Measurement-based characterization of the load of a Mobile IPv6's Home Agent

Cristian Mihai
Albert Cabellos-Aparicio
Jordi Domingo-Pascual

TABLE OF CONTENTS

Measurement-based characterization of the load of a Mobile IPv6's Home Agent

Mihai CRISTIAN, E-mail: cristian.mihai2007@yahoo.com

ABSTRACT

Wireless technologies are rapidly evolving and the users are demanding the possibility of changing their point of attachment to the Internet (i.e Access Router) without breaking the IP communications. This can be achieved by using Mobile IPv6. However mobile clients must forward their data packets addressed towards their home network through a special entity, the Home Agent (HA). This HA is a key point when considering the performance of Mobile IPv6-based networks. This paper presents the first steps towards characterizing the load of a HA. This may be useful both for researchers that aim to propose novel architectures that improve the performance of the HAs and for ISPs willing to deploy Mobile IPv6. To achieve our goals first we analyze the internal traffic of a medium-size department. Then we review existing models and evaluate their applicability to this particular scenario. Our results show that the estimated load of a HA serving a medium-size department (around 1500 hosts) is high with a maximum throughput of 262Mbps. Additionally we show that existing models of Wireless LAN networks can be applied to this scenario.

Index Terms—Mobile IPv6, Home Agent, Characterization, Modeling

1. INTRODUCTION

Wireless technologies have rapidly evolved in recent years. IEEE 802.11 is one of the most used wireless technologies and it provides up to 54Mbps of bandwidth in an easy an affordable way. In the current Internet status a user can be connected through a wireless link but he cannot move (i.e. change its access router) without breaking the IP communications. That's why IETF designed Mobile IP which provides mobility to the Internet. With "mobility", a user can move and change his point of attachment to the Internet without losing his network connections.

In Mobile IP a Mobile Node (MN) has two IP addresses. The first one identifies the MN's identity (Home Address, HoA) while the second one identifies the MN's current location (Care-of Address, CoA). The MN will always be reachable through its HoA while it will change its CoA according to its movements. A special entity called Home Agent (HA) placed at the MN's home network will maintain bindings between the MN's HoA and CoA addresses.

The main limitation of Mobile IP is that communications between the MN and its peers are routed through the HA. This means that a HA may be responsible of multiple MNs on a Home Link. The failure of a single HA may then result in the loss of

connectivity of numerous MNs. Thus, HAs represent the possibility of a single point of failure in Mobile IP-based networks. Moreover MN's communications through the HA may also lead to either the HA or the Home Link becoming the bottleneck of the system. In addition, the HA's operation such as security check, packet interception and tunneling might not be as optimized in the HA's software as plain packet forwarding.

The transmission of datagrams to the mobile node while it is away from its home network is achieved through a mechanism called tunneling. At the home network, the HA intercepts datagrams that are destined for the mobile node and tunnels them to the CoA addres it has in it's binding. This CoA address can be directly the node itself or another mobility agent situated in the visited network called Foreign Agent(FA). Mobile IP also makes use of reverse tunneling which implies that another tunnel is created from the FA back to the HA, when the mobile node sends datagrams. This mechanism however leads to an abnormal routing scheme and increased delay in delivering datagrams. Also it adds to the load of the HA as mentioned above.

Mobile IP comes into two flavors, Mobile IPv4 [13] and Mobile IPv6 [14]. Mobile IPv6 outperforms Mobile IPv4 in many aspects because of the improvemens that the new IPv6 addressing scheme and IPv6 headers present. The IPv6 protocol and two new protocols designed for IPv6: *Neighbour Discovery* and *Stateless Adress Autoconfiguration* form an almost perfect protocol basis for mobile networking. The basic functionality of the Mobile IP protocol remains the same. What has changed is that the mobile node now has an ensured capability to obtain a CoA by using the above mentioned address configuration protocols. Thus the need for a Foreign Agent to assign CoA adresses is greatly reduced. Moreover, in order to provide route optimization, IPv6 offers the posibility of maintaining binding updates, that can be sent not only to the HA, but also to correspondent nodes. This feature can be implemented using the newly defined *destination options* of IPv6. Since destination options are only inspected by the destination, there is no performance penalty at intermediate routers allong the transmission path for using them. The use of such options that can be placed in extension headers defined by IPv6 reduces the quantity of overhead information involved in sending binding updates to correspondent nodes. The binding update can be included in a normal packet that the mobile node sends to a correspondent node anyway. These binding updates allow a mobile node that uses Mobile IPv6 to communicate directly with its peers. This means that the packets sent to corresponding nodes are no longer routed through the HA in the home network. Thus the Mobile IPv6 HA now only has to deal with packets sent between mobile nodes that belong to the same home network. This paper focuses on a Mobile IPv6's HA.

The research community has focused on solving these issues proposing novel architectures that improve both the performance and the reliability of the HAs[15,16,17,18]. Although they are very effective, Mobile IPv6 has not been deployed yet. This means that the load of a Mobile IPv6's HA is unknown. Hence the proposed architectures might have been evaluated with an unrealistic load. On the other hand ISP's willing to deploy Mobile IPv6 need an estimation of the expected load that the deployed Home Agents will have. That is why we believe that modeling the load of a Mobile IPv6 HA is important for the research community and for the industry.

This paper presents the firsts steps towards characterizing the load of a Mobile IPv6 Home Agent. First we have analyzed the internal traffic of a medium-size department of the UPC. We have assumed that all the hosts inside the department are Mobile IPv6 nodes that are away of their Home Network. As explained earlyer all the internal traffic must be processed by an hypothetical Mobile IPv6's HA. Second we have reviewed existing models of traffic that can be applied to this particular scenario. Specifically we have focused on the models that characterize the load of Wireless LAN networks [1,2]. We have evaluated its goodness-of-fit for our particular case. Finally we have analyzed at which granularity these models can be applied.

Our results show that high processing power is needed when deploying a Home Agent in a medium size scenario. Also we show that existing models for flow-level variables such as flow size and flow inter-arrival times apply to our data on the aggregated traffic level, even if these models have been estimated under different circumstances. Finally we find that these models would also apply to the subnetwork traffic if the number of clients within the subnets is high enough.

The rest of the paper is organized as follows, Section II describes the tools used for this particular study, Section III presents an overall look of the datasets used in terms of internal vs external traffic, transport and application protocols, Section IV contains an empirical characterization of the load of our Home Agent, Section V reviews other models that fit with our estimates for the aggregate traffic, Section VI looks to the traffic generated per sub-net and Section VII contains our conclusions.

2. TOOLS USED FOR THE STUDY

In this secition we describe the tools used for gathering the data and obtaining our results. These comprise NetFlow records, flow-tools which is a collection of applications used for processing NetFlow traces, PERL scripts developed for computing several parameters such as average flow size, inter-flow arrival times, throughput,etc and MatLab scripts for plotting our results and fitting of distributions.

2.1.NetFlow

NetFlow is an open but proprietary network protocol developed by Cisco Systems to run on Cisco-IOS enabled equipment for collecting IP traffic information. The data used for this study comprises NetFlow records from a department router at UPC for six days of traffic.

