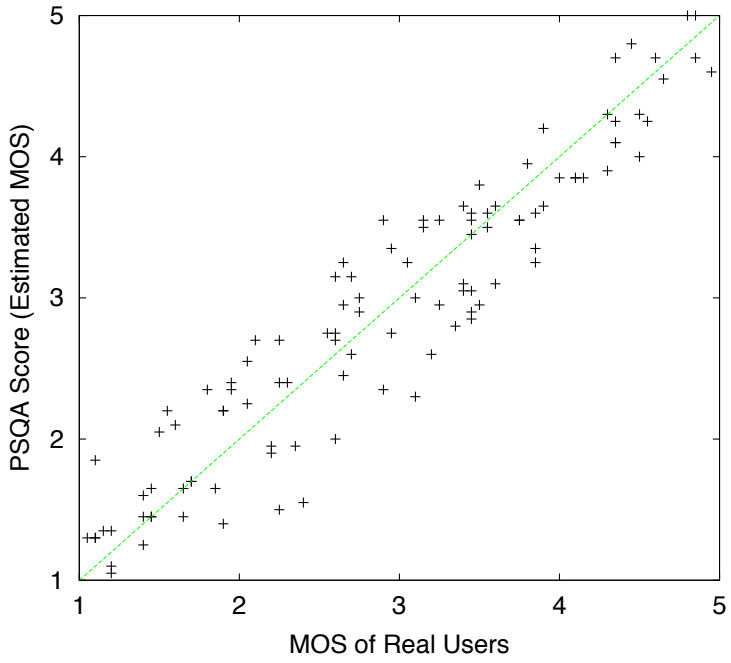

Figure 3.2: Real scores vs Estimated QoE scores with PSQA

In general, there are some QoE measurement solutions that require full reference

or reduced reference to measure QoE and that is the case with PSNR, VQM, NVFM, MPQM, SSIM, etc. In fact there are some methods that have high complexity and take lot of CPU resources such as those that need to process the video signal, for example MPQM. PSQA has the advantage of being no-Reference and of consuming very few CPU resources because of its parametric approach since it doesn't use video signal processing but it uses the specific parameters that have an impact on QoE.

PSQA uses a black box approach that permits it to consider network parameters as well as parameters related to video characteristics. About video characteristics there is a trade-off regarding the complexity. For the moment we have considered some video encoding parameters that are very easy to obtain from the video stream. We are also considering to use some parameters related to video characteristics/encoding for VIPEER because network parameters like packet losses are not very useful when we use TCP.

We can see some examples comparing the accuracy of PSQA as compared to other approaches. For example, Figure 3.1 shows the estimated value of MOS vs. the real value of MOS with ITU G.1070 model [6] which like PSQA is also a parametric, no-reference approach with very light computational complexity. It can be seen that the model is not precise and this is mainly because it does not model the impact of all the important parameters. On the contrary, PSQA is more precise as shown in Figure 3.2. Figure 3.2 shows the scatter plot with estimated MOS vs real MOS obtained from the subjective tests. The scatter plot shows the good accuracy of the estimation by PSQA.

Moreover, it should be noted that comparison of PSQA with commercial approaches is difficult because commercial algorithms are not publicly available.

3.5 Conclusions

The current section introduced a state of the art on QoE measurement and positioned our solution, which is named PSQA, to related work. Besides, some details are given on the PSQA method, which is a pseudo-subjective quality assessment approach as it tries to emulate the human perception to produce the MOS without using any reference. In contrast with the existing approaches PSQA offers an accurate QoE measurement for both UDP and TCP flows[1].

[1]Note that none of the existing approaches offers a solution to measure the QoE of TCP flows.

4 Traffic classification

4.1 Scope of traffic classification

The evolution of the Internet in the last few years has been characterized by dramatic changes to the way users behave, interact and utilize the network. The rapid introduction of new categories of applications such as network games and peer-to-peer, the increasing presence of malicious traffic, and the widespread use of encryption techniques, make the measurement, analysis and classification of Internet traffic a challenging task. The research community and many network operators are responding to these changes by designing and deploying traffic measurement and classification architectures of increasing complexity.

In the framework of VIPEER it is considered to locate some contents close to the final user, for example in the public part of the set-top box. In that case accessing the content will consume some bandwith of the users access link. This might impact the QoS offered to the applications in the customers premises. It is useful to obtain some information about which usage is done of the access bandwith. We identify this as a traffic classification problem.

We think that a fine grained classification of the traffic is maybe not necessary. To be more precise we would be interested into identifying some "categories of applications". The notion of category of application is fuzzy. How we should define the categories depends essentially on how we will use the classification results. In our case we need to identify which QoS is required by the applications in the customers premises. For example if the end user is using a HD visioconference service the dCDN should not use his access bandwith in order to upload/download some contents. To push these considerations to the extreme one could even think that the definition of two categories of applications is maybe sufficient: interactive real-time applications and bulk data transfer. Interactive real-time applications are sensitive to both the delay and, for some of them, to the available bandwith. Whereas bulk data transfers are only sensitive to the available bandwith. The question of the precise definition of VIPEER categories of applications is still open.

Traditional methods for traffic classification are based (i) either on the analysis of port numbers (ii) or on the analysis of the application layer payload. Table 4.1 presents usual port numbers and payloads for a few applications. Port numbers and payload analysis can be used in many cases in order to identify applications. But it is well known that these methods are not any longer fully reliable. Some applications use dynamic port numbers or do some tunneling in order to "hide" themselves behind other applications. Deep Packet Inspection (DPI, that is to say payload

Service	Port	Message de début de connexion
FTP (File Transfer Protocol)	21	220 (vsFTPd 2.0.6) USER anonymous
SSH (Secure Shell)	22	SSH-2.0-OpenSSH-4.7 SSH-2.0-OpenSSH-5.1p1...
SMTP (Simple Mail Transport Protocol)	25	220 smtp.xxxx.xxxx SMTP Ready HELLO client
HTTP (Hypertext Transfer Protocol)	80	HTTP/1.0 200 OK

Table 4.1: Port number and payload for some applications

based filtering) is extremely demanding on high bandwith links and cannot be used if the applications cipher their traffic. Consequently some methods have been designed in order to identify applications without port numbers or payload inspection. These methods are based on a statistical analysis of some traffic descriptors such as packet or flow-level characteristics (size, timestamps). These methods are using data mining methods such as classification methods, time series analysis, estimation theory, decision making tests, etc...

In the past few years there has been a bunch of publications on this topic, mainly from the academic world but also from the industry. Partner Institut Telecom organizes every year in collaboration with the university of Pisa the International Workshop on TRaffic Analysis and Classification (TRAC), co-located with IWCMC. The first edition of the workshop was in Caen (France) in july 2010 and the next edition will be in Istanbul (Turkey) in july 2011: http://netserv.iet.unipi.it/trac2011/.

4.2 Classification methods

In this section we are going to discuss some statistical methods that can be used in order to identify applications or categories of applications. The next section will be devoted to the implementation of these methods over a laboratory testbed deployed by partner Institut Telecom.

The traffic classification methods that we have implemented are making use of two kinds of traffic descriptors:

- either the TCP flags of the first packets of a flow
- or the packet lengths of the first packets of a flow

Typically, for some flows, the first packets are captured. The number of packets that should be captured in order to classify a flow is a parameter that we can set. By experience we know that it is not possible to classify flows reliably on the basis of less than 5 packets since the first packets correspond to some handshake procedure. We also know that it is not necessary to wait for more than 15-20 packets; using more packets would conduct to an additional delay and would not improve the accuracy of the classifier. The way flows are reconstructed (from packet level measurements)

is inspired from the work of the IPFIX working group at the IETF and adressed in Chapter 5.

Once TCP flags or packet lengths of the first N packets of a flow have been gathered a decision is taken by the traffic classifier. The traffic classifier decides in favor of a category of an application or a category of application. It could also decide that it has not accumulated enough information in order to take an accurate decision.

For the moment we have been using supervised classification methods. This means that a "signature" of the applications has been obtained from a previous analysis of some traffic traces with established groundtruth. The signature can be in form of, for example, a probability density function for the first N packets of the flows. By groundtruth we mean that, for learning the signatures of the applications, we have been using labeled datasets. These labeled datasets are traffic dumps in which each flow record is labeled with the application (or protocol) that has generated that flow. The way groundtruth is established is discussed in section 4.5.

We now assume that signatures of the applications have been established in the learning phase. In the detection phase the traffic is then classified by "comparing" each new flow to the set of signatures and by taking a decision in favor of the signature that best "fits" the new flow. Different methods can be used in order to take that decision. For example we can use the so-called GLR (Generalized Likelihood Ratio) test which is a classical decision making algorithm in the case of composed hypotheses or a ML (Maximum Likelihood) test which decides in favor of the signature under which the flow is the most likely. This is discussed in section 4.2.3.

