Zoo: A tool for traffic analysis and characterization User manual

Nicolas LARRIEU, Philippe OWEZARSKI
LAAS – CNRS
7, avenue du Colonel ROCHE
31077 TOULOUSE Cedex 4
{nlarrieu, owe}@laas.fr

1 Introduction

This paper introduces the Zoo tool that has been designed in the French Metropolis project, explaining why Zoo speaks French; we will soon teach Zoo how to speak English. More technically speaking, this tool has several contributions:

- It makes possible to model traffic by computing statistical moments of the traffic. In theory, all moments from first to infinite orders are required. Practically, first and second orders are sufficient to have a good representation of traffic process characteristics;
- It allows users to analyze the characteristics of different classes of traffic. These classes can be defined as we want: as concrete applications, depending on the size, or duration of flows, etc. This can provide very important information for handling these different traffic classes differently, and then adapt the protocol and mechanisms to their real requirements;
- **measure the quality of service provided by the network in terms of dynamic traffic parameters**. The principle of this QoS evaluation is then based on the oscillation nature, LRD and self-similarity levels (that are quite close notions for Internet traffic), given that the more oscillations and LRD, the lowest the QoS. It then will be important in EuQoS to check that the LRD of the resulting traffic in the EuQoS system is low. It would mean that the oscillation level is reduced and then that the utilization of resources is not far from being optimal.

2 Traffic trace manipulation

2.1 Traffic trace format

The native format for traffic analyzed in Zoo is ERF (Extensible Record Format). This is the standard packet format developed by Endace, a company providing passive monitoring hardware called DAG cards [2] [3]. Traffic traces at this format can be obtained from PCAP data (captured with TCPDUMP [7] tool for instance), using a specific tool, called DAGCONVERT, developed by LAAS and LIP6 (two French laboratories), that converts the pcap format into ERF.

Usage for this tool is:

dagconvert -i pcap_data.pcap -o erf_data.dag pcap:erf
where:

- pcap_data.pcap is the input trace to translate from pcap to erf format,
- erf_data.dag is the output trace translated in erf format.

2.2 Record length

Record length for microscopic passive captures is a really important problem. Indeed, this length impacts the complexity degree that can be made in the future analysis. For working efficiently, Zoo needs to get the full TCP/IP header for each analysed packet. So, each capture must record for each packet at least 54 bytes of data (14 bytes for Ethernet header (without the 4 CRC bytes) + 20 bytes for the IP header + 20 bytes for the TCP header without option field). Once collected in a ERF trace, each record represents 70 bytes of data (54 + 16 more bytes for ERF header).

For interested readers, figure 1 represents each field of an ERF header.

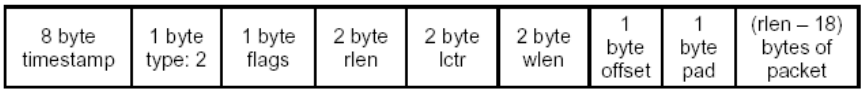

Figure 1: Details of ERF header for packets captured with DAG systems (for TCP/IP architecture over 10/100 Mbps Ethernet)

Details for each field are given below:

- timestamp: arrival date for captured packet,
- type: kind of frame collected at the link level (Eth, ATM, PoS),
- flags: various information on capture state (interface number, record not completed, . . .),
- rlen (record length): full length for the record exchanged between capture card and storage device,
- lctr (loss counter): number of packets lost between DAG card and storage device (if PCI bus is overloaded),
- wlen (wire length): real length for captured packet,
- offset / pad: number of bytes not captured at the beginning of data frame (not implemented yet in DAG cards).

3 How to use Zoo?

3.1 Software and hardware requirements

Because of the algorithm complexity and the size of traces, Zoo has to work on a powerful computer with a large memory. A standard hardware configuration is Pentium IV at 2 Ghz with 1 Go of

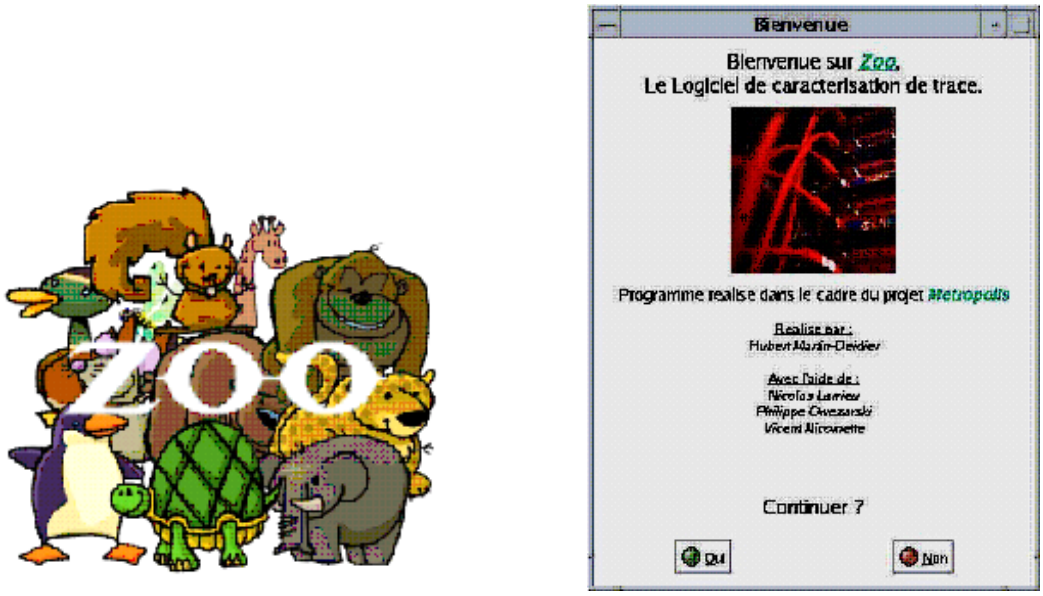

Figure 2: Introduction window

RAM. This software was developed under Linux operating System and needs a Kernel 2.2. Moreover, graphical user interface (GUI) is based on GTK 2 (Gimp Tool Kit [4]). Thus, GTK library must be installed otherwise Zoo will not work well.

