Multi-Layer Protocol Analyzer: A Performance Monitoring Framework for Unbundled ICT Environments

P.J. Meulenhoff [a], R.D. van der Mei [a,b], B.M.M. Gijsen [a]

[a]TNO Telecom, Knowledge Innovation Center, 2260 AK Leidschendam, Netherlands
[b]Vrije Universiteit, Faculty of Exact Sciences, 1081 HV Amsterdam, Netherlands

Abstract

The unbundling of the telecom industry, separating functional roles such as access and core providers, service providers and content providers, has become reality. This has led to new services offered over distributed, heterogeneous, multi-domain infrastructures, and an increasing importance of Service Level Agreement between administrative domains. These trends have raised the need for new tools that provide insight in both the per-domain performance and the end-to-end performance experienced by the end user. In this paper we present the Multi-Layer Protocol Analyzer (MLPA), a new monitoring framework that provides these insights in real time. The MLPA concept has been implemented in an easy-to-install and user-friendly software package. The practical applicability of the MLPA is illustrated with results from a number of practical cases for real customers in the telecom industry.

1. Introduction

Over the past few years the landscape of the information and telecommunication industry has been subject to dramatic changes, with major consequences for management of traffic and performance of emerging services. First of all, unbundling of the telecom world is becoming reality, separating functional roles in the telecom and information domain, such as access and core network providers, service providers and content providers. This leads to a heterogeneous, multi-domain infrastructure over which new services are supported, and consequently, an increasing importance of Service Level Agreements (SLAs) between administrative domains. Also, the multi-domain infrastructures complicate the control over end-to-end performance. Secondly, boosted by the success and opportunities enabled by the

Internet, the variety of offered services and user-devices has grown tremendously. Typically, in order to stay in business service providers have to offer a wide range of these services, such as e-mail, information downloading, ticketing, and streaming services. Since each of these services has its own traffic and performance characteristics, it is important to be able to control the traffic and performance of these services both in isolation and on aggregate level. Third, the increasing usage of E-commerce applications illustrates that performance of information and communication infrastructures are becoming more business critical. For example, in the e-commerce context customer dissatisfaction about overly long response times or unavailable servers will directly cause a decrease in revenues for e-commerce applications. In short, controlling traffic and performance of modern services is becoming more important and more complex at the same time.

These trends have raised the need for new tools that provide insight in traffic and end-to-end performance of mixtures of services in multi-domain environments. In particular, there is a critical need for tools that enable service providers to monitor the end-to-end performance at the end-user level, and to correlate measurements at different protocol layers and from different administrative domains. These features are offered by commercial monitoring packages only to a limited extent. Some products provide end-to-end performance metrics based on remote softwareagents that measure the responsetime while executing a recorded web-transaction at regular intervals (SiteGuardian, HP OpenVew Web Transaction Observer). Other products are focussed on the performance metrics of a servers or applications (Microsoft Performance monitor, NetIQ AppManager). Network Analyzer like products (RADCOM, Sniffer Technologies) concentrate on network troubleshooting and lower layer network performance metrics.

To this end, in this paper we present the Multi-Layer Packet Analyzer (MLPA), a new monitoring tool that provides this insight in real-time. The MLPA will be illustrated with results from a number of practical cases where we used to MLPA for analysing traffic and performance of i-mode and MMS services for real customers.

This paper is structured as follows. Section 2 is a general description of the event analysis framework that is the basis of MLPA. Section 3 is a description of the details and implementation of MLPA. Section 4 describes the application of MLPA in the i-mode environment. Section 5 concludes the paper.

2. Event Analysis Architecture

The MLPA is based on an event-based architecture. The typical environment where MLPA operates is usually distributed over several independent network domains, which favours the usage of loosely coupled component. In such an environment, components exchange information using asynchronous communication. The most important element of this architecture is the Event-Processor. The majority of components are based on the element. A schematic overview of the Event-Processor is given in Figure 1. Events received are stored in a queue (EventQueue) and subsequently processed by an Event-Handler. After processing the Event, it is sent (to another Event-Processor).