Different datasets are used for establishing the signatures of the applications (learning phase) and for measuring the accuracy of the classifier. Some discussions about the datasets is available in section 4.5.

4.2.1 Traffic descriptors and models

Markov model of TCP flags

This method is based on a paper that partners France Telecom and Institut Telecom have published at MineNet, a Sigmetrics 2007 workshop dedicated to traffic data mining [2]. The idea developed in this paper is that the applications might have a measurable impact on the dynamics of the connection at the transport level. The method is usable for TCP traffic only but the composition of traffic in terms of transport layer protocol is such that TCP represents a majority of flows.

The method is based on a Markov model for the succession of packets in the TCP connection. In order to take into account possible lost or out-of-delay packets as well as packet reordering the sequence number of the packets is used to sort the packets of the TCP connection. This is a limitation of this method since it forces the measurement point to maintain a state for each TCP connection under monitoring which limits the scalability of the method. Hence we think that this method is usable in monitoring points which are relatively "high" in the network or would

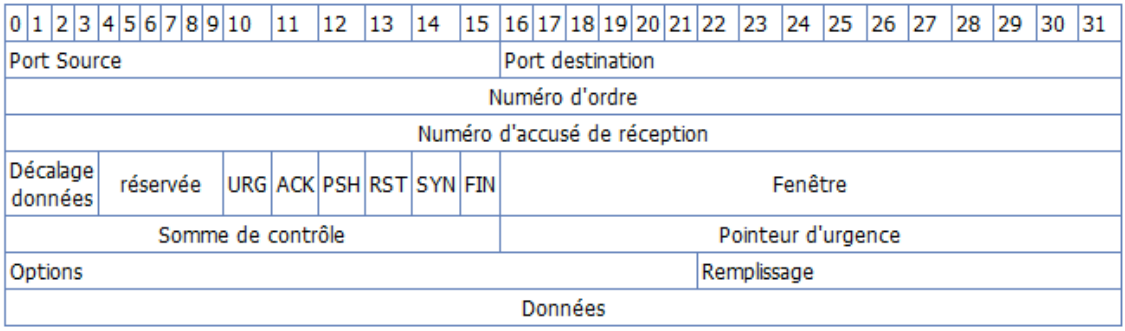

Figure 4.1: Structure of a TCP segment

require severe subsampling of flows in the core.

The traffic descriptors that are used are the so-called TCP flags. There are 6 flags: URG, ACK, PSH, RST, SYN, FIN. URG is used for URGent data. ACK means that the packet is an ACKnowledgment. PSH means that the data should be delivered immediately (PuSH). RST corresponds to an anormal interruption of the connection (ReSeT). SYN asks for the SYNchronization or establishment of the connection. FIN requests the end of the connection. Figure 4.1 is a reminder of the structure of a TCP segment.

In our method each flag is associated to a power of 2 as follows: SYN=1, ACK=2, PSH=4, RST=8, URG=16, FIN=32. Any combination of the 6 TCP flags is associated to an integer in between 1 and $64 = 2^6$. For example a SYN-ACK packet is associated to the integer $3 = 1 + 2$ (SYN+ACK).

A Markov model is assumed for the succession of flags of the connection. This model is characterized by transition probabilities P_{ij} which represent the probability that the flag of the next packet will be j given that the flag of the previous packet in the TCP connection is i. These transition probabilities can be displayed in the form of a state transition diagram. An example of state transition diagram is given on Figure 4.2.1; it corresponds to the signature of uTorrent for outgoing traffic.

Gaussian or semi-parametric models for packet lengths

In another method the used traffic descriptors are the length (in number of bytes) of the first packets in a flow. In order to model the distribution of the packet lengths we have used two different models, a Gaussian model and a semi-parametric Gaussian model. In both cases, to simplify treatments, we have assumed that the packet lengths of the first packets in a flow were independent.

- In the Gaussian model the distribution of the packet length is assumed to be a Gaussian with mean μ and variance σ^2, with probability density function

$$K_{\mu,\sigma}(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2\sigma^2}(x-\mu)^2) \tag{4.1}$$

  The mean μ_i and variance σ_i^2 is different for each packet depending on its position i in the flow.

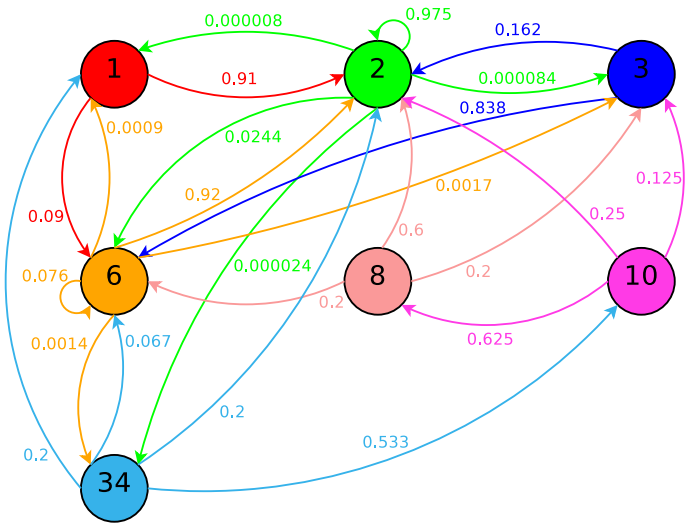

Figure 4.2: uTorrent signature (outgoing trafic)

- In the semi-parametric Gaussian model the distribution of the packet length is assumed to be a mixture of T Gaussians with the same variance σ^2 and different means x_i. The probability density function has the following expression:

$$f(x) = \frac{1}{T}\sum_{t=1}^{T} K((x - x_i)/\sigma) \tag{4.2}$$

where K is the probability density function of a Gaussian distribution with mean 0 and standard deviation 1:

$$K(x) = \frac{1}{\sqrt{2\pi}} \exp(-x^2/2) \tag{4.3}$$

The semi-parametric Gaussian distribution is illustrated on Figure 4.2.1.

4.2.2 Signatures learning

As we have already explained we are using a supervised learning approach. This means that a labeled dataset is used in order to build the signatures of the applications. In this section we are going to explain briefly how the signatures are obtained.

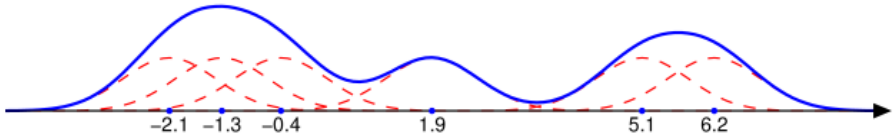

Figure 4.3: Semi-parametric probability density function

We assume that in the learning dataset there are enough flows for each application (or category of applications) in order to compute reliable statistics. In the case of applications which generate few flows but very large flows (FTP for example) the Markov model of TCP flags can still be trained. Indeed in that case the number of packets hence the number of transitions beween the different flag combinations is large in the training dataset (although the number of flows might be small). This is not true for the Gaussian or semi-parametric Gaussian model of packet lengths since in that model only the first packets of each flow are useful in order to train the classifier.

Markov model for TCP flags

We assume that the Markov model is homogeneous, that is to say that the transition probabilities P_{ij} do not depend on the position of the packet in the TCP connection. We remind the reader that P_{ij} represent the probability that the TCP flag value of the next packet will be j given that the TCP flag value of the current packet is i. The transition probabilities P_{ij} are obtained by simply counting the number of transitions from TCP flag combination i to TCP flag combination j in the learning dataset, considering all the flows which label corresponds to the considered application.

$$P_{ij} = \text{Nb. of Transitions from Flag i to Flag j / Nb. of Packets with Flag i}$$

Gaussian model for packet lengths

If we consider the Gaussian model the signatures are obtained by computing, for each application and for each position of the packet in the flow, the mean μ_i and variance σ_i^2 of the packet sizes from the learning dataset. Please note that the mean μ_i and variance σ_i^2 is different for each position i of the packet in the flow (and obviously different for each category of applications). We used the so-called empirical estimators of the mean and variance:

$$\hat{\mu} = \frac{1}{T}\sum_{t=1}^{T} x_t \qquad \hat{\sigma}^2 = \frac{1}{T}\sum_{t=1}^{T}(x_t - \hat{\mu})^2 \tag{4.4}$$

where $(x_t; t = 1, \ldots, T)$ are the samples of the learning dataset from which the empirical estimators are computed.

