SNMP Trace Analysis: Results of Extra Traces

Gijs van den Broek, Sietse ten Hoeve, Giovane C. M. Moura, Aiko Pras
University of Twente
Enschede, The Netherlands
Email: j.g.vandenbroek, s.n.tenhoeve@student.utwente.nl,
g.c.m.moura, a.pras@utwente.nl

December 15, 2009

Abstract

The Simple Network Management Protocol (SMMP) was introduced in the late 1980s. Since then, several evolutionary protocol changes have taken place, resulting in the SNMP version 3 framework (SNMPv3). Extensive use of SNMP has led to significant practical experience by both network operators and researchers. Since recently, researchers are in the possession of real world SNMP traces. This allows researchers to analyze the real world application of SNMP. A publication [1] of 2007 made a significant start with this. However, real world trace analysis demands a continual approach, due to changing circumstances (e.g., regarding the network and SNMP engine implementations). Therefore, this paper reports on a lot more traces than in the mentioned paper, which are also more recent.

Contents

Chapter 1

Introduction

The Simple Network Management Protocol (SNMP) [2] is the *de facto* management standard for TCP/IP networks. It was firstly introduced in the late 1980s and since then it has been used in various scenarios and deployed in most of the network devices, such as routers and switches. Despite being widely deployed, it is not enterily clear how SNMP is effectively used in different production networks.

To address this, in 2007 Schönwälder *et al.* [1] presented an inital analysis on SNMP usage. To do that, the authors have captured and analyzed eight SNMP traces from seven different locations, including national research networks, university and faculty networks, and a point of presence. The main conclusions were that SNMP is used more for monitoring than configuring purposes. Moreover, despite versions 1 and 2 of SNMP have been declared as historic by IETF and SNMPv3[3] as the standard, the authors could not capture many SNMPv3 packets. Finally, the authors have pointed that their research was "work in progress", and more trace analysis from different locations was necessary.

In this context, this report presents an extesion of the research conducted in [1]. We have used the methodoly and tools presented in the aformentioned paper to analyze 17 different SNMP traces from various locations. Moreover, we have presented a comparison between our results and the results obtained in the previous paper. To the best of our knowlegde, this technical report presents the most complete analysis of SNMP usage in real life production networks.

The report is organized as follows. Chapter 2 presents more details about the SNMP traces used in our analysis. Next, the results of our analysis are described in chapter 3 and, finally, Chapter 4 present our conclusions and future work. Appendix A presents extra graphs obtained in our analysis wich presents more fine-grained details of SNMP usage.

Chapter 2

Traces

The traces used as input data for analysis were obtained from several locations. For this report we have analyzed 17 different traces from 9 different locations, including a private telecommunications company. In order to easily identify traces and locations, we have adopted the same naming scheme for traces described in [1]. This scheme contains two numbers, in which the first one describes the location and the second one the trace number. For example, *l01t01* refers to trace 1 recorded at location 1.

Table 2.1 contains an overview of the traces discussed in the remainder of this paper, including a location description of the location each trace was obtained from, the start time of the trace and the trace length in hours.

trace	**location description**	**start time**	**hours**
l01t01	national research network	2005-07-20	13.5
l01t03	national research network	2006-04-14	24.0
l05t02	regional network provider	2006-05-28	196.1
l11t01	networking lab network	2006-01-07	614.1
l13t01	university network	2006-10-11	0.2
l14t06	telecommunication company network	2006-09-26	85.1
l14t07	telecommunication company network	2006-09-30	181.3
l14t08	telecommunication company network	2006-10-08	1705.1
l14t09	telecommunication company network	2006-12-19	33.3
l14t10	telecommunication company network	2006-12-31	90.0
l15t02	national research network	2006-09-21	81.7
l15t03	national research network	2006-09-27	308.7
l15t04	national research network	2006-10-17	185.8
l16t01	university network	2007-06-20	59.3
l16t02	university network	2007-06-23	934.4
l17t01	university network	2007-06-20	118.9
l18t01	national research network	2007-06-22	336.0

Table 2.1: Overview of the analyzed SNMP traces

It should be observed that we had 10 different traces for location # 14. Since the first 5 of this location cover really short periods of time and they took place at a time close to theother traces, they were not taken into account. The same

reasoning applies to the decision to location #15, where the trace l15t01 was discarded.