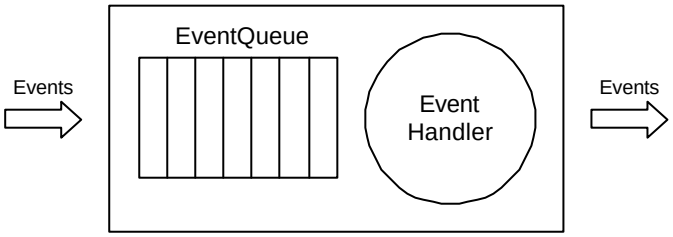

Figure 1. Logical overview of the Event-Processor.

The function of the Event-Processor is primarily defined by the function of the Event-Handler and the configuration of input and output of events: some Event-Processors can transmit and receive events, some can only transmit, etc. Multiple Event Processors, each with its own function, can be couple together to form a loosely coupled environment, in which events can be routed, processed, analysed, stored, etc.

In this event-based architecture we group Event-Processors into four different classes. The Event-Processors in each class all have a common function. These four classes are: (1) collectors, (2) processors, (3) storage and (4) presentation. An overview of the relations between the Event-Processors in each of these four classes is given in Figure 2 below.

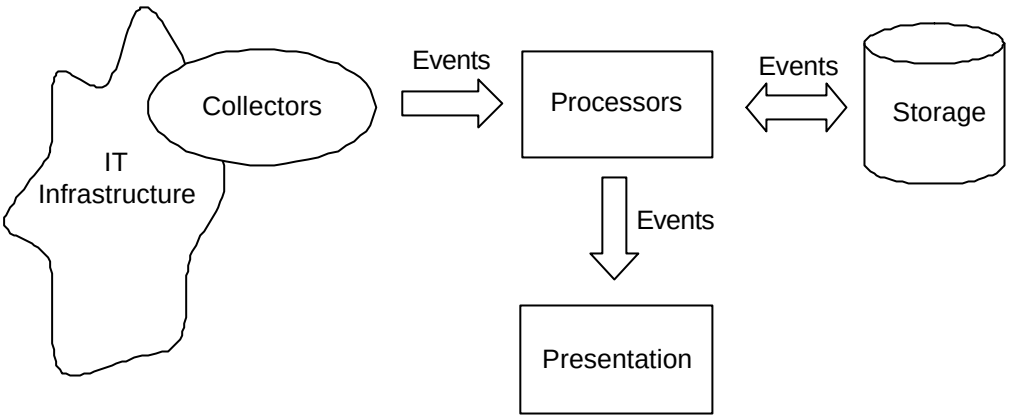

Figure 2. High level overview of the four Event-Processor classes and their relation.

These four Event-Processor classes are described by the following characteristics:

1. *Collectors* create events. Event-processors in this class gather performance information from the underlying ICT infrastructure, convert them into event-format, and send them to another event-processor. A typical example of a collector is a software module that gathers the CPU/Memory/Disk/Network/Webserver load-data from a webserver and transmits this data at regular intervals.

2. *Processors* route and convert incoming events. These class of event-processors can either be used to route the stream of incoming events towards the desired direction. For example, an Event-Processor can make a copy of all incoming events: send one to a persistent storage facility and send another event away for further analysis by another event-processor. These Event-Processors can also be used to convert events where one or more incoming events of type A are converted into one outgoing event of type B. An example of a such a processor is one that receives CPU/Disk/Memory/Network/Web-server load events, calculates an average and sends this data at a lower interval to another event-processor.

3. *Storage* is the class of event-processors that is responsible for storing and retrieval of events. The event-processor cab either store incoming events into a database, or retrieve previously stored events from that database and send them to another event-processor.

4. *Presentation* converts incoming events into another (external) format. Event-processors in this class can be used to: convert incoming events to a graph on a Web site, convert incoming event into a SQL query, export a calculate performance parameter to a network management system, etcetera.

3. The Multi-Layer Protocol Analyzer

The Multi-Layer Packet Analyzer (MLPA) is a software product we built on the basis of the event-based architecture described in this section 2. The primary goals of MLPA are to (1) monitor, (2) calculate and (3) report the performance of TCP/IP based Internet services such as HTTP. Important features of MLPA are that we use a method of non-intrusive network monitoring to gather performance data and that we calculate end-to-end performance indicators that give insight in the quality experienced by the end-user.