The principle of Netflow is as follows: when the router receives a packet, its NetFlow module scans the source IP address, the destination IP address, the source port, the destination port, the protocol type, TOS field, the login input (or output) port of the network equipmet of the IP packets, judges whether it belongs to a flow that already exists. If so, update the flow record; otherwise, a new flow record is created in cache. The expired flow records in cache are exported periodically to a destination IP address using UDP. The most common export format is NetFlow V5 which was used in our case as well. This format contains the following information for each flow: source and destination IP address, source and destination port, protocol type, start and end timestamps, the number of octets and packets within each flow.

2.2.Flow-tools

Flow-tools is a collection of applications used to collect, send, process and generate reports from NetFlow data. We used flow-tools especially for filtering the data and generating some reports about the make-up of the traffic. The following applications have been used:

- Flow-print : **flow-print** will display flow data in ASCII using predefined formats that can be selected using the –f parameter. The synopsis is the following:

  ```
  Flow-print -f5<flow-file
  ```

  The output file created using the number 5 format will contain information about each flow on one line with the same fields contained in the NetFlow V5 format

- Flow-filter : The **flow-filter** utility will filter flows based on user selectable criteria. The IP address filters are defined in `flow.acl` or by the filename specified by -f. The synopsis is the following:

  ```
  Cat flowfile|flow-filter -f filter_config>output
  ```

  A typical filter configuration file contains the IP addresses and masks along with the *permit* or *deny* parameter to specify how to handle flows. An example is shown below:

  ```
  Ip access-list standard name permit IP address mask
  Ip access-list standard name deny IP address mask
  ```

- Flow-nfilter: the **flow-nfilter** utility is simmilar to flow-filter but allows for more complex filtering using several criteria such as start and end times or source-destination ports. Filters are defined in a configuration file and are composed of primitives and a definition. Primitives define the type of the filter and the values that are to be allowed or denied. Definitions contain match lines grouped to form logical AND and OR operations on the flow using the selected primitives. An example is shown below for filtering based on IP address and port number:

  ```
  filter-primitive adr
  type ip-address
  permit 10.0.0.1
  deny 10.0.0.2

  filter-primitive port
  type ip-port
  permit 80

  filter-definition test
  match ip-source-address adr
  ```

```
match ip-destination-port port
or
match ip-source-address adr
match ip-source-port port
```

After such a file is created the filter can be applied to the NetFlow record using the following command:

```
Cat flowfile|flow-nfilter -f config_file -F definition>outputfile
```

- Flow-cat: flow-cat is used for concatenating flow files. The following command can be used:

```
Flow-cat flowfiles*>output_file
```

- Flow-stat: The **flow-stat** utility generates usage reports for flow data sets by IP address, IP address pairs, ports, packets, bytes, interfaces, next hops, autonomous systems, ToS bits, exporters, and tags. The type of the report desired is specified using the –f parameter. The following command is used:

```
Cat flowfile|flow-stat -f report_type(0-32)>output
```

Other parameters can aslo be specified such as options for sorting the output, adding a title and header to the report or reporting the results in percent values.

- Flow-report: the **flow-report** utility is also used to generate reports for netflow files but allows for more options including choosing which fields a report should contain, including filters and combining different reports. Reports are definied in a configuration file by the 'stat-report' keyword followed by a report name. Each report has a type defined below and other commands. Reports are grouped into a definition with the 'stat-definition' keyword followed by a definition name. Each definition can invoke a filter and optionally apply tags. A configuration file example is shown below:

```
stat-report t1
type summary-detail
scale 100
output
        format ascii
        options +header,+xheader,+totals
        fields +other

stat-report t6
type ip-source-port
output
        format ascii
        options +header,+xheader,+totals,+names,+percent-total
        sort +pps
```

```
stat-definition test
report t1
report t6
```

The report can then be used on a NetFlow file using the command:

```
Cat flowfile|flow-report -s config_file -S definition
```

2.3. PERL scripts

In order to evaluate certain parameters for the traffic we created two PERl scripts: **final.pl** and **diff.pl**. These scripts work on output files provided by the flow-print utility and provide results that can be stored in normal files for plotting.

The first script is used for evaluating the following parameters: flow-arrivals, number of packets, number of octets, bps, pps, number of flows per second and average flow size. The interval on which to average can be modified. For our study we used one hour intervals to determine flow-arrival patterns for the aggregated traffic for all six days, and one minute intervals for computing the parameters for each day of traffic. A time counter is used to achieve this. It is initialized with the timestamp of the first flow inside the NetFlow record and compared to the timestamps of the flows as the file is being processed. If the difference between the counter and a timestamp is bigger than one minute(or another selected interval) the counter is incremented. For each given minute several counters are incremented in order to determine the number of flows that are active in that interval, and the number of octets and packets they carry. Then for each interval the set of mentioned parameters is computed and printed.

The input parameter of the script is the input file which as stated is a flow-print produced file or a text file which contains the following fields:

```
start end sif sourceip srcport dif destip destport protocol Flag octets packets
```

The script is run like any PERL script using the following command:

```
Perl final.pl input_file>output_file
```

and it produces an output file containing the following fields for each of the intervals found: number of active flows, number of octets, number of packets, bps, pps, active flows per second and average flow size.

The script can be modified easily to work with other formats for the imput and output files and any intervals can be used for averaging.

The second script uses the same input file described and computes the inter-flow arrival times for a given NetFlow trace. The way this is done si by simply computing the difference between the start timestamps of two consecutive flows. The syntax for running the script is the same as the one used for the first script and the output file is a one column file containing the computed inter-flow arrivals.

2.4. MatLab

Matlab is a numerical computing enviroment and programming language that allows easy matrix manipulation, plotting of functions and data, implementation of algorithms, creation of user interfaces and interfacing with programs in other languages. Matlab also contains a set of toolboxes that can be used to plot and interpret data.

In our case Matlab was used to plot the parameters obtained using the above mentioned scrips. The plots used vary from simple plots, to bar graphs and CDF functions which are already implemented as functions in Matlab libraries.

In order to produce such plots the output files of the PERL scripts have to be first loaded into Matlab using a simple load command. This produces a data structure that contains the data from the output files divided into columns. Each column can then be plotted using the plot functions implemented in Matlab: plot(data), bar(data), cdfplot(data), histogram(data). We have written a couple of simple scripts for loading and plotting data in Matlab.

Matlab also offers a Distribution Fitting Tool that was used to fit currently existing mathematical distributions to our empirical data in our effort to model it. The application is started from the Matlab command line by typing **dfittool.** A user friendly graphical interface will the appear.