3.2 User manual

Each feature of Zoo is described in this section and it also gives its GUI details.

- Introduction window

  When launching Zoo, an introduction window (cf. figure 2) is opened in front of the main window. It introduces the and proposes to go ahead or quit Zoo. If “Non” button is selected, the software is closed whereas one click on “Oui” button close the introduction window and put the focus on the main one.

- Main window

  The main window (cf. figure 3) is composed of:

  – One menu bar: users can launch every feature of Zoo. It presents the following menus: “Fichier”, “Analyse” and “Aide”,.Each of them will be detailed in next parts.

  – One tool bar: it presents the main features of Zoo that users can quickly launch: “Nouveau”, “Ouvrir”, “Enregistrer”, “Analyser”, “Afficher”, “Options” and “Quitter”.

  – One notebook: this printing zone is used to display graphics and statistic results of analysis.

  – A state bar: it is used to display some messages for users: help, error message, etc.

- “Fichier” menu

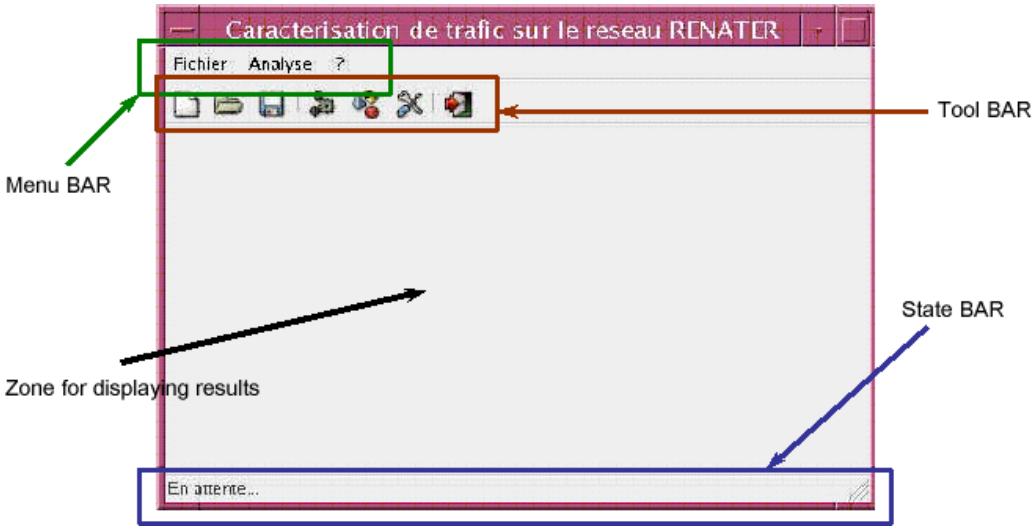

Figure 3: Main window

(a) "Fichier" menu

(b) "Analyse" menu

Figure 4: Main menu bar details

This menu (cf. figure 4(a)) manages the whole analysis and is also used to quit the software. Users can find the following sub-menus: "Nouveau", "Ouvrir", "Enregistrer", "Fermer" and "Quitter".

"Nouveau"

This sub-menu allows the creation of a new project by detailing the directory and each file name in which the results will be recorded. Moreover, the creation of a new project makes an initialization of all option values.

By clicking on "Nouveau" and if any other project is already opened, the window depicted in figure 5 is displayed.

On top of this window, a field is used to enter the directory name (where results will be saved) either directly, or with the "Parcourir" button that helps to select a specific directory in the file system. Then, user can enter names for all files in which results will be saved. For easiness reasons, default values are initially proposed.

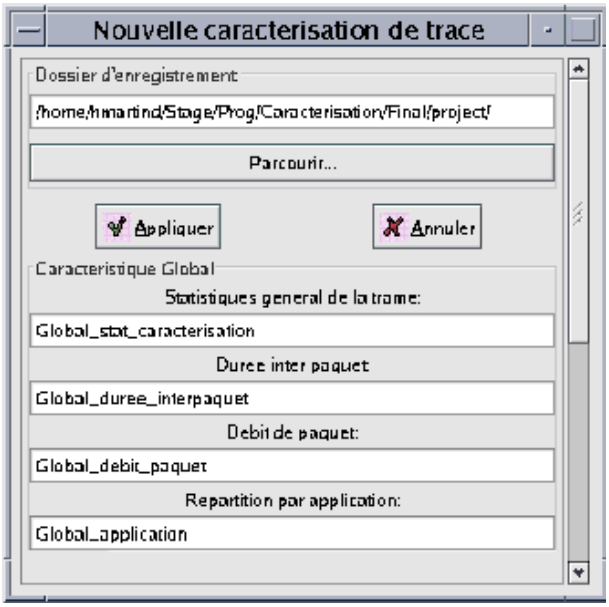

Figure 5: “Nouveau” window

Finally, the user must click on “Appliquer” button to apply all its choices. If “Annuler” button is selected, he comes back to the main window without saving the modified parameters.

“Ouvrir”

The sub-menu “Ouvrir” is about an already saved project. Saved data are contained in the name of the output directory, file names and the whole options selected by user. Once the project opens, all values are initialized with the ones read in the selected file. It is then possible to launch a new analysis or display old results.

