Exploiting Hybrid Measurements for Network Troubleshooting

Stefano Traverso[a], Edion Tego[b], Eike Kowallik[c], Stefano Raffaglio[c],
Andrea Fregosi[c], Marco Mellia[a], Francesco Matera[b]
[a] DET, Politecnico di Torino, Italy – {lastname}@tlc.polito.it
[b] Fondazione Ugo Bordoni, Roma, Italy – {lastname}@fub.it
[c] Fastweb, Milano, Italy – {firstname.lastname}@fastweb.it

***Abstract*—Network measurements are a fundamental pillar to understand network performance and perform root cause analysis in case of problems. Traditionally, either active or passive measurements are considered. While active measurements allow to know exactly the workload injected by the application into the network, the passive measurements can offer a more detailed view of transport and network layer impacts. In this paper, we present a hybrid approach in which active throughput measurements are regularly run while a passive measurement tool monitors the generated packets. This allows us to correlate the application layer measurements obtained by the active tool with the more detailed view offered by the passive monitor.**

The proposed methodology has been implemented following the mPlane reference architecture, tools have been installed in the Fastweb network, and we collect measurements for more than three months. We report then a subset of results that show the benefits obtained when correlating active and passive measurements. Among results, we pinpoint cases of congestion, of ADSL misconfiguration, and of modem issues that impair throughput obtained by the users.

I. Introduction

Since the early days of the Internet, network measurements have always constituted a pillar to understand the behavior of the network and to find the possible root cause in case problems are highlighted. On the one hand, active measurements have been defined to gauge end-to-end delay, e.g., via `ping`, path properties, e.g., via `traceroute`, or to measure the application layer throughput, e.g., via a speed-test. On the other hand, passive methodologies have been defined to observe details and correlate messages generated by the application with transport and network layer protocols. Tools like TCPDump [1] or Wireshark [2] are considered the swiss-army knives for troubleshooting sessions, and more advanced passive monitors are available to automatically extract information about the status of the network [3], [4], [5].

Active measurements allow to exactly define the workload the network is subject to, and to measure the desired quantity. In the case of a speed-test for instance, a client is instrumented to download a well defined amount of data from a server, allowing to compute the application layer throughput as perceived by the end-user.

Passive measurement tools instead observe the packets flowing on a link, and extract information from protocol headers. They allow thus to potentially extract information without injecting any data in the network, and often allow to get a more detailed view of the network status. For instance, it is possible to passively observe the retransmissions faced by TCP, and to monitor the RTT [6], but only in case some client actually generates traffic toward the targeted server.

In this paper we demonstrate the feasibility of having a hybrid approach, in which active measurements are used to generate the desired traffic, while passive measurements augment the information exposed to the analyst. We define a simple speed-test use case: active probes scattered in an ISP network are instrumented to periodically download (upload) via FTP some predefined files from (to) a server, logging the application layer throughput. The machine hosting the FTP server runs also Tstat [3], a passive sniffer that extracts transport-layer information about the TCP flows.

The whole system has been implemented following the guidelines defined in the mPlane architecture [7]. mPlane is a FP7 Integrated Project that aims at defining a measurement plane for the Internet. It defines an architecture to allow the collection of measurements coming from mPlane compliant probes. Measurement results are then stored into repositories, where they can be correlated and processed. Thanks to this flexibility, the mPlane architecture allows to easily glue together for instance active measurements results with passive measurements, i.e., to get the hybrid measure we are interested in.

We deployed active probes in more than 30 locations scattered in the operational country-wide network of the Italian operator Fastweb. Active probes are connected to the network using the same technology offered to customers, e.g., an ADSL or an FTTH access link, while the FTP server has been installed in a test-plant in Milan. Active probes are then instrumented to periodically run speed-tests by sending and receiving files of predefined size, measuring the application layer throughput. Tstat runs on the server, and create a log for each TCP flow. We run the system for more than three months, observing the time series of application throughput. When some unexpected behavior is observed, information is correlated with metrics extracted by Tstat, using domain knowledge to find the root cause of the anomaly.

