# MGRP - passive aggressive measurements

Ferdinand Mayet
Supervisors: Dipl.-Ing. Dirk Haage, Dipl.-Inf. Johann Schlamp
Advanced Seminar - Innovative Internettechnologien und Mobilkommunikation WS2010/2011
Chair for Network Architectures and Services
Department of Informatics, Technische Universität München
Email: mayet@in.tum.de

## ABSTRACT
This work presents the Measurement Manager Protocol (MG RP), an in-kernel service supporting flexible, efficient and accurate measurements. MGRP schedules probe transmissions on behalf of active measurement tools and reduces the monitoring overhead by reusing application traffic. A small benchmark experiment demonstrates the potential of this passive aggressive measurement before an evaluation is carried out. In this context, another sophisticated approach, namely TCP Sidecar, is presented and compared with MGRP and other traditional methods. At the end, some analysis about the usage and application of both concepts are discussed.

## Keywords
active, passive, aggressive, network, traffic, measurements, MGRP, TCP Sidecar

## 1. INTRODUCTION
The Internet evolved in the last thirty years from text based utilities to a platform used for multimedia streaming, online conferencing and other services of the World Wide Web. The majority of the users grasp the Internet as a medium which provides the connectivity between their applications and distributed information and data. The end-users do not need to know any background of how the Internet works, such as the processes that are triggered after the user clicks on a hyperlink or how packages are routed on their way through the network [1].

However, network researchers aim to understand the networks infrastructure and the protocols used to communicate with other instances of the network. A major methodology researchers use to collect and analyse information about networks are end-to-end measurements. Due to measurements, interesting network properties could be estimated which help to improve applications and protocols in order to gain a good user experience. A good user experience could for example be reached by selecting the nearest and fastest server to download from. A different application might need a low round-trip-time (RTT) and a high path capacity. Therefore network applications need to discover the current network conditions and adapt accordingly. Since there is no possibility to gather information about the state of the network by asking other network devices, traffic analysis has to be carried out between the endpoints.

The research area of network traffic measurements also aims at evaluating a given network in order to be able to understand its topology and to identify the available bandwidth between different hosts. This topic is steadily gaining popularity since video streaming becomes more and more important to individuals (e.g. watching videos on YouTube) as well as to companies using the Internet as a online meeting platform. Detailed information about the network would enable applications or even protocols to change their behaviour and adapt to the networks state. For example, if a video is hosted on multiple servers the application could choose the best connection between client and server. This might be the nearest server but it could also be the case that this specific one is too busy to satisfy the users requirements. In order to achieve a higher user experience the application should be able to discover such shortages and choose an appropriate way to solve them. Furthermore, measurement techniques are used to reveal security issues like firewall misconfigurations or to locate problems occurring during communication.

In the following, a short introduction to the topic of network measurements will be outlined and a variety of traditional measurement approaches will be discussed. Thereafter, two sophisticated techniques will be presented and analysed. At the same time, their specific advantages and drawbacks will be elaborated. Finally, some related work are presented.

## 2. BACKGROUND
Network measurements are applied whenever information about a network and its current state is necessary. Therefore, four main reasons for network measurements will be presented and their benefits will be explained:

**Network Troubleshooting** The purpose concerning traffic measurements in the area of network troubleshooting is to discover defective hardware and misconfigurations of endpoints and intermediate devices. For example, the Internet Control Message Protocol (ICMP) can be used to send messages to a desired endpoint. If these messages do not arrive at the endpoint an error message is triggered which indicates that something is wrong in the network. More precise evaluations in combination with other protocols can then be used to identify defects or misconfigurations in the network [1].

**Protocol Debugging** Protocol Debugging is necessary if new protocols are developed. Thereby, measurement techniques ensure the standard compliance of a protocol by for example analysing the traffic. Furthermore, it is possible to prove the backward compatibility of a newer protocol version to its predecessor. In order to prove backward compatibility a variety of approaches are available such as establishing a communication between two endpoints with different protocol versions and examining the transferred messages [1].

**Workload Characterisation** Another area where network measurements are applied is the field of workload characterisation. This domain analyses the exchanged traffic between endpoints and creates a ranking of the protocols which transfer data. On the basis of this ranking applications can be optimised for the most frequently exchanged data. This is very important to multipurpose applications which use several protocols to communicate with other hosts. Workload characterisation also focuses on the protocol layer and supports the improvement of newer protocol versions with regards to the monitored workload [1].