Semi-parametric Gaussian model for packet lengths

The probability density function is the convolution of the samples with a Gaussian kernel:

$$f(x) = \frac{1}{T}\sum_{t=1}^{T} K((x - x_t)/\sigma) \tag{4.5}$$

where $K(x) = \frac{1}{\sqrt{2\pi}} \exp(-x^2/2)$ is the probability density function of the standard normal distribution. In this equation the samples $(x_t; t = 1, \ldots, T)$ are the elements of the learning dataset which are used to build the probability density function. Typically these samples represent all the packets in position i over all flows which label corresponds to the considered category of application. The standard deviation σ of the Gaussian kernel can be varied in order to smoothen the density. A large value of σ would end up returning a very smooth probability density function with a large overlapping of the different Gaussian components. A small value of σ would on the contrary return a probability density function with many "peaks" located around positions x_t.

4.2.3 Decision making

After the analysis of the learning dataset a set of signatures is available and characterizes the different categories of applications. Note that the analysis of the learning dataset is performed once and for all. This analysis is performed off line.

Once the signatures have been obtained the system uses these signatures in order to classify new flows. This classification can be performed online. For example, in our system, the classification of flows is based on the first (10 to 15) packets of a flow. As soon as a sufficient number of packets has been captured a decision can be made in order to classify that flow. We recognize this problem as a decision making problem, in which the system has to decide in favor of one hypothis. Each hypothesis corresponds to a given category of applications and is characterized by a stochastic model, either a Markov chain in case the traffic descriptors are TCP flags, or a Gaussian or semi-parametric Gaussian model in case the traffic descriptors are packet lengths.

The decision is based on the likelihood (or log-likelihood) value of the observations $(x_1, x_2, \ldots, x_N)$. N is the number of packets that are taken into account in order to take a decision. The "observations" $x_t, t = 1, \ldots, N$ represent either the TCP flags values or the packet lengths.

We have been using a non sequential decision making approach: the number of observation N is assumed to be fixed; typical values of N are in between 10 and 20. On the contrary, in a non-sequential decision making approach, the number of observations would not be fixed in advance. The classifier would wait until enough information had been accumulated in favor of one hypothesis to take a decision.

The likelihood $L(x_1, \ldots, x_T)$ and log-likelihood $LL(x_1, \ldots, x_T)$ have the following expressions:

- Markov model of TCP flags

$$\begin{aligned} L(x_1, \ldots, x_N) &= \prod_{t=1}^{N-1} P_{x_t x_{t+1}} \\ LL(x_1, \ldots, x_N) &= \sum_{t=1}^{N-1} \log P_{x_t x_{t+1}} = \sum_{i,j} \log P_{ij} n_{ij} \end{aligned} \tag{4.6}$$

  where n_{ij} is the number of successive observations such that $x_t = i$ and $x_{t+1} = j$ that is to say $n_{ij} = \sum_{t=1}^{T-1} \mathbb{1}_{x_t=i} \mathbb{1}_{x_{t+1}=j}$.

- Gaussian or semi-parametric Gaussian model of packet lengths

$$\begin{aligned} L(x_1, \ldots, x_N) &= \prod_{t=1}^{N} f_t(x_t) \\ LL(x_1, \ldots, x_N) &= \sum_{t=1}^{N} \log f_t(x_t) \end{aligned} \tag{4.7}$$

  $f_t(\bullet)$ is the probability density function (pdf) of packet number t in a flow. The form of this pdf is given either by Equation 4.1 in the case of a Gaussian model or by Equation 4.2 in the case of the semi-parametric Gaussian model.

A possible decision making algorithm is the GLR (Generalized Likelihood Ratio) test. The GLR test is a decision making test to choose between two hypotheses, one of them being composite. A GLR test can be used in the case when one wants to detect one particular application among many other applications. The assumption $H_1 = \{\theta = \theta_1\}$ corresponds to the application that one wants to detect. Assumption $H_2 = \{\theta \in \Theta - \{\theta_1\}\}$ is composite and corresponds to all the other applications. The GLR test decides in favor of H_1 in case the GLR statistics Λ is greater than a given threshold h, where the GLR statistics is computed as:

$$\Lambda(x_1, \ldots, x_N) = \frac{L(x_1, \ldots, x_N; \theta = \theta_1)}{\max_{\theta \neq \theta_1} L(x_1, \ldots, x_N; \theta)} \tag{4.8}$$

The value of the threshold h depends on the value of the False Alarm Rate (FAR) α that one is ready to accept. The lower the threshold h the greater the power β of the test (percentage of flows under H_1 which are truly detected as H_1) but also the greater the FAR α (percentage of flows under H_2 which are detected as H_1).

Another very simple classifier is to decide in favor of the model under which the sequence of observations is the most likely. That is to say that the classifier attempts to maximize the likelihood (Equation 4.6 or 4.7) by picking out one of models corresponding to a given category of applications.

4.3 Applications clustering

During testings some classification errors became usual. Searching for an explanation for these events we discovered a very close relationship between some applications. In some cases this was very logical, since the basic functioning of both applications was the same. This is why we thing that without payload analysis it might be possible to identify reliably some categories of applications but maybe not the applications themselves in all the cases.

That is how the term of clustering got in the way. Clustering is a term usually used when performing spacial analysis to refer to a group of items placed very close, and apart from other items, or other clusters as well. So the idea of 'Applications Clusters' refers to the different groups of applications which have a similar functioning.

An example could be the Flash and the FTP applications. Traffic generated by Flash players such as the ones embedded in Internet sites as YouTube, basically consists of a single connection to download the video. This is not real-time streaming as P2P Streaming. It is buffered video so it can be seen as a file download. FTP is also an application type that consists of a single long connection to download a file. As it can be seen, the applications are very different but their behavior is very similar.

4.3.1 Küllback-Leibler divergence

In order to measure the similarity between the signatures of different applications we can use the notion of Küllback-Leibler divergence. Küllback-Leibler divergence is an information-theoretic measure of dissimilarity between random variables. The Küllback-Leibler divergence between two probability distributions with probability density functions (pdf) $p(x)$ and $q(x)$ is defined as:

$$D_{KL}(p(x), q(x)) = \int_x p(x) log\Big(\frac{p(x)}{q(x)}\Big). \tag{4.9}$$

This quantity is always positive and it is equal to zero only if $p = q$.

The Küllback-Leibler divergence is usually defined in introductory courses to information theory as a measure of dissimilarity between two probability distributions. But it can be generalized to the case of stochastic processes. For example, the Küllback-Leibler divergence between two Markov chain models is equal to:

$$D_{KL}((\pi_1, P_1), (\pi_2, P_2)) = \sum_{i,j} \pi_1(i) P_1(i, j) \log \frac{P_1(i, j)}{P_2(i, j)} \tag{4.10}$$

where (π_1, P_1) are respectively the stationary distribution and transition matrix of the first Markov chain, and (π_2, P_2) are those of the second Markov chain.

4.3.2 Neighbor joining

Neighbor-joining is a tree-based clustering method. The method reconstructs the relation between items by iteratively joining pairs of nodes until a single node remains, giving birth to a tree. The criterion for which pair of nodes to merge is based on both the distance between the pair and the average distance to the rest of the nodes.

This method requires as an input a symetric distance matrix between the different items that one wants to cluster. The Küllback-Leibler divergence is not a distance; the symetry property of a distance is indeed not satisfied. In order to

have a symetric measure of dissimilarity we have used the following measure, the transformation being introduced in order to symetrize the divergence:

$$D_{KL}(Model1, Model2) + D_{KL}(Model2, Model1) \tag{4.11}$$

To process the matrices, a freeware application called T-REX was used, in particular the web-version. The program takes the matrix of dissimilarity values as text with a specific format and exits the graph of the formed tree, an example of output can bee seen on Figure 4.4. The software also implements other clustering algorithms.

4.4 Traffic Classification testbed and the Nicofix software

4.4.1 Architecture of the Institut Telecom testbed

In order to develop and test in our own laboratory some methods for traffic classification we are using a local measurement testbed that had initially been settled down in the framework of the OSCAR project funded by the ANR. This testbed is displayed on Figure 4.5. It includes:

- a measurement station equipped with a DAG card and a GPS synchronisation toolkit
- a few client stations running different Operating Systems and generating traffic of different applications
- a bridge between the measurement station and the Internet access (main router of the faculty)

The DAG card is hosted by the measurement station and is installed in between the client stations and the Internet access. The DAG card is replicating without any modification all the traffic received on one of its ports onto another port. A copy of a parametrable number of bytes of each packet is sent to the host measurement station through the PCI bus. Accurate timestamping of the measurements is obtained thanks to the GPS synchronisation toolkit. The bridge has been inserted between the DAG card and the access to the Internet in order to introduce some perturbations such as delay, jitter or losses and observe their impact on the stability of the classifier.