Results show the benefit of complementing active measurements with fine grained details obtained from passive measure-

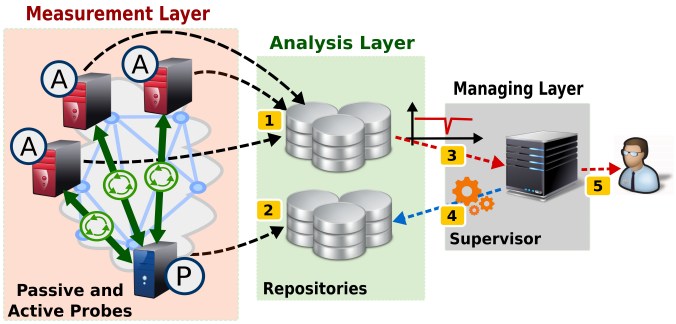

Figure 1. The distributed measurement platform inspired to mPlane for ISP network troubleshooting, and its workflow. The green double-headed arrows correspond to active measurements performed by active probes (depicted with "A") towards the passive probe (depicted with "P"). The black arrows correspond to measurement data that are exported from the probes to the repositories. The red arrows correspond to anomaly notification reports. The blue arrow depicts the request made by the supervisor to trigger deeper data analysis.

ments. For instance, we are able to pinpoint congestion at the access link for some lines (curiously, when the PC accesses the Fastweb network using a Virtual Leased Line instead of when the home gateway is directly connected to a Fastweb DSLAM), problems on a home gateway that causes packet loss, impact of ADSL physical impairment, etc.

We believe the hybrid approach defined in this paper shows promising benefits and we plan to integrate it with more complicated reasoning processes, e.g., by automatically correlate features which show abrupt changes, by triggering further measurements when anomalies are detected, and by making the root cause analysis as much as possible automated.

The remainder of the paper is structured as follows. Sec. II provides a general description of the mPlane architecture and its components. In Sec. II-A we describe how we instantiated mPlane to address the problem of troubleshooting the network, and we resume the steps which compose the measurement procedure. Sec. II-B and Sec. II-C describes IQM and Tstat, respectively, the active and passive probes we employed in our test-bed. Sec. III provides details about the test-bed and the dataset we obtained from it. In Sec. IV we presents three classes of anomalies we have been able to detect during our experiments. Finally, Sec. V discusses the related work and Sec. VI concludes the paper.

II. System Description

This section describes the distributed platform we employ to perform measurements with the aim of performing network measurements in wireline access networks. The platform is an instance of the mPlane architecture described in [7], [8]. The mPlane monitoring system is composed by three different entities, namely *probes*, *repositories* and *supervisors*. They interoperate thanks to a standard protocol, and are logically organized into three main layers, as depicted in Fig. 1:

- **The Measurement Layer:** it consists of a set of probes located at several vantage points within the administered network, and which typically generate large amounts of measurement data. The system supports different kinds of measurements: i) *active*, for instance, the output of a simple `traceroute`, or ii) *passive*, e.g., the packet level trace of traffic flowing on a link. Measurement campaigns may be triggered on demand, with results returned as soon as the measurement is done, or be run continuously, with results that are periodically exported to limit the storage utilization at the probe.
- **The Analysis Layer:** this layer consists of *repositories* which collect and aggregate data generated by probes. Apart from the storage capacity, the Analysis Layer is provided with a set of analysis routines which process the data imported from the probes. Such processing may involve filtering, grouping, aggregation of the raw data imported from the probes. The result is a higher level of aggregated, more refined measurements, and queryable.
- **The Management Layer:** this layer consists of the *Supervisor*, a smart component which orchestrates probes and repositories. In particular, it executes high level procedures and algorithms. The supervisor may trigger alarms when unexpected phenomena are observed, and thus consider or initiate additional operations, e.g., perform on-demand measurement to drill down to the root cause of an anomaly.

In the following we present how we instantiated this generic architecture for the goal of detecting QoS degradations at the network access.

A. The Workflow

In general, anomaly detection is a continuous process. The supervisor instructs the probes to periodically perform measurements, passively or actively, and, thanks to correlation algorithms available at the repository, it compares measured features to baseline parameters. Periodically generated results are then forwarded by the supervisor to the analyst who can then take further actions.

The numbers in Fig. 1 refer to the steps which compose the workflow for the detection of anomalies. Each operation is mapped to the proper measurement component.

Step 1 - active probing: Active probes continuously perform tests i) to monitor specific network parameters (e.g., they can perform `traceroute` towards given nodes), or ii) to resemble the behavior of actual users (e.g., they can emit requests for web-pages containing YouTube videos). Measurement results are transferred then to the repository.