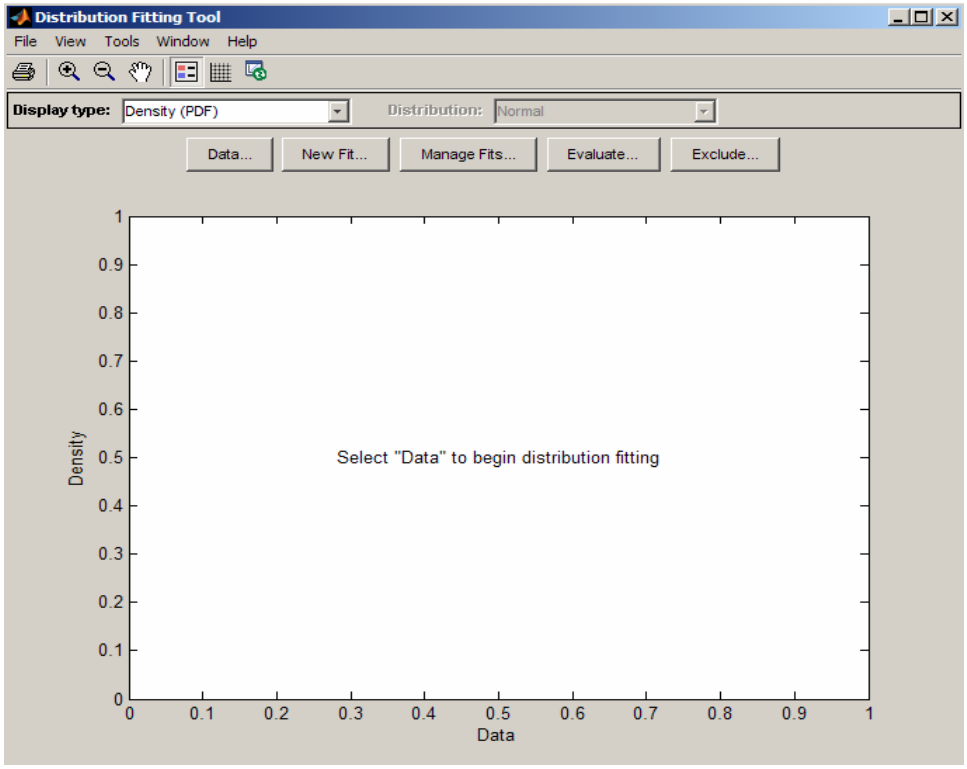

Figure 1: Distribution Fitting Tool Interface

First a user must load the data into the distrbution fitting tool. This is done by clicking the Data button. Any arrays present in the current Matlab workspace can be loaded. Once the data is loaded a display type must be selected. The given choices are: Density(PDF), Cummulative Probability(CDF), Quantile(Inverse CDF), Probability plot, Survivor function and Cummulative hazard function.

Then, by choosing New Fit, the user can select the distribution that is to be fitted on the loaded data. Choices include Normal, LogNormal, Generalized Pareto, Exponential, Extreme Value and other distributions. After a distribution is fitted it will be plotted against the loaded data in the main window of the application. The parameters of the distribution such as mean and variance are also shown.

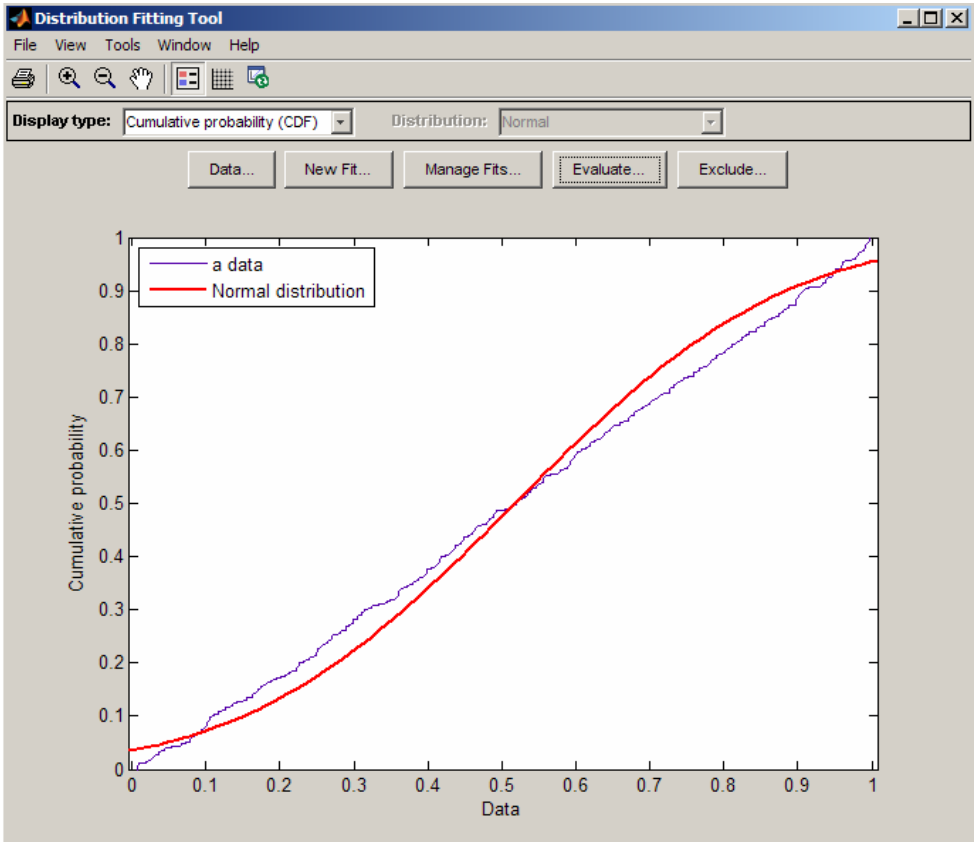

Figure 2: Distribution Fitting example

The fit can be evaluated by clicking the Evaluate button. The application computes the confidence bounds for the fitted distribution. If the data is within these confidence bounds than the fit can be considered optimal. More than one distribution can be selected in order to determine the best fit for the data.

3.DATA ACQUISITION AND TRAFFIC BREAKDOWN

In this section we present an overlook of the datasets used for this study and focus on the structure of the traffic that is relevant for the goal of this study. We describe how the traffic was divided in terms of Internal an External traffic and examine the main components of the traces on the transport and application layer.

3.1. Internal versus External

The data analyzed comprises NetFlow records from a department router at UPC Barcelona, for six days of traffic, from March 9 to March 14, 2007 both inside the department and to/from the Internet. For the 144 hours of traffic monitored we found a total of 903.7 Gigabytes. Figure 3 shows that 95% of all flows, octets and packets belong to traffic to and from the Internet and only 5% represents the traffic between hosts inside the department.

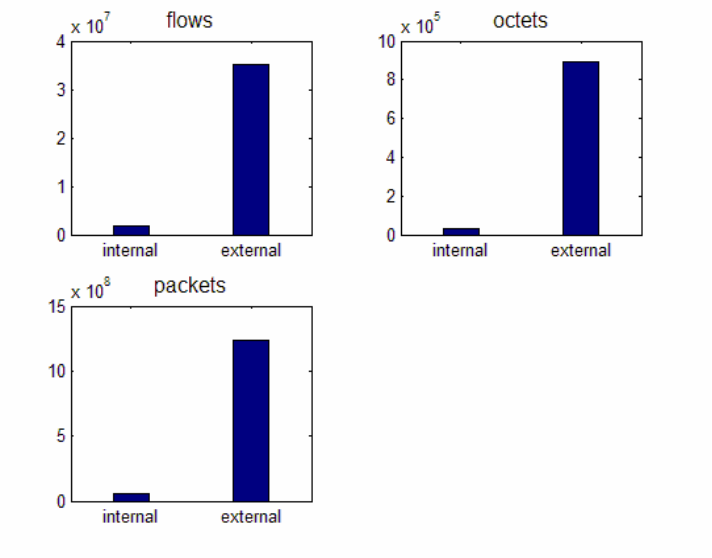

Figure 3: Internal versus External traffic

We base our study on the assumption that all clients inside the department are Mobile IPv6 clients. Following this assumption the traffic between clients inside the department (labeled internal) should be routed by the Home Agent. Traffic to and from the Internet (labeled external) is delivered to its destination directly using IPv6 extension headers and the Return Routability procedure [11]. As our aim is to characterize the load of the Home Agent we disregard the external traffic.