If this sub-menu is selected and if any other project is open, a window is displayed. Thus, the user can choose a name for its project.

“Enregistrer”

The sub-menu “Enregistrer” is used to save an open project. Saved data deals with the ouput directory name, result file names and the whole options chosen by the user.

“Fermer”

This sub-menu is about the end of an open project. It refreshes the display of main window. The file closing allows users to open another project. The simultaneous opening of two different projects is impossible with Zoo.

“Quitter”

This sub-menu is for terminating the Zoo session. A confirmation message is then displayed for the user to confirm its initial choice.

- “Analyse” menu

  This menu (cf. figure 4(b)) is about the management of trace analysis. The following sub-menus are proposed: “Lancer”, “Afficher” and “Préférences”.

  “Lancer”

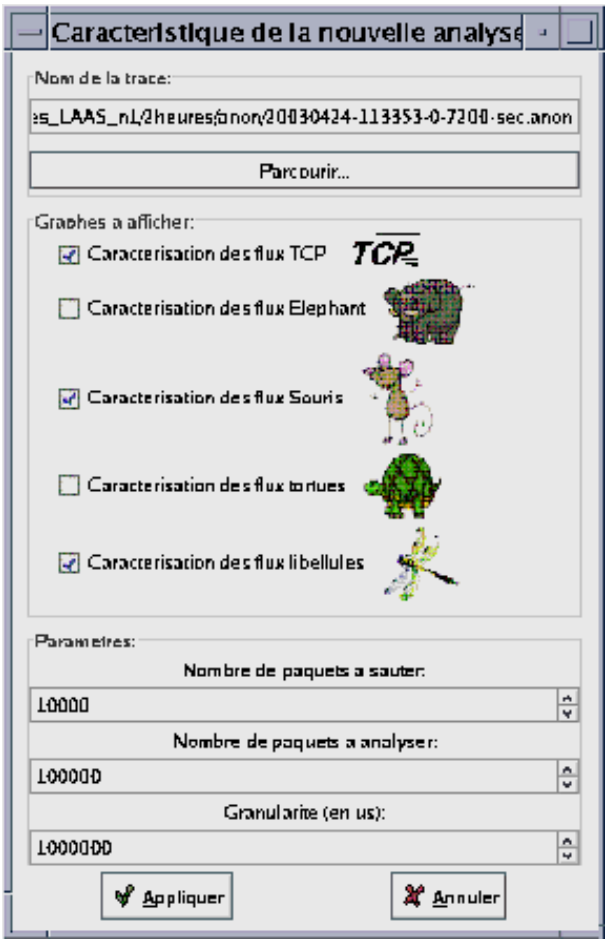

Figure 6: Window to launch an analysis

This sub-menu (cf. figure 6) launches a new analysis if a specific project has been already opened by the user. One click on this menu displays a new window where main analysis parameters can be defined and selected:

- Trace name to analyze: a field is used to enter the trace name either directly or with the “Parcourir” button that helps to select a specific file in the file system.
- Figures to display: user can click on the checkbox and then activate or not the analysis for a specific kind of flows. Disabling this option shortens the time of analysis (which can be rather long on a computer with little memory and low CPU speed).
- The main parameters for the analysis.

Once analysis parameters have been selected, the user can choose to launch it with the “Appliquer” button. If “Annuler” button is selected the window is closed and the analysis is not started.

“Afficher”

This sub-menu displays resulting figures and statistics for an old analysis if a project is already opened. Note that every result figures (PNG files) are saved in the directory read at the opening of a specific project, otherwise an error message is displayed.

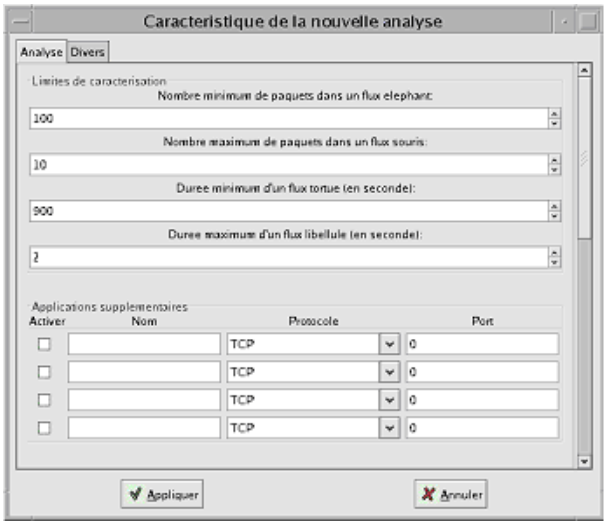

Figure 7: Analysis options

"Préférences"

This sub-menu displays and modifies some specific parameters of an analysis. These options are divided into two groups:

- Analysis options (cf. figure 7): user can define lower and upper bounds for the analyzed flow, others applications he wants to characterize and the maximum value for correlation computing.
- Miscellaneous options (cf. figure 8): user can define a zooming ratio for displaying figures, and if he wants, files to be saved at the exit of Zoo.

- "Aide" ('?') menu

  This menu displays the user help as well as some miscellaneous information on Zoo.

- After an analysis

  Once the analysis ends, the notebook is updated and selected figures are displayed. Figures 9 and 10 below show a typical display after a classical analysis.

  First notebook pages introduce global analysis statistics and then, detail results got for every type of applications. User applications are added after main applications. Next pages display figures based on the kind of decomposition (TCP, Elephant, Mice, Dragonflies or Tortoises) and based on observation level (packet and flow). If this analysis has run on a not sufficient number of packets or flows, figures are not displayed for the concerned classes. In this case, an error message informs the user, and the related pages are not added to the notebook.