Step 2 - augmenting information through passive monitoring: Passive probes run in parallel to active tests (for instance, on the same machine running the active measurements) and complement them with a continuous collection of traffic captures. The resulting measurement data is also periodically transferred to the proper repository.

Step 3 - detection of anomalies: Once measurement data is available at the repository, a set of selected features is sampled and aggregated into timeseries, and analysis modules embedding anomaly detection routines are run on them. As

soon as a sudden change in the timeseries is detected, an alert is raised to the supervisor.

Step 4 - correlating multi-source measurement data: When alarms about unexpected degradations are detected, the supervisor runs correlation analysis (e.g., Factor Analysis) to investigate which features, or classes of features, show a similar abrupt change. Correlated features are then compared against a catalogue of known anomaly patterns, and if a match is found, an alert is raised to the supervisor.

In this paper we focus on a simple scenario in which we aim at detecting QoS degradations. More precisely, the system verifies that the available bandwidth parameters offered by the physical access link of the ISP customers are actually respected. Following the workflow in Fig. 1, active probes measure the application layer throughput on the path towards a given server. This is done by simply downloading (uploading) a file from (to) a central FTP server. Results of the speed-tests are pushed to a repository running a standard SQL database (Step 1). At the same time, the passive probe installed on the FTP server periodically exports its logs to the repository (Step 2). Then, the supervisor continuously looks for degradations in the throughput measurements, and when a glitch is found (Step 3), it queries the TCP-level statistics in the passive logs. The supervisor then runs correlation and classification algorithms to guess the cause behind throughput degradation, and reports the most probable cause(s) to the network administrators and/or to the Service-Level Agreement (SLA) certifier (Step 5).

In the following we provide a brief description of the active and passive probes we employ. Sec. III describes the deployment of our prototype and the dataset of measurements we obtained from it.

B. Active Probes

As depicted in Fig. 2, we rely on the Internet QoS Measurement (IQM) probe developed by Fastweb. IQM probe consists of several components glued together by a modular architecture. Each component is oriented to a specific end-to-end QoS measurement task. More specifically, the current version of IQM can support the following active measurements:

- Running speed-tests to measure the available bandwidth on a network path. This is done by embedding an FTP client which uploads (downloads) files to (from) an FTP server.
- Performing network connectivity tests using `ping`.
- Performing network tomography tests via `traceroute`.
- Using a headless browser (e.g., PhantomJS) to gather HTTP/HTTPS performance metrics, and to run QoE tests of popular content providers and services such as YouTube.

Furthermore, the IQM probe offers measurement scheduling capabilities based on crontab, exports data in several different formats, and provides the user with a simple web graphical interface.

As shown in Fig. 2, depending on the kind of meaurements, the IQM probe may require to involve both the client and the server (e.g., FTP file transfers), or the client only (e.g., `traceroute`).

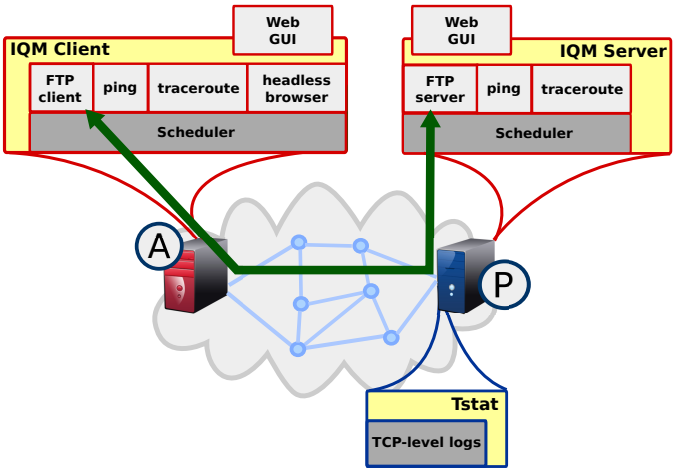

Figure 2. Schematic view of the deployment, and the interaction between active and passive probes (only one active probe is depicted here for clarity). The green double-headed arrow corresponds to the example in which FTP file transfers are performed between the IQM client and server.

