6th Workshop on High Speed Networks (HSN '97)
Stuttgart, October 8-9, 1997

Web-Based Internet Traffic Analysis Using Flows

Siegfried Löffler, Paul Christ, Martin Lorang
University of Stuttgart, Germany

<loeffler@cdi.cdi.fr>
<paul.christ@rus.uni-stuttgart.de>
<lorang@ind.uni-stuttgart.de>

Abstract

The term „flow" is being used in at least three different contexts in the Internet environment: It is used to describe traffic for resource reservation protocols like RSVP. Flows are also considered as a unit for traffic switching. Finally, flows are a rather new category in network measurement and analysis.

In our work, we focussed on using flows for traffic measurement and analysis.

This field has become important because of the rapid growth of the Internet and the growing demand for multi-media applications which require high bandwidth network resources. New tools have to be developed for the analysis and measurement of traffic at high line speeds. These tools have to provide the information necessary for network planning and configuration, resolution of congestion problems as well as for user accounting and charging.

In this paper, we describe the use of a flow-based methodology as a means to analyze and monitor traffic. Following the trend to integrate network management technologies into a World-Wide-Web (WWW) framework, a Java based traffic analyzer is presented as a contribution to the Internet Engineering Task Forces (IETF) Real Time Traffic Flow Measurement (RTFM) architecture.

Existing Network Analysis Tools

Many developments have been made in the past to assist the network manager in his tasks. The first were primarily focussed on the management of networking entities, i.e. the change of parameters (routes, settings, etc...) of the switches. The "Simple Network Management Protocol" (SNMP) [3] was developed to provide a standardized interface for controlling the networked devices.

A very basic mechanism for traffic analysis was soon developed using the SNMP: Routers keep state information about the usage of the ports (byte counts) and make them available as SNMP counters. Such counters, when traced over longer periods of time, can be used for a number of purposes. A long-term analysis over the counters permits the network manager to get information about trends for the total load of the network segment. This information allows him to plan a changeover to higher networking speeds or a fragmentation of the network segment before overload effects will occur due to too much traffic on the segment. Shorter-term measurements (i.e. over one day) can be used to gain information about usage characteristics, maximum transfer rates etc.

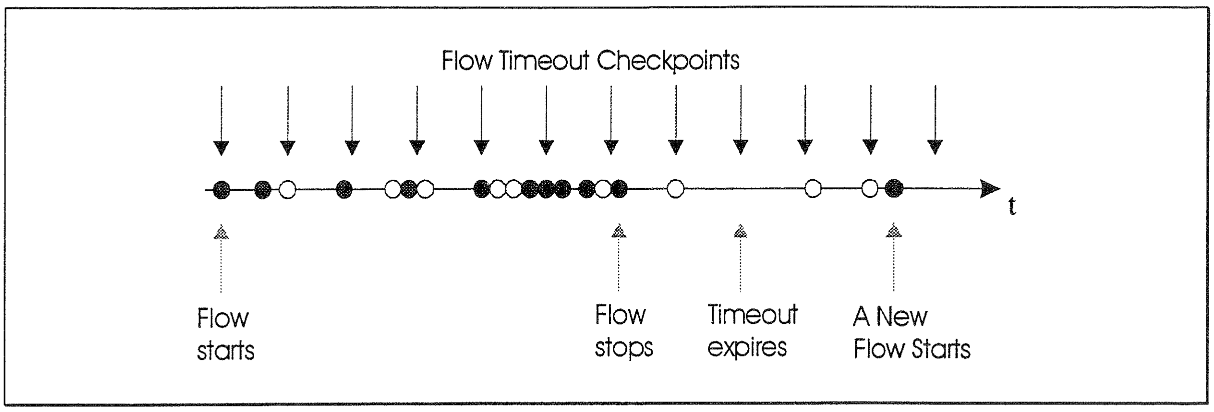


Figure 1: Defining a flow based on the timeout during idle periods. The full circles mark incoming packets that match the flow criteria.

However, it is often the case that displaying the progress of a counter over the time is not sufficient to answer all questions. A first effort to improve measurement possibilities was implemented with the RMON ("Remote Monitoring") standard [4] for network monitoring, which itself uses the SNMP protocol for the transfer of the data. RMON systems consist of special RMON probes that can be put in interesting points of the network to analyze the traffic at those points. The focus is however more on monitoring than on analysis.

Studying the latest products that have been released to the network management market, one will find that most of them are now integrated in a World-Wide-Web framework. However, they still usually are limited to display charts over counters or current network status.