**Performance Evaluation** Another important usage of measurements is performance evaluation. Network traffic measurements are utilized to determine the performance of the network. A detailed analysis may help to identify performance bottlenecks. Once these problems are identified the results might be used to further improve protocols or the network infrastucture itself. Performance evaluation is often used in combination with the workload characterisation process described earlier [1].

These four reasons are just a few examples of motivations for monitoring network traffic. Detailed information about protocols and processes of a network are very important to achieve a higher usability and to enhance the users experience. Hence, network measurements must be performed. However, the crucial part is to pay attention to the realization of such measurements because they should be transparent to the user and should not change the network. In the following, a short introduction to traditional methods for monitoring the network is given.

## 3. METHODOLOGY

Software network measurement tools can be classified in two major monitoring concepts, passive and active. These two concepts can again be subdivided in offline and online measurements. In this case online describes a technique where packets are analysed on the fly whereas offline denotes a mechanism that first captures information and evaluates the data afterwards. A common representative of offline monitoring are log files or dump files, for example created by *tcpdump*[1].

### 3.1 Passive Measurements

Passive measurement describes a mechanism which collects the observed traffic of the network. The term “passive” states that no additional workload is introduced into the network and only available traffic is captured and analysed. In order to obtain information about a given link, such as time dependent references, the observation of the traffic has to be applied at different network locations (see Figure 1) [2].

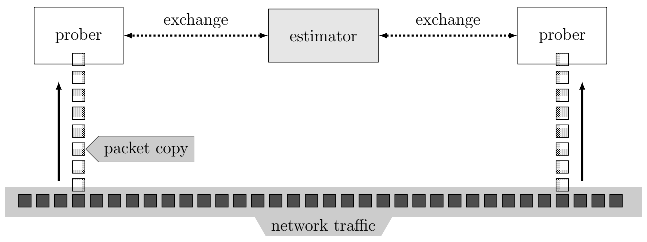


**Figure 1: Concept of passive measurements**

Claise ([3]) categorises passive measurements into two groups:

**Full collection** This process captures every single packet which passes the metering point. The main advantage of a full collection is accuracy as the collected data is exactly equal to the passed traffic. However, a drawback is the large number of packets that have to be stored and analysed, which may require very fast metering hard- and software.

**Partial collection** Most of the time it is not possible to perform a full collection due to high speed interface technologies which send a huge amount of data in a very short time period. Therefore a partial collection process which filters or samples the collected data is necessary. For example, filtering mechanisms may select a specific flow of data (e.g. TCP traffic) to reduce the workload of the monitoring unit. In contrast, sampling uses statistical methods (e.g selecting 1 of $N$ packets) in order to reduce the load of the measurement systems.

Passive measurements are applied if the exact network state is important and interference with live traffic is not wanted. However, the disadvantage of monitoring is that desired traffic types may not be present in the traffic passing the observation point. This purpose could be solved using active measurement techniques.

### 3.2 Active Measurements

In contrast to passive measurement, active measurements require explicit requests that generate synthetic traffic with a desired type and workload [2]. Active measurements involve two systems into the process, a sender and a receiver. The sender creates the desired traffic and sends it to the receiver which collects all packets at their arrival and evaluates each (see Figure 2).

[1]tcpdump: a Linux tool which dumps traffic on a network `http://www.tcpdump.org/`

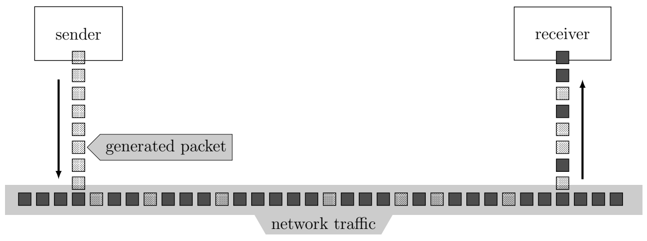


**Figure 2: Concept of active measurements**

Performance evaluations, network troubleshooting and protocol debugging is mostly done using active measurements since it is possible to generate an extremly high workload, malformed packets and special traffic. This might be necessary if special information about the network must be collected. Compared to passive techniques active probing has the advantages of maximum traffic control, independency of the current traffic and the ability to detect the maximum available bandwidth. Furthermore, active measurements are easier to implement than passive ones. Nevertheless, synthetic traffic may cause collsions in the network und thus change the network's behaviour. Hence, active measurements are commonly an estimation of the real network state and throughput [3].

