Paris Traceroute

SRINikhil Mamadapalli Junnutula Meghanath Reddy

February 16, 2014

Contents

1 Introduction

Traceroute is a computer network diagnostic tool for displaying the path and measuring transit delays of packets across an Internet Protocol (IP) network. The history of the route is recorded as the round-trip times of the packets received from each successive host in the path; the sum of the mean times in each hop indicates the total time spent to establish the connection. It utilizes the IP protocol TTL field and provides the user an ICMP TIME_EXCEEDED response from each gateway/router along the path to the host. Unfortunately, traceroute measurements can be inaccurate and incomplete when the measured route traverses a load balancing router, or load balancer.

Paris Traceroute is a basic upgrade to the standard Traceroute tool. Paris Traceroute takes into account the load balancing routers that the Standard Traceroute neglects. These are widely deployed in today's Internet at all the levels. The standard Traceroute is not aware that there are often multiple paths between a source and a destination in the Internet: the paths will split at a load balancing router at one point along the route and then converge at some point in the path. Standard traceroute, unlike the Paris Traceroute is incapable of furnishing this information to the user and reports a single path which is actually consisting of multiple paths all along the source to destination. Paris Traceroute is aware of the multiple paths as well as the single paths and can report them accurately

1.1 Aim of the project

This project aims at the development of a Traceroute that employs detection of Multipath routing in source to destination routing of IP packets. It is a rigorous modification of classical traceroute algorithm, taking into account load balancing properties of the modern routers. Moreover, the projects also deals with mapping the actual path of the packets considering per-flow routing as per Paris-traceroute algorithm.

1.2 Methodology

In designing the Paris Traceroute tool the following steps have been considered:

- Feasibility Study:
  Understanding and identifying of existing Standard Traceroute and associated study of the the routing algorithms and required protocols(ICMP) that are necessary to develop such a Traceroute.

- Analysis:
  Proper analysis using the busy servers available and building a Paris Traceroute using the structures and functions will be considered.

- Design:
  Designing a Paris Tracroute will be achieved through basic system programming using the c language features and implementations.

1.3 Requirements for Paris Traceroute

- Hardware Requirements
  - 512 MB RAM or more
  - Windows or Linux
- Programming Languages
  - Basic C
- Backup Media
  - Hard Disk

The traceroute tool develoepd in this project has been developed using Windows and tested using linux as system software and is tested on all operating systems mentioned above.

2 Generic Description

The logic behind the implementation of traceroute is fairly simple and follows a simple algorithm based on the TTL value. Each IP packet consists of an entry known as the TTL or Time to live which is responsible for recording the remaining lifespan of the packet in the system, measured in number of router hops, and functions to prevent routing loops from consuming an infinite amount of network resources by setting a finite limit on the number of hops that a packet can be routed through. As part of the IP routing process, each router which handles a packet will decrement the value of the TTL field by 1. If the TTL value ever reaches 0, the packet is dropped immediately from the system at that router, and an ICMP TTL Exceed message is returned to the original sender letting it know that the packet was dropped. Traceroute is based on this inherent behavior of the IP routing process to map out each router that a packet if forwarded through, by sending out a series of probe packets which are intended to expire before reaching their final destination, and capturing the resulting ICMP TTL Exceed messages.

Some of the simple steps that every traceroute follows:

- The traceroute tool launches a probe packet towards the final destination, with an initial TTL value of 1.
- Each router that handles the packet along the way decrements the TTL by 1, until the TTL reaches 0.
- When the TTL value reaches 0, the router which discarded the packet sends an ICMP TTL Exceed message back to the original sender, along with the first 28 bytes of the original probe packet.
- The Traceroute utility receives this ICMP TTL Exceed packet, and uses the time difference between the original probe packet and the returned ICMP packet to calculate the round-trip latency for this router “hop”
- This process again from step 1, with a new initial TTL value of N+1.

- The final destination receives the Traceroute probe packet, and sends back a reply packet other than an ICMP TTL Exceed.

The Traceroute utility uses this to know that the path-trace is now complete, and ends the process.

2.1 Load Balancing

Load balancing is a common technique used within complex traffic sources across multiple physical links between a particular source and destination. This is done to increase reliability and resource utilization. There are two primary ways to accomplish this, a layer-2 based Link Aggregation protocol (often called a LAG, such as 802.3ad), and layer-3 based Equal Cost Multi-Path (ECMP) routing. Layer 2 based LAGs are invisible to Traceroute, but layer-3 based ECMP is often detectable. When multiple paths are included in Traceroute results, it can significantly increase the difficulty of correctly interpreting the results and diagnosing any potential problems.