The main process of MLPA is to capture Network traffic from one or more network links, calculate the correlation between multiple layers of that traffic, and extract performance indicators from the correlated data. Typical performance indicators calculated by MLPA are application- and network-layer response times, traffic overhead, request rates, and bandwidth and application usage. In addition, the correlation between several Key Performance Indicators (KPIs), such as HTTP response times and HTTP request rates, is calculated. The architecture of MLPA is depicted in Figure 3.

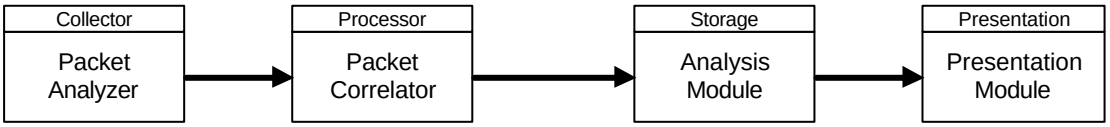

Figure 3. Architecture of the MLPA.

MLPA consists of four modules: (1) a Packet Analyzer (PA) that passively captures network traffic, (2) Packet Correlator (PC) that calculates the correlation between performance metrics at the HTTP, TCP and IP protocol layers, (3) Analysis Modules (AMs) to conduct specific analysis and calculate performance indicators, and (4) a Presentation Module (PM) to convert

the calculated performance indicators into web-based format. The four modules are discussed in more detail below.

PacketAnalyzer

The Packet Analyzer (PA) is comparable to standard Network Analyzer software like ethereal, `tcpdump` [PCAP] that can passively monitor the connected network. The core process of the PA is to capture Ethernet frames from a network interface, convert the data in these frames into events and send them to the Packet Correlator (PC) process. The PA can be configured to capture specific network traffic using a standard filter expression [PCAP].

Packet Correlator

The Packet Correlator (PC) is the process where the correlation between TCP/IP and the application layer (HTTP) of network frames is calculated. During this phase all details of a application-layer transaction are calculated. For HTTP such a transaction is defined by the HTTP request by the client, the HTTP response by the server and the acknowledgement(s) from the client that the response was received. The acknowledgements(s) from the client refer to the TCP packets sent by the client that confirm that a packet sent during the HTTP response from the server has been successfully received. The most relevant performance indicators calculated are (in case of HTTP): timestamps of the HTTP request, HTTP response, TCP acknowledgements, Number of bytes transferred at each layer (IP, TCP, HTTP), retransmitted data, HTTP request header, HTTP response header. These performance indicators are stored in a HTTP Transaction object, a subclass of a TCP Connection object, and subsequently sent to one or more Analysis Modules.

Analysis Modules

Analysis Modules (AM's) are the software modules that carry out the actually desired performance analysis. The Packet Correlator (PC) is a process that calculates Key Performance Indicators (KPI's) for TCP connections. An AM uses this information to calculate for exmaple the avarege response a specific Web site. The input of the AM's are the TCPC onnection events. During analysis these events are processed and converted into

graphs/tables/histograms objects containing the desired performance statistics. These objects containing the performance statistics for a specific range in time are persistently stored by a storage processor, a component of this module. The current implementation of MLPA uses one analysis module that calculates performance statistics for URL's based on an XML configuration. Using this XML configuration it is straightforward to add, modify or delete specific statistics. The screen shot in Figure 4 gives a brief overview of the partially collapsed XML configuration that is used during analysis.