To conclude, both techniques have multiple advantages but also several disadvantages. Obviously, new mechanisms are necessary to minimise the newly introduced traffic on the one hand and to keep the control of the transferred traffic as high as possible on the other hand. In the following sections two more sophisticated measurements techniques are presented which try to counter the observed problems and combine all advantages.

## 4. MGRP - PASSIVE AGGRESSIVE MEASUREMENTS

This section introduces the Measurement Manager Protocol (MGRP) which addresses the shortcomings of traditional approaches by using a *hybrid* concept. MGRP is an in-kernel service that enables probes to reuse application traffic transparently and systematically. As described in section 3 passive probing is efficient but unable to detect improvements of network conditions and active probing affects the current traffic on the link. MGRP permits the user to write measurement algorithms as if they are *active* but be implemented as if they are *passive*. Hence MGRP can be more aggressive without harming the performance of an application [4].

MGRP piggybacks application data into probes in order to minimise newly introduced traffic. Piggybacking is a process that is aware of probes which mostly consist of empty padding. Empty padding is necessary because probes have to reflect the behaviour of real application traffic that carries useful payload. The piggybacking mechanism replaces the empty padding of a single probe with payload that should be transmitted to the receiver and thus prevents the prober from sending unnecessary packets.

### 4.1 MGRP architecture

MRGP is a kernel-level service which extends the transport layer of the ISO/OSI model. This protocol is basically accessed using two application programming interfaces (APIs), the probe and payload API. The payload API extracts useful data from other transport protocols like TCP or UDP and hands it to the MGRP service. In collaboration with the probe API MGRP generates a *hybrid* of a probe and application data. Instead of sending single probes directly to the receiver, the sender uses the probe API to specify an entire *train*. A *train* is defined by the size of each probe, the number of probes, the amount of padding and the gap between probes [5].

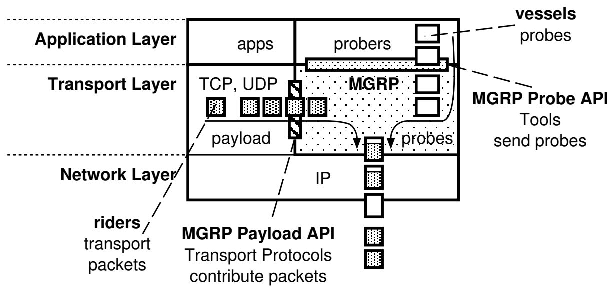


**Figure 3: MGRP architecture [4]**

Once defined a *train* MGRP starts piggybacking application traffic and sends these merged packets to the receiver. By filling most of the empty padding with payload MRGP nearly behaves like a passive algorithm since it omits the overhead generated by active measurements. Figure 4 indicates the transition from active mechanisms with probes and empty padding (white/black checkerboard) to a MRGP like traffic with no padding. Figure 4 also shows the newly introduced MGRP header (illustrated as small light gray box).