4.4.2 The Nicofix software

Partner Institut Telecom has developed Nicofix, a trafic classification software, based on statistical methods. This software has been developed in the framework of some projects with Telecom Bretagne students and some internships of students in our laboratory. The software is still under development.

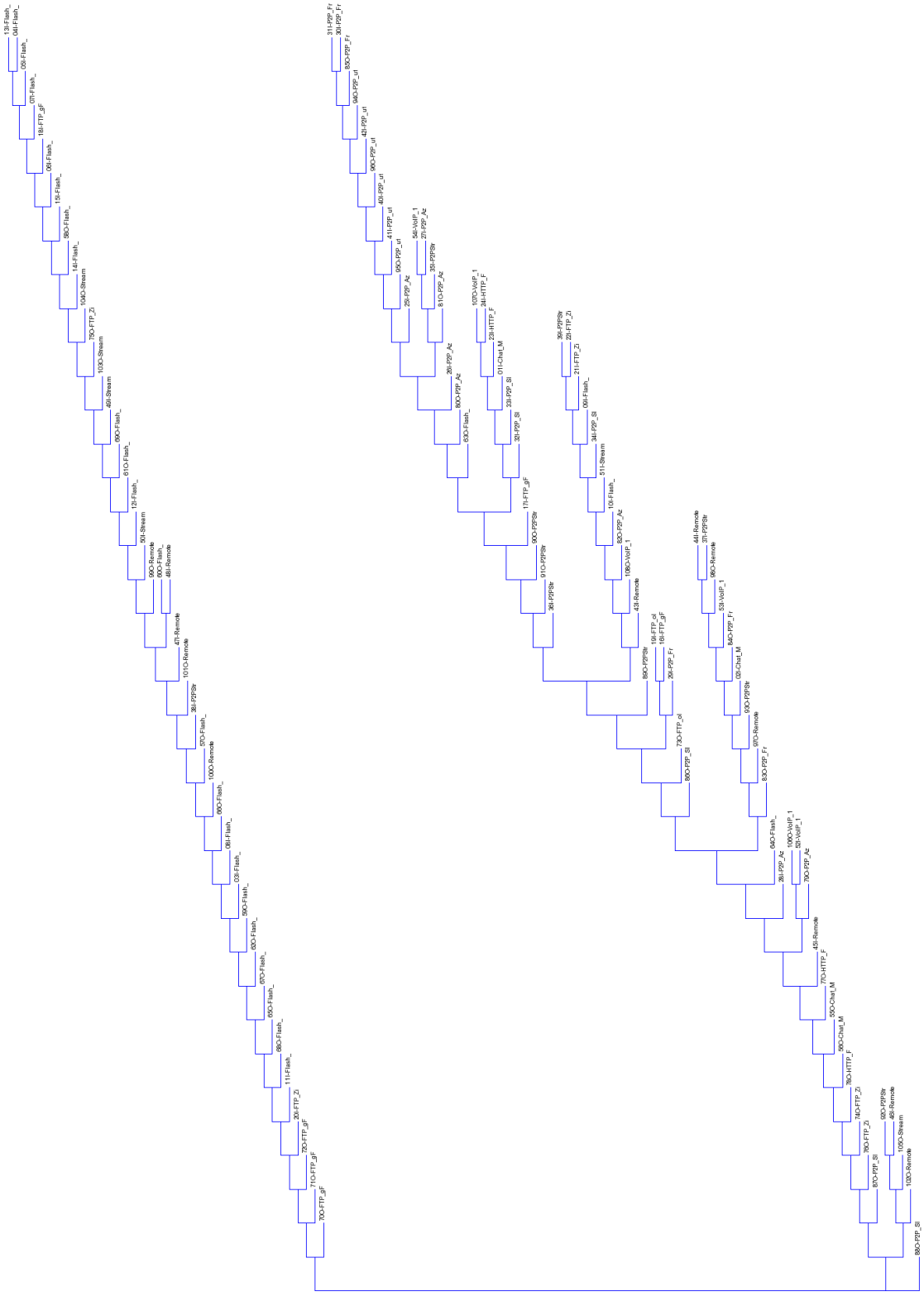

Figure 4.4: Neighbor Joining Analysis for Markovian signatures

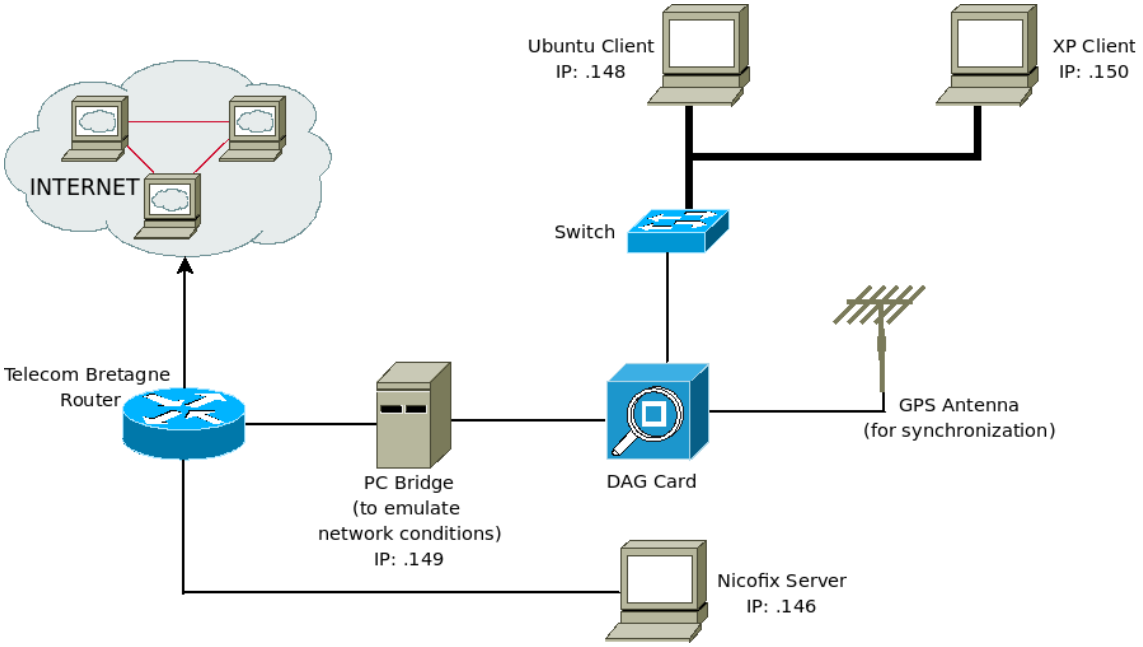

Figure 4.5: Metrology platform @ Telecom Bretagne

4.4.3 The three modes of Nicofix

The Nicofix software has been designed to analyze the network traffic using either a Markov Chain model for TCP flags or a Gaussian or semi-parametric Gaussian model for packet lengths. Nicofix offers different functionalities which can be divided into three modes:

- the learning mode,
- the analysis mode,
- the detection mode.

For each of these modes, some functions have been developed into separate modules. The three modes of the Nicofix software are schematically displayed on Figure 4.6 for the particular case of a detection based on TCP flags values.

In learning mode, the program reads the traffic from a source (DAG card, *.isd files or a directory depending of the running parameters) and builds a signature. The signature is a transition matrix of TCP flags between consecutive TCP fragments, or a Gaussian or semi-parametric Gaussian model for the packet lengths.

These two steps are separated into two modules:

- Read flow Module : the objective of this module is to build flows from the readen traffic. Figure 4.7 illustrates this process (the format of a flow-list is displayed on Figure 4.8). This module is also used in the detection mode. A flow is detected as finished only if no new packet has been detected after a given timeout.

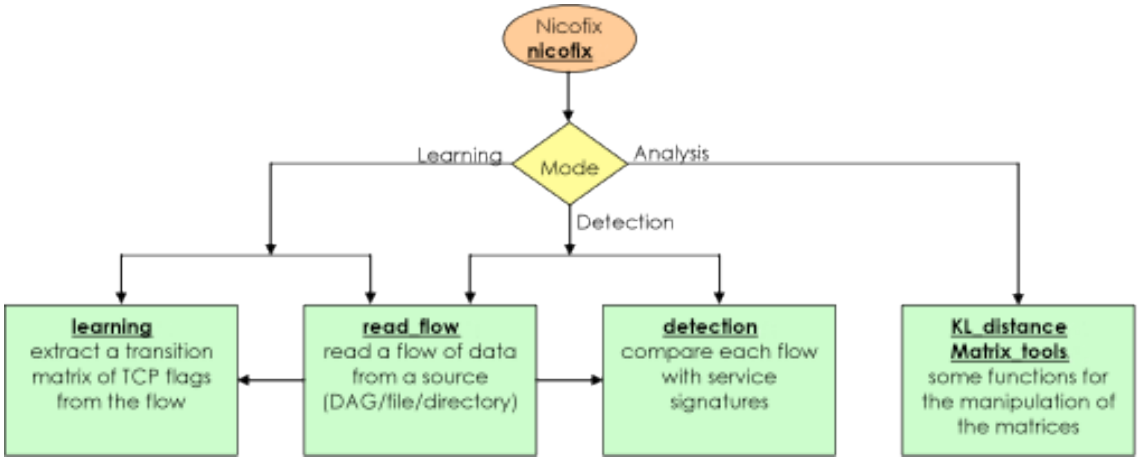

Figure 4.6: The 3 modes of the Nicofix program

- Learning Module : this module receives the flows from the other module and builds the matrix containing the number of transitions of the TCP flags between consecutive packets (in the case of a classification based on TCP flags) or computes some statistics about packet lengths (in the case of a classification based on packet lengths).