Flows

The idea to aggregate multiple packets on the connectionless-style Internet into flows is nothing new. In [1] Jain presented "Packet Trains" as an alternative model for arrival processes. In [2] Claffy, Braun and Polyzos present the "Flow" methodology for computer network traffic. In this model, a "flow" is defined as a sequence of packets matching certain criteria, exchanged between two entities on a network.
Figure 1 shows the timings that are relevant for this definition of a flow. The circles on the time axis depict the arrival of data packets. The data packets which match the *flow criteria* are shown as full circles. The arrows on top mark the "flow timeout checkpoints". At those points it is checked whether information that matches the *flow criteria* has been received since the last checkpoint. If information was received, the flow is called "*current*" or "*active*". If none of the packets that have been received matched the flow criteria, the flow timeout expires. The flow is then called "*inactive*". Any new packets that match the same flow criteria will then belong to a new flow.

A very interesting aspect of this methodology is that it is completely left open what a flow specification exactly is. A flow specification might for example be given by the condition that source and destination IP addresses for all packets of the flow are identical. It might as well be the condition that all packets are telnet packets, i.e. TCP port 23. By not further specifying what a flow specification is, the methodology itself can be used for a variety of applications.

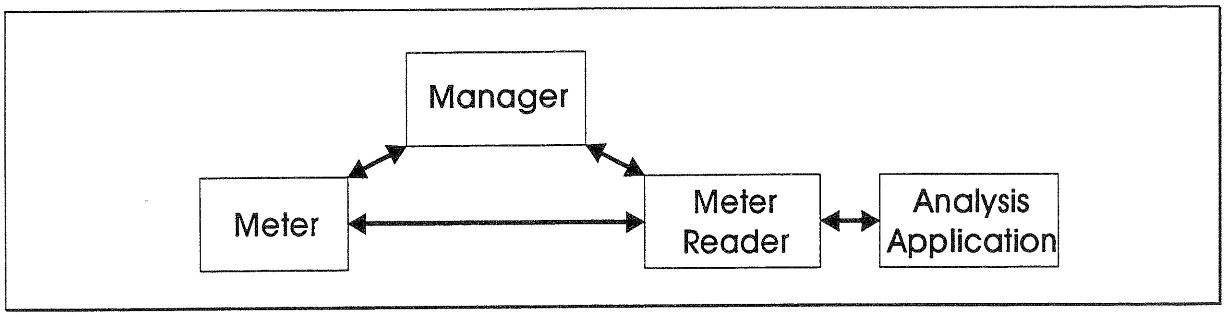


Figure 2: The IETF Reat Time Traffic Flow Measurement Architecture: The measurement or analysis process is separated into several independent applications.

Flows in Traffic Measurement: The RTFM Architecture

So how can flows be used for Network Traffic Measurement? Since quite a while, a working group within the Internet Engineering Task Force (IETF), the so called „Real Time Traffic Flow Measurement" (RTFM) Working Group, has been developing a new architecture for traffic measurement [6]. This architecture was initially designed mainly for accounting issues [5] but has more and more been extended for traffic analysis purposes. Figure 2 depicts the basic structure of the architecture.

The following terms are defined within the RFC and form the architecture:

Manager – An application which configures „Meter" Entities and controls „Meter Reader" Entities. It uses the data requirements of analysis applications to determine the appropriate configurations for each meter and the proper operation of each meter reader.

Meter – The meter is the heart of the traffic flow measurement architecture. Meters are placed at measurement points determined by network operations personnel. Each meter selectively records network activity as directed by ist configuration settings. It can also aggregate, transform and further process the recorded activity before the data is stored. The processed and stored results are called the „**usage data**".

Meter Reader – A meter readers task is to reliably get the usage data from meters and make it available to analysis applications.

Analysis Application – The analysis application finally processes the usage data in order to provide information and reports which are useful for network engineering and management purposes. Examples could be traffic flow matrices, flow rate frequency distributions and traffic volumes sent or received by particular hosts.[1]

The RTFM architecture has been implemented in a set of example applications by Nevil Brownlee at the University of Auckland, New Zealand. In his application suite, the meter (called **NeTraMet**) is a program running on UNIX or DOS hosts, which put the network adapter into the "promiscuous mode" to capture all packets. The captured packets are then aggregated to flows, corresponding to the flow specifications that are written in so called "ruleset files" and are uploaded to the meter by combined manager / meter reader applications (**NeMaC**, **Nifty**, **nm_rc**). Those sample applications are in use in a large number of places all over the world, mainly for accounting purposes, and have shown to work very well.