C. Passive Probes

The passive probe is co-located with the IQM server (as shown in Fig. 2). It runs Tstat [3] an open-source traffic monitoring tool that analyses packets exchanged with the FTP server. Tstat is a powerful monitoring tool that rebuilds TCP flows, and reports more than 100 indexes, among which the average and standard deviation of RTT, and the number of retransmitted TCP segments by the server.[1]

Recently, Tstat has been modified to be mPlane-compliant: following the mPlane standard protocol, it can be instrumented to start captures on demand, to consider different kinds of traffic, as well as to periodically export its output logs through several different bulk transfer applications.

III. Deployment and Dataset

As a preliminary testbed, we consider the case in which IQM probes periodically perform FTP-like file transfers, running, de facto, speed-tests. The IQM server was installed within Fastweb test-plant in Milan. On the same machine we installed Tstat to monitor and log all incoming TCP/UDP connections. More than 30 IQM probes were uniformly distributed within Fastweb edge network, taking care of considering as many portions of the network as possible. To simulate the QoS experienced by Fastweb customers, probes access the network with the same technologies in Fastweb's offer: three configurations for ADSL (U-1Mbps/D-16Mbps, U-1Mbps/D-12Mbps and U-0.5Mbps/D-8Mbps) and one for FTTH (U-10Mbps/D-10Mbps). Each IQM probe was instrumented to run a 10 second long speed-test every 4 minutes.

We run the testbed for three months, since February 1st to April 30th 2014, thus obtaining a total dataset of 1.2 million speed-test reports produced by the IQM probes, and to as many TCP entries in the log generated by Tstat.

[1]See http://www.tstat.polito.it for a complete list.

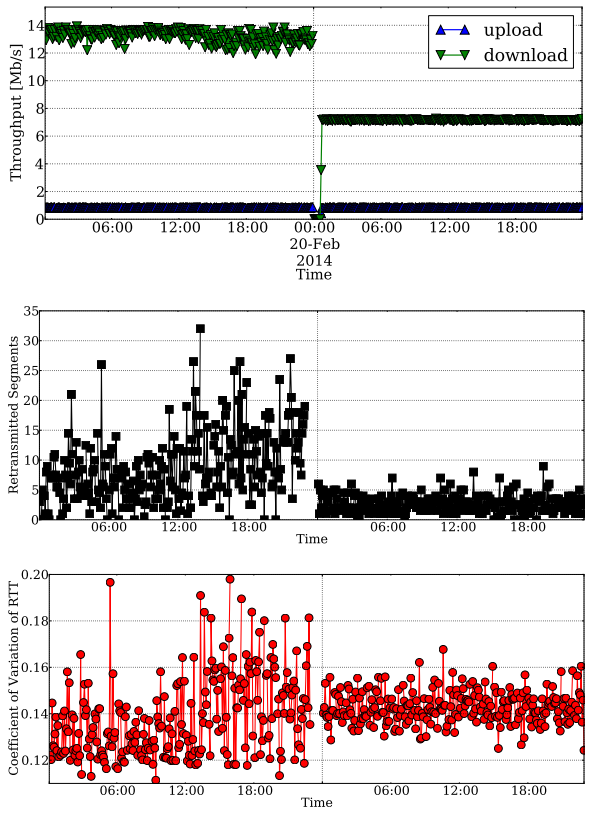

Figure 3. Evolution of time of the throughput measured by an active probe (top), the number retransmitted segments (center), and the coefficient of variation of the RTT (bottom). U-1Mbps/D-16Mbps ADSL probe.