This causes anomalies in the standard traceroute, and the values obtained can be erroneous and imprecise as the packets traverse through the load balancer.

2.2 Role of the header

Every unique value in the probe header ensures a different response from the tool. Thus, changing any field in the first four octets of the trasport layer header amounts to changing the flow marker for each probe packet. The destination port field lies in the first four octets of the UDP header. Althought the sequence number lies in the second octet of the ICMP Echo header the change in it will incur a change in the checksum field, which is in the first four octets of the header. Thus, the traceroute still needs to be able to match response packets to their corresponding probe packets.

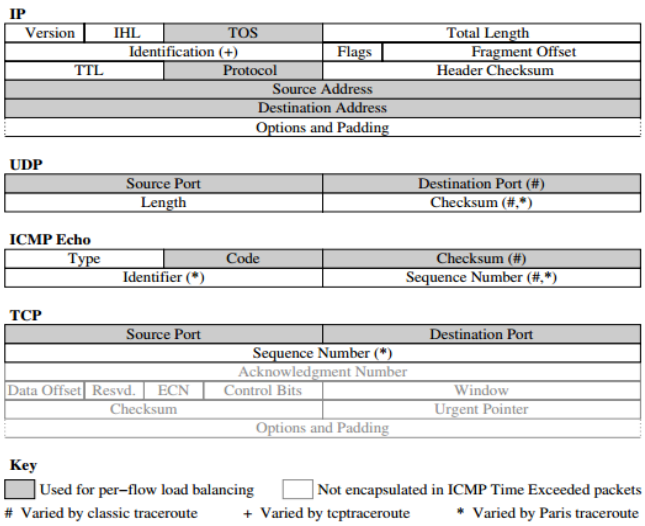

Figure 1: Different headers and its effect

2.3 Need for Paris Traceroute

Earlier discussion shows that the traceroute does not match the values of its corresponding packets. Thus, arises the need of a new upgraded version that varies the header fields that are within the first eight octets of the transport layer as they are not used for the load balancing. This keeps the checksum similar to the previous value. Paris route does this operation for the user, eliminating the anomalies of the standard traceroute. The packets also require manipulating the payload to yield the desired value, as packets with an incorrect checksum are liable to be discarded.

The Paris traceroute varies the Sequence number field like the standard traceroute but also changes the identifier field unlike its counterpart keeping the Checksum field constant throughout.

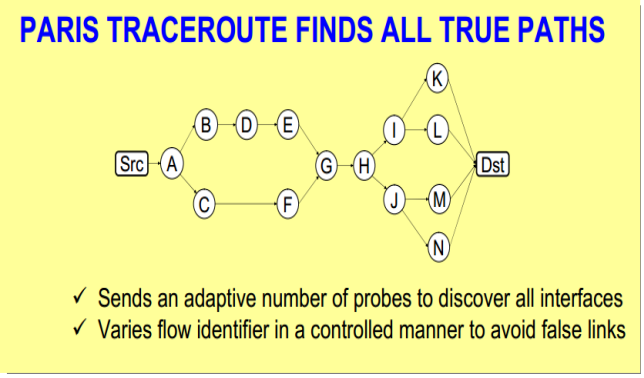

Figure 2: Different headers and its effect

3 Implementation

The project uses the ICMP (Internet Control Message Protocol) probe packets to get the information about the network topology. ECHO request packets of around 10 to 20 were sent for every TTL to get the information from all the intermediate nodes between source and destination. As in the implementation of the code the user does not directly deal with IP header, a function named setsockopt() is used to set the TTL value in the IP header. This setsockopt() function was used with proper pre-defined flags to set the TTL value at the proper IP header field.

Also, the corresponding Berkeley select() function along with fd_set functions to check if the information was received accordingly when the functions implements from the destination or intermediate routers. As explained above standard traceroute always ends up transmitting the packet in different paths. The standard ICMP ECHO requests sent above for every TTL value returned with ECHO reply from different intermediate routers for every probe packet which is due to the load balancing, a factor which plays an important role in deciding the packet-path in the project. For every ICMP probe packet sent, ICMP checksum is calculated and put it in the appropriate ICMP checksum field.