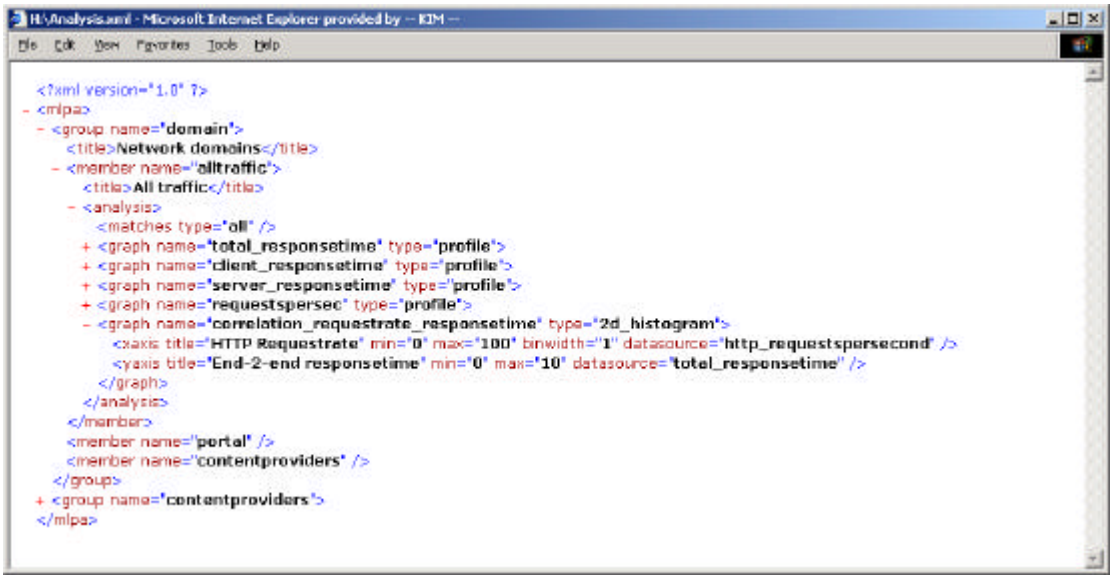

Figure 4. Screenshot of a partially collapsed XML configuration file of the AM of MLPA.

The configuration shown in Figure 4 specifies what type of graphs are created and the performance indicators that are used to fill the graph. The graphs can be grouped to a specific subject. The tag with name matches specify which range of URL's should be used to calculate the statistics of this group of graphs. One graph (`correlation_requestrate_responsetime`) that is not collapsed is specified to contain the correlation between the total HTTP response time and the HTTP request rate.

Presentation Module

The presentation module of MLPA is a software program that converts the statistics objects calculated by the Analysis Module into a Web site including graphs, tables, histograms etc. The presentation module uses a storage processor to retrieve the previously stored graphs

objects from file and convert them in the desired output format. The presentation module also uses the XML configuration file (Figure 5) to add application-specific details (e.g., grouping of results, layout and descriptions). A screen shot of the output Web site of the Presentation Module is given in Figure 5 below.