The most significant difference between the traces discussed in this report and in [1], is that the traces in this paper are newer and are also of different locations. We should also point that different traces from locations 1, 3, and 5 were also analyzed in the aformentioned paper.

Chapter 3

Analysis

This chapter presents the analysis results of the traces described in Chapter 2.

3.1 General Characterization

Table 3.1 provides a general characterization of the traces used in this research. It list number of messages in each traces and the distribution of the SNMP messages according to each version available. Percentages smaller than 0.0% are expressed as 0.0% and traces where none of the messages were of a particular version result in a dash in the respective column.

trace	**messages**	**SNMPv1**	**SNMPv2c**	**SNMPv3**
l01t01	71.501	100.0%	-	-
l01t03	11.332.845	99.9%	0.1%	-
l05t02	9.333.212	100.0%	-	-
l11t01	6.566.061	38.6%	61.4%	-
l13t01	6.455	16.5%	83.5%	-
l14t06	835.706	99.9%	0.1%	-
l14t07	3.215.572	100.0%	0.0%	-
l14t08	131.036.004	100.0%	0.0%	0.0%
l14t09	1.576.897	100.0%	0.0%	-
l14t10	1.037.907	100.0%	-	-
l15t02	1.844.512	15.8%	84.2%	-
l15t03	11.852.127	15.9%	84.1%	-
l15t04	8.980.759	25.9%	74.1%	-
l16t01	19.300.858	100.0%	0.0%	-
l16t02	306.598.103	100.0%	0.0%	-
l17t01	1.056.276	8.1%	91.9%	-
l18t01	30.487.406	19.2%	80.8%	-

Table 3.1: General characteristics of the traces

The table shows that all of the considered traces, except l13t01, are quite large in terms of the number of recorded SNMP messages. Another observation is that in the case of 10 out of these 17 traces 99.9% or more of the SNMP messages are of version SNMPv1. On the other hand, 7 traces have a majority

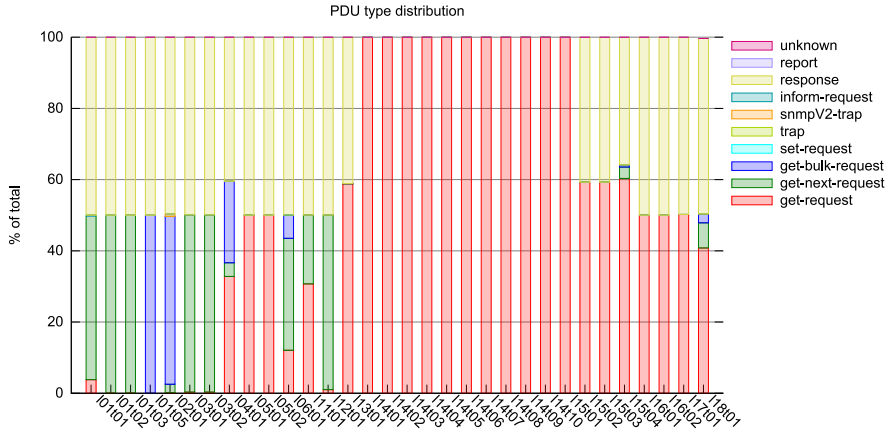

Figure 3.1: Distribution of PDU types of all traces

of SNMPv2c messages. Finally, only one of these 17 traces had any SNMPv3 messages. However, since the number of SNMPv3 messages in l14t08 is so very small (just 3 messages), the percentage still remains far below 0.1%. These messages are probably the result of someone experimenting with SNMPv3.