In the paris traceroute implementation of the mentioned code, we are focusing on the sending ICMP packets with a constant checksum for a particular load flow, while gathering the packet ttl information form the sequence number. The checksum can be chosen and the 16 bit data can be added to the ICMP message to satisfy the initial checksum chosen. This prevents the packets from giving a checksum error and reduces its probability of being dropped from the network.

Now the checksum of the ICMP packets had to be controlled and this was done by sending the constant value in the probe packets in order to make probe packets to follow the same path. The figure shows how exactly the node information is received from the entire topology between source and destination. This is how the Paris Traceroute implements its algorithm from the source to the destination giving the user the intermediate paths of travel.

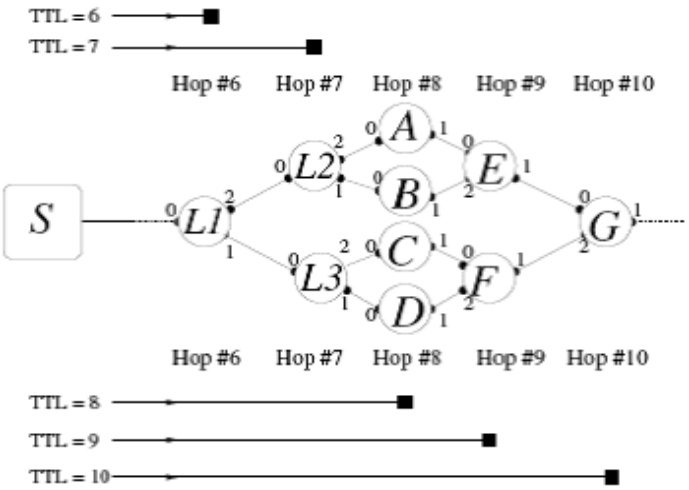

Figure 3: Path of the Traceroute

The source code has been divided into two parts; the first part of our source code consists of all the node information including its IP addresses, return TTL value, checksum, IMCP code and its type saved in array of structure instances. Once all the information of node in the topology between source and destination are received the implementation of the second part would begin that is; second part of the code is handled. In the second part of the code the checksum and ICMP code of every probe packet particular TTL is controlled and constant checksum are sent to the network flow.

Now, when these controlled probe packets are sent into the network all the packets flow in the same path. To confirm that all the packets follow the same path, first part of the code would be repeated and this would confirm the same, where we start TTL with value 1 and then keep incrementing constantly till the destination is reached. The ICMP type from the ECHO reply has been used to know if destination was reached. If returned ECHO ICMP type is “8” then it is not the destination, now the TTL would be incremented and then send its probe packets.

As seen through our analysis the probability of probe packets following the same paths increases with Paris Traceroute implementation which is also explained through the implementation and design of the code. Once all the probe packets are received from the same IP address or host after setting the constant checksum and ICMP code, the single path that Paris Tracerout uses will be displayed. The code is attached in the appendix for future reference.

4 Observations and Analysis

The observations and proof of the code correctness has been made by considering www.facebook.com which is one of the busiest servers available. The analysis shows that the code has been implemented correctly and has rigorous results with very good load balancing traits achieved with the Paris Traceroute source code designed through the project. The facebook website was reached in 15th TTL from Texas A and M University- College Station using the University's network.

The figure above also shows various Autonomous sytems as detected by the projects traceroute tool. Also, the IP addresses and their ranges have been depicted. The IP WHOIS tool and the ASN lookup tools have been used to analyse and detect the paths given by the Paris Traceroute. It also shows how the load is balanced at certain routers along a path dividing into multiple paths when traced from facebook.com.

Also, through the code load balancing was not effective in reaching some of the sites like www.weather.com and www.youtube.com as the path was reached within 6-7 TTL's and Paris Traceroute couldn't be applied in those cases and thus it remains in scope of future analysis and what the project would like to extend to in analyzing how the path would be derived appropriately for smaller and less busier networks.