It is worth noting here that Mobile IPv6 clients can also communicate directly with its peers, even if they are at its Home Network. However these communications must be secure and this is achieved by routing them through the Home Agent. Additionally this route does not affect the performance of the communications since the Home Agent and the peers of the Home Network are very close.

Internal traffic is the traffic between Mobile IPv6 clients and their Home Network (the servers in the department). The number of hosts (Mobile IPv6 clients) inside the department sending and receiving packets is 1547. They account for a total of 32.95 Gigabytes of traffic on 1773751 flows. We provide an analysis of this traffic the next sections. These hosts are divided into 7 different subnets each with a number of hosts ranging from 140 to 220 (subnets 1 to 7).The addresses of the hosts have been anonimized.

3.2. Traffic breakdown

In order to efficiently characterize the load of the Home Agent we must first understand the makeup of the traffic that we are analyzing. To do this we chose a similar approach to the one presented in [3, 8], that is examining the traffic on the transport and application layer.

In Table 1 we break down the traffic by transport protocol in terms of flows, octets and packets.As expected most of the octets are sent using the TCP protocol. The percent of the ICMP traffic increases during the second and third day, Saturday and Sunday, but this is only because the total traffic on these days is less then the average (less staff is present form the department during the weekend). The TCP and UDP traffic remains fairly constant during all the days. We find that the bulk of the traffic is sent using the TCP protocol for reasons explained below.

Protocol	flows	octets	packets
17(udp)	28.75%	0.81%	3.73%
6(tcp)	67.87%	99.13%	95.84%
1(icmp)	3.37%	0.05%	0.42%

Table 1: Fraction of flows, octets and packets for different transport protocols

Next we take a look to the application layer. To categorize the traffic inside the department we grouped the applications into several high-level categories.

Table 2 shows the main applications found. The applications have been identified using the flow destination port. Only the most important applications are shown, the ones that account for most of our flows (minor applications have been ignored). The main applications found are consistent with the deployed services in the department.

category	**Protocols**
Bulk	ftp,https,tftp,rtip
Email	smtp,imap,pop,brutus,pop3s
interactive	telnet,ssh
Name	Dns,netbios-ns
net-manage	dhcp,ntp,epmap,snmp,timed
Web	http,https
Windows	netbios-ssn,netbios-dgm
authentication	Ident
Printing	lpp
streaming	hp-pdl-datastr

Table 2: Applications and their respective protocols

Figure 4 shows the application usage for the aggregated traffic (all seven subnets). The percentage ratio changes depending on the week day, as explained earlier. We can clearly see that the most used applications both in terms of flows and packets are email and interactive (especially SSH in our case).In terms of flows email applications represent roughly 50% and SSH only 5%, whereas in terms of octets SSH accounts for almost 60% percent of the traffic, which indicates that it was used for large file transfers in the period monitored. Other applications show a normal ratio between the flows and octets percentage.

The figure also reflects the findings in Table 1. Most of the application layer protocols found use the TCP protocol (web, email, SSH) which explains the 99% octets transmitted using TCP, whereas in terms of flows only 67% percent used TCP, mainly because of the name and net management applications which use the UDP protocol.

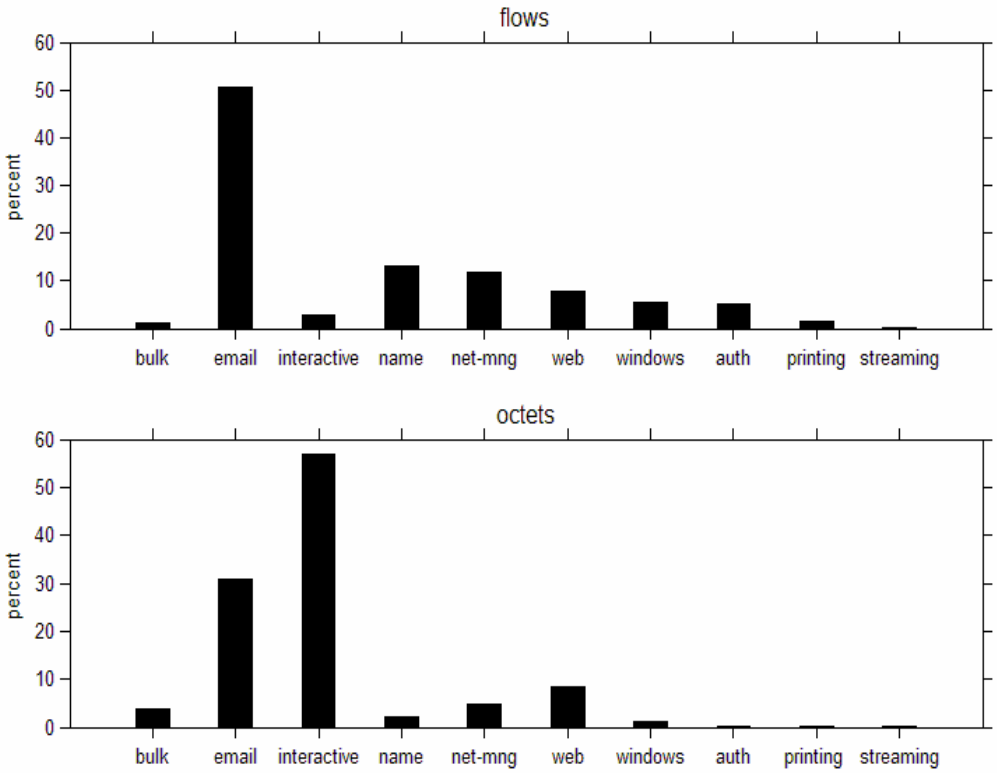

Figure 4:Application usage in terms of flows and octets

Finally it is worth noting here that [3] shows a similar analysis regarding internal enterprise traffic and finds the same pattern of large numbers of bytes traversing the network on a small number of flows.

4. EMPIRICAL ANALYSIS OF THE LOAD OF A HOME AGENT

In this section we aim to provide an empirical characterization of the traffic that a Home Agent in a medium size department would have to process. The department contains a number of 1547 hosts (assumed as IPv6 clients), which account for a total of 32.95 Gigabytes of traffic during the one week interval. In order to characterize this load we have computed a series of parameters such as number of flows per second, flow size, flow inter-arrival times, throughput in bites per second and packets per second.

We start by looking at the flow arrival process to see if we can observe any patterns that could be helpful in our characterization. Figure 5 plots the time series of the flow arrivals for the entire network using one hour bins. The plot shows sharp increases in the number of flow arrivals in the morning with peaks at 28000 flows per hour during weekdays and 9000 flows per hour during the weekend.