In order to have better performances, we have decided to run these two modules in separate processes which communicate through a pipe as it is displayed on Figure 4.9.

In the future versions of Nicofix, these modules will be able to run on separate machines and to communicate via messages through the network. One can imagine a machine running the flow construction module and sending it to a collector machine that uses it for the detection. This architecture with a centralized point that take the decisions and many monitoring points that collect flow level records and export them to the collector is inspired by the work of the IPFIX group at IETF and discussed in section 5.4.

An important features about our architecture is its modularity. It is constructed independently of the method used for learning or for the detection. One can use different methods and add its own methods by writing a few lines of codes while profiting of the general services provided by the architecture.

4.4.4 User Interface and Development Environment

The graphical user interface is not mandatory to use the main functions of the Nicofix program, but it is useful for a non expert user. It provides a friendly and intuitive interface to analyze the signatures, set the program parameters, run the detection mode, instead of typing long command lines.

An IDE (Integrated Development Environment) called “Anjuta” has been chosen. It provides many tools for C projects and it includes a “Glade” environment that is really useful for the development of a graphical interface.

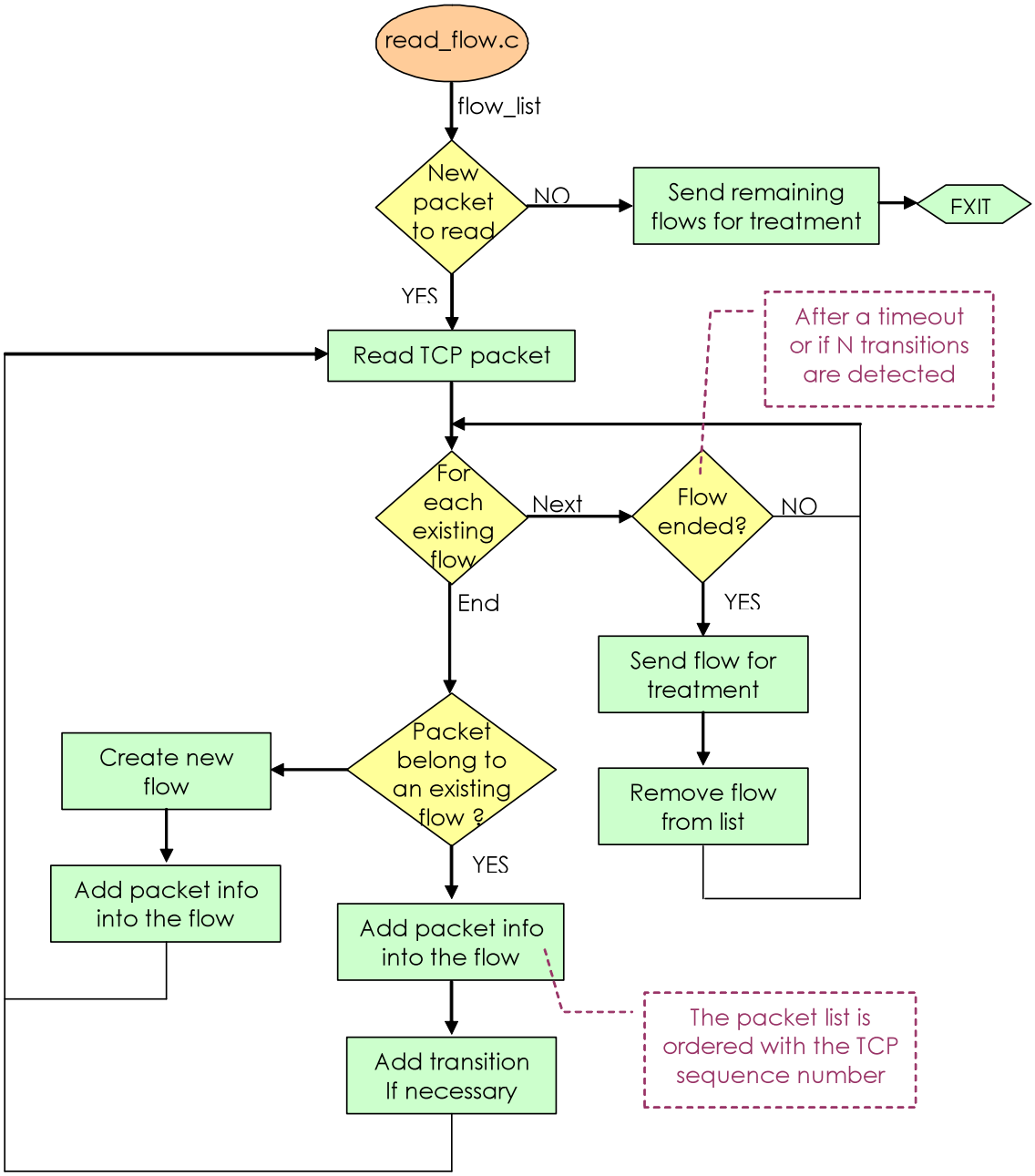

Figure 4.7: Learning Mode

The program uses 3 libraries in addition to the standard C library:

- GSL (GNU Scientific Library) for the matrices manipulations,
- GTK 2.0 to build the graphical user interface,
- Glade for simplifying the development of the GUI using a XML file which describes the interface.

Some snapshots of the graphical interface are displayed on Figures 4.10, 4.11, 4.12 and 4.13.