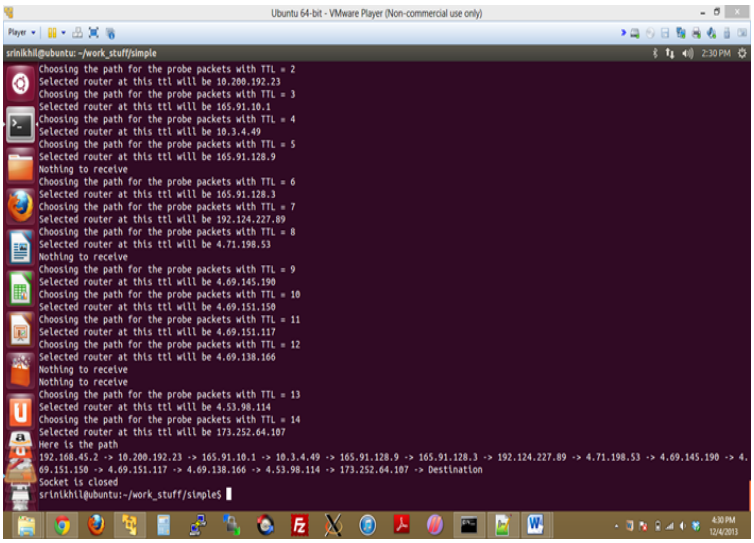

Figure 4: Path given by the code for www.facebook.com

5 Results

The results have been considered and obtained by using the Paris Traceroute tool and the destination as the facebook server. The results of the Paris Traceroute have been plotted using the tables from the observations section. The figures below represent the path that has been given by the tool developed in the project. It shows how the load balancing have taken precedence from the implementation of the tool as soon as the route entered the facebook autonomous system.

5.1 Flow diagram

The path flow of the multiple route that have been plotted with the IP addresses:

