# Distributed Network Analysis Using TOPAS and Wireshark

Gerhard Münz, Georg Carle
Computer Networks and Internet
Wilhelm Schickard Institute for Computer Science, University of Tuebingen, Germany
Email: {muenz|carle}@informatik.uni-tuebingen.de

***Abstract*—Distributed network analysis deals with the inspection of traffic observed at various locations in the network. The conventional approach is to deploy a full-fledged network analyzer at every observation point, which allows exhaustive examinations, but at the same time is a very costly solution. In this paper, we present an alternative approach using packet data exported by PSAMP and Flexible Netflow devices, such as routers, switches, and monitoring probes. Exported packet records are received by the real-time network analysis framework TOPAS and examined by the open-source network analyzer Wireshark. Monitoring devices are configured with a Monitor Manager in order to export only data needed to achieve the analysis goal. Apart from an architectural description, this paper contains the results of experimental performance evaluations and a discussion on the advantages and limitations of our approach.**

## I. Introduction

Network monitoring is an important means for network administrators for supervision and fault diagnosis. In some cases, simple traffic statistics are insufficient, and deep packet inspection is necessary to trace and understand a certain occurrence or behavior. Network monitoring and analysis on packet level is also deployed by protocol and system engineers in order to test and debug new protocol implementations.

*Wireshark* [1], formely known as *Ethereal* is probably the most popular open-source network analyzer tool. Since the project was started by Gerald Combs in 1998, more than 500 authors have contributed their own pieces of code extending Wireshark's packet and protocol inspection capabilities. For real-time packet capturing, Wireshark makes use of the pcap library (libpcap) [2]. The analysis functions comprise stream reassembly, decoding and dissection of hundreds of protocols, as well as collecting traffic and protocol statistics.

When used for real-time traffic analysis, Wireshark suffers from the limitation that it has to run on the same machine that also performs the packet capturing to supply the data via the libpcap interface. For example, Wireshark can be deployed on a machine that is connected to the monitoring port of a switch or router, yet this configuration requires physical access and proximity to the observation point. On the other hand, Wireshark cannot be used to analyze traffic at remote observation points without any technical gimmicks, such as traffic redirection or the experimental remote capture functionality of WinPcap [3].

In this paper, we present a solution to this problem based on the real-time *Traffic flOw and Packet Analysis System* (TOPAS) [4]. TOPAS integrates a collector which receives monitoring data exported by routers, switches and monitoring probes using Cisco Netflow [5], IPFIX (IP Flow Information eXport) [6] or PSAMP (Packet SAMPling) [7] protocol. Furthermore, TOPAS provides a framework for modules that process and analyze the received monitoring data in real-time. In order to provide an interface to Wireshark, we implemented a module which transforms packet data received from PSAMP and Flexible Netflow [8] exporters into a stream of frames in pcap format. Wireshark is able to read this pcap stream from a Unix pipe and to perform continuous packet inspection and protocol analysis just as if the program was running directly at the observation point.

Capturing and exporting per-packet information cause a significant processing load at the monitoring devices. Also, the resulting amount of packet data must not exceed possible bandwidth limitations on the path to the collector. Appropriate filters can be installed at the monitoring devices to export no more than the data actually needed for the analysis. For this purpose, we developed a Monitor Manager offering a comfortable user interface for creating, editing and deploying configuration data and applying it to the remote monitoring devices using the Netconf protocol [9].

In the next section, we start with a brief overview on the state of the art in distributed network monitoring. Section III gives an introduction to PSAMP and Flexible Netflow and describes the architecture and implementation of TOPAS, the pcap writer module, and the Monitor Manager. Section IV presents the results of our performance evaluation, before we discuss benefits and limitations of our concept in Section V. Section VI concludes this paper.

## II. State of the Art

For the examination of local problems in a small network, monitoring at a single observation point may be sufficient. In these cases, we can use a network analyzer, such as a machine running Wireshark, which is directly connected to the network segment or the monitoring port of a switch or router. In larger networks, it is often necessary to perform simultaneous monitoring at multiple observation points. Therefore, distributed network analysis solutions have been developed, consisting of multiple distributed network analyzers that send their analysis results to a centralized management consol. Examples for high-end commercial products are *ClearSight*

978-1-4244-2067-4/08/$25.00 ©2008 IEEE.