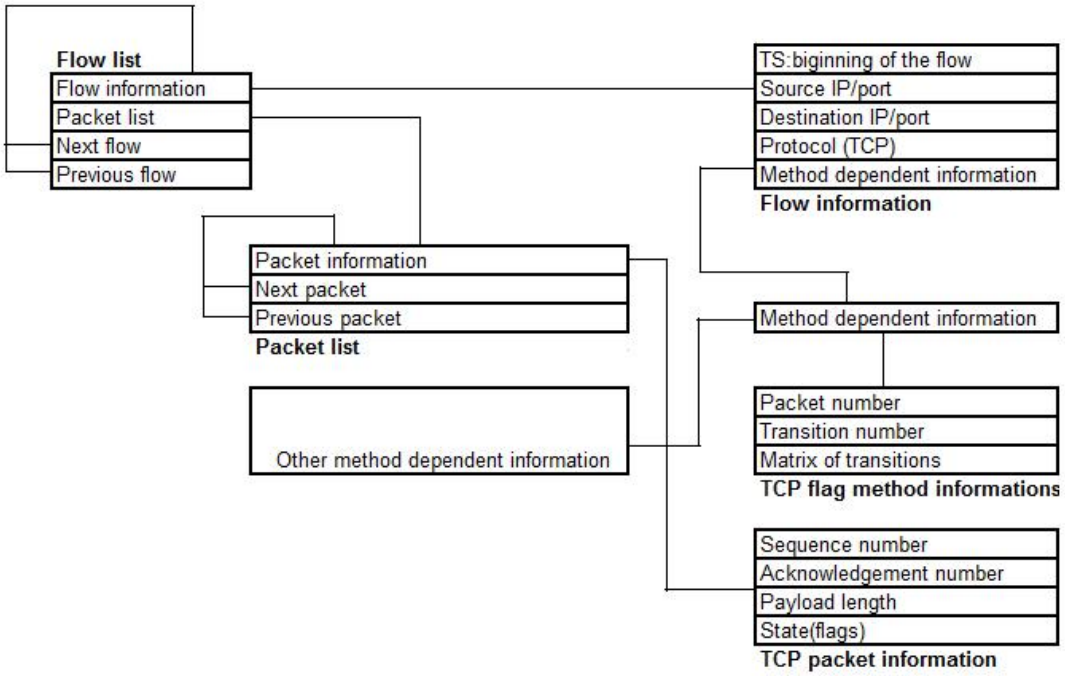

Figure 4.8: Flow list structure

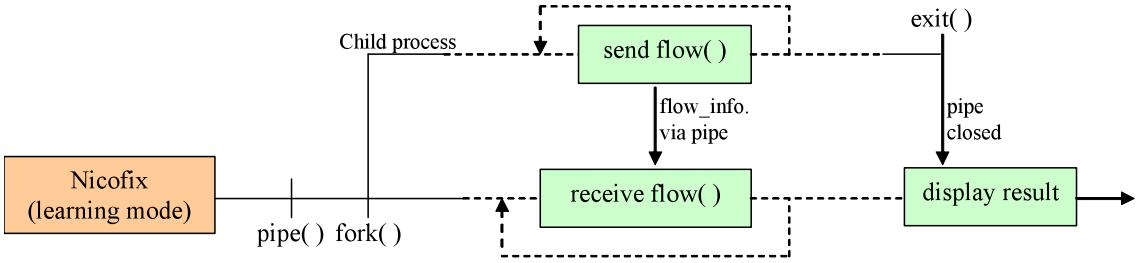

Figure 4.9: Communication between processes

4.5 Datasets and ground truth validation

4.5.1 Groundtruth validation

The question of groundtruth validation is difficult [4]. By groundtruth we mean a reliable method that can identify applications and/or protocols. The idea is to link groundtruth meta-data to Internet traffic traces. The results of statistical classification can then be confronted with that "groundtruth" in order to evaluate the classification accuracy. There does not exist a real consensus on how this groundtruth should be established but there are some main methods.

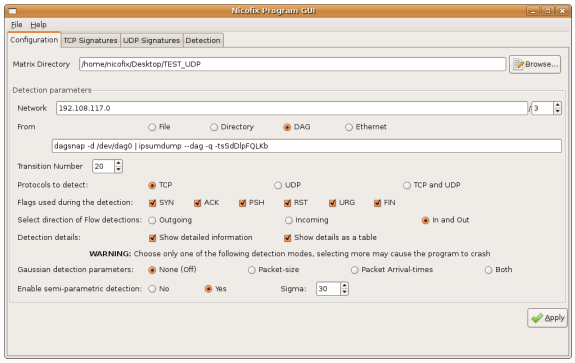

Figure 4.10: Configuration panel

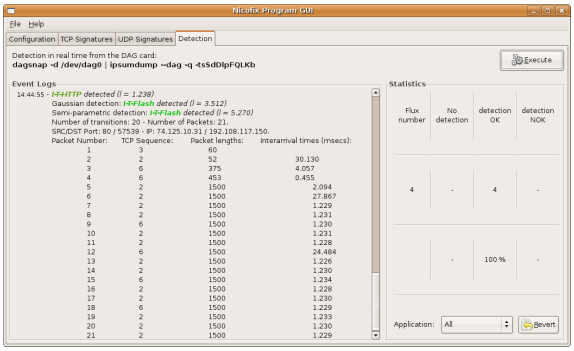

Figure 4.11: Detection panel

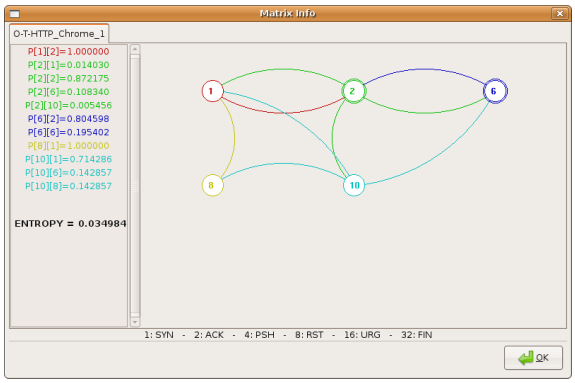

Figure 4.12: A Markovian Signature

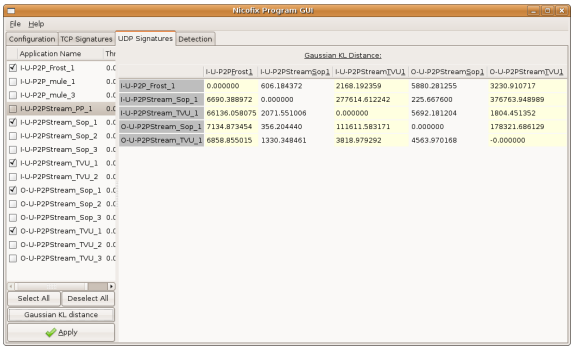

Figure 4.13: KL divergence for Gaussian signatures

A manual generation of traffic could be considered. This is a naive approach since it cannot prevent daemons or other background applications generating their own traffic. The dataset is consequently polluted by other protocols, a typical example being the case of DNS requests. Wireshark (previously Ethereal) is useful in order to observe the traffic and debog applications.

Analysis of port numbers is not always reliable since some applications negociate dynamically their port numbers or hide themselves behind other applications using some classical port numbers. Classification methods which are based on port numbers do not classify correctly more than 30% of the flows. In particular P2P applications cannot be identified on the basis of port numbers.

We think that Deep Packet Inspection (DPI) is still today the primary mechanism used to derive the ground truth about network traffic at the protocol level. It is based on regular expression matching (filters) on the application layer data to determine what protocols are being used. This is for example how L7-filter, a packet classifier for Linux, works.

L7-filter was initially designed within the Linux kernel's QoS system. It is now implemented in Netfilter, the packet filtering framework inside the Linux 2.4.x and 2.6.x kernel series. Filters are publicly available for many protocols. One can also coin new regular expressions for other protocols. DPI (payload inspection) cannot be used in the case of applications that use encryption or obfuscation mechanisms to hide their traffic, such as Skype and P2P file sharing applications. Even in the

case of clear traffic DPI encounters some protocol classification errors.

GT (Grountruth) is a method that has been designed by the university of Brescia (together with Politecnico di Torino and CAIDA) [7]. GT has also been released as an open source software. GT gathers groundtruth at the application level by probing the host's kernel. The principle of GT is to run a daemon on the monitored host and regularly probe the kernel in order to track changes in active sockets. In parallel GT records flows on a gateway. An accurate synchronization between flow records and sockets tracking permits to correlate both and to associate the flows with the applications that own the sockets. GT also includes DPI (protocol identification by payload inspection) in order to tag the traffic with both protocol information and application identification.

In our search for groundtruth we have used L7-filters and GT in order to identify some protocols and applications.

4.5.2 Datasets

There is still a need for best practice and clearly articulated standards in Internet measurements [8]. There is also a need for publicly available datasets with meta-data. Those meta-data make it possible for researchers to confront the results of their analysis to some groundtruth. This could be the groundtruth about security related events (ongoing attacks) or groundtruth about the applications that generated flows and the protocols that are being used. Up to our knowledge there are very few publicly available datasets with groundtruth for the traffic classification problem. This stems from privacy concerns about end users of Internet Service Providers and also from the competition between operators that pushes them to reveal as few information as possible about their traffic.

The authors of GT have released an anonymized Internet trace. This trace corresponds to a measurement campaign that has been performed at the university of Brescia. We have used this dataset in order to evaluate the performance of our classifiers.

Partner France Telecom has also provided a proprietary dataset with identification of some categories of applications. This dataset corresponds to a measurement campaign of 41 minutes over a 1 Gbits/sec. ADSL link. The classification per category of application is based on some Deep Packet Inspection. The identified categories are: Web, P2P, Download (for example FTP), News, Mail, Database, Others (ICMP, DNS, etc.), Control (SSH, Telnet, etc.), Games, Streaming, and Chat.

It is worth noting that the exact definition of classes change from one dataset to another and that this definition is relatively fuzzy. For example does Flash traffic (such as Youtube) correspond to Web or to Download? This depends mainly on the method that has been used in order to identify the category of application. It consequently makes it difficult to check the stability of a classifier by, for example, training the classifier on one dataset and testing its accuracy on another.

We also proceeded to some measurements on our monitoring platform at Institut Telecom. In that case the groundtruth was established by using L7-filters and the GT tool.

4.6 Conclusion

In this chapter we have described the current development of a tool that has been implemented for on-line traffic classification on a measurement testbed settled down by Institut Telecom.

5 Architecture of the VIPEER monitoring infrastructure

5.1 Scope of the VIPEER monitoring testbed

Our goal is to integrate several functions in the measurement platform:

- real-time determination of the network parameters such as the available bandwidth, the packet loss rate, the jitter and the delay by actively probing the access link,
- determination of the activity of the end-user in real-time that is to say the composition of traffic generated by this end-user in terms of applications,
- automatic estimation of the Quality of Experience that can be delivered to the different end-user applications, taking into account the network state in real time.