The distribution of the messages over the three versions is not significantly different compared to the distribution in [1], where there were also hardly any SNMPv3 messages found in the traces.

3.2 Protocol Operations

Figure 3.1 shows the distribution of different SNMP operations for each trace. As in [1], the most observed operations were GetRequest, GetNext, and GetBulk operations and their responses, which are used for monitoring purposes. Very few set messages were observed in our traces.

Varbind distribution

Figure 3.2 shows the varbind distribution over the SNMP messages we have analyzed. As one can see, most used only one varbind, which means that requests are most commonly used for short requests and do not make much use of the (new) possibilities to request large amounts of data within a single request.

In the same figure, it is possible to observe a cluster of points on the bottom right side of the graph. There is a significant number of messages with a high amount of varbinds used. This can be identified as behavior expected when `get-bulk-request`(s) are used. According to figure 3.1, one can see that indeed about 20% of the PDUs is a `get-bulk-request` message.

The traces from location 14 often make use of 3 or 6 varbinds in the majority of the responses. This is probably due to the manager application, which is generating static requests.

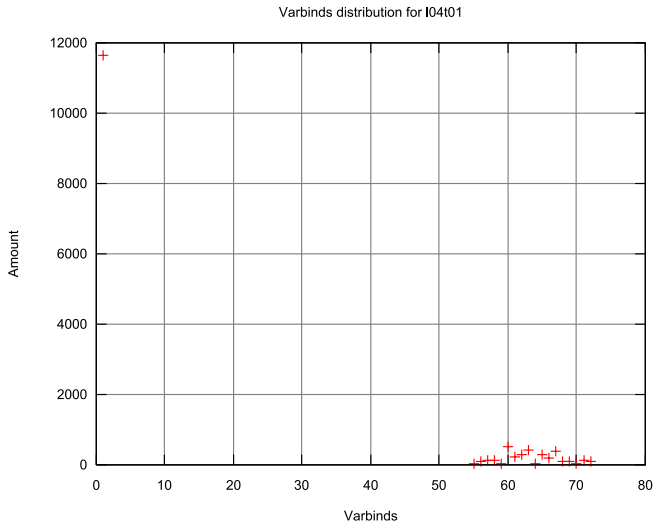

Figure 3.2: Varbind distribution of l04t01

Response Messages Size Distribution

For the analyzed traces, we could observe that, for most of the messages, the response message size was approximately 50 bytes. A number of traces also contain quite of lot of responses with a size varying between 50 and 300 bytes. Large messages were found but in a smaller number.

However, in trace l04t01 a different pattern was observed. Figure 3.3 shows the response message size distribution for this trace. As one can see, half of the responses has a size around 1300 bytes. The large size of the responses is due to the usage of `get-bulk-requests` in this trace.

The results observed for the response message size distribution are similar to those observed in [1].

3.3 Management Topology

The management topology is defined by the relationship between the number of managers and agents in a SNMP-enabled network. Figure 3.4 presents this relation for each trace we have analyzed.

As one can see, for most of the traces the number of managers is larger than the number of managed devices, or agents, which is somehow expected. However, traces l03t01, l03t02, l14t09, l14t10 and l15t01 have a relatively high proportion of managers in relation to managed devices (more than 40%).

3.4 Traffic Patterns

As mentioned in [1], it is expected to have regular traffic patterns for SNMP traffic, since it is assumed that most of the traffic is generated by tools such as Cacti, which fetch MIB variables at regular intervals. This was, as in [1],