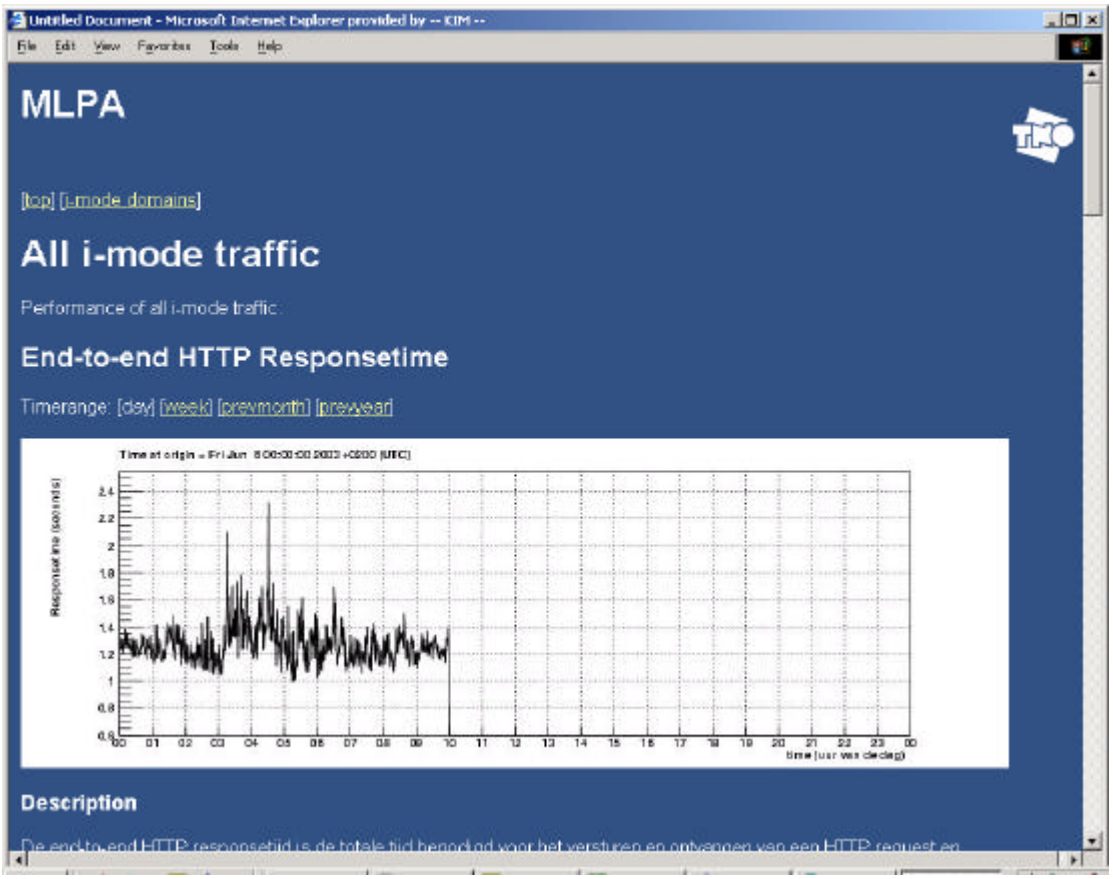

Figure 5. Screenshot of the output Web site of MLPA as created by the Presentation Module.

Implementation

The current implementation of our event-analysis architecture and MLPA is largely based on ROOT [ROOT2001]. ROOT is an Objected Oriented Data Analysis Framework that was created in the context of the NA49 experiment at CERN. NA49 generates an impressive amount of data, about 10 Terabytes of raw data per run. Therefore, NA49 is an ideal environment to develop and test the next generation data analysis tools and to study the problems related to the organization and analysis of such large amounts of data. More information about ROOT, its development and use can be found at ROOT Web site (`http://root.cern.ch`). In our architecture ROOT is primarily used for storage of

events. All event-objects are a subclass of the ROOT TObject class. The TObject class is a base class that implements functionality such as object I/O, error handling, sorting, inspection, printing and drawing.

4. Application of MLPA

Currently, MLPA is being used in a production environment to monitor the end-to-end performance of one of the largest wireless Internet services in the Netherlands. This Internet service is built on top of GPRS, a packet switched wireless network using the GSM infrastructure. A simplified overview of this wireless Internet service and the setup of MLPA is given in Figure 6 below.

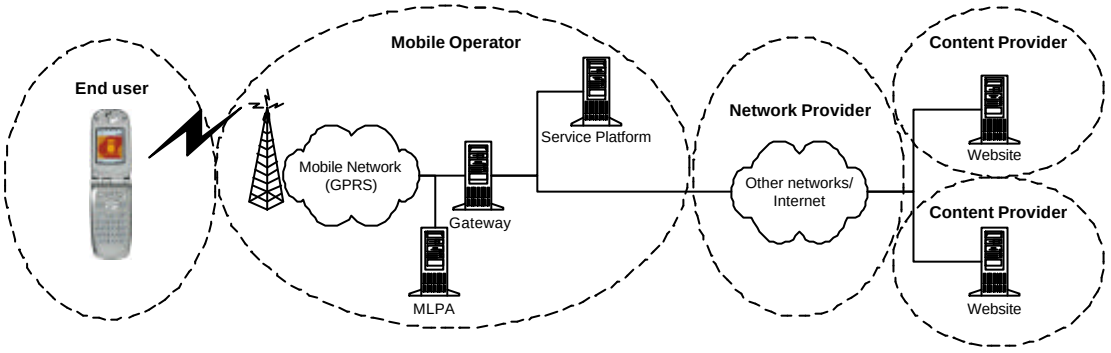

Figure 6. Schematic overview of the wireless Internet service and setup of MLPA.