[1] The architecture was initially specifically designed for accounting purposes. In accounting scenarios, the analysis application corresponds to the program that prints the bills for the customers.

Combining "Web based Management" and the Flow Methodology

Having on the one hand seen the flow based applications for the RTFM architecture by Nevil Brownlee, on the other the trend to integrate all kinds of management and measurement systems into a World-Wide-Web framework to push their acceptance and ease their usage, we thought about a way how measurement and analysis could be brought closer to the user. The main difficulty we experienced when studying this is that it is really *hard to predetermine in which kinds of measurements the user later may be interested in.* It is obvious that users will want to know about how much data has been transferred over a certain link – this data can already provided by the existing tools we saw. We felt unable to predetermine all kinds of questions and issues that users might want to ask about. However, the RTFM working group has experienced the same difficulty in their work, and their framework offers a solution. By focussing on the flow methodology itself and not further defining what kind of flows are interesting (this stays configurable using "rule sets") it is made possible for the user to define new measurement criteria at any time.

Our Contribution: A Web Frontend for RTFM meters written in Java

We therefore decided to work on a web interface for this architecture. For web based programming, several techniques exist. The most common is the use of the "Common Gateway Interface" (CGI). CGI programs are always run on the web server, they are called every time a user sends an HTTP request to the server. Parameters may be passed to the CGI program, and the program can return its results as HTML output. After that output is transmitted to the users web browser, it cannot be modified any more, so this technique is only usable to display static information. Still it is quite easy to write such programs and it will be a field for further work to write basic CGI programs to display static information read from RTFM entities.

A more interesting challenge for web integration however is a second technique: Java[2] programs [8]. Those, in contrary to CGI, are transferred to and then executed within the web browser of the user who accesses the web server. They run on all kinds of different platforms for which a Java Virtual Machine (VM) Interpreter exists. The Java VM is implemented within the Netscape browser, and therefore is available on all major platforms.

Our idea for a new kind of analysis application was to provide the network manager with a tool that allows him to gain insight into the current status of the network. Given that the existing tools were all focussing on traces of byte or packet counters, we felt what would be needed next was some kind of replacement for tools like "tcpdump", which are usually used to see what exactly is happening in real time. We do not think of replacing "tcpdump", we merely want to fill the gap between tools that produce raw dumps of the traffic, and tools are used to generate charts over traffic counters.

For this purpose, we needed to write a program that communicates via SNMP in real time. Since Nevil Brownlees example applications for the RTFM architecture were already well developed and available for a number of network technologies (Ethernet, FDDI, ATM OC3), we decided to focus on writing an application that works with **NeTraMet** - Brownlees RTFM meter implementation. This also has the advantage that we do not have to implement the whole set of programs, but we can start with just implementing the "meter reader" and "analysis application" functionality and use the existing applications for the rest.

Having started to develop in Java, we found that the task was not so easy as it

[2] Java should not be mixed up with „JavaScript" – a scripting language invented by Netscape which is a completely different technique.

might seem at first sight. Java has some very strict security limitations, which do not allow a program inside a web browser to communicate with other machines on the Internet than the web server the program was loaded from or the machine the browser is running on. For management programs this is a severe restriction, since they need to communicate to entities like routers etc that do not fall into those two categories. Additionally, Suns Java Development Toolkit (JDK) did at that time not offer an API for SNMP programming[3].

However, we were lucky to discover a commercial SNMP library[4] which does not only offer a broad variety of SNMP methods, it also solves the security problem by providing a so called "secure applet server" that is running on the web server host and transparently redirects all SNMP requests to any other entity we want to communicate with.

Our final implementation is shown in Figure 3. The applet is displaying information about the current state of the network in an XY-diagram (in more or less the same manner as "nifty", one of the applications by Nevil Brownlee, does it). The X axis shows the flow duration, the Y axis the number of packets seen for the particular flow. Depending on the kind of flow different symbols can be used for the display. The most common use probably will be to use characters to distinguish between different protocols. In the example, an "S" in the XY-diagram represents a secure shell (SSH) flow, the X-axis value for it shows how long the total flow duration was (the time between *flow starts* and *now* (or *flow stop*s in case the flow is no longer active)), the Y axis shows the total of PDUs seen for this flow in both directions. Inactive flows are also shown, but their color is slowly fading to white and they disappear from the graph after a while.

[3] Meanwhile a first experimental Version of such an API, the JMAPI, is available from Sun.