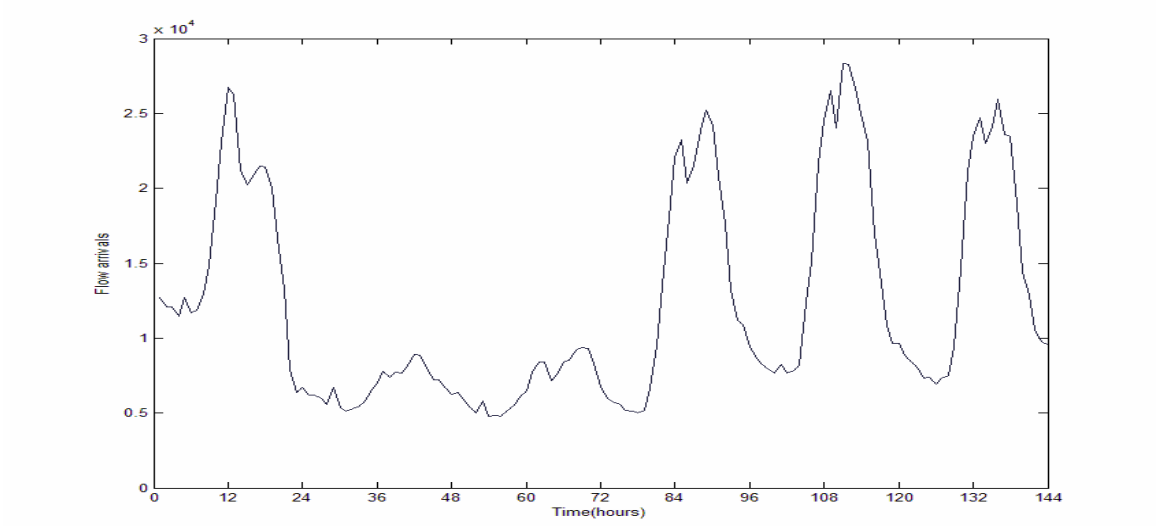

Figure 5: Flow arrival pattern

Another important flow-level variable for the traffic load is the number of flows per second. This parameter represents the number of active flows that, for each second, a HA has to handle. This allows us to see what kind of loads our Home Agent should expect to process when deployed in a similar environment to that of the department. Figure 6 plots the CDF of the average flows per second for each of the six days of traffic. The plots show a clear distinction between the weekdays and the weekend as observed before. For the weekdays we find a mean value of 4.67 flows per second and a maximum of 35.5 flows per second, while during the weekend the mean drops to 1.9 flows per second. The large difference between the weekday and weekend plots also indicates that modeling this variable using a single parametric distribution is rather difficult.

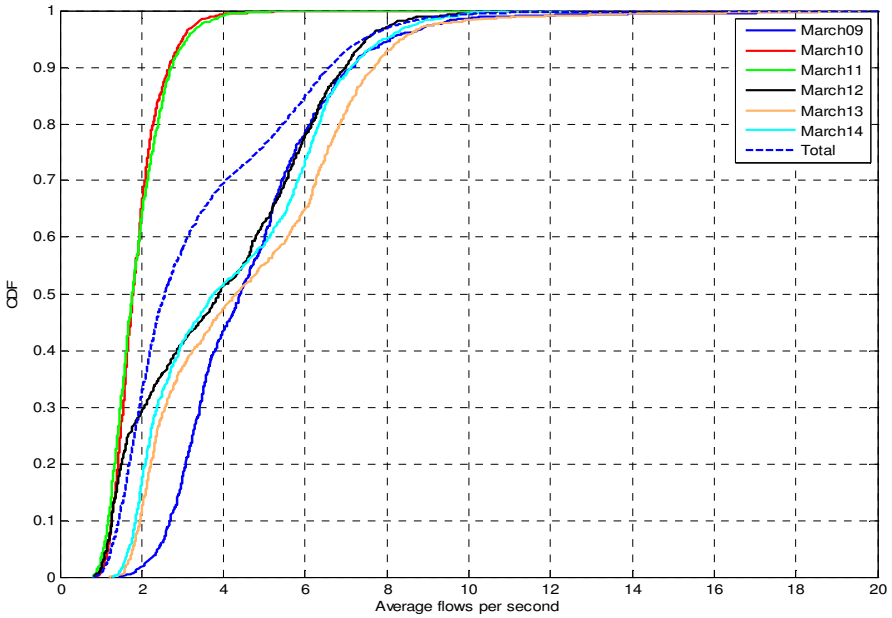

Figure 6:Distribution of average flows per second

Throughput parameters have also been computed for this scenario for all the days using one minute intervals for averaging. Table 3 summarizes our findings regarding these particular parameters. For each day we compute the mean and maximum throughput in bits and packets per second (bps and pps) and the mean and maximum active flows per second (fps). The mean and maximum values for the throughput are shown in Megabits per second.

Day	**Mean Mbps**	**Max Mbps**	**Mean pps**	**Max pps**	**Mean fps**	**Max fps**
1	1.39	102.95	217.3	12.66K	4.73	20.8
2	0.03	3.28	18.2	0.29K	1.9	18.41
3	0.04	4.35	22.36	7.54K	1.89	14.98
4	0.68	61.58	123.69	7.33K	4	35.75
5	0.63	262.24	124.26	24.66K	4.67	19.7
6	0.9	123.48	191.58	14.61K	4.27	12.46

Table 3: Throughput values for the aggregated traffic

We can see that during the six day monitoring period our hypothetical HA would have to process mean values of up to 1.39Mbps and 217.63 pps. Maximum values reach 262.24Mbps and 24.66 Kpps (Kilopackets per second) on day 5. We can also distinguish a pattern that holds for all the mean values in the table. Both bps and pps mean values start from a low value in the weekend (days 2 and 3) and increase gradually, peaking on the last day of the week (day 1).

The next component we look at is the flow inter-arival time distribution. This parameter is essential for estimating a Kendall queuing model [12] for our deployed HA and was computed in milliseconds. Figure 7 plots the distribution for the six days monitored. We can see that there is still a difference between the weekdays and weekends but it is much smaller than in the case of flows per second or throughput. Mean values for the flow inter-arrival time are 279 ms for the weekdays and 618 ms for the weekend, which is consistent with the plot in Figure 5 where we see a smaller number of flows arriving during the weekend.

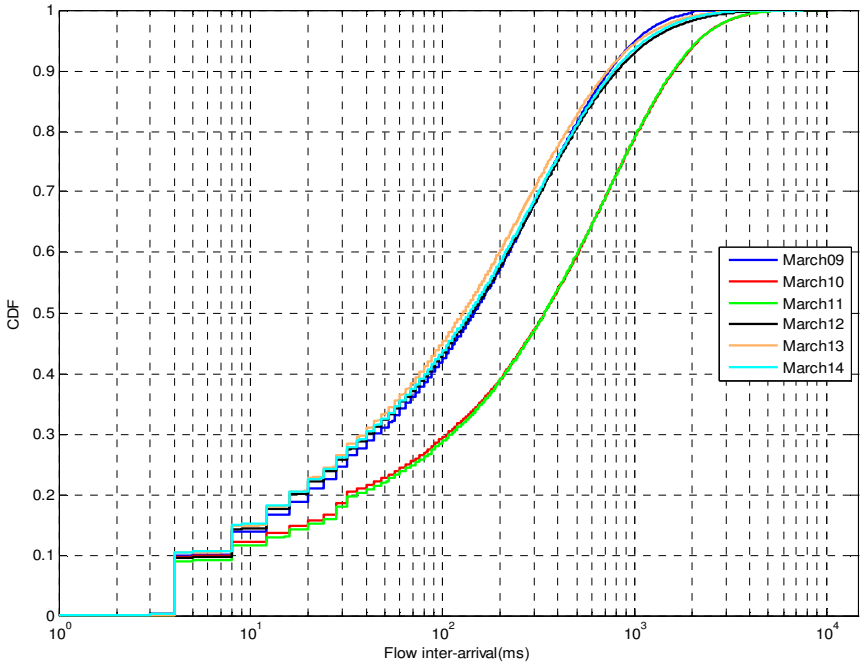

Figure 7:CDF plot of flow inter-arrivals

In order to provide a good characterization of the load of our hypothetical home agent it is not enough to look at the flow arrival process but also at the flow sizes to find out how much each flow carries in terms of bytes.