4 Decomposition example: Mice vs. Elephant traffic characterization

We are going to illustrate the features of Zoo on a simple example. We have analyzed a trace captured on RENATER. We are going to divide traffic into two different well known flow classes: mice and

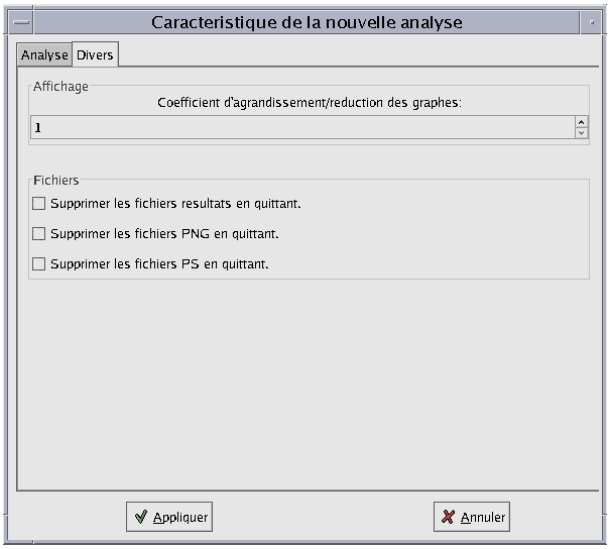

Figure 8: Miscellaneous options

elephants. We are going to analyze which differences can be highlighted for these two classes by taking into account the following parameters:

- Flow arrival distribution,
- Packet arrival distribution,
- Correlation level for flow arrivals,
- Correlation level for packets arrivals,
- LRD level for flow arrivals,
- LRD level for packet arrivals,

In this section we are going to start with mice flows. Then, we will consider elephants. For characterizing each flow class, we are going to consider two different analysis levels (flows and then packets). This approach brings two kinds of modeling results. Indeed, if we consider an upper level of observation (application level for instance), it is rather interesting to study flow behaviors. At the opposite, if the goal is to model transport protocols for example, resource provisioning, QoS and performance evaluation, it can be more interesting to consider a low observation level with packet level modeling results.

4.1 Mice traffic characterization

- Mice flow characterization

  Mice are defined as flows with a short number of packets [6]. A good example of an application exchanging a lot of mice is web browsers that open a lot of mice connections to download the different objects included in a HTML page. It is really interesting to model arrival times of these flows because they are most related to application level and give a more accurate idea on user and application behaviors.