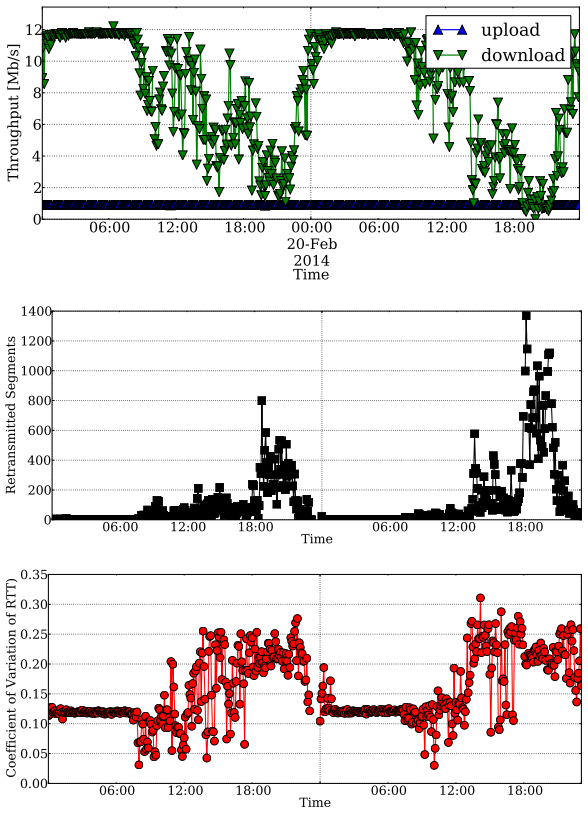

Figure 4. Evolution over time of the throughput measured by an actibe probe (top), the number of retransmitted segments (center), the coefficient of variation of the RTT (bottom). U-1Mbps/D-12Mbps ADSL probe.

IV. Results

In this section we illustrate some examples of anomalies detected by our system. We also show that by correlating the results obtained by the active tests with those gathered from the passive probe, we can pinpoint the cause(s) behind the glitches.

By analyzing our dataset, we could identify three main classes of anomalies. We report them in the following.

A. Low SNR on ADSL lines

In top plot of Fig. 3 we report the evolution over time of the throughput measured by an active probe accessing the ISP network through a U-1Mbps/D-16Mbps[2] ADSL interface for a period of two days. Observe that the download throughput curve appears to be noisy during the first day, while after midnight, the ADSL line was re-calibrated to U-1Mbps/D-8Mbps. Since then, speed-test measures are much more stable over time. By correlating such output with the statistics provided by Tstat, we could notice during the first day, a fairly large rate of retransmitted segments in the flows[3] (center plot), and a constant coefficient of variation of the RTT (bottom plot). The absence of evident day-night patterns let us exclude that this situation might be due to network congestion, since this typically emerges only during peak periods (see the next case).

The most probable cause of this anomaly is the presence of a low signal-to-noise ratio (SNR) at the physical link, which can lead to large bit error rate (BER). Losses due to noise then causes TCP congestion control to (randomly) slow down the download. The confirmation of this hypothesis is given by the second half of the plots in Fig. 3, when the ADSL modem automatically tunes the ADSL downlink capacity to improve the SNR, i.e., negotiating 8Mb/s instead of 16Mb/s, thus considerably reducing the packet loss rate, and making RTT measurement more stable.

B. Congestion in the Network

As before, top plot of Fig. 4 reports the evolution over time of the throughput measured by an active probe (U-1Mbps/D-12Mbps ADSL interface in this case). During the 48h observation window, a clear degradation of the available throughput is detected. We notice that no degradation is observable during the night, i.e., when the network is typically lightly loaded. Conversely, during peak time available capacity decreases. This suggests that there might be some congestion

[2]We remark that the throughput reported in the plots is below the nominal bandwidth value since it is measured at application level, therefore, netting the overheads of transport, network, and data-link layers.

[3]Tstat detects retransmissions in TCP flows by tracking the sequence number of transmitted packets.

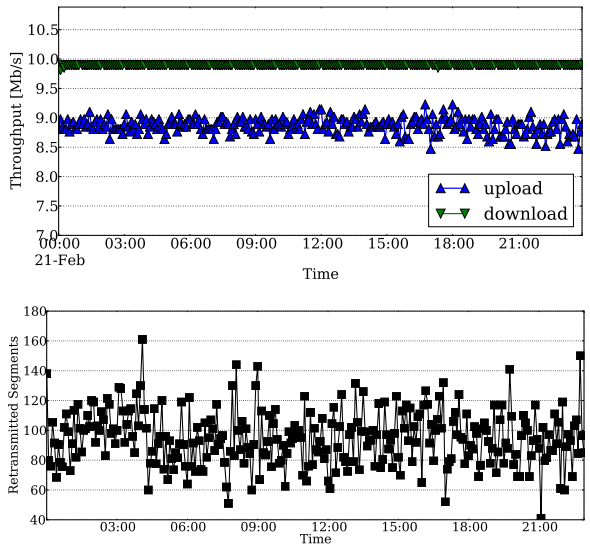

Figure 5. Evolution over time of the throughput measured by an actibe probe (top), and the retransmitted packet rate as measured by Tstat (bottom). U-10Mbps/D-10Mbps FTTH probe.

in the network. By inspecting the statistics provided by Tstat at the server side, we could confirm this intuition. Indeed, notice how the coefficient of variation of the RTT (bottom plot) and the rate of retransmitted packets (center plot) measured on TCP connections carrying FTP file transfers considerably increase during the network utilization period.