We shall deploy a measurement testbed on which the QoE evaluation tool and the traffic classifier specified in Tasks 2.2 and 2.3 will be integrated and tested on live traffic generated in our own laboratory. We first of all will have to settle down this measurement testbed; and then the automatic methods for evaluating the QoE and for service recognition will be integrated in the testbed.

This measurement platform will include computers with dedicated hardware and software (as DAG cards of the Endace society) for capturing the traffic of specific links. A precise time stamping will be obtained by synchronizing the machines clocks by mean of a GPS equipment. A bridge will be added between the measurement station and the access to one of the main routers in our institute; we will emulate the properties of wide area networks (variable delay, loss, duplication, and re-ordering...) by running network emulation tools (as netem for instance) on the bridge station. We will also include in the testbed a router or a computer with several network cards in order to establish IPSec tunnels and consider the case of traffic ciphered at the network level in our studies.

To emulate the properties of wide area of users, different client workstations will be installed. On these workstations we will run various applications such as Web browsing, file transfer, torrents, streaming audio, streaming video, mail, P2P video streaming, etc... Different audio and video players will be installed on the client workstations. We will also install different versions of servers in the measurement testbed.

This testbed has already been deployed by partners Institut Telecom (Brest) and Orange Labs (Lannion). These partners profit from the expertise that they have acquired in the framework of the OSCAR project (funded by the ANR) to which both of them have participated. A description of the platform at Telecom Bretagne has already been provided in section 4.4.1.

In the VIPEER context, one of the goals is to recognize applications or categories of applications "on the fly" by analyzing the traffic at the packet/flow level. Another goal is to automatically evaluate the QoE as a function of network conditions and multimedia parameters. For traffic classification we need passive measurements that is to say that we are going to passively listen to the traffic in different monitoring points. For netwok level QoS parameters (losses, delay, etc...) monitoring we will most likely use active probing methods.

The proposed passive measurement platform has to satisfy some constraints in order to provide relevant and reliable information about the traffic. The first constraint is to be non intrusive i.e. not to perturbate the network (no introduction of losses or delay for example). The second one concerns the ability of the platform to collect traces which are (i) complete (the information needed is collected for every packet that transit on the link) and (ii) with accurate timestamping.

A very strong and difficult requirement when one performs flow level monitoring is the scalability of the solution. Depending on the bandwith at monitoring point not all flows can be monitored. What is important for the traffic classifier module is that we are sure not to miss any packet of the monitored flows. But a subsampling strategy at the flow level can be considered. By flow level subsampling we mean that only 1 flow out of N is monitored. Ideally we would like to be sure that, for that particular monitored flow, none of the first (10 to 20) packets of the flow are missed. Indeed the classification is normally based on the first packets of the flow.

5.2 Choosing good locations for monitoring the traffic

Traffic can be probed more or less high in the network that is to say close to the end user or close to the content provider or somewhere in the network of the NSP (Network Service Provider). In case the traffic is monitored close to the end user there is less traffic to look at at each probe, but plenty of probes (monitoring points). In case the traffic is monitored close to the content provider there are less probes but a larger traffic to look at at each probe. This choice depends on the expected usage of the monitoring activity, and on the CDN topology of the Vipeer solution.

Intuitively, monitoring traffic can help CDN strategy to decide where to upload/download chunks of contents. For example if the observed traffic corresponds to non-interactive application, the CDN can decide to use the link for transferring its content. At the opposite, if the observed traffic corresponds to real-time applications, the CDN will then avoid using this link and choose another one if possible. The goal is to help the CDN to evaluate how the CDN generated traffic affects end-users playing with their favorite network applications if it decides to use one link instead of another for transferring its contents.

The Vipeer topology is not yet defined. However, one can predict that there

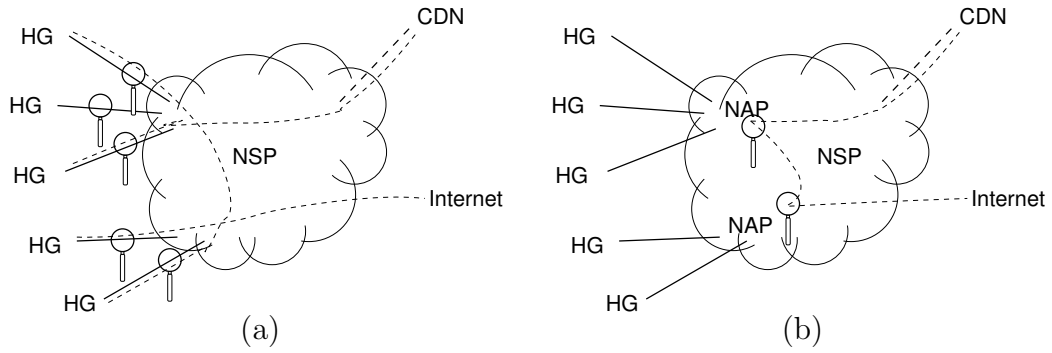

Figure 5.1: Location of probes on two possible CDN topologies

will be caches of chunks of contents at different locations in the network. Choosing good locations for CDN's caches is also an open question at this time of the study. A user playing a video is informed by the CDN of which of these caches provide chunks of the wanted video. Moreover a CDN may decide to push popular chunks by anticipation. Intuitively, traffic probes should be located on links used by these caches.

Figure 5.1 illustrates two possible architectures (over several). On figure 5.1a caches are embedded onto *home-gateways* (e.g. the set-top-box of the end-user). On this proposal, probes (depicted by magnifying glasses) are located near NSP's network access points (NAP). On figure 5.1b caches are located somewhere into NSP's network. The topology is almost the same except that it is *one level higher*. Probes are located between CDN's entities.

5.3 DAG card

To deal with our needs concerning passive measurements (collected traces are complete and with an accurate timestamping), one solution is to use the DAG cards developed by the university of Waikato in New Zealand and now marketed, maintained and improved by the ENDACE company.

The working principle of a DAG card is depicted on Figure 5.3. The first advantage of this kind of card is that they are installed in derivation to the link to be treated. So, in case of a fiber network, one just has to put a splitter which separates the optical power: 80% for the original fiber (normal way) and 20% for the DAG probe. Thus, the traffic is not perturbed at all, no delay is introduced at the splitter level and the traffic still has the same characteristics. As a result, the measurement system is totally transparent.

A DAG card is a dedicated card which realizes, in real time, the extraction of the header of all packets that transit on the link. The header size is specified during the card configuration for the capture. In our case, we want to capture IP and TCP headers only. An advantage of doing so is that we are compliant with the personal data protection law: no personal data are captured.

Figure 5.2: DAG card

Finally, for each captured packet, the card adds a timestamp encoded in 64 bits to the captured header. All data collected are then stored on a disk. All these elements (real time dedicated card, high-capacity system bus, large memory and hard disks of big capacity) are the required elements to guarantee a well-dimensioned system able to capture a trace made up of all packets that transit on the measured link.

In order to timestamp all captured packets and store this timestamp with the packet header a GPS reference is used. For this, the card is directly connected to a GPS antenna through a synchronization toolkit. And the host station, on which the DAG card is installed, has a clock which is resynchronized every second with a GPS signal that transports the universal time (the time from reference atomic clocks). The drift clock is almost nonexistent, allowing highly accurate measurement times. Furthermore, all the probes are synchronized to the universal time and thus between each other.

The DAG card produces .erf files as records of the data capture; these files can be filtered using IPSUMDUMP to extract only the packets we are interested in, and finally converted to tcpdump (.pcap files) or to ASCII files. A library has been developed in order to extract from these files the information that are considered as necessary in order to recognize Internet applications. The format of a flow has been displayed on Figure 4.8.

Once collected, the information can be processed in two different ways:

- Decentralized analysis: in a decentralized analysis, the information is analyzed in the monitoring point itself. The first drawback is that doing so one only has a partial view of the network (local view at the monitoring point). Another drawback is that for maintenance reasons one has to update all the monitoring probes in order to enable a local processing of the monitoring data which could be difficult for large deployments.