Average flow sizes have been computed for one minute intervals, for everyday of traffic, by dividing the total number of octets to the total number of flows inside the bin. The average flow size distribution is plotted in Figure 8. Flow sizes show mean values of 10.4 Kilobytes per flow during the weekdays and 2.47 Kilobytes per flow in the weekend.

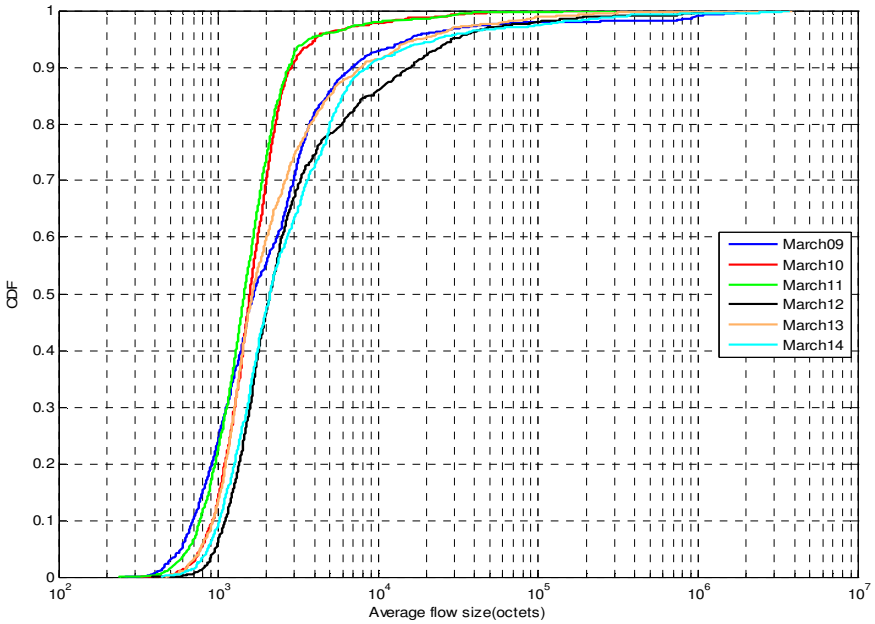

Figure 8: Average flow size distribution for all days of traffic

Summarizing, this section attempts to provide an empirical characterization of the load of a HA. With a total of roughly 1500 Mobile IPv6 nodes, a HA would have to process a mean value of 3.57 active flows per second and should handle mean throughput values of 0.67Mbps and 120 pps. Maximum throughput values of 262.24Mbps indicate that high processing power is needed to deploy as HA in a similar scenario. Also we have spotted patterns in the the flow arrival process, flow inter-arrival times and the average flow-sizes for hour traffic which indicate that these variables can be modeled using parametric distributions.

5.MODELLING THE DATA

To the best of our knowledge no studies exist aiming to characterize the load of a Mobile IPv6 Home Agent. Nevertheless other researchers have characterized similar loads. In this section we aim to evaluate if these existing models fit on our empirical data. A similar analysis to ours is found in [1]. However the authors attempt to model the traffic for a campus WirelessLAN, not for a Home Agent as is our case. Their approach is based on two levels of modeling: the session and the flow level. Flow arrivals in [1] are considered as a cluster process triggered by session arrivals which is not the case for our study. We focus our analysis entirely on the flow level. Additionally the load characterized both in [1] as well as our analyzed data are highly dependent on the applications deployed. Application usage differs when looking at a Wireless LAN or a HA load which could lead to different load characteristics. We will try to see if the models proposed fit with our empirical data.

We find that our flow arrivals pattern (Figure 5) is very similar to the session arrivals pattern depicted in [1] which leads us to believe that models for other flow

variables might apply in our scenario as well. Variables on the flow level have been modeled for WLAN traffic using the following distributions: flow-size using a Pareto distribution and flow inter-arrival time using the log-normal distribution.

We will evaluate the fits of the proposed distributions to see if they match our findings. Figure 9 plots the flow inter-arrival empirical distribution against the proposed LogNormal distribution with the following parameters: mean=365.72, mu(log location)=5.77 and sigma(log scale)=0.49, with it's 95% confidence bounds. The empirical distribution remains within the confidence bounds of the LogNormal distribution, but in the tail section, which contains mostly extreme values, it varies from lognormality.

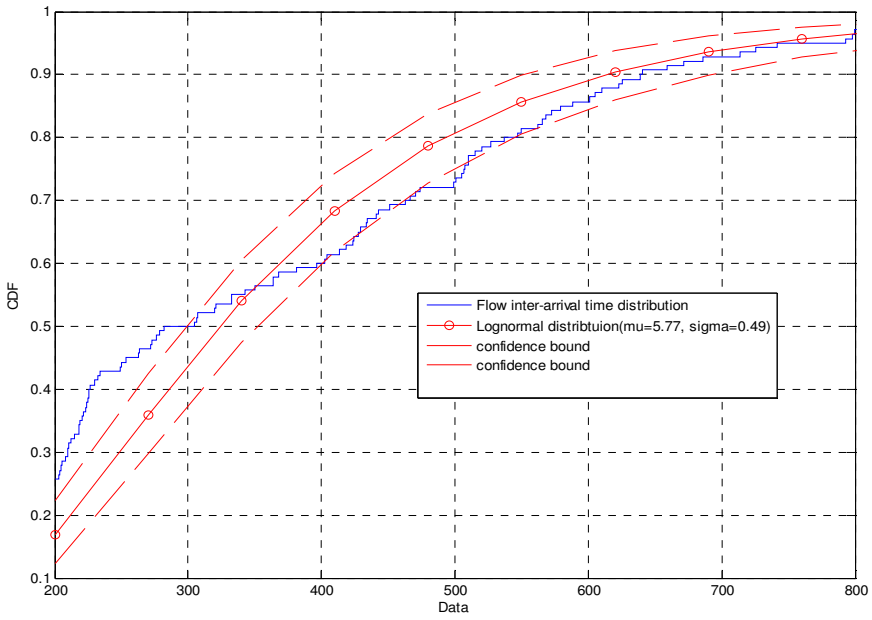

Figure 9: CDF plot of Lognormal distribution plotted against empirical distribution of flow inter-arrival times

When attempting to fit the proposed distribution in [1] for the flow size empirical data we come across the same issues. Figure 10 plots the empirical distribution of the flow sizes with is confidence bounds against the proposed Pareto distribution with the following parameters: mean=3099.9, k(shape)=0.63, sigma(scale)=1234.18 and theta =635. Again we can see that the proposed model provides a good fit for our empirical data, with the same problem in the tail of the curve where the empirical distribution has a higher skew.

We also found that the generalized extreme value distribution provides a slightly better fit especially for the mentioned tail section because this section contains mostly extreme values of the plotted data.