By manually checking, we could find out that such probe accesses the Internet through a bottlenecked Virtual Leased Line, and its available bandwidth is out of the control of the operator.

C. Congestion at the Uplink

In top plot of Fig. 5 we report the evolution over time of the throughput measured by an active probe accessing the ISP network through a U-10Mbps/D-10Mbps FTTH interface. Another kind of unexpected result is illustrated. Indeed, the upload throughput is below the expected value, and its profile is very noisy. While this may be common for an ADSL interface (see the first case), it is rather unexpected for the case of a fiber access link, which is not affected by low SNR issues. By inspecting Tstat statistics, we observe that the rate of retransmitted packets (bottom plot) is relatively high and constant in time, meaning that there is some time-independent loss on the path towards the server. We compared these results with others obtained from other probes accessing the network through FTTH, but we could not notice any similar loss pattern at the uplink. This hints for problems at the FTTH home gateway, where possibly some self-congestion is induced when packets sent by the FTP client (connected via a 100Mb/s fast Ethernet) are queued at the FTTH uplink buffer (running 10Mb/s). We instrumented then the IQM probe to flood the server via MTU-sized ICMP echo requests. This generates a burst of packets that enters the buffer. By observing the first packet that appears to be lost, we estimate the buffer size. Thanks to this further experiment we verify that such hag presented some issues in the management of its output buffer, thus reducing its actually employed size and causing losses during the upload speed-test.

D. Other Kinds of Anomalies

Finally, during our analysis we could also identify a link failure event happened at the core of the network. Such failure involved all probes in a specific portion of the network. The reports obtained from active probes together with the exceptional nature of the event were enough to reconstruct the cause behind such distributed degradation. However, even if aggregating active and passive measurements was not necessary in this case, we could detect it thanks to the distributed design of the mPlane architecture. Indeed, apart from enabling the correlation of measurement data generated from different kinds of sources, mPlane design allows to *spatially* correlate measurements coming from different points of the network.

V. Related Work

In general, the task of monitoring and troubleshooting the network can be accomplished by means of i) active measurements, i.e., by injecting packet probes into the network to gather measurements of varied nature, or ii) passive measurements, i.e., by passively monitoring traffic generated by users or by observing diagnosis reports obtained by monitoring protocols running on network devices.

Many papers propose methodologies based on active measurements. `ping` and `traceroute` are daily employed by network administrators to perform reachability and tomography tests. `iperf` is commonly employed to test the available bandwidth on specific network paths. Considering the literature, a recent trend in the research community is to combine different active measurement techniques in distributed frameworks. For instance, authors of [9] build an active measurement system to address the problem of spatially identifying faults in the network. Similarly, authors of [10] propose a complex multi-objective methodology based on a plethora of active measurements to specifically identify Service-level Agreement (SLA) violations. However, active measurement come at the cost of injecting artificial load into the network, and thus must be carefully designed to not overload the system. Moreover, it becomes complicated to define active measurements that involve information related to the transport and network layer. For instance, during a speed-test run using `iperf`, it is difficult to retrieve information about the TCP internal states, e.g., the maximum segment size, or the number of retransmissions; similarly, it is difficult to have information about the network path, e.g., the Round Trip Time (RTT) faced by packets, or the number of traversed hops.

The passive approach to network troubleshooting has inspired the development of many tools, such as TCPDump [1], NetFlow [11] and Wireshark [2] just to name the most popular ones. These kind of tools analyze flowing packets on a network interface, rebuild the conversations at different levels, and

provide statistics and classification results about observed flows. Other approaches instead, consider the diagnosis logs and reports collected by the network devices, e.g., SNMP [12]. Similarly to the case of active measurements, some recent papers investigate the feasibility of distributed platforms for the passive monitoring of the networks [13]. The passive approach however, suffers from at least three non-negligible issues. First, it fails when no traffic is exchanged with some targeted service, or if the traffic does not conform with the measurement assumption, e.g., if the flow does not carry enough data. Second, passive monitoring tools produce in general a large amount of data that must be carefully processed, so that the troubleshooting procedure dramatically slows down. Third, detecting performance degradations by analyzing traffic traces is a task that is sometimes at best frustrating, due to the attention and time it requires.