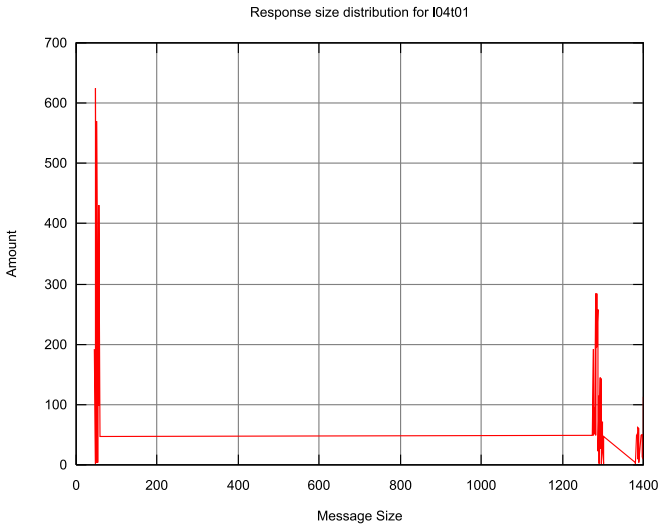

Figure 3.3: Response Messages Size Distribution of l04t01

observed for most of the traces. Figure 3.5 shows this traffic pattern for the trace trace l06t01.

However, this is not always the general rule. For example, Figure 3.6 shows an interesting shift in the usage of SNMP for trace lt03t02. Before this shift, one could observe a relatively constant pattern on the number of messages per minute. However, after that, the number of message increases up to 10 times, and it oscillates in a non linear way.

Another execption can be observed in Figure 3.7 for the trace l05t01. After approximately 15 days after the beginning of the capture of the traces, there was a increase for a short time on the number of messages sent (from around 15 to more than 100). We belive this indicates an non expected event, such as power outage, or malfunctioning hardware. However, we do not have more information from the source to identify the exact event that caused this behavior.

3.5 Data Types

The results obtained for the data types are similar to those found in [1]. For example, in most traces the data type `null` is found. The data types observed for all traces can be found in the histograms in Appendix A.

3.6 Managed Objects

There is a significant diversity in the MIBs used among the traces we have analyzed. Still, we can observe that the `interfaces` and `IF-MIB` (both descendants from the `MIB2` tree) are by far the most used MIBs. This is slightly different than observed in [1], where the `IF-MIB` is dominating. Another observation is that the majority of the used MIBs are part of the standardized MIBs, even

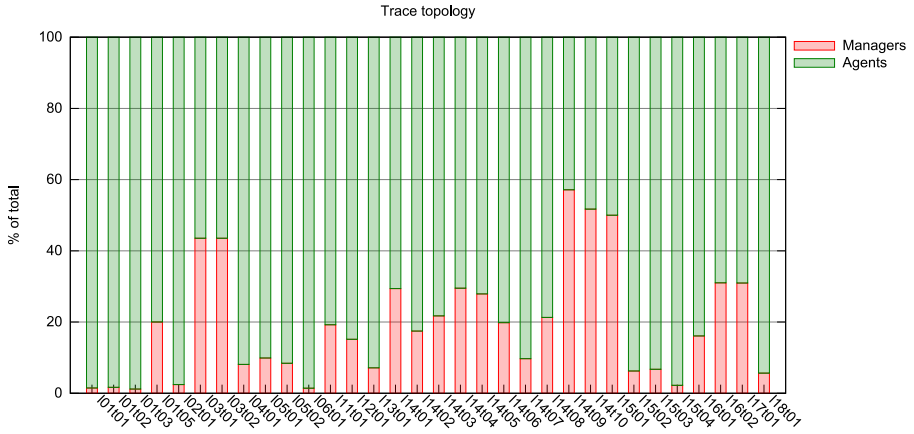

Figure 3.4: Relationship between managers and agents for each trace

though it is known that network vendors have designed their own MIBs and thus those are found as well.

3.7 Notifications

Only 3 traces contain `trap` messages and none of the traces contain `inform` messages. l05t02 contains 83 `trap` messages of SNMPv2c of which 78 signal Cisco related TCP session teardowns and 5 that are related to Cisco configuration changes. The `trap` characteristics of this trace are the same as that of a trace previously analyzed from this same location, as described in [1].