*Distributed* [10] and *WildPackets OmniAnalysis Platform* [11]. These systems are very powerful with respect to the offered analysis capabilities, but they are very costly as well since a full-fledged network analyzer has to be deployed at every observation point.

The Remote Network Monitoring (RMON) Management Information Base (MIB) [12] enables distributed network analysis using monitoring devices of different manufacturers in a standardized way. Switches and routers supporting RMON generate traffic statistics that can be queried using SNMP (Simple Network Management Protocol). Fully instrumented RMON devices are able to capture packets matching a given filter as well. A remote analyzer can access the corresponding packet data from a MIB group and use it for deep packet inspection or protocol dissection. Thus, distributed network analysis is possible at reasonable costs thanks to the separation of monitoring and analysis. However, RMON is not adapted for real-time network analysis because of the restricted real-time capabilities of SNMP.

In contrast to these state of the art methods, our approach is based on the novel packet monitoring capabilities offered by PSAMP and Flexible Netflow which are introduced in the next section.

## III. Analyzing Packet Records with Wireshark

In this section, we give a brief introduction to PSAMP and Flexible Netflow before presenting the architecture and implementation of our distributed network analysis system.

### A. PSAMP and Flexible Netflow

The PSAMP working group at the IETF has been developing techniques for selecting individual packets at an observation point and exporting packet data to a remote analyzer [13]. Several filters and sampling mechanisms have been standardized [14] enabling deterministic as well as probabilistic packet selections. For targeted packet inspection or protocol dissection, packet filters based on packet header fields can be applied, for example in the case that only packets directed to a specific destination or with a specific port number are of interest.

PSAMP [7] makes use of the IPFIX protocol [6] to export packet records including header and payload information of the selected packets. Similar to flow records, a packet record contains a time stamp that indicates when the packet was observed, and a set of packet header fields. In addition, a packet record may include continuous sections of packet data of variable length, e.g. the first 100 octets of IP payload or even the entire packet. As the record structure can be defined in a flexible way using templates, it is possible to only export those parts of a packet which are needed for the analysis.

Depending on the transport protocol (UDP, TCP, or SCTP), TLS (Transport Layer Security) [15] or DTLS (Datagram Transport Layer Security) [16] can be deployed to authenticate and encrypt the exported packet data. This is an important property since it prevents data manipulation and eavesdropping between the exporter and the analyzer, and enables secure transport of packet data through an untrusted network.

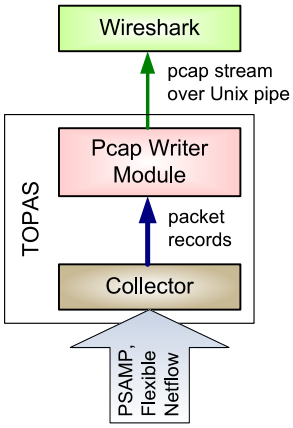


Fig. 1. TOPAS framework and Wireshark

The newest version of Cisco Netflow, called Flexible Netflow [8], implements some of the PSAMP concepts and enables the export of packet records. At the moment, Flexible Netflow only supports a limited number of packet selection options and deploys Netflow.v9 [5] instead of IPFIX protocol for exporting packet records. However, we expect that Cisco will support IPFIX and PSAMP specifications as soon as the standardization has been finished.

### B. Running Wireshark in the TOPAS Framework

TOPAS [4] was originally developed in the European project *Diadem Firewall* [17] where it has been deployed for real-time attack and anomaly detection based on flow records. To do so, a collecting process is receiving monitoring data from IPFIX and Netflow.v9 exporters and passing it to one or more detection modules which perform different kinds of attack and anomaly detection algorithms.

In order to run Wireshark within TOPAS, we extended the collector to receive and process packet data. As mentioned, PSAMP and Flexible Netflow use the IPFIX and Netflow.v9 protocols respectively, thus no changes had to be made to the protocol stack. Yet, we developed a pcap writer module for TOPAS that transforms packet records into frames in pcap format [2]. The module prepends a layer 2 header, as layer 2 information is usually not included in the exported data, as well as the mandatory pcap header which contains the observation time stamp from the packet record. The assembled pcap frame is then passed to Wireshark through a Unix pipe. Figure 1 illustrates the architecture of the system.