Our idea combines the two approaches: we leverage active measurements to generate limited and fully controlled traffic, while, in parallel, we passively monitor such traffic with Tstat to enrich active monitoring reports with detailed transport-level information. In this way, we reduce the amount of traffic data to process, but we still have enough information to possibly investigate the root cause of detected anomalies.

VI. Conclusions and Future works

In this paper we investigated the potential of "hybrid" measurements for network troubleshooting. Inspired to mPlane architecture, we designed, deployed and tested a measurement system where the traffic generated by active probes is passively monitored by a passive monitoring tool. This approach let us overcome some of the common limits one encounters when using either active or passive measurements. We show that hybrid measurements allow to enrich application-layer measurements with transport- and network- layer information, thus obtaining a much more insightful view of the status of the network. Second, since passive traces are limited to the traffic that is synthetically generated by the active probes, we can reduce the amount of data to analyze, and, thus, considerably speed up the network troubleshooting process.

We analyzed the measurement data obtained by running our system for a period of three months in the operational network of Fastweb. We could illustrate three main classes of problems affecting ISP customers accessing the Internet through two different kinds of network access technologies, i.e., ADSL and FTTH. We could thus validate our above statement: indeed, our results demonstrate that the hybrid measurements can actually ease and, thus, accelerate the network troubleshooting procedure.

In our ongoing efforts we are working to automate as much as possible the whole hybrid measurement platform, as recommended by the mPlane reference design. In particular, we are focusing our attention on automatic root cause analysis techniques. For instance, we are exploring many correlation analysis methods (e.g., Factor Analysis, Copulas, etc.) to group together metrics sharing similar abrupt changes and immediately pinpoint the root causes behind anomalies whenever they occur.

Acknowledgments

This work was supported by the European Commission under the FP7 IP Project "An Intelligent Measurement Plane for Future Network and Application Management" (mPlane).

References

[1] [Online]. Available: www.tcpdump.org/
[2] [Online]. Available: www.wireshark.org/
[3] [Online]. Available: www.tstat.org/
[4] [Online]. Available: www.snort.org/
[5] [Online]. Available: www.ntop.org/
[6] M. Mellia, M. Meo, L. Muscariello, and D. Rossi, "Passive analysis of {TCP} anomalies," vol. 52, no. 14, 2008, pp. 2663 – 2676. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128608001795
[7] B. Trammell, P. Casas, D. Rossi, A. Bär, Z. Ben-Houidi, I. Leontiadis, T. Szemethy, and M. Mellia, "mPlane: an Intelligent Measurement Plane for the Internet," *IEEE Communications Magazine, Special Issue on Monitoring and Troubleshooting Multi-domain Networks using Measurement Federations*, Accepted for publication. [Online]. Available: https://www.ict-mplane.eu/sites/default/files//public/publications//756mplanerevised.pdf
[8] B. Trammell, M. Mellia, A. Finamore, S. Traverso, T. Szemethy, B. Szabo, D. Rossi, B. Donnet, F. Invernizzi, and D. Papadimitriou, "mPlane Architecture Specification," no. D1.3, Nov 2013.
[9] M. Natu and A. Sethi, "Active probing approach for fault localization in computer networks," in *End-to-End Monitoring Techniques and Services, 2006 4th IEEE/IFIP Workshop on*, April 2006, pp. 25–33.
[10] J. Sommers, P. Barford, N. Duffield, and A. Ron, "Accurate and efficient sla compliance monitoring," in *Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '07. New York, NY, USA: ACM, 2007, pp. 109–120. [Online]. Available: http://doi.acm.org/10.1145/1282380.1282394
[11] [Online]. Available: http://www.cisco.com/c/en/us/products/ios-nx-os-software/ios-netflow/index.html
[12] J. Case, M. Fedor, M. Schoffstall, and C. Davin, "A simple network management protocol (snmp)," 1989.
[13] C. Thomas, J. Sommers, P. Barford, D. Kim, A. Das, R. Segebre, and M. Crovella, "A passive measurement system for network testbeds," in *Testbeds and Research Infrastructure. Development of Networks and Communities*, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, T. Korakis, M. Zink, and M. Ott, Eds. Springer Berlin Heidelberg, 2012, vol. 44, pp. 130–145. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-35576-9_14