Trace l11t01 contains just 4 `trap` messages of SNMPv2c which inform the manager of a link up/down event at the respective agent. l18t01 Contains the most `trap` messages. It contains 6602 `trap` messages of SNMPv1. By far the most of the OIDs in these trap messages are BPG routing notifications (63%), 35% are interface related notifications (e.g., link up/down notifications). Remaining are a few DS1 line status notifications, Juniper specific and some IPv6 interface table OIDs.

3.8 Flow Analysis

Besides general trace and message analysis, we have also separated all of the traces into flows. A flow is a set of particular messages in a trace that have occurred between two network layer endpoints. For instance, all the messages of a particular category that have been exchanged between a manager and an agent that is being polled every few minutes.

At the moment there are two definitions for a flow being used. One of them has been used in the previously mentioned `snmpdump` tool [1]. However, a recent publication [4] includes the first formal publication of a flow, which differs slightly from the previous one. The most important difference is that requests and responses that are to be considered related to each other in a flow,

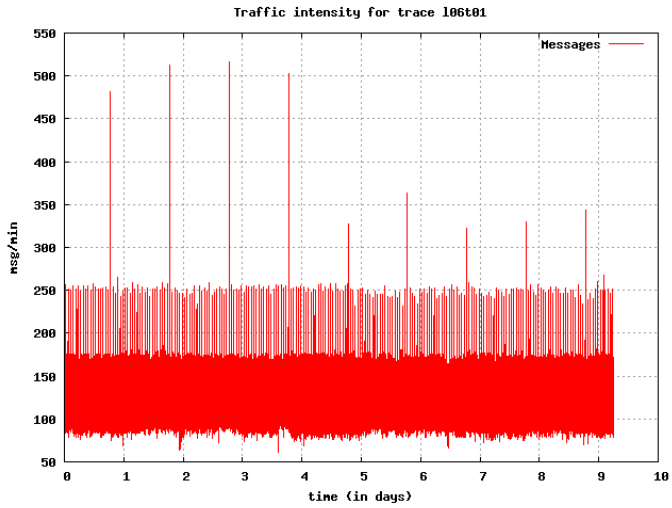

Figure 3.5: Expected behavior of SNMP traffic

must use a single transport layer endpoint pair. The previous 'definition' of a flow only required a single network layer endpoint pair, which is less strict.

Table 3.2 contains an overview of the identified traces in the various traces, based on the definition used in [4]. The first column lists the trace under consideration. The second column shows all the command group flows (i.e., all the flows that involve `get-, get-next-, get-bulk-, set-request` PDU type messages and their related response messages (i.e., PDU types `response` and `report`). The third column shows all notification group flows, which involve PDU types `trap` and `inform` and their related response messages (where applicable). Since all flows are either command or notification group flows, the sum of columns two and three results in the total number of flows identified for that particular trace. The fourth column lists the number of command generators (identified network layer endpoints that have sent at least one of the mentioned request messages for a command group flow). The fifth column lists the number of network layer endpoints that sent response messages (either response or report). Finally, column six and seven do the same for the notification group flows, where 'no' signifies the case of notification generators and 'nr' for the responders.

Due to the fact that the only published definition is quite new, we considered it to be interesting to know to what extent flows, according to the two different definitions, in real world traces would differ between the two cases. After applying a script that can separate a trace into flows according to the two different definitions, we noticed only a difference of 3 messages in all of the traces. All of these 3 messages are response messages. A close inspection shows that all of these were the result of an improper binding with their respective request message. It appears that the respective manager in these cases uses a fixed request identifier for at least a large number of requests sent within a relatively short timeframe (possibly including retransmissions). The result of this, is that it is not possible to properly relate a response to a request in the case of the old definition, which only binds them using the network layer endpoints. The