Processing the pcap stream from the pipe, Wireshark shows the decoded packet and protocol information just as if it was running at the observation point. Note that an inevitable but small delay is introduced by the data transport from the exporter to the collector, yet the displayed timing information corresponds to the observation time as included in the packet record. Furthermore, packet losses may occur due to insufficient bandwidth between the exporter and the collector, or limited processing resources of the analyzer machine running

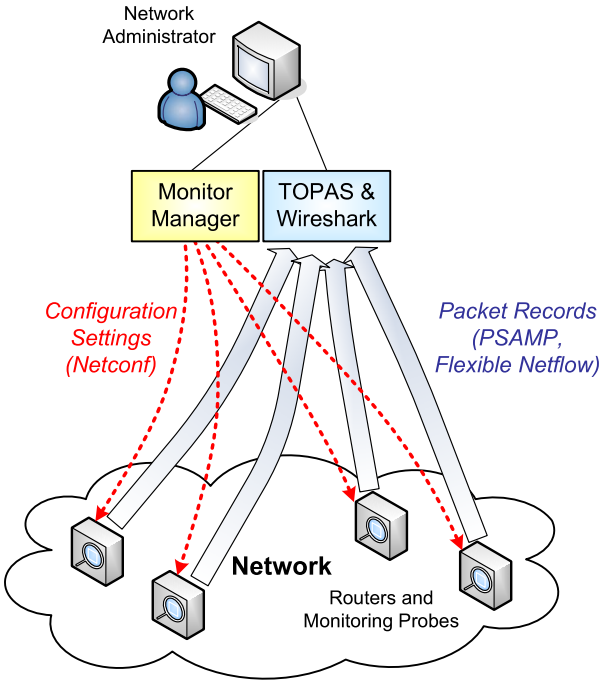


Fig. 2. Deployment example

TOPAS and Wireshark. Yet, such packet losses are recognized by the collector thanks to message sequence numbers, which inhibits misinterpretations of the analysis results.

### C. Monitor Manager for IPFIX/PSAMP Devices

Packet inspection and protocol dissection are usually deployed on specific packet streams or connections, for example, on those directed to a particular server or established between two given end-points. If deployed locally, Wireshark performs the corresponding packet classification and filtering itself or with help of the filtering capabilities of the pcap library. When analyzing exported packet records, packet filtering can already take place at the monitoring device in order to reduce the amount of data sent to the collector. Apart from saving bandwidth, early filtering also reduces the processing load at the exporter and collector. These two aspects are of particular importance in high-speed networks, where unfiltered packet monitoring would require enormous resources or result in numerous packet losses.

In [18], we presented a solution for remote configuration of IPFIX and PSAMP monitoring devices based on the Netconf protocol [9]. As a frontend, we have now implemented a Monitor Manager which offers a comfortable user interface to create, edit and deploy configurations on distributed monitoring devices. As illustrated in Figure 2, the Monitor Manager complements TOPAS and Wireshark with the capability to configure the capturing and export of packet information according to the requirements of the network analysis.

Currently, our software monitoring probe Vermont [19] is the only PSAMP implementation that supports remote configuration with Netconf. The Netconf agent running on the probe is based on the EnSuite framework [20] which provides the Netconf protocol functionalities. A broader deployment of our monitor management and configuration concept requires that Netconf is supported by more IPFIX/PSAMP and Netflow exporters. In addition, a device-independent data model is needed for configuring devices of different manufacturers in a unified way. Therefore, we are currently working on the standardization of a configuration data model for IPFIX and PSAMP devices [21] within the IETF.

## IV. Performance Evaluation

In [4], we evaluated the performance of TOPAS with respect to the maximum number of IPFIX packets and flow records that can be processed without losses, depending on the number of active detection modules. With one active module, we successfully tested rates of 20,000 IPFIX packets and 190,000 records per second. For evaluating the performance of the TOPAS/Wireshark setup presented in this paper, we conducted additional experiments using PSAMP data. In the test setup, Vermont captured traffic from a monitoring port of a Gigabit Ethernet switch and exported packet records containing a variable length field with the first 128 bytes of each captured packet (or less if the packet was shorter), and a timestamp indicating when the packet was observed. An exported PSAMP packet included a maximum of 10 packet records in order to avoid IP fragmentation (transport protocol was UDP). Vermont and TOPAS were running on two dual-processor Linux PCs.