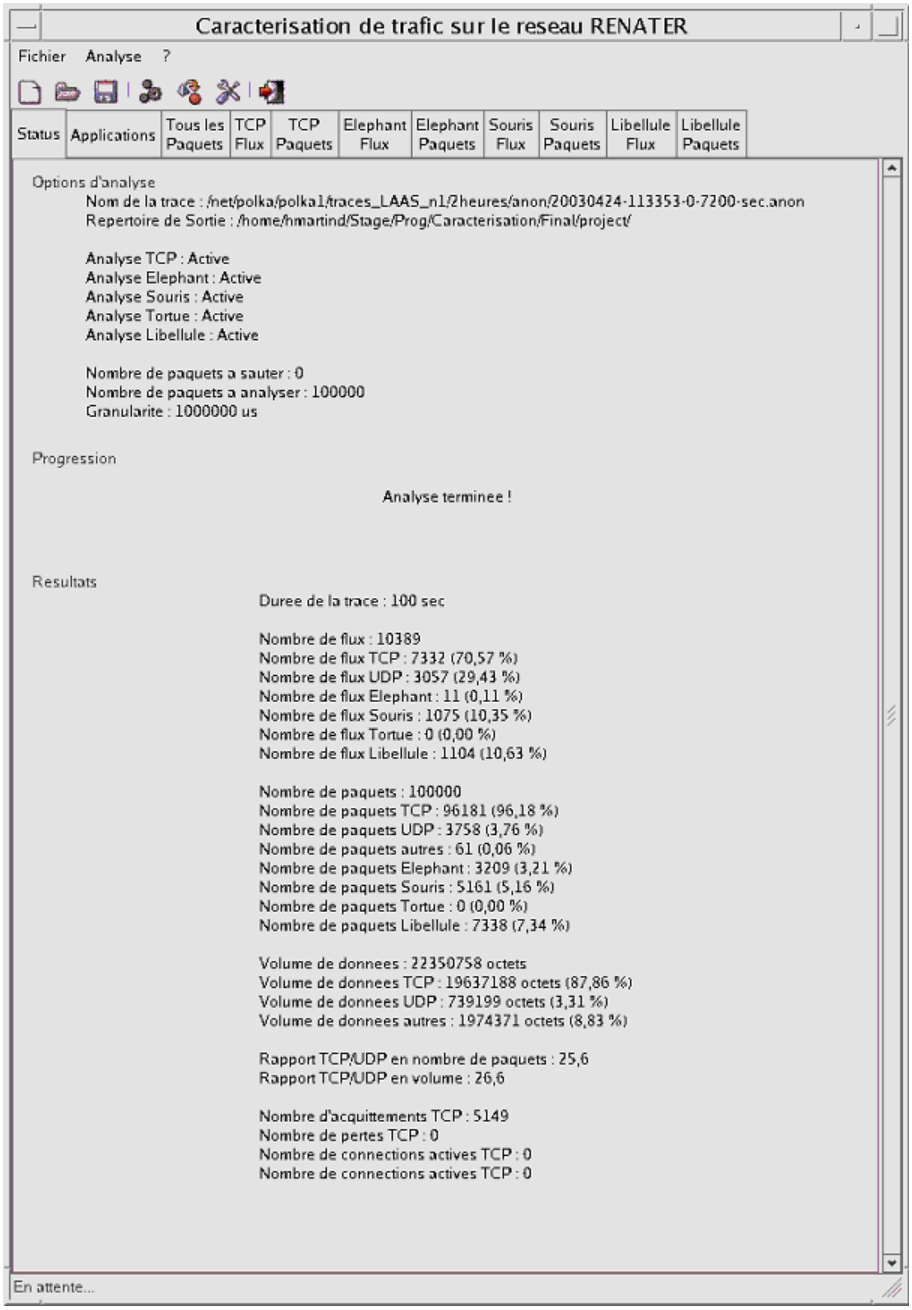

Figure 9: Display of analysis statistics

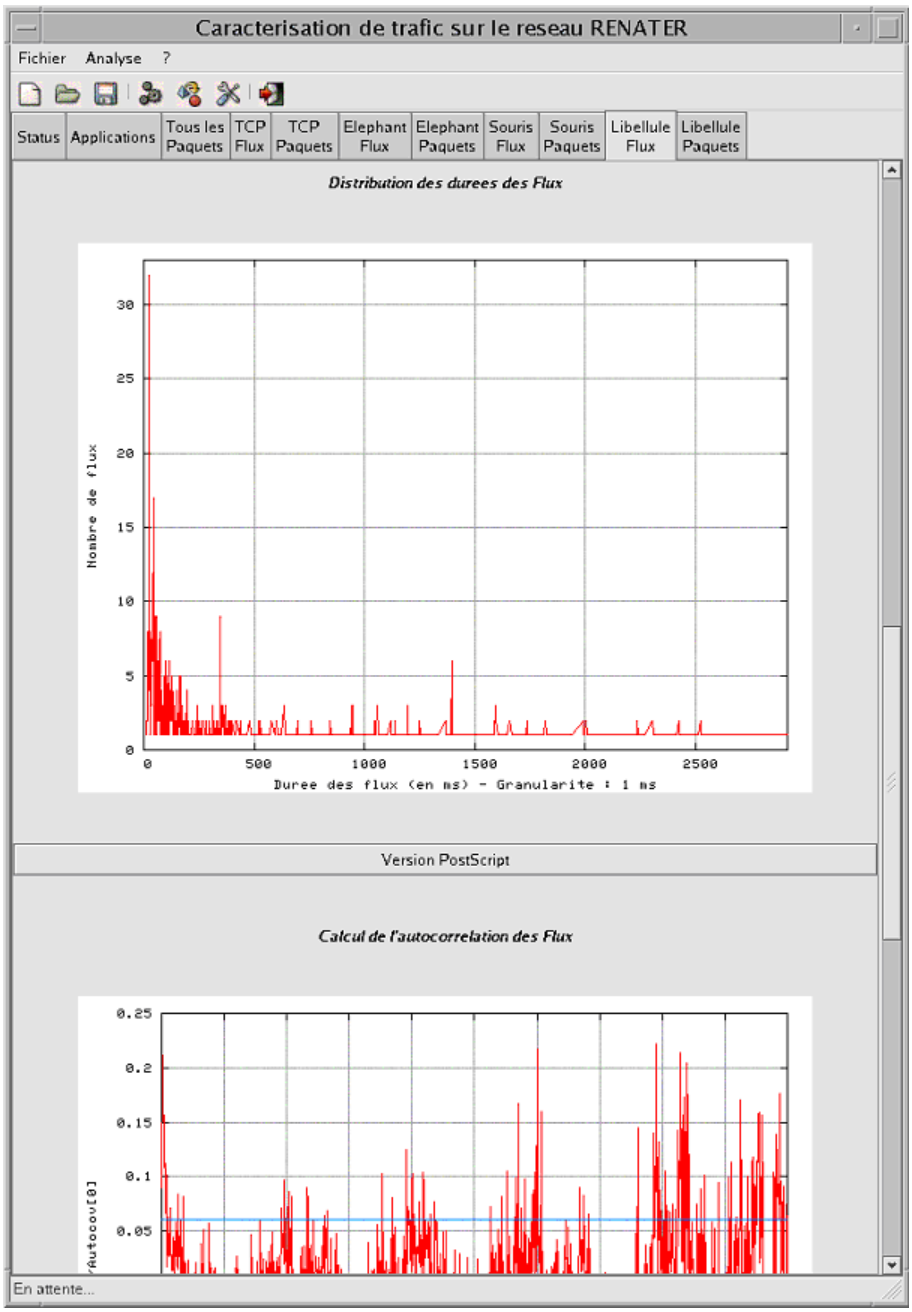

Figure 10: Display of figures produced after analysis

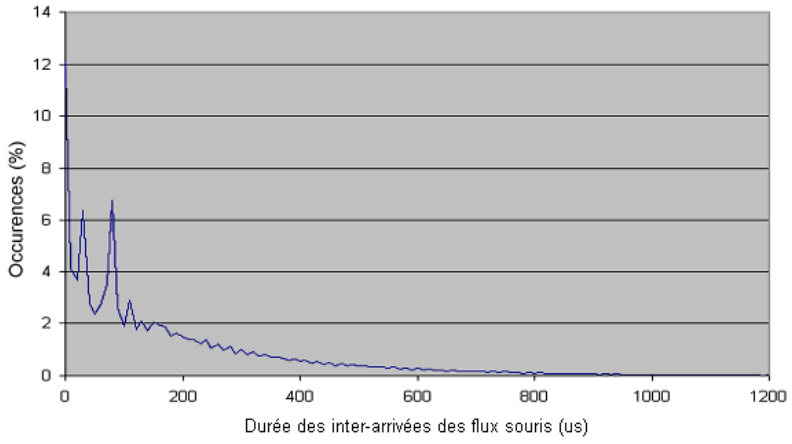

Figure 11: Arrival distribution for mice flows (Granularity = 50 us)

Arrival process for mice flows

First step for this characterization is about the computation of arrival distribution for mice flows (cf. figure 11).

hen, we can not conclude whether the arrival process for mice flows can fit a Poisson process. For this purpose, we need to analyze correlation levels for these arrivals.