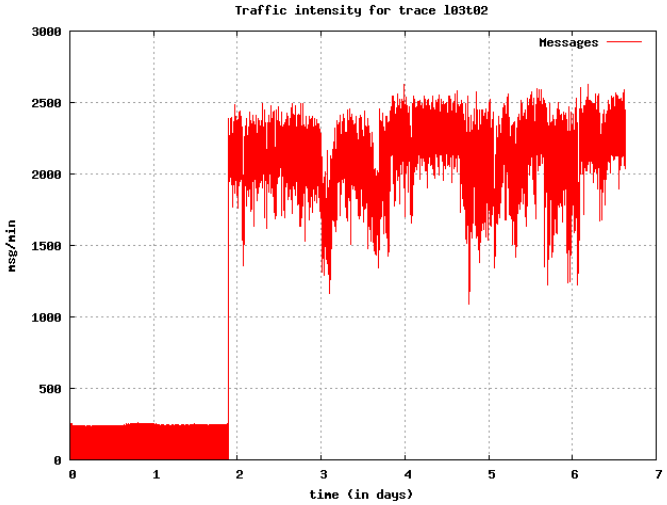

Figure 3.6: Exceptional behavior in trace l03t02

trace	cg/cr flows	no/nr flows	cg	cr	no	nr
l01t01	134	-	2	134	-	-
l01t03	274	-	2	169	-	-
l05t02	80	3	4	73	2	3
l11t01	21	1	5	21	1	1
l13t01	55	-	4	52	-	-
l14t06	105	-	21	85	-	-
l14t07	95	-	10	93	-	-
l14t08	180	-	46	170	-	-
l14t09	46	-	32	24	-	-
l14t10	25	-	15	14	-	-
l15t02	45	-	3	45	-	-
l15t03	139	-	8	111	-	-
l15t04	477	-	10	440	-	-
l16t01	148	-	24	125	-	-
l16t02	127	-	85	189	-	-
l17t01	35	-	13	29	-	-
l18t01	99	29	6	99	29	1

Table 3.2: Flow characteristics of the traces

difference then occurs when the transport layer bindings are used, like in the case of the new definition.

Table 3.3 shows an example segment of a trace, which results in the observed difference when applying the two definitions individually to it. Assume that the maximum time between a request and response is set to 25 seconds and that all of the messages have a request identifier equal to 0. Then, when the old definition of the flow is applied, all `get-requests` and all `responses` are considered part of the same flow, because it is not known to which request a specific response exactly relates when only considering the network layer endpoints. For

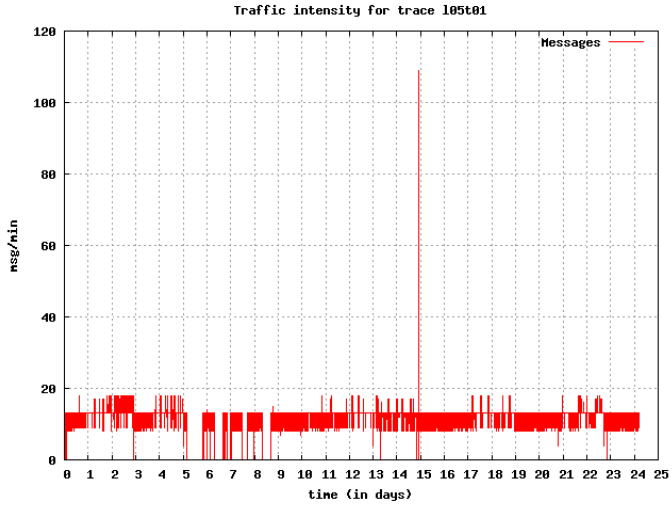

Figure 3.7: Exceptional behavior in trace l05t01

instance, the last `response` shown can be linked to the last `get-request`, resulting in even the last response message being part of the same flow (i.e., the last `response` message is still occurring within 25 seconds after the last `get-request` message). However, when the new definition is applied, the transport later endpoints are being used. This results in, for instance, the only possible relation of the first `response` message to the first `get-request`, which is not allowed with a maximum time between a request and its response of 25 seconds (i.e., the difference between the first `get-request` and the first `response` message is larger than 25 seconds). This results in the first `response` message being rejected from assignment to the same flow. The same can be said about the second `get-request` message and the second `response` message. In the case of this example scenario, we would see a difference of two messages in the two possible flows resulting from applying the two definitions of a flow. Based on the insignificant number of such messages observed in all of the traces under consideration and actual scenario in which they occur (like shown above), we state that the new definition seems to be a correct one, resulting in a proper splitting of traces into flows.