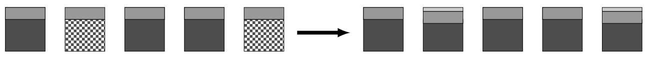
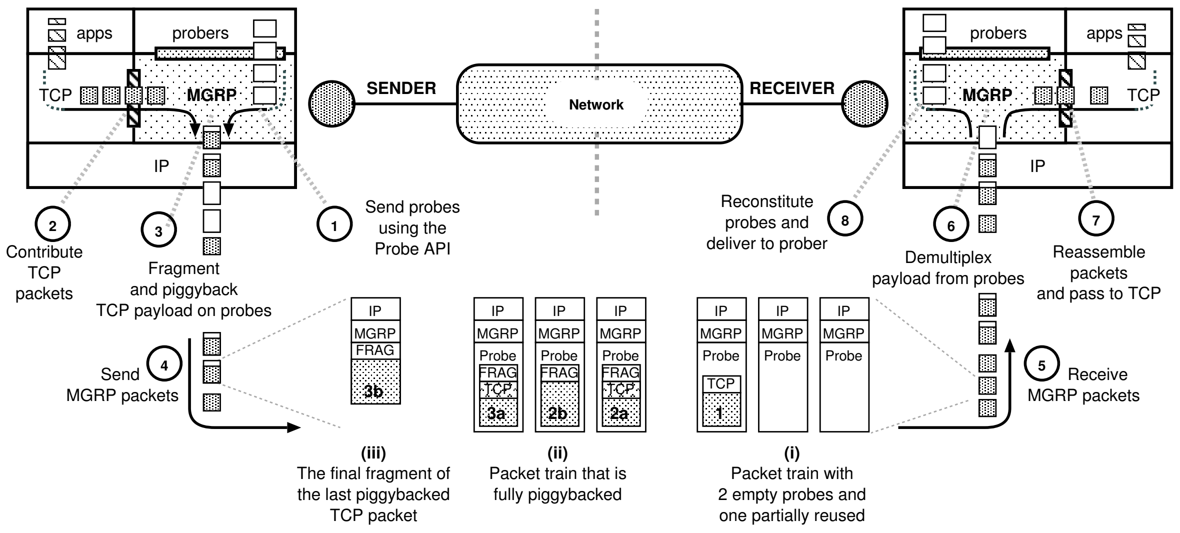


**Figure 4: Transition from active measurement traffic (left) to MGRP traffic (right)**

At the receiver side the payload is separated from the measurement data. The payload output is handed over to the standard transport layer and the measurement data is transferred to the monitoring system. The receiver side adds a second timestamp to the MGRP header and delivers the packet to the prober. The MGRP header mainly consists of two timestamp header fields, one timestamp is entered by the sender when the packet is sent. The other one is entered by the receiver and contains the reception time [4].

### 4.2 Probe Transaction by Example

The example described in this section is illustrated in Figure 5 and will be walked through from step ① to ⑧. Consider the following case: The sender is streaming multimedia data to a destination $D$ and at the same time MGRP is used



apps
probers
TCP
MGRP
IP
SENDER
Network
RECEIVER
Send probes using the Probe API
Contribute TCP packets
Fragment and piggyback TCP payload on probes
Send MGRP packets
Reconstitute probes and deliver to prober
Demultiplex payload from probes
Reassemble packets and pass to TCP
Receive MGRP packets
IP
MGRP
FRAG
Probe
TCP
3b
3a
2b
2a
1
(iii) The final fragment of the last piggybacked TCP packet
(ii) Packet train that is fully piggybacked
(i) Packet train with 2 empty probes and one partially reused

**Figure 5: Demonstration of MGRP operation [5].**

to measure the network condition between the sender and *D*.

At step ① the prober calls the `sendmsg` function to send the first probe. The first `sendmsg` call also defines several options and the ancillary data. For example, the gap between probes and the barrier flag is set. The barrier flag is used to indicate that a whole *train* of probes should be created. As long as the barrier flag is set to 1 all probes are buffered until the flag is unset. Afterwards the probes are sent as a *train*. Packets generated by the streaming application with same destination *D* as the probes are collected by the payload API ②. At step ③ the TCP packets are fragmented and piggybacked into the probe. In some cases fragmentation might be necessary as the payload could exceed the MGRP payload size. Due to additional header information of MGRP the payload size is smaller then the one of e.g TCP. Afterwards MGRP sets the kernel timestamp in the MGRP header field and hands it to the IP layer for transmission ④. During transmission three different kinds of packets may occur: As illustrated in (i) of Figure 5 MGRP provides the possibility to completely disable piggybacking or only partially. In the partially disabled case, MGRP sends out probes with empty padding if no suitable rider was found. If it is completely disabled MGRP behaves like a traditional active measurement tool. The ideal case of piggybacking is shown in (ii) where all probes are reused to transport application data. The label 2a and 2b indicate that the original TCP packets had to be fragmented to fit into the probes data field. If MGRP was unable to piggyback the payload before the buffer timeout exceeds additional MGRP packets containing the remaining chunks are sent (iii). The buffer timeout determines the available time for buffering application data until it must be transmitted by MGRP. As the packets arrive at destination *D* ⑤ the payload is demultiplexed ⑥ from the measurement data. Next, the original TCP packets are reassembled and delivered to the application ⑦. At the same time, MGRP reconstructs the probing packets by zeroing the padding and setting the reception time ⑧. Finally, MGRP buffers the probes for delivery to the prober [5].