In Figure 6 the dashed lines indicate the independent organisations that together form the complete service: end-user, mobile operator, network provider and content providers. MLPA is attached to the network of the mobile operator and captures all network traffic that is exchanged between end-user terminals and the different service platforms or Web sites.

The end-user is connected to the GPRS network of the mobile operator using a mobile phone or terminal that is equipped with an embedded Web browser and E-mail client. This client can either connect to a Service Platform (SP) or a Content Provider (CP) Web site. The SP provides functionality like: fetching E-mail, changing passwords, reading help, (un)subscribing to CP services. The CP Web sites deliver the actual content, e.g.,. real-time stock-information, weather, traffic, news, games, ring tones, music or image downloads and images. A CP Web site is usually restricted to users which have a subscribed to the Web site

for a monthly fee. Besides a group of official CPs, the end-user is able to visit content that is available on the Internet.

A gateway is located on the path between the end-user and the SP or CP Web site. This gateway, comparable to a HTTP Proxy server, is responsible for controlling access to the various Web sites. Users that have not subscribed to CP Web sites are blocked access by the gateway and redirected to the SP so that they can subscribe to the specific CP service.

The complete service is distributed over several organisations, which are strictly separated. In such an environment it is difficult to get detailed insight in the end-to-end performance and behaviour of the complete service and its constituent parts using standard network management systems. In this environment, an instance of MLPA is installed on the network of the Mobile operator located between the gateway and the GPRS network and configured to monitor all network traffic exchanged by the end-user mobile terminal and the different servers/Web-sites in the network. The complete MLPA system (excluding the Web site where the results are displayed) is installed on a single Linux OS based server. A Dell PowerEdge 2650 rack system equipped with 2 1.8Ghz Intel Xeon processors and 1Gb of memory.

The analysis modules used in MLPA are configured in such a way that performance statistics are gathered of the complete service and for each of the individual components of the service. This enables systems management to inspect the total performance of the operational system as to possible cause of errors or slower end-to-end performance.

Output samples

One of the most useful performance metrics is the HTTP response time. This metric is a close approximation of time the end-user has to wait when he/she is retrieving a URL from the i-mode platform. The two graphs in Figures 7a and 7b give a typical overview of the HTTP response time. Both graphs display the total response time together with its constituent parts: the delay in the GPRS network, and the delay in the Internet network. It can be seen from both graphs that both the average and the distribution of the response time in each compartment of the service shows a different behaviour.