time [s]	**direction**	**PDU type**	**req. ID**
0.65	1.1.1.1:3146 -> 2.2.2.2:161	get-request	0
0.68	1.1.1.1:3201 -> 2.2.2.2:161	get-request	0
22.21	1.1.1.1:3315 -> 2.2.2.2:161	get-request	0
22.25	1.1.1.1:3370 -> 2.2.2.2:161	get-request	0
30.49	2.2.2.2:161 -> 1.1.1.1:3146	response	0
30.50	2.2.2.2:161 -> 1.1.1.1:3201	response	0
30.52	2.2.2.2:161 -> 1.1.1.1:3315	response	0
30.54	2.2.2.2:161 -> 1.1.1.1:3370	response	0

Table 3.3: Example scenario showing difference between flow definitions

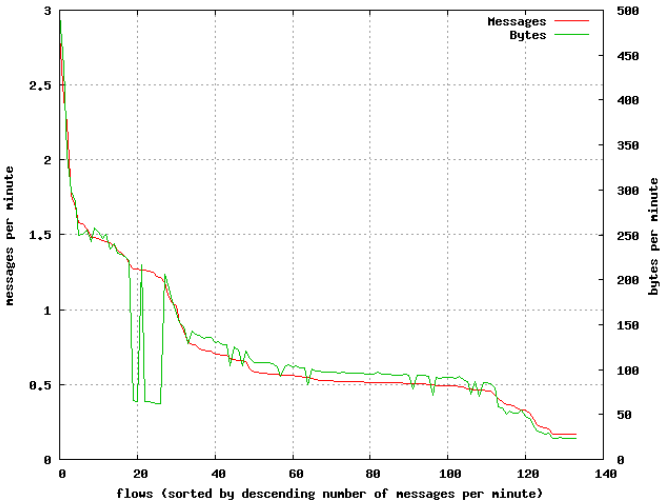

Figure 3.8: Traffic intensity of the flows in trace l01t01

As stated at the beginning of this subsection, all of the traces have been separated into flows. Of two very different traces is the traffic intensity of the flows plotted. Figure 3.8 shows the separation result for trace l01t01, where the number of messages per minute within the various flows are quite close to each other. Still, it can be seen that a few flows have a higher number of messages per minute that most others. Another interesting observation regarding this trace, is that a reduction of messages per minute (reading the graph from left to right), the number of bytes per minute do not decrease necessarily. This could be caused by a manager requesting more information from one agent than another, resulting in varying message sizes.

Another extreme of trace separation into flows, is shown in figure 3.9. This trace shows the typical distribution of the messages per minute in the identified flows of a trace. Namely, just a few flows have a very high messages per minute rate and the greater majority has a far smaller rate. This could simply be the result of an agent being queried more often and/or being queried more date than others, or it could be the result of an erroneous situation. Another possibility is the time-filter mechanism, which can be used to retrieve only those rows of a time filtered table that have changed since a particular time [1]. Appendix A contains the results for the other traces.