The monitored traffic was generated by replaying pcap trace files from two different public repositories: the repository of the Enterprise Tracing Project of the Lawrence Berkeley National Laboratory (LBNL) and the International Computer Science Institute (ICSI) [22], and the Traffic Measurement Data Repository of the University of Twente [23]. The first repository contains anonymized packet header traces (up to transport header) captured at the LBNL from which we selected a trace file recorded on a weekday between 8am and 9am. The average packet rate was about 880 packets per second with a maximum of 1,556 packets per second. As expected, this quite low packet rate did not represent a real challenge for our implementation; TOPAS and Wireshark were able to process all packet records without any losses.

From the second repository, we chose a trace file captured on a weekday at 5pm at a Gigabit link connecting a college with more than 1,000 students and staff members to the Dutch academic and research network. The average packet rate was 3,850 packets per second with a maximum of 4,981 packets per second. In multiple tries, the ratio of lost packets stayed below 0.05 percent. This is acceptable since in practice, one would not be interested in applying deep packet inspection to the entire traffic, but only specific packet streams. By configuring appropriate filters at the monitoring device, TOPAS and Wireshark would be operating at a packet lower rate without any losses.

## V. Discussion

Our approach allows profiting from Wireshark's extensive packet inspection and protocol dissection capabilities for distributed network analysis, which represents an interesting alternative to the deployment of multiple distributed network analyzers as done by the commercial products presented in Section II. In comparison, our approach is supposed to be less expensive since it partly builds on existing network devices, such as routers and switches supporting PSAMP or Flexible Netflow. If necessary, additional monitoring probes can be installed, yet at much lower costs than full-fledged network analyzers. The fact that we draw on existing and upcoming standards is another advantage with regard to cost-effective deployment. Hence, our approach represents a useful and economic solution for large network operators running millions of Internet access lines.

The major drawback of our solution is that in high-speed networks, we need to restrict the traffic analysis to specific packet streams. Otherwise, the amount of monitoring data risks exceeding the available bandwidth between the exporter and the collector, resulting in packet losses or unacceptable high delays. In this respect, distributed network analyzers have the advantage that they only export reports with analysis results which can be much smaller than the examined packet data.

Apart from limiting the number of exported packet records, we can also reduce the size of each record to decrease the overall data volume. This can be achieved by exporting only selected header fields and payload sections. Thus, fields which are not required for the analysis can be omitted. Moreover, reduced size encoding as specified in [6] can be applied to encode small integer and float values with fewer octets than the original field length. For example, low port numbers (0 to 255) can be encoded within a single octet instead of two.

Concluding the discussion, let us denote that the proposed architecture and implementation is not limited or specific to the utilization of Wireshark. There are many other programs and tools which are able to receive and process pcap data generated by the pcap writer module. For example, we have successfully deployed Snort IDS within the TOPAS framework in order to perform signature detection on packet records [24].

## VI. Conclusion

We presented an architecture and implementation that enables using the popular network analyzer Wireshark to inspect packet data exported by routers, switches, and monitoring probes. We described the system components and their realization, evaluated the system performance with pcap trace files captured in two different networks, and discussed the advantages and limitations of our approach compared to conventional distributed network analysis systems.

Our solution broadens the field of application of Wireshark and overcomes its conceptual limitation of only being able to inspect local traffic. As PSAMP is in the final phase of standardization, and as Flexible Netflow is already available on the market, we expect capturing and exporting packet information to become a commonplace functional extension of devices used for flow information export today. If this assumption becomes true, using the new capabilities for distributed network analysis will be a straightforward and inexpensive alternative to conventional systems consisting of multiple distributed and full-fledged network analyzers.

## Acknowledgments

We gratefully thank our students Lothar Braun, Nico Weber, and Maximilian Hütter for contributing to the design and implementation of TOPAS and the Monitor Manager.