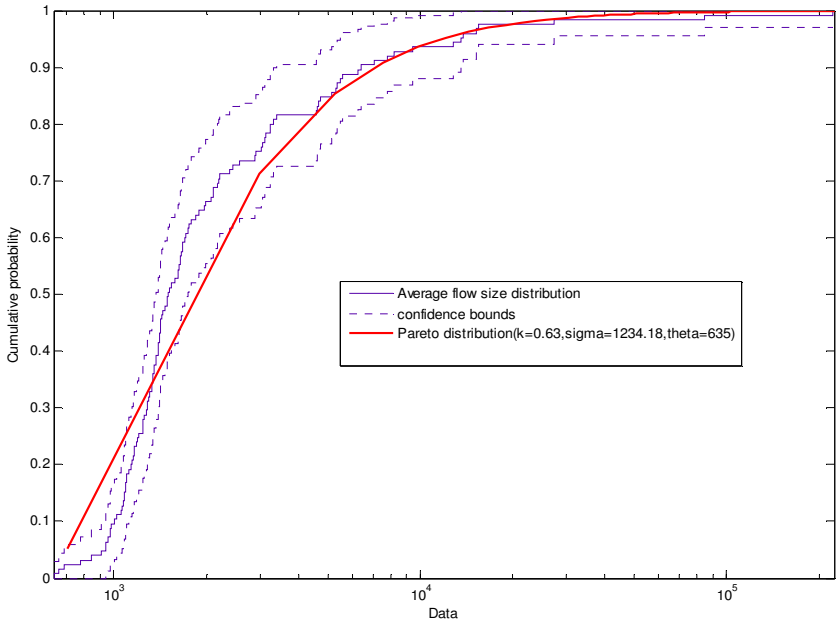

Figure 10: Generalized Pareto distribution plotteg against empirical distribution of average flow sizes

Our findings indicate that flow variables for the load of a HA can be modeled using statistical distributions. Existing models found in[1] provide a good fit for our empirical data and can be used to model the load of a hypothetical HA in a scenario with parameters similar to ours (number of hosts and deployed services).

6.SUB-NETWORK TRAFFIC

In this section we try to find patterns in the per-subnet traffic and fit them to the above mentioned models. This could be useful for adapting the models to suit a scenario with any given number of subnetworks. The same parameters presented in Section III have been computed for each of the seven subnetworks in the department using one minute intervals as stated before.

Figure 11 plots the average flow size distribution for each of the seven subnetworks using dotted lines and the average flow size for the aggregated traffic using a solid line. For this parameter we can see that subnet 1 and 4 follow the Pareto distribution of the aggregate traffic. Other subnets can be grouped together according to their distribution, for instance subnets 5 and 3 have similar flow size distributions, but subnet 6 and subnets 7 and 2 do not resemble the model of the aggregated traffic.

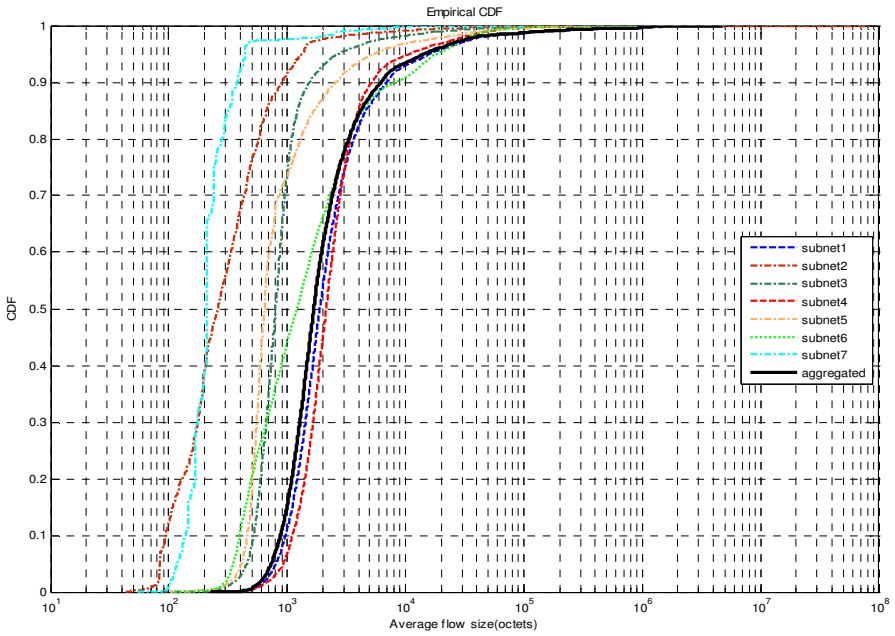

Figure 11: Average flow-size distributions for all subnets

In terms of flow inter-arrival times, plotted in Figure 10, we see a higher resemblance between the distribution of the subnets and the LogNormal distribution of the aggregated traffic. As before subnets 1 and 4 show the closest match whereas subnet 2 shows a completely different distribution.

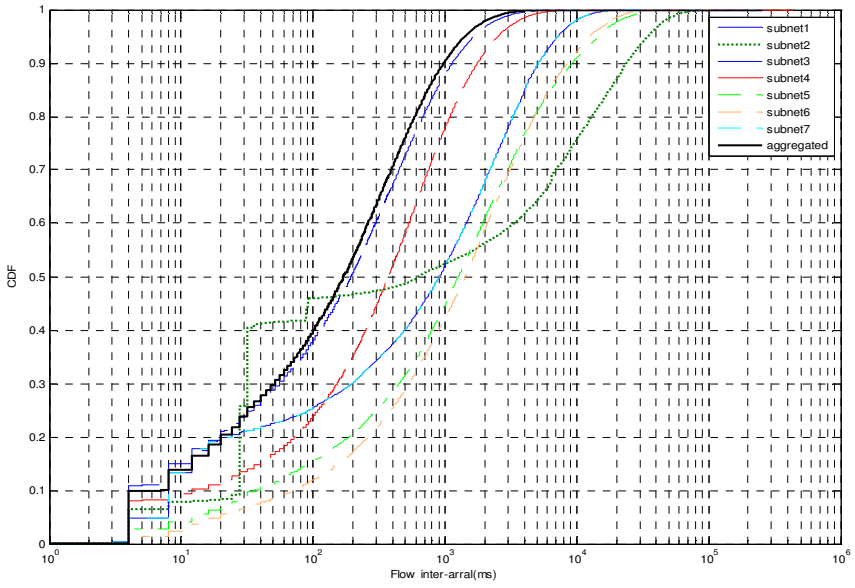

Figure 12:CDF plot of flow inter-arrivals per sub-network

The large difference between distributions for sub-networks can be explained by the fact that they have different application usage patterns. A similar analysis to the one in Section II revealed that subnets 1 and 4 present the highest resemblance to the overall

application usage (Figure 2), whereas other subnetworks show a completely different pattern. Also subnet 1 and subnet 4 have the highest number of clients.

This leads us to believe that the flow-level traffic models presented for the aggregated traffic should work for subnetworks with a higher number of clients. A higher number of clients leads to a smoother curve for the evaluated parameters and an application usage pattern that complies with the one presented for the aggregated traffic.

7.CONCLUSIONS

In this study we presented the first steps towards characterizing the load of a Mobile IPv6's HA. In order to do this we have measured the internal traffic of a medium-size department and assumed that all the hosts are Mobile IPv6 clients away from their home network.

In the first part of the study we evaluate the load that a Mobile IPv6 HA would have to process. Our results show that in a medium size network comprising roughly 1500 clients a HA is looking at mean values of 3.57 active flows per second and maximum throughput values of 262.24 Mbps and 24.6 Kpps. This indicates that high processing power is needed for a HA deployed in a similar scenario.

