Project #2 : Traceroute Next Generation

CS155: Computer and Network Security
Due: May 4th, by 11:59pm (Part 1),
Due: May 11th, by 11:59pm (Part 2)

Spring 2009

Contents

1 Introduction 3
- 1.1 Instructions . . . 3
- 1.2 Submission and Grading . . . 3
- 1.3 Exercise Overview . . . 5
- 1.4 Recommended Reading . . . 5

2 Standard Traceroute 6
- 2.1 Sample Invocations . . . 6
- 2.2 Sample Output . . . 6

3 Versatile Traceroute 7
- 3.1 Sample Invocations . . . 7
- 3.2 Sample Output . . . 7

4 Statistics Reporting 8
- 4.1 Sample Invocations . . . 8
- 4.2 Sample Output . . . 9

5 Path Diagnostics 11
- 5.1 Loss . . . 11
- 5.2 Route Load Balancing . . . 12
- 5.3 Server Load Balancing . . . 12
- 5.4 Sample Invocations . . . 12
- 5.5 Sample Output . . . 13

6 Firewall Handling 14
- 6.1 Firewalk . . . 14
  - 6.1.1 Sample Invocation . . . 14
  - 6.