## References

[1] Wireshark Homepage, http://www.wireshark.org, 2008.
[2] Libpcap Homepage, http://www.tcpdump.org, 2008.
[3] WinPcap Homepage, http://www.winpcap.org/, 2008.
[4] G. Münz and G. Carle, "Real-time Analysis of Flow Data for Network Attack Detection," in *Proc. of IFIP/IEEE Symposium on Integrated Management (IM 2007)*, Munich, Germany, May 2007.
[5] B. Claise, G. Sadasivan, V. Valluri, and M. Djernaes, "Cisco Systems NetFlow Services Export Version 9," RFC 3954 (Informational), Oct. 2004.
[6] B. Claise, S. Bryant, G. Sadasivan, S. Leinen, T. Dietz, and B. H. Trammell, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information," RFC 5101 (Proposed Standard), Jan. 2008.
[7] B. Claise, J. Quittek, and A. Johnson, "Packet Sampling (PSAMP) Protocol Specifications," Internet-Draft, work in progress, draft-ietf-psamp-protocol-09, Dec. 2007.
[8] Cisco Systems, "Introduction to Cisco IOS Flexible NetFlow," White Paper, Jun. 2008. [Online]. Available: http://www.cisco.com
[9] R. Enns, A. Bierman, K. Crozier, T. Goddard, E. Lear, P. Shafer, S. Waldbusser, and M. Wasserman, "NETCONF Configuration Protocol," RFC 4741 (Standards Track), Dec. 2006.
[10] ClearSight Networks, Inc. Homepage, http://www.clearsightnet.com, 2008.
[11] WildPackets, Inc. Homepage, http://www.wildpackets.com, 2008.
[12] S. Waldbusser, "Remote Network Monitoring Management Information Base," RFC 2819 (Standard), May 2000.
[13] N. Duffield, D. Chiou, B. Claise, A. Greenberg, M. Grossglauser, P. Marimuthu, J. Rexford, and G. Sadasivan, "A Framework for Packet Selection and Reporting," Internet-Draft, work in progress, draft-ietf-psamp-framework-12, Jun. 2007.
[14] T. Zseby, M. Molina, N. Duffield, S. Niccolini, and F. Raspall, "Sampling and Filtering Techniques for IP Packet Selection," Internet-Draft, work in progress, draft-ietf-psamp-sample-tech-10, Jun. 2007.
[15] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1," RFC 4346 (Proposed Standard), Apr. 2006.
[16] E. Rescorla and N. Modadugu, "Datagram Transport Layer Security," RFC 4347 (Proposed Standard), Apr. 2006.
[17] Diadem Firewall Homepage, http://www.diadem-firewall.org, 2007.
[18] G. Münz, A. Antony, F. Dressler, and G. Carle, "Using Netconf for Configuring Monitoring Probes," in *Proc. of IEEE/IFIP Network Operations & Management Symposium (NOMS 2006), Poster Session*, Vancouver, Canada, Apr. 2006.
[19] R. T. Lampert, C. Sommer, G. Münz, and F. Dressler, "Vermont - A Versatile Monitoring Toolkit for IPFIX and PSAMP," in *Proc. of IEEE/IST Workshop on Monitoring, Attack Detection and Mitigation (MonAM 2006)*, Tuebingen, Germany, Sep. 2006.
[20] EnSuite Homepage, http://ensuite.sourceforge.net, 2008.
[21] G. Münz and B. Claise, "Configuration Data Model for IPFIX and PSAMP," Internet-Draft, work in progress, draft-muenz-ipfix-configuration-04.txt, Feb. 2008.
[22] LBNL/ICSI Enterprise Tracing Project, http://www.icir.org/enterprise-tracing/Overview.html, 2008.
[23] R. van de Meent, "M2C Measurement Data Repository," University of Twente, Enschede, The Netherlands, M2C Deliverable D1.5, Dec. 2003.
[24] G. Münz, N. Weber, and G. Carle, "Signature Detection in Sampled Packets," in *Proc. of IEEE Workshop on Monitoring, Attack Detection and Mitigation (MonAM 2007)*, Toulouse, France, Nov. 2007.