## 4.3 Experiments with MGRP

This section elaborates on an experimental setup in order to demonstrate the behaviour and performance of MGRP. The network topology is given in Figure 6.

In the experiment a constant 4 Mbps stream is transmitted from $m3$ to $m5$ representing a multimedia stream hosted on $m3$ and requested by $m5$. The data rate of 4 Mbps was chosen because it is commonly used for high definition multimedia streams [4].

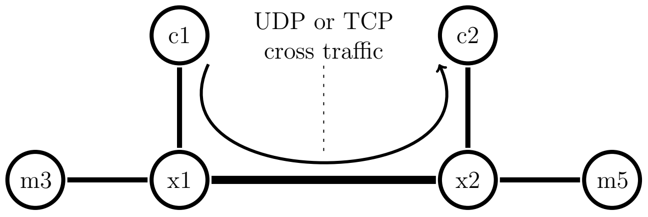


**Figure 6: Experimental setup (based on [5])**

While streaming from $m3$ to $m5$ MGRP is used to determine the actual bandwidth between those two nodes. In order to obtain more realistic results, the link $x1x2$ is throttled to a maximum data rate of 10 Mbps as there are only 4 participants in the network.

As shown in Figure 7 the multimedia stream in disturbed by UDP cross traffic which is transmitted from $c1$ to $c2$. The cross traffic is stepwise increased until it reaches the maximum spare throughput of 6 Mbps and is decreased afterwards. Each interval lasts 45 seconds and transmits constant rates of 1, 3, 5, 6, 4 and 2 Mbps. The most interesting interval is between 135 and 180 seconds because at this point in time no additional traffic like measurement probes are able to pass from on side to the other without harming one of the two streams on link $x1x2$ [4].

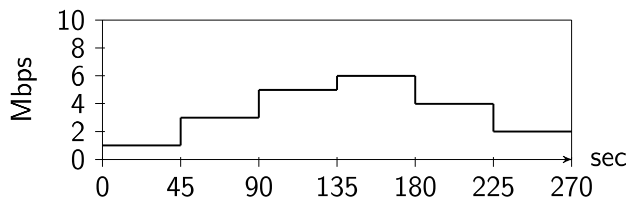


**Figure 7: Cross traffic from $c1$ to $c2$ (based on [5])**

Figure 8 presents the results of the experiment described before. Two cases are considered in the plots:

1. active probing with piggybacking disabled (upper plot)
2. reuse of application payload as a rider using MGRP (lower plot)

**Case 1:** After about 70 seconds the two streams and the additionally introduced payload of the packets start to interfere as there is not enough bandwidth available to satisfy all network participants. Unfortunately, UDP has no congestion control and keeps sending as much cross traffic as possible. However, the data flow between $m3$ and $m5$ uses TCP and recognises that the link is overwhelmed which forces TCP to enter the congestion avoidance phase. The algorithm of this phase decreases the maximum segment size ($MSS$) (e.g. $MSS = \frac{MSS}{2}$) and therefore limits the transmission rate to 3 Mbps respectively to 2 Mbps later on [6]. If both hosts would have used UDP as transport protocol a dramatic packet loss would have occurred. Figure 8 also indicates that the monitoring traffic consumes approximately 2 Mbps. To conclude, the user viewing the multimedia data on location $m5$ will experience stuttering or in the case that the hosting application is aware of the congestion the stream quality will be downgraded.

**Case 2:** In this case MGRP utilises the application traffic as riders and nearly all probes carry application data. Only approximately 0.2 Mbps are used to send probes without piggybacked payload. Hence, the TCP connection experiences only little congestion and the stream nearly stays at 4 Mbps. To sum up, MGRP reduces the measurement overhead to a minimum and lowers interference with other network communications while monitoring constantly. The viewer of the video stream might only suffer small or even no changes.