Auto-correlation structure for mice flows

This function (cf. figure 12) extracted from mice flow arrival process gives a qualitative information on the degree of LRD in the traffic. Indeed, this figure depicts a complex process (rough oscillations and slow decrease). Moreover, every values are not included in Gaussian bounds ($\pm$ 0,0038). So, these constant values highlight some dependency in the arrival of mice flows, then exhibiting that the arrivals of mice do not follow a Poisson process.

- Mice packets characterization

  Mice packet arrival process

  After considering only mice flows, we now take into account the whole packets generated by these short flows. For this purpose, figure 13 illustrates the distribution for packet inter-arrival time in microseconds for this class of traffic.

  It is obvious that this distribution does not bring enough information to characterize this part of traffic. We need, as in the case of mice flows, to consider the auto-correlation function for inter-arrival times of mice packets.

  Auto-correlation structure for mice packets

  This function (cf. figure 14) extracted from mice packet arrival process depicts a complex process (slow decrease) and all values are not included in Gaussian bounds ($\pm$ 0,0020). So, these constants highlight a high level of dependency in the arrival of mice packets, thus showing it does not follow a Poisson process.

  Hurst parameter and wavelet based method

  To quantitatively characterize the LRD level of current Internet traffic, we realized a wavelet based decomposition of traffic traces. Interested readers can refer to [1] for details on the

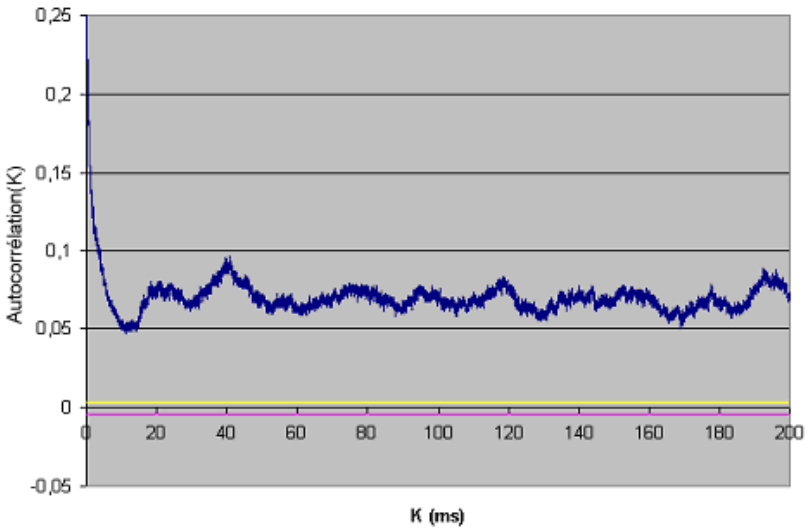

Figure 12: Auto-correlation function for mice flow arrival process

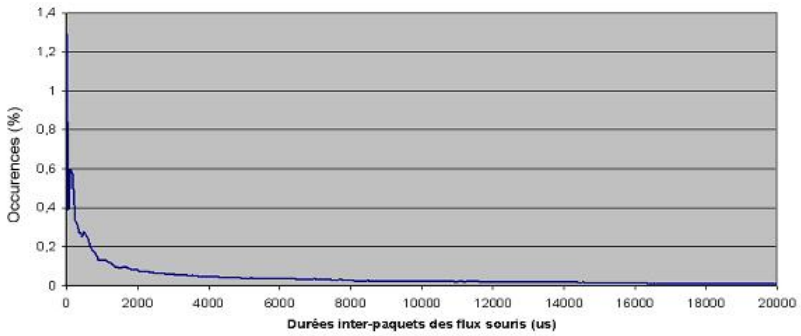

Figure 13: Arrival distribution for mice packets (Granularity = 50 us)

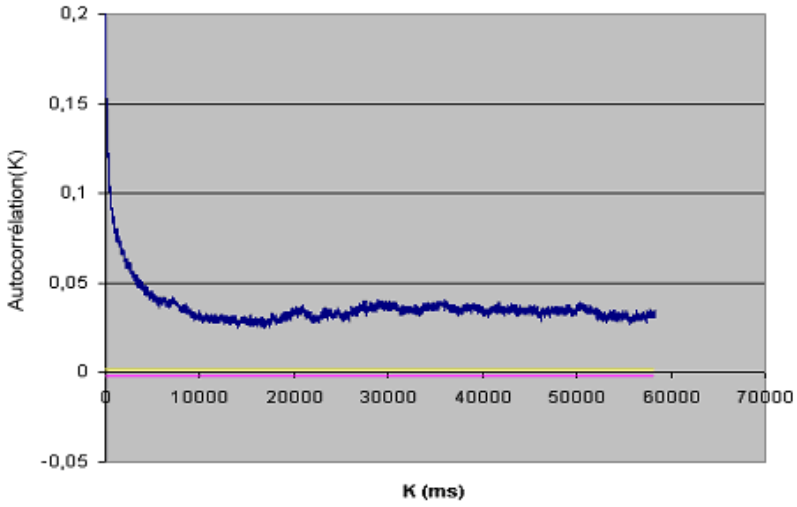

Figure 14: Auto-correlation function for mice packet arrival process

wavelet based method as well as the LDestimate tool that provides the LRD diagrams illustrating this section. It shows the signal with different granularities of observation and then gives information on variability degree of the analyzed signal (here inter-arrival durations) according to the range of observations.