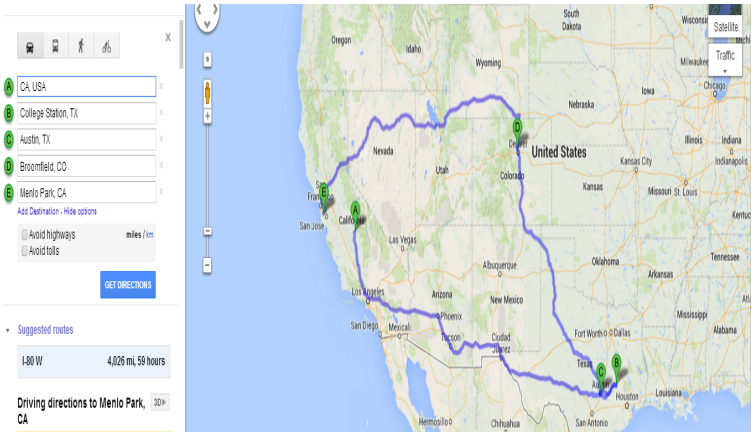

Figure 5: Results of first phase

Figure 6: Results of second phase

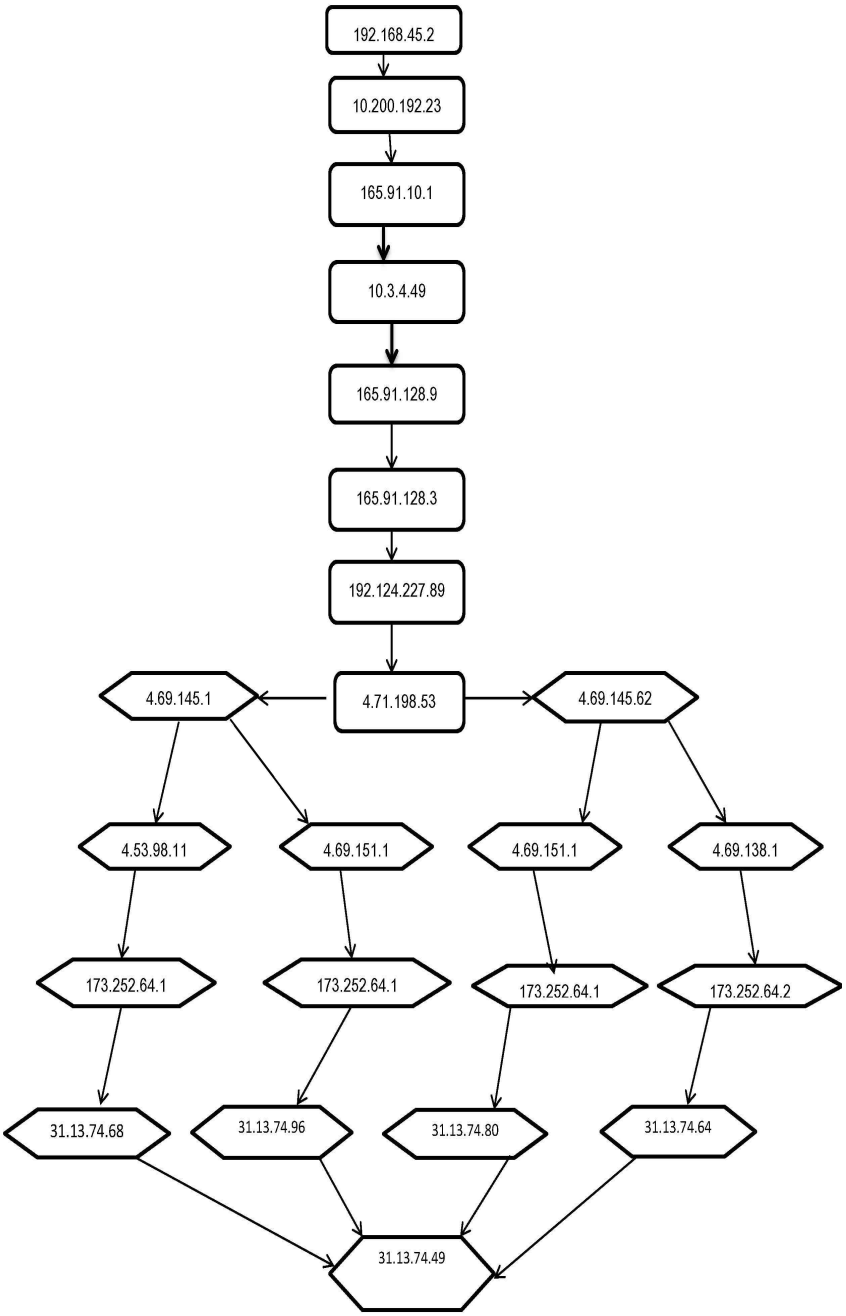

Figure 7: Flow by Paris Traceroute

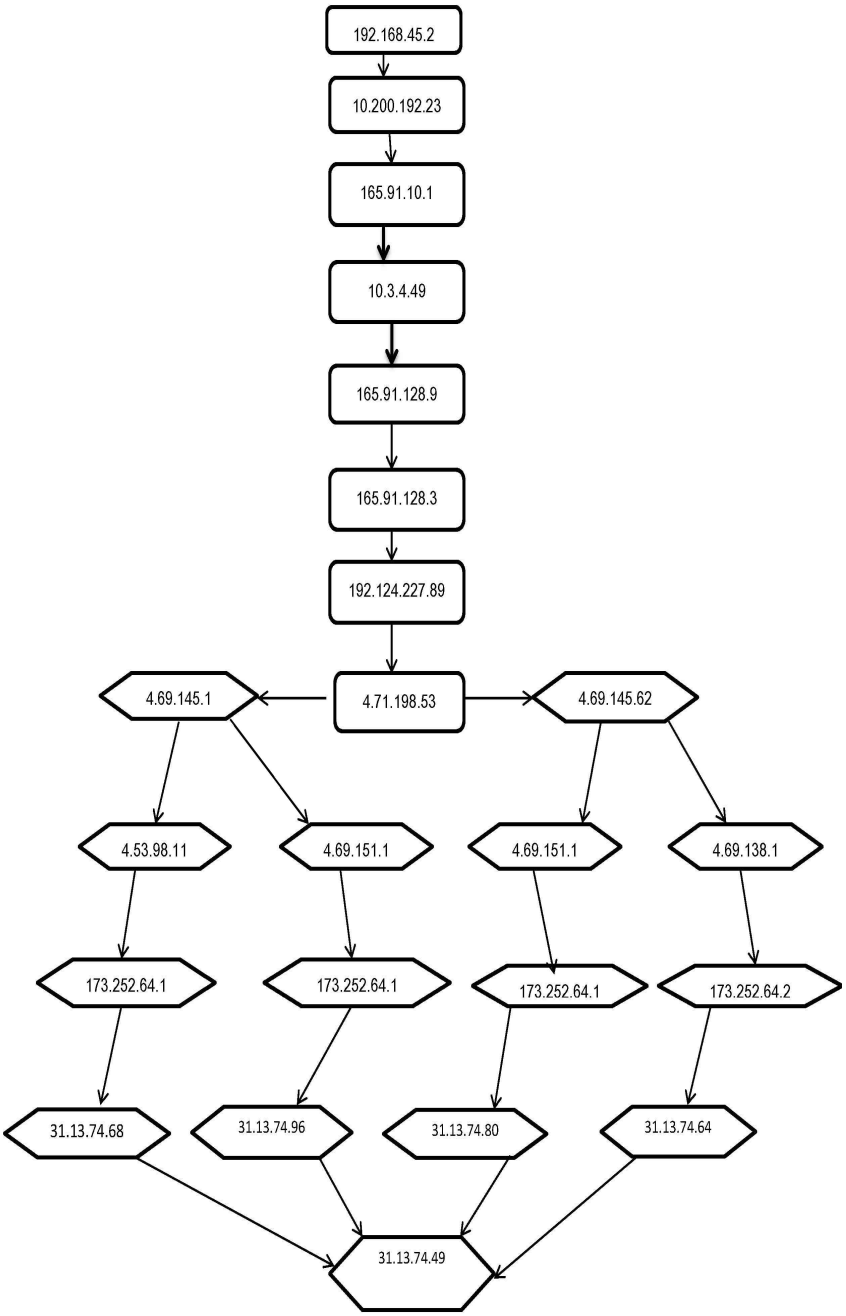

Figure 8: Flow by Paris Traceroute

6 Conclusion

In this project, a new characterization of a route between a source and a destination taking load balancing factor into account has been introduced. The implementation has been done on IP level datagrams by creating specific ICMP echo packets with varying TTL and data along with it to vary the checksum. This was the packets arriving will have different flow. An algorithm for the Multipath detection has been proposed whose aim is to enumerate all load-balanced paths from the source to destination. From, the source used in this project, the routes to 30-40 percent of the destinations followed multiple paths. Also, the aim of this project that is to establish a tool to detect Multipath routing in source to destination by the routing of IP packets has been accomplished.

7 Future Scope

Although this algorithm represents an important basic step, it is clearly not the end. The project can be improved in a number of ways. It could be possible to send fewer probes/packets, given better knowledge of the hash algorithms that per-flow load balancers use. Also, the project can be extended to find where the load balancing and certain other network traffic properties. The observed rare cases of uneven load balancing can be resolved and the multiple paths can be found out.No systematic measurements and analysis to detect it has been done edue to the time constraint, but we believe it is a very simple extension to the current algorithm.