- Centralized analysis: in a centralized analysis, all the collected information is sent to a central point of analysis (what we call the "collector"). In the

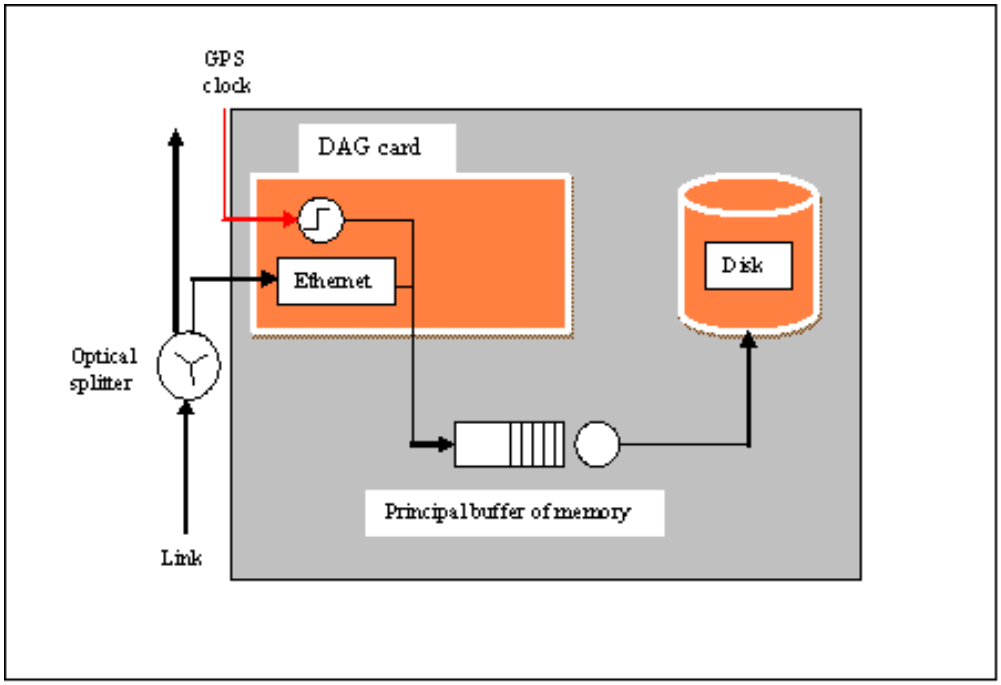

Figure 5.3: Working principle of a DAG card

VIPEER context, the dCDN will deal with all its own elements. With this approach, only one more component will be added. In contrast, using the first approach, the dCDN has to deal with all the probes, implying a higher complexity.

This centralized scenario will be described into more details in the next section.

5.4 Collector

The proposed measurement platform is composed of local data collection at the location of every partner involved in WP2 (INRIA in Rennes, Telecom Bretagne in Brest and France Télécom in Lannion). The collected traces are then converted in real time to flow-level traces containing the required information, and finally sent to a central collector (see figure 5.4) which performs the traffic classification. We will now detail more this mechanism.

The collector is the central point where the collected information is sent. The reports are sent periodically (typically every 60 seconds or less depending on our needs) using bursts of UDP packets. The collector merges all the local reports using their timestamps, which provides a global view of the network. In practice, the

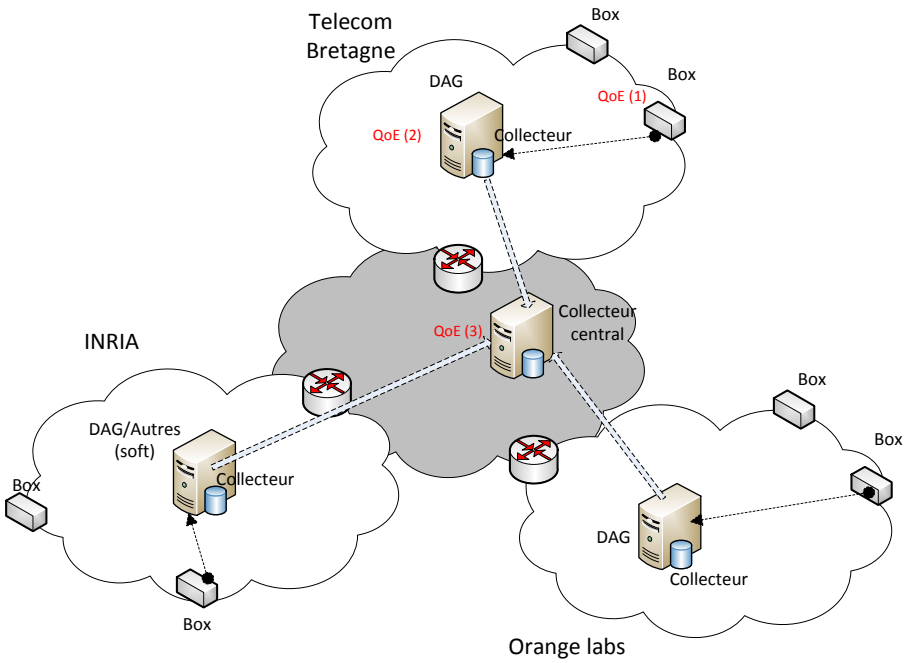

Figure 5.4: Proposed architecture

collector will work with sockets for the reception of the UDP packets. Then it is listening and waiting for a new packet.

Next, it decodes the data format, it identifies the local probe (thanks to a field in the data), it determines the number of flows described in the UDP packet burst, and finally extracts the flows themselves.

Data are stored temporarily until all the reports from the different local probes are received. At this stage, the collector has all the information that has been cumulated during the last period of time. One of the collector's goals is to share pretreatments and provide the needed information to the algorithms. Thus, if there are several algorithms, the collector also manages the execution of all algorithms, the data sharing and the allocation of all needed resources (CPU, memory...).

The QoE measurement can be handled at the different possible locations indicated in the figure 5.4. However, the closest to the user we are, the more accurate is the measurement. Thus, if we consider the QoE measurement at the Set Top Box level, the QoE values will be sent through UDP or SNMP trap messages to the collector.

To summarize, the collector will handle : the algorithm for automatic service recognition, and the decoding of all data formats while the automatic methods for evaluating the QoE will be probably be handled by the Set Top Boxes or at the terminal level.

In practice, the collector is a server that will be installed at France Telecom in

Lannion. The results of the two algorithms will be sent to an element of the dCDN and may then be used when choosing a sender for a special content.

6 Conclusion

In this deliverable we have detailed two building blocks of the VIPEER measurement infrastructure: the QoE monitoring block and the Traffic Classification block. A state of the art on network level QoS monitoring has been performed. We also have described how the main blocks of the monitoring infrastructure will be integrated in a platform distributed between the different partners. The interaction between the monitoring infrastructure and the CDN has been briefly adressed.

Bibliography

[1] P. Belzarena. Lecture notes on QoS monitoring, IIE/FING/UdelaR (Uuguay).

[2] H. Dahmouni, S. Vaton, and D. Rossé. A markovian signature based approach to IP traffic classification. In *MineNet 2007: ACM Sigmetrics Workshop on Mining Network Data, San Diego*, June 2007.

[3] Niranjan Damera-Venkata, Thomas D. Kite, Wilson S. Geisler, Brian L. Evans, and Alan C. Bovik. Image quality assessment based on a degradation model. *IEEE Transactions on Image Processing*, 9(4):636–650, April 2000.

[4] M. Dusi, F. Gringoli, and L. Salgarelli. Quantifying the accuracy of the ground truth associated with Internet traffic traces. *Computer Networks*, 2011.

[5] Institut für Rundfunktechnik. http://www.irt.de/en/activities/online/audiovideo-quality.html, 2010.

[6] ITU-T Telecommunication G.1070. Opinion model for video-telephony applications, Apr 2010.

[7] F. Gringoli, L. Salgarelli, N. Cascarano, F. Risso, and K.C. Claffy. GT : picking up the truth from the ground for Internet traffic. *ACM SIGCOMM Computer Communication Review*, 39(5), Oct. 2009.

[8] B. Krishnamurthy, W. Willinger, P. Gill, and D. Arlitt. A socratic method for validation of measurement based networking. *Computer Communications*, 34:43–53, 2011.

[9] S. Mohamed and G. Rubino. A Study of Real-time Packet Video Quality Using Random Neural Networks. *IEEE Trans. On Circuits and Systems for Video Tech.*, 12(12):1071–1083, Dec. 2002.

[10] K. Singh and G. Rubino. Quality of Experience estimation using frame loss pattern and video encoding characteristics in DVB-H Networks. In *Packet Video Workshop, Hong Kong*, Dec. 2010.

[11] C.J. van den Branden Lambrecht. Color Moving Picture Quality Metric. In *IEEE International Conference on Image Processing*, September 1996.

[12] C.J. van den Branden Lambrecht. *Perceptual Models and Architectures for Video Coding Applications*. PhD thesis, EPFL, Lausanne, Swiss, 1996.

[13] S. Voran. The Development of Objective Video Quality Measures that Emulate Human Perception. In *IEEE GLOBECOM, Phoenix, AZ , USA*, pages 1776–1781, December 1991.

[14] Z. Wang, L. Lu, and A. Bovik. Video quality assessment using structural distortion measurement. *Signal Processing: Image Communication, IEEE. Special issue on "Objective video quality metrics*, 19(2):121–132, February 2004.

[15] Zhou Wang and A. C. Bovik. A universal image quality index. *Signal processing letters IEEE*, 9(3):81–84, March 2002.