The estimation of Hurst factor gives $H = 0.622$. This value is greater than 0.5. Thus, there is LRD in the traffic of mice. Moreover, this value helps us to quantitatively compare LRD degree generated by each class of flows: mice and elephants. Figure 15 depicts the LDestimate diagram that helps us to study scale laws in the traffic.

4.2 Elephant traffic characterization

- Elephant flow characterization

These last years, Internet traffic knew an important change. New applications such as peer-to-peer for instance appeared. And their big success impacts in an important way the profile of exchanged file length. Thus, traffic has more and more long flows (elephants such as audio or video files). It is then very important to better understand the behaviors of these flows that represent nowadays the most important part of data volume in the Internet.

Elephant flow arrival process

Figure 16 illustrates the distribution of inter-arrival for elephants.

Auto-correlation structure for elephant flows

The auto-correlation function computed on elephant traffic shows that data series have lowest correlation levels. Indeed, most of the points are between the two Gaussian bounds ($\pm$ 0,048).

This analysis on elephant flows shows that their arrival process seems to fit a Poisson process (with an exponential distribution that does not show dependency). This result is not surprising as we have already said that elephant traffic can be represented by very heavy transfers of very

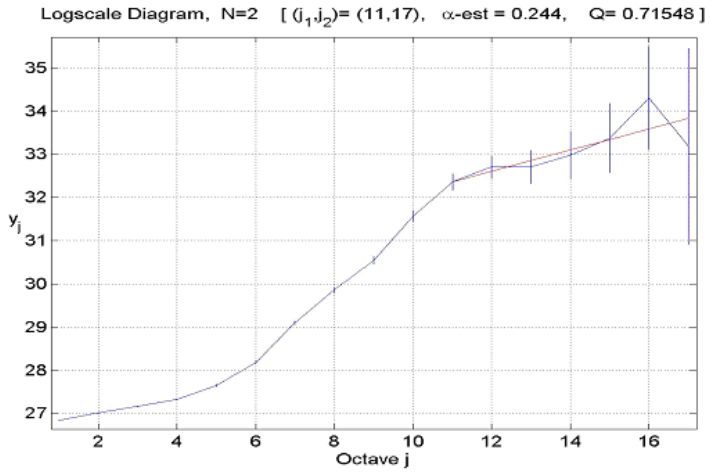

Figure 15: Logscale diagram computed with wavelet based method on inter-arrival times of mice packets

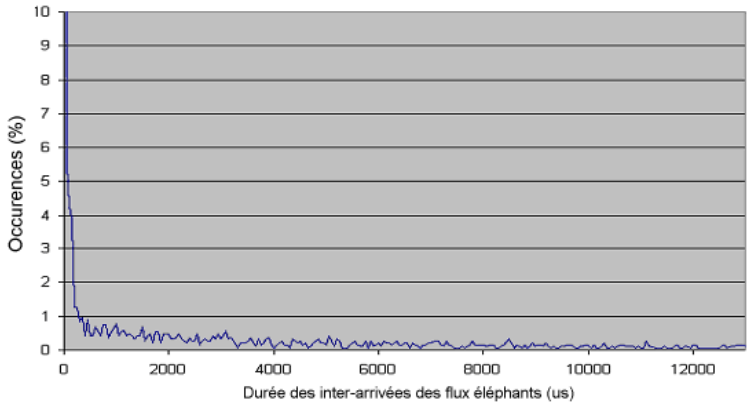

Figure 16: Distribution for elephant flows (Granularity 50 us)

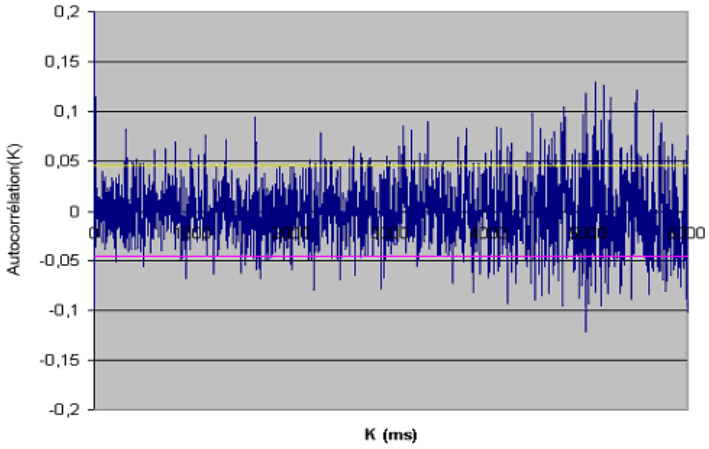

Figure 17: Auto-correlation function for elephant flow arrival process

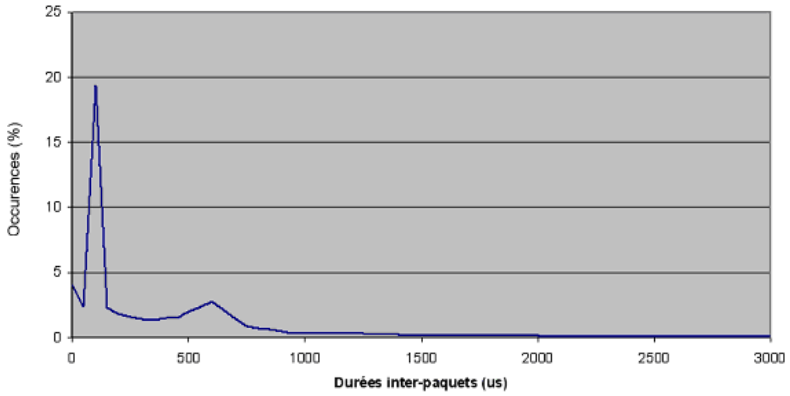

Figure 18: Distribution for elephant packet arrivals (Granularity 50 us)