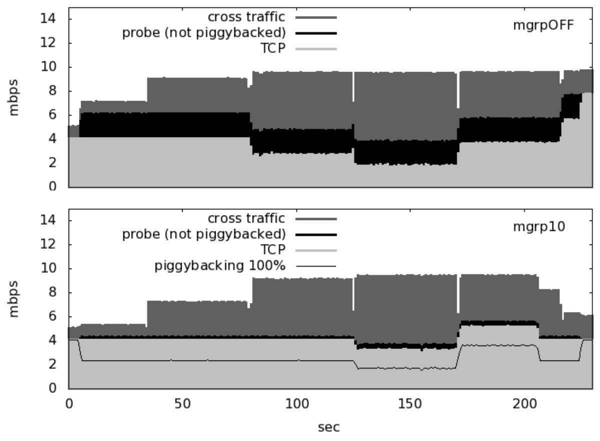


**Figure 8: Results of the experiment [5]**

## 4.4 Evaluation

This section provides a overview on MGRP and discusses problems that may occur using MGRP for measurement purposes. As described earlier, MGRP facilitates the reuse of existing traffic. Hence, the networks condition is just slightly modified and the amount of newly introduced collisions is minimal. Furthermore, MGRP is traffic independent since it switches to an active measurement like mode if no riders are available. Due to the fact that the number of probes can be specified, MGRP is able to determine the maximal available bandwidth.

But there are several reasons why MGRP is not used every and all the time: First of all, MGRP inserts delay into the network as application traffic is buffered and multiplexed at the senders side and demultiplexed at the receivers side. Secondly, MGRP changes the behaviour of all TCP connections on the link as collisions may appear more likely. Since TCP implements a congestion avoidance algorithm multiple TCP connections influence each other and try to share the maximum available bandwidth equally. In a scenario where MGRP is used to measure the network additional overhead is added to a single connection which leads to a worsening of all other connection sharing the same link. Even if these modifications are small an excessive usage of MGRP on multiple connections might yield a large overhead which restricts the performance of the network in contrast to passive measurements. Furthermore, sophisticated delay calculation have to be done by the prober since there are several timeouts and buffering/fragmentation delays have to be considered. Additionally, measurement packets are not able to traverse firewalls and NATs (network address translations) which also prevents MGRP from being used universally. This problem occurs as firewalls and NATs do not understand the MGRP header format. However, the most important drawback relates to the implementation of MGRP. As described in section 4 MGRP is an in-kernel service. This indicates that the network stack has to be modified in order to add MGRP to the transport layer. Currently no operating systems integrates MGRP by default. Hence, the monitoring systems stack has to be changed which might not be feasible in many cases. The fact of the matter is that this prevents MGRP from being used by peers globally. Streaming or peer-to-peer applications like Skype can not use it as well, since there is no guarantee of having MGRP available on all participatory systems.

The following section will present another measurement technique. This approach is based on the traditional active measurement process and extends it in several ways.

## 5. TCP SIDECAR

TCP Sidecar is a monitoring platform for injecting packets into a network and follows the principle that the network provides enough bandwidth to handle additional traffic caused by probes. The main goal of TCP Sidecar is to circumvent intrusion detection systems (IDSs) and firewalls since synthetic traffic is most often considered being extraordinary and potentially malicious. Hence, most firewalls will block measurement traffic and IDSs will trigger alerts and abuse reports. Therefore, carefully designed measurement probes and responses have to be generated. Thereby, TCP Sidecar does not restrict the source and destination nor the time of measurement since the platform does not want to force any extraordinary behaviour [7].

### 5.1 Architecture and Probe Transaction

The concept behind TCP Sidecar is to generate probes consisting of replayed data segments. Therefore Sidecar uses passive measurements to collect traffic that is passing by and retransmits it to destination. Figure 9 outlines the standard procedure of a Sidecar measurement. The prober (Sidecar) can be positioned freely in the network. Often it is placed at the senders side but it is possible to place it at every other node in the network as long as both the forward and reverse path are observed by Sidecar [8].

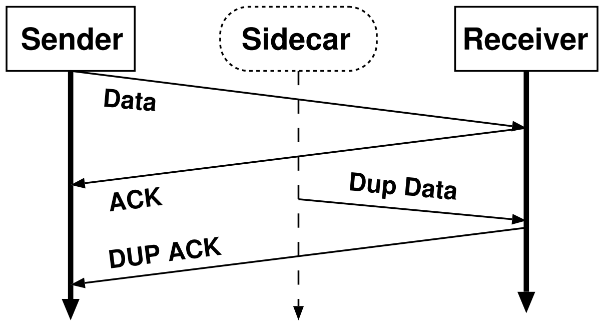


**Figure 9: TCP Sidecar procedure [7]**

TCP Sidecar is also used to modify the captured data before replaying it. This has to be tampered very accurately as firewalls and IDSs should not notice any difference between real and replayed packets.