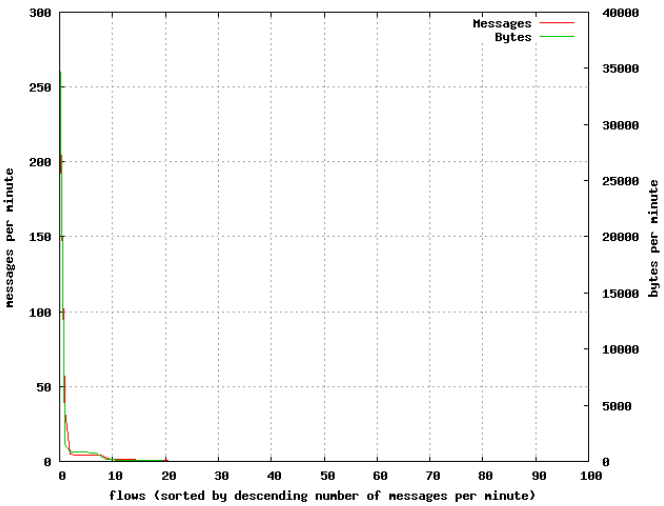

Figure 3.9: Traffic intensity of the flows in trace l14t07

Chapter 4

Conclusions and Future Work

A publication [1] made a significant start with the analysis of real world SNMP trace analysis, by describing a specific toolset for analysis, as well as some statistics that were distilled from some traces. However, real world trace analysis demands a continual approach, due to the changing circumstances of networks and SNMP implementations. This paper reports on a lot more traces than in the mentioned paper, which are also more recent. From our results it can be concluded that no significant changes in the use of SNMP can be detected in the traces considered in this paper.

One of the most prominent observations made in this paper, is regarding the use of the various SNMP versions and the PDU types. Based on the analysis of the 17 traces in this paper, it can be stated that SNMPv3 is still hardly used. Of the 17 traces only 1 contained some SNMPv3 messages, despite the fact that the IETF declared SNMPv1 and SNMPv2 as historic and SNMPv3 as standard. Regarding the use of certain request PDU types, we also do not see significant differences compared to the results in [1]. The same can be stated about the most commonly referenced MIB: `IF-MIB`. However, a difference can be found in the data type distribution, where this paper finds relatively more occurrences of `null` types. Another one is the average response size, which is about 50 bytes in our results, where it is around 300 bytes in [1]. The difference is likely the result of specific trace characteristics, where traces might (not) have `get-bulk-request` messages, which affect this value.

Besides that, some new categories, compared to the previously mentioned paper, were presented in this paper. These are the varbind distributions, the impact of two different flow definitions and a more extensive look at traffic intensity in traces, but also in flows. Interesting observations in these categories are firstly that it appears that a varbind of 1 is most common, which means that requests are generally very small. Secondly, the impact of the two definitions of a flow have been compared, which resulted in the finding that the introduction of the flow definition given in [4] seems correct (with respect to the made assumptions) and therefore does not result in a different separation of traces into flows compared to the previously, non-formalized definition of a flow.

Even though the results of this paper suggest that no significant changes can

be stated compared to the previous paper ([1]), the analysis of SNMP traces should remain a continual process. This, because network environments change, but also SNMP engine implementations.

Acknowledgment

This research work has been supported by the EC IST-EMANICS Network of Excellence (#26854).

Bibliography

[1] J. Schoenwaelder, A. Pras, M. Harvan, J. Schippers, and R. van de Meent. Snmp traffic analysis: Approaches, tools, and first results. In *Proceedings of the Tenth International Symposium on Integrated Network Management, Munich, Germany*, pages 324–332, Piscataway, May 2007. IEEE Computer Society Press.

[2] R. Presuhn. Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP). RFC 3416 (Standard), December 2002.

[3] D. Harrington, R. Presuhn, and B. Wijnen. An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks. RFC 3411 (Standard), December 2002. Updated by RFCs 5343, 5590.

[4] G. van den Broek, J. Schoenwaelder, A. Pras, and M. Harvan. Snmp trace analysis definitions. In *Resilient Networks and Services, Bremen*, volume 5127 of *Lecture Notes in Computer Science*, pages 134–147, London, July 2008. Springer Verlag.

Appendix A

Appendix A