The second part of our study looks to fit existing traffic models to our empirical data. We found that variables such as flow-size, inter-arrival times and flow arrivals fit currently deployed models for the aggregated traffic even if these models have been developed for characterizing Wireless LAN traffic and not the load of a HA. The following variables have been modeled: average flow sizes using a Pareto distribution and flow inter-arrival times using a Lognormal distribution.

In the final part we look at the per subnet traffic for similar patterns. We find that the same distributions can be used to model the traffic at the subnet level for subnetworks with a high number of clients. Further work is needed in this direction to match subnet traffic patterns with aggregated traffic patterns.

8.REFERENCES

[1] F.Hernandez-Campos, M.Karaliopolus, M.Papadopouli, H.Shen. "Spatio-Temporal Modelling of Traffic Workload in a Campus WLAN", WICON 2006

[2] M. Karaliopolus, M. Papadopouli, E.Raftopolous, H.Shen. "On scalable measurement-driven modelling of traffic demand in large WLANs", LANMAN 2007

[3] R.Pang, M.Allman, M.Bennet," A first look at modern enterprise traffic", IMC 2005

[4] A.Sinha, K.Mitchell, D.Methi. "Flow level upstream traffic behaviour in broadband access networks: DSL versus Broadband fixed wireless", IPOM 2003

[5] C.E. Perkins." Mobile IP', IEEE Communications Magazine, May 1997

[6] Mark Fulmers flow-tools

[7] Z.Sun, D.He, L.Liang. "Internet QoS and traffic modelling", IEE 2004

[8] M.Ploumidis, M.Papadopouli, T.Karagiannis. "Multi-Level application-based traffic characterization in a large scale wireless network", WOWMOM 2007

[9] K.Thompson, G.J.Miller, R.Wilder."Wide-Area Traffic Patterns and Characteristics", 1997

[10] C.E.Perkins, D.B. Johnson. "Mobility Support in IPv6", MOBICOM 1996

[11] K.J.Lee, J.Park, H.Kim. "Route optimization for mobile nodes in a mobile network based on prefix delegation", VTC 2003

[12] N.Vandaele, T.Van Woensel, A.Verbuggen. "A queueing based traffic model", March 2000

[13] D.Johnson et al. "Mobility Support in IPv6", RFC 3775, 2004

[14] C. Perkins: IP Mobility Support for IPv4 RFC 3344 (2002)

[15] F. Heissenhuber et al. "Home Agent Redundancy and Load Balancing in Mobile IPv6" I.Conf Broadband Communications 1999

[16] H. Deng et al "Load Balance for Distributed Home Agents in Mobile IPv6", IEEE PIMRC 2003

[17] R. Wakikawa et al "Virtual mobility control domain for enhancements of mobility protocols" INFOCOM 2005

[18] A. Cabellos-Aparicio et al., "A flexible and Distributed Home Agent Architecture for Mobile IPv6-based Networks" IFIP Networking 2007

APPENDIX: Manual for using the applications

In this appendix we describe the steps that were taken in order to obtain the results. This manual can be used in order to conduct a simmilar study.

1. The first step for our study was the separation of internal traffic from the captured NetFlow traces. This is done by using either the **flow-filter** or the **flow-nfilter** applications from flow-tools. We consider that flow-nfilter is better as it also allows for time criteria to be added to a filter. The implementation of filters has been described in section 2.2. The filters have to be applied on the NetFlow version 5 records and the results will be in the same format.
2. The second step was determining the make-up of the internal traffic in terms of protocols and applications that are used. This is done with either **flow-stat** or **flow-report** applications. For the protocol statistics we used flow-stat report number 12 and for the port statistics report number 5:

   ```
   Cat flowfile|flow-stat -f12>output_file
   Cat flowfile|flow-stat -f5>output_file
   ```

   The output files will be text files that contain the total of flows, octets and packets sent using each protocol, or sent to each destination port in the second case.

3. The third step was computing the flow parameters for the internal traffic. This was done using the PERL scripts described in 2.3. In order to run the scripts they have to be inside the same folder as the files they process and the command syntax is given in section 2.3. The input files have were files obtained with flow-print from the NetFlow traces or files with the same format. An example is shown below:

```
Start             End               Sif   SrcIPaddress    SrcP  DIf   DstIPaddress    DstP    P Fl Pkts       Octets

0308.23:59:44.709 0308.23:59:46.037 68    10.10.30.189    50396 63    10.10.33.188    143   6   0  8          476
0308.23:59:45.341 0308.23:59:45.401 68    10.10.30.189    34561 63    10.10.33.188    143   6   0  6          371
0308.23:59:45.889 0308.23:59:46.137 68    10.10.30.189    59810 63    10.10.33.188    143   6   0  7          420
0308.23:59:09.821 0308.23:59:45.817 70    10.10.30.249    68    0     10.10.32.29     67    17  0  4          1312
0308.23:59:46.829 0308.23:59:46.829 70    10.10.30.163    60251 0     10.10.30.197    53    17  0  1          72
0308.23:59:46.829 0308.23:59:46.829 69    10.10.30.197    53    0     10.10.30.163    60251 17  0  1          129
0308.23:59:47.157 0308.23:59:47.157 70    10.10.30.163    60252 0     10.10.30.197    53    17  0  1          63
0308.23:59:47.189 0308.23:59:47.189 69    10.10.30.197    53    0     10.10.30.163    60252 17  0  1          313
0308.23:59:47.373 0308.23:59:47.373 63    10.10.3Z3.188   46754 0     10.10.30.197    113   6   2  1          60
```

4. The output files obtained after running the PERL script have the following format: first timestamp in bin, numer of flows in bin, number of octets in bin, number of packets in bin, bps, pps, flows per second, average flow size. An example is show below:

```
0310.01:01:01.743 123 347990 852 5799.83333333333 14.2 2.05 2829.18699186992
0310.01:02:03.103 89 107506 484 1791.76666666667 8.06666666666667 1.48333333333333 1207.93258426966
0310.01:03:03.964 134 362184 930 6036.4 15.5 2.23333333333333 2702.86567164179
0310.01:04:04.720 103 253696 726 4228.26666666667 12.1 1.71666666666667 2463.06796116505
0310.01:05:04.580 100 93276 631 1554.6 10.5166666666667 1.66666666666667 932.76
0310.01:06:08.828 107 142183 677 2369.71666666667 11.2833333333333 1.78333333333333 1328.81308411215
0310.01:07:02.012 97 190053 576 3167.55 9.6 1.61666666666667 1959.30927835052
0310.01:08:05.861 118 148588 580 2476.46666666667 9.66666666666667 1.96666666666667 1259.22033898305
0310.01:09:04.089 84 125895 539 2098.25 8.98333333333333 1.4 1498.75
```

5. These results can then be imported in MatLab for plotting. The main types of plots used are cdfplots. These are already implemented in Matlab. An example is shown of plotting the CDF of one parameter.

```
#first load the data;the working directory for Matlab has to be
#the directory which contains the data files. This will create a
#matrix variable with the same name as the data file
load datafile;
#choose a parameter for printing, a column from the file
fps=datafile(:,number_of_column);
#then plot the data
cdfplot(fps);
```

6. The final step is fitting the proposed distributions for the empirical data. This is done using the Matlab Distribution Fitting Tool as described in section 2.4.
7. For the subnetwork traffic the same steps were taken after filtering the traffic for each particular subnetwork