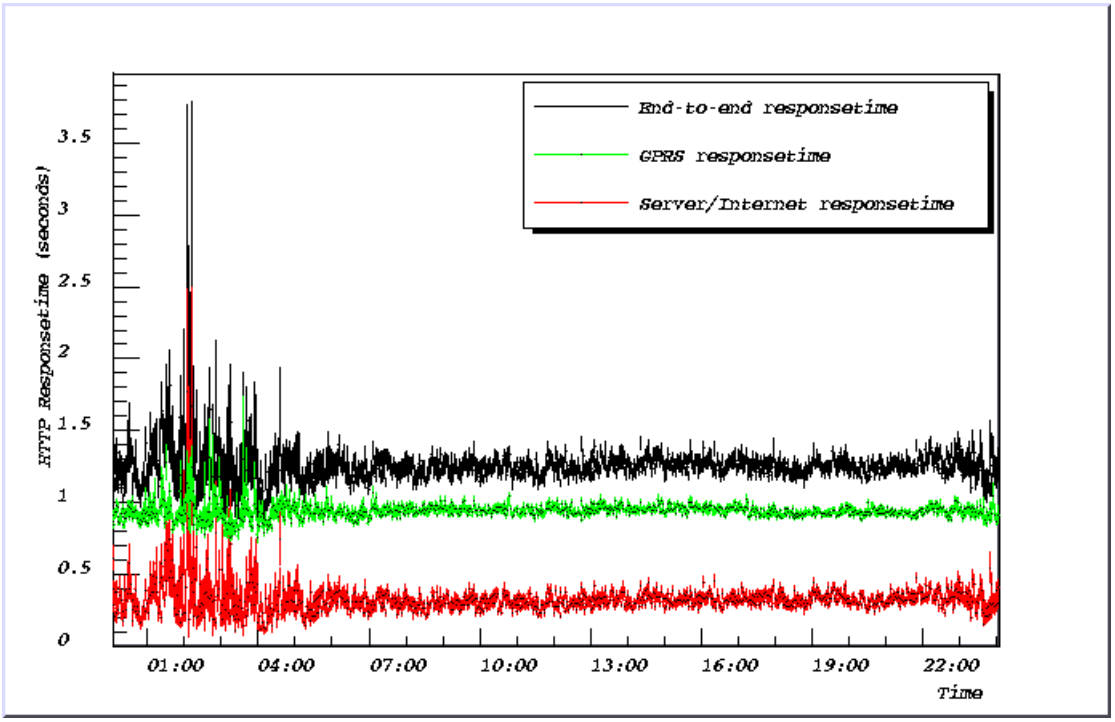

Figure 7a. Plot of the total response time together with its constituent parts (in this case GPRS network and Internet).

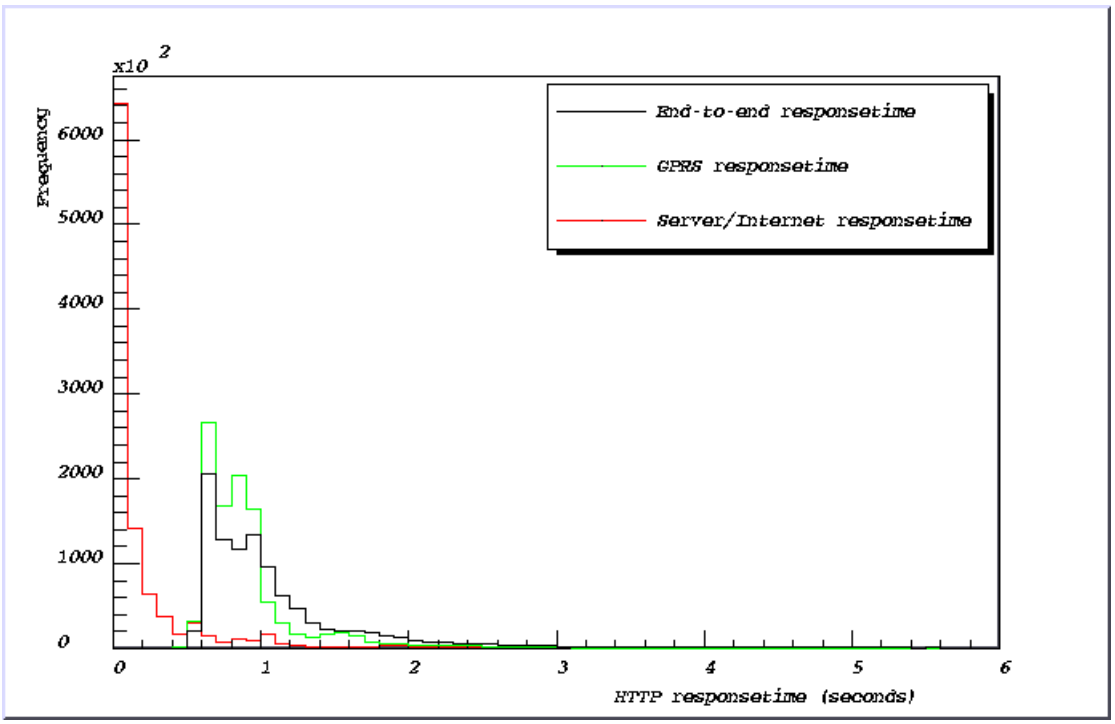

Figure 7b. The distribution of the total response time together with its constituent parts.

Another useful performance metric is displayed in Figure 8 below. This graph displays the correlation between the server part of HTTP response time and the HTTP Request rate of a selected content provider. The correlation is plotted as the 1-minute averages of HTTP

response time as a function of the 1-minute averaged HTTP request rate. This type of plots can give direct insight in the scalability and thus possible sources future problems in the network.