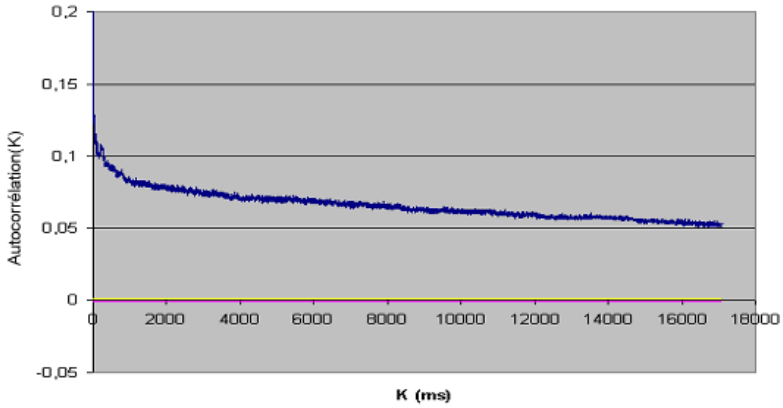

Figure 19: Auto-correlation function for elephant packet arrivals

big files: for instance, downloading of audio or video files. They are then independent from each others.

- Characterization for packets of elephant flows

  Elephant packet arrival process

  Figure 18 depicts this arrival process.

  Auto-correlation structure for elephant packets

  The auto-correlation function (cf. figure 19) shows a very strong correlation (every points are out of Gaussian bounds ($\pm$ 0,011) and its behavior shows a very slow decrease).

  Hurst parameter and computation of LRD level

  This diagram (cf. figure 20) extracted with wavelet based method allows us to analyze scale laws hidden in elephant traffic. The Hurst value computed with this method is $H = 0.840$. This value is very significant because it highlights that elephant packets are responsible of the most important part of LRD in the traffic. First, we analyze a Hurst factor really higher than

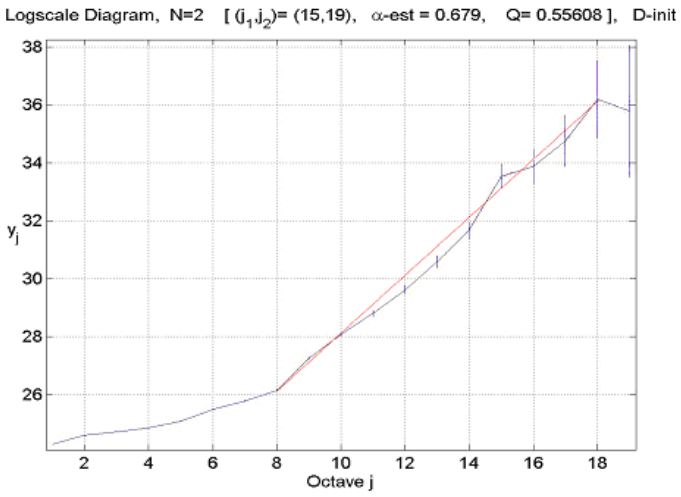

Figure 20: Logscale diagram computed with wavelet based method on inter-arrival times of elephant packets

0.5. Thus, this value is very close to Hurst parameter of global traffic ($H = 0.844$). We can recall that this factor was 0.622 for mice packets. So, this observation is raising the following question: How is elephant traffic contributing so much to the LRD degree of current Internet traffic?

This answer has been already addressed. The conclusion is generally that TCP is not suited for transmitting long flows on high speed networks. Indeed, congestion control mechanisms introduce long range dependence in the traffic [9] [5] [8], what creates large oscillation and non stability issues in the network, thus making stable QoS difficult, quite impossible, to enforce in the presence of such a traffic.

References

[1] P. Abry, D. Veitch, *Wavelet Analysis of Long Range Dependent Traffic*, Transaction on Information Theory, Vol.44, No.1, January 1998.

[2] S. Donnelly, I. Graham, R. Wilhelm, *"Passive calibration of an active measurement system"*, in Passive and active measurements workshop, Amsterdam, April 2001.

[3] Endace Web Site, http://www.endace.com

[4] GTK Web Site, http://www.gtk.org

[5] Guo L, Crovella M. and Matta I., *How does TCP generate pseudo-selfsimilarity ?*, 9 pages, Proc. the International Workshop on Modeling, Analysis and Simulation ofComputer and Telecommunications Systems (MASCOTS'01), Cincinnati, OH, Août 2001.

[6] Papagiannaki K., Taft N., Bhattacharyya S., Thiran P., Salamatian K., Diot C., *A pragmatic definition of Elephant in Internet backbone traffic*, 2001.

[7] Site web du logiciel TCPDUMP : http://www.tcpdump.org.

[8] Sikdar B. & Vastola K., *On the contribution of TCP to the selfsimilarity of network traffic*, Proceedings of the 2001 Tyrrhenian International Workshop on Digital Communications: Evolutionary Trends of the Internet, Springerv 2001.

[9] Veres A. & Boda M., *The chaotic nature of TCP congestion control*, in Proceedings of IEEE INFOCOM'2000, 9 pages, March 2000.