Furthermore, the Multipath detection algorithm will also allow us to build more complete and accurate maps of the internet. It is clear that our probing technique does not signicantly improves the coverage of the internet, in terms of number of discovered interfaces. However, it adds valuable information on load balancing, by grouping a set of paths, which seem independant in our traditional view of the internet map, into a multipath that can indifferently be selected by an end-user to reach a destination. Finally, the study of the path diversity that the network natively provides might lead to innovations in the optimization of end-to-end connections.

8 Bibliography and References

1. "Socket Programming Online resources", Internet: http://beej.us/guide/bgnet/output/html/singlepage
2. "Traceroute", Internet: http://en.wikipedia.org/wiki/Traceroute
3. J. Moy, "OSPF Version 2," RFC 2328, April 1998
4. B. Huffaker, D. Plummer, D. Moore, and k claffy, "Topology discovery by active probing," in Proc. Symposium on Applications and the Internet (SAINT), January 2002.
5. Z. M. Mao, D. Johnson, J. Rexford, J. Wang, and R. H. Katz, "Scalable and accurate identification of AS-level forwarding paths," in Proc. IEEE INFOCOM, March 2004.

6. "Multipath tracing with Paris traceroute", Internet: http://www-rp.lip6.fr/ augustin/augustin07multipath.pdf.

7. "Exhaustive path tracing with Paris traceroute ", Internet: http://www-rp.lip6.fr/ augustin/augustin07multipath.pdf.

8. "Load Balancing(Computing)", Internet: http://en.wikipedia.org/wiki/Load_-balancing_(computing)

9. J. Postel, "Internet control message protocol," RFC 791, September 1981.

10. Cisco, "How does load balancing work?", Internet: http://www.cisco.com/en/US/tech/tk365/technolog tech note09186a0080094820.shtml

11. Article from http://cluepon.net/ras/traceroute.pdf

12. "Configuring load-balance per-packet action,", Internet: http://www.juniper.net/techpubs/software/ju policy/html/policy-actions-config11.html.

9 Appendix

The code developed for the Paris Traceroute through this project is attached below for reference. It contains all the structures, implementations and functions used for the development of the Paris Traceroute.