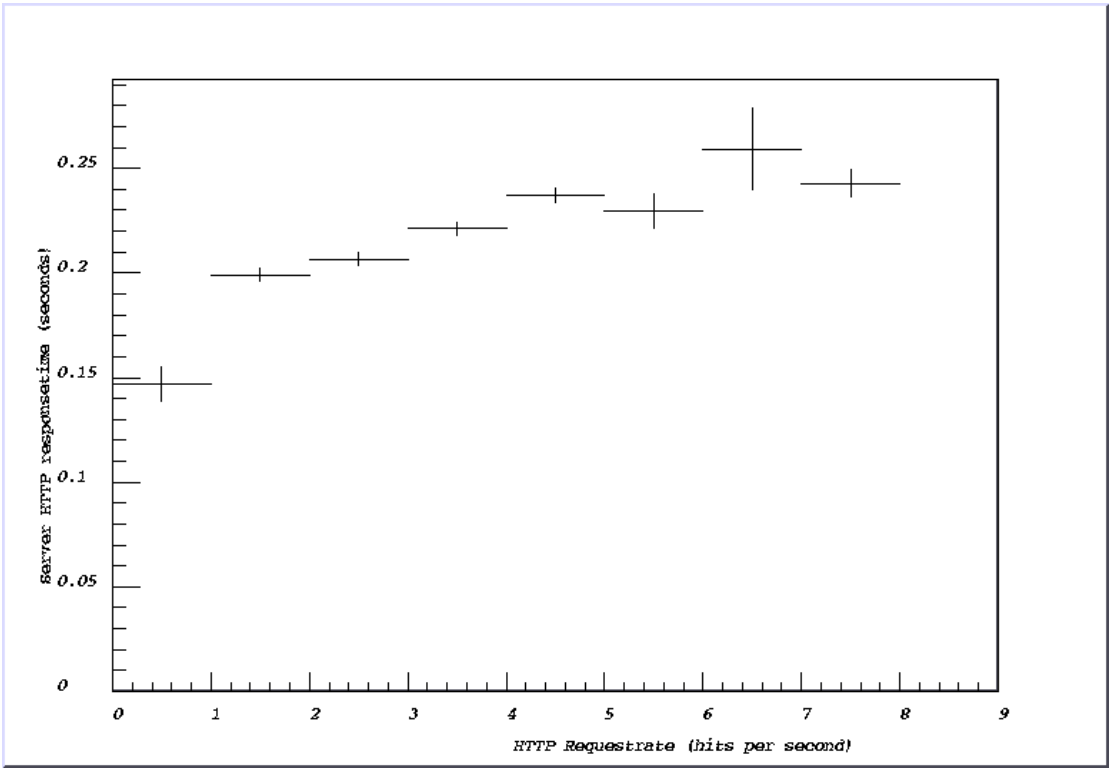

Figure 8. HTTP server responsetime as a function of the HTTP request rate. Plotted for one contentprovider.

5. Conclusions

This paper presents an architecture for capturing, storing, processing and presenting streams of events that carry performance data originating from telecommunications and computer infrastructures. The Multi-Layer Protocol Analyzer (MLPA) is a performance monitoring tool built using this architecture. MLPA uses network analyzer technology to passively monitor network traffic and calculate performance metrics from the correlated traffic. Differences with respect to other monitoring software: MLPA uses a non-intrusive passive monitoring technology in contrast to active polling techniques. In its current form, MLPA is a lightweight solution that can be installed on a standard PC technology server system. The practical applicability of the MLPA has been demonstrated in a variety of practical cases for real customers in the telecom industry.

References

[PCAP] Packet Capture Library, Van Jacobson, Craig Leres and Steven McCanne, all of the Lawrence Berkeley National Laboratory, University of California, Berkeley, CA, website http://www.tcpdump.org

[ROOT2001] René Brun, Fons Rademakers, Suzanne Panacek, Damir Buskulic, Jörn Adamczewski, and Marc Hemberger. The root users's guide. Technical report, CERN, 2001.