For example, the time-to-live (TTL) option of the TCP header can be adjusted freely because the receiver does not necessarily need the information contained in the replayed packets. Hence, these packets can be dropped before they reach the receiver. This modification might be used inside TCP Sidecar to detect NATs. The detection mechanism simply uses ICMP and varies the TTL. If a TTL exceeded message is returned by a network node and the source address of this error message is the same as the original destination address of the packet, a NAT can be assumed [8].

Once a duplicate packet arrives at the receiver no warning or even error message is generated since TCP considers the reception of duplicate packets. Shortly after the reception the receiver generates a duplicate acknowledgement (ACK) and sends it to the source address. This packet is then again captured by TCP Sidecar for measurement purposes [8].

### 5.2 Evaluation

TCP Sidecar is a platform for unobtrusive measurements and enables measurements throughout firewalls, and NATs. Furthermore, Sidecar is able to detect NATs and perform several kinds of measurements without alerting IDSs. Especially large network service like PlanetLab [8], CoDeeN [9], OpenDHT [10], Meridian [11], and CoralCDN [12] might benefit from TCP Sidecar since most of these services already perform network measurements and might struggle with altering IDSs.

However, Sidecar has several drawbacks: Firstly, the platform depends on existing traffic which might be applicable to large networks but may be a problem in smaller networks. Since no measurement will be forced (only the amount of replayed data can be set) the results in smaller network may not be as accurate as with using other active measurement tools. A second problem might be the placement of the monitoring system as both communication channels must pass the metering point. The most significant problem is the fact that duplicate ACKs are generated. A duplicate ACK can be regarded as network problems by the sender. Therefore TCP adjusts the congestion window size which is a state variable that limits the maximal amount of unacknowledged TCP packets. For example, if the congestion window size is 2, TCP can only send two TCP packets with an outstanding acknowledgment. Each duplicate ACK decrements the congestion window by one. This might get even worse if a third duplicate ACK reaches the sender because in this case TCP enters the slow start phase and halfs the maximum segment size (MSS). Hence, the data rate is reduced and the communication is violated by measurements. Nevertheless, this problem might be solved by selectively grabing duplicate ACKs and discarding them if they are not important to the sender. Regarding to Figure 9 the Sidecar node must be able to not only generate duplicate data but also to analyze duplicate ACKs on their way back to the sender. If the the duplicate ACK is considered being non-essential to the sender (e.g a duplicate ACK with the same sequence number has already been transferred to the sender) the packet must be dropped by the Sidecar node. However, the classification of these packets into categories like important to the sender or not might quite challenging but would potentially increase the performance of TCP Sidecar.

## 6. RELATED WORK

MGRP is similar to many approaches (Periscope [13], Scriptroute [14], pktd [15]) in that it serves the possibility to define measurement probes and schedules. MGRP differs from these approaches as it is the first tool, which is fully integrated on layer 4 in the IP protocol stack. Thereby, MGRP reduces the measurement overhead by reusing probes as riders for application data. Furthermore, MGRP is a protocol and not a standalone application. Thus, it can be integrated into a big amount of existing applications and help to improve their performance by reducing unnecessary overhead.

The following collection of related work should give a short overview of similar project:

- Sidecar has the advantage that it support ICMP messages which enable NAT detection. But as a consequence of the problems described in 5.2 the measurement intervals have to be kept low which makes the usage of Sidecar difficult.
- MAD [16] is a Multi-user Active Measurement service that generates probes on behalf of probers and is also implemented in the Linux kernel in order to gain a higher accuracy. In contrast to MGRP MAD does not use piggybacking but provides a interface for self-measurements of the system which again enhances the accuracy.
- Scriptroute [14] is a public Internet measurement facility that conducts remote measurements for users. Measurements are written in a special script language and uploaded to a server. Afterwards, the server performs the desired measurement in a secure way by providing several mechanisms to the user which ensure that a measurement does not exceed a given bandwidth or no bad packets are generated.

Obviously, there is a large amount of measurement tools available online [17, 18]. Combining the features of these tools with the Manager Protocol might lead to even more sophisticated applications for network measurements.