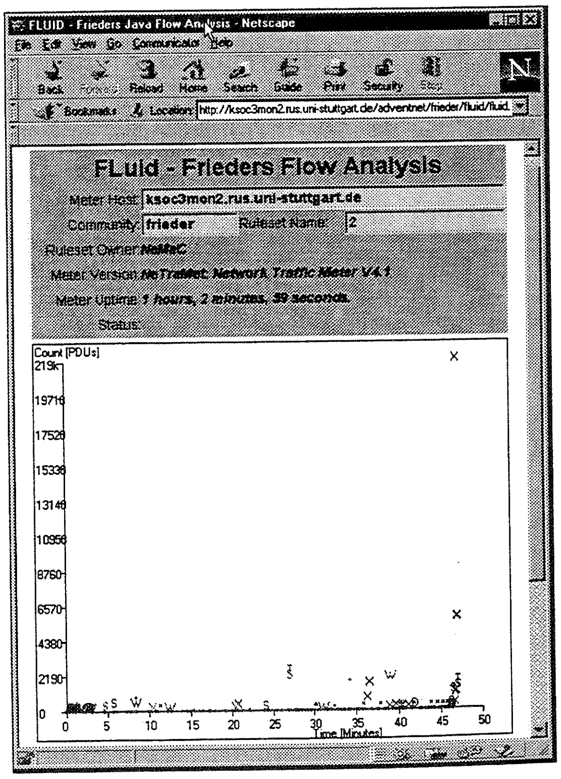
[4] By AdventNet Corp., see http://www.adventnet.com

Conclusions

We found that flows are a very interesting concept for traffic measurement and analysis. The main advantages we see in using flows in this field is that we reduce the amount of data we have to deal with at the earliest possible stage and that thanks to the flow specifications (i.e. the rulesets) the methodology itself does not limit the user by pre-assuming which kinds of measurements he might be interested in.

The well matured RTFM architecture offers a very useful framework for developing flow based measurement and analysis applications.

The web-based effort for network management seems to be very promising to us. Not only because it makes it easy to integrate the tools into an intranet style framework which will make the results


FLUID - Frieders Java Flow Analysis - Netscape

Location: http://ksoc3mon2.rus.uni-stuttgart.de/adventnet/frieder/fluid/fluid.

FLuid - Frieders Flow Analysis

Meter Host: ksoc3mon2.rus.uni-stuttgart.de

Community: frieder Ruleset Name: 2

Ruleset Owner: NeMaC

Meter Version: NeTraMet: Network Traffic Meter V4.1

Meter Uptime: 1 hours, 2 minutes, 39 seconds.

Status:

Figure 3: Our Java Applet („Fluid") displaying Information about current traffic flows.

accessible for a larger community but also because it is no longer needed to install the management software on multiple machines.
We also found that Java is suitable for network management applications. It is fast enough, even if, as in our case, quite a large number of data has to be transferred over the network by the application.
Of the three flow contexts - "Resource Reservation", "Switching" and "Measurement - we focussed our work on the latter. We feel this to be necessary in order to make the flow paradigm eventually applicable to Resource Reservation and Switching, since those will need to use measure traffic for their purposes as well.

References

[1] Raj Jain, Shawn A. Routhier: "*Packet Trains – Measurement and a New Model for Computer Network Traffic*", IEEE Journal On Selected Areas In Communications, Vol. SAC4, No. 6, September 1986

[2] K Claffy, H.-W. Braun, G. C. Polyzos: "*A parametrizable methodology for internet traffic flow profiling*", IEEE JSAC Special Issue on the Global Internet, 1995

[3] M. Schoffstall, M. Fedor, J. Davin, J. Case: "*A Simple Network Management Protocol (SNMP)*", Request for Comments (Standard) RFC 1157 (STD15), Internet Engineering Task Force, May 1990

[4] S. Waldbusser: "*Remote network monitoring management information base (RMON MIB)*", Request for Comments (Experimental) RFC1757, Internet Engineering Task Force, February 1995

[5] D. Hirsh, C. Mills, G. Ruth: "*Internet accounting: Background*" Request for Comments (informational) RFC1272, Internet Engineering Task Force, November 1991

[6] N. Brownlee, C. Mills, G. Ruth: "*Traffic Flow Measurement: Architecture*", Request for Comments (Experimental) RFC2063, Internet Engineering Task Force, January 1997

[7] Siegfried Löffler "*Using Flows for Analysis and Measurement of Internet Traffic*", Diploma Thesis, Institute of Communication Networks and Computer Engineering (IND) at the University of Stuttgart, 1997

[8] Ken Arnold, James Gosling: "*The Java Programming Language*", Addison Wesley Longman, 1996