## 7. CONCLUSION

This work presented traditional measurement techniques and the Measurement Manager Protocol, a flexible and efficient monitoring protocol. Based on an experiment MGRP's performance was demonstrated and proved the potential of this approach. Furthermore, TCP Sidecar was introduced which presented a security oriented way of measuring networks and introduced new features like NAT detection.

Both, MGRP and TCP Sidecar provide a interface to collect information about the network. Subsequently, this feedback can be used to improve applications and protocols. Especially MGRP has great potential to be used by streaming applications to enhance the user experience without harming the network. Furthermore, both mechanisms are able to detect improving network conditions. This information is very important to all kinds of applicatons since it enables them to leave the congestion avoidance phase earlier. The information could also be used to replace the slow start phase of TCP after a congestion occured because the application is aware of the maximal available bandwidth.

To conclude, measurements have the ability to solve a large range of problems - e.g. performance issues. However, the resulting measurement overhead has to be taken into consideration to prevent network exhaustion and co-occuring delays. Moreover, the whole measurement process must be transparent to the user to keep Internet usage as simple as possible.

## 8. REFERENCES

[1] C. Williamson, "Internet traffic measurement," *IEEE Internet Computing*, vol. 5, no. 6, pp. 70–74, 2001.

[2] D. Verma, *Principles of Computer Systems and Network Management.* Springer-Verlag New York Inc, 2009.

[3] B. Claise, "Network Management - Accounting and Perfomance Strategies," *Cisco press*, p. 672, 2007.

[4] P. Papageorgiou, "The Measurement Manager: Modular and Efficient End-to-End Measurement Services," *Doctor*, no. 1, 2008. [Online]. Available: http://drum.lib.umd.edu/handle/1903/8900

[5] P. Papageorge, J. McCann, and M. Hicks, "Passive aggressive measurement with MGRP," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 4, p. 279, Aug. 2009.

[6] W. Stevens, M. Allman, and S. Paxson, "RFC 2581: TCP Congestion Control," 1999.

[7] R. Sherwood and N. Spring, "Touring the internet in a TCP sidecar," *Proceedings of the 6th ACM SIGCOMM on Internet measurement - IMC '06*, p. 339, 2006.

[8] R. Sherwood, "A platform for unobtrusive measurements on PlanetLab," *Proceedings of the 3rd conference on*, 2006.

[9] L. Wang, K. Park, R. Pang, V. Pai, and L. Peterson, "Reliability and Security in the CoDeeN Content Distribution Network," in *Proceedings of the USENIX 2004 Annual Technical Conference.* USENIX Association, 2004, p. pp 14.

[10] S. Rhea, B. Godfrey, B. Karp, J. Kubiatowicz, S. Ratnasamy, S. Shenker, I. Stoica, and H. Yu, "OpenDHT: A public DHT service and its uses," *Interface*, vol. 35, no. 4, pp. 73–84, 2005.

[11] B. Wong, A. Slivkins, and E. G. Sirer, "Meridian: a lightweight network location service without virtual coordinates," in *Proceedings of the 2005 conference on Applications technologies architectures and protocols for computer communications*, R. Guérin, R. Govindan, and G. Minshall, Eds., vol. 35, no. 4. ACM, 2005, pp. 85–96.

[12] M. J. Freedman, E. Freudenthal, and D. Mazières, "Democratizing content publication with Coral," in *NSDI.* USENIX Association, 2004.

[13] K. Harfoush, A. Bestavros, and J. Byers, "An Active Internet Probing and Measurement API," 2002.

[14] N. Spring, D. Wetherall, and T. Anderson, "Scriptroute: A Public Internet Measurement Facility," *Proc USENIX Symp Internet Technologies and Systems USITS Mar*, 2002.

[15] J. Gonzalez, "pktd: A packet capture and injection daemon," *Passive and Active Measurement Workshop*, 2003.

[16] J. Sommers and P. Barford, "An active measurement system for shared environments," *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement IMC 07*, p. 303, 2007.

[17] Caida, "Performance Measurement Tools Taxonomy."

[Online]. Available: http:
//www.caida.org/tools/taxonomy/performance.xml
[18] L. Cottrell, "Network Monitoring Tools," 2010.
[Online]. Available: http:
//www.slac.stanford.edu/xorg/nmtf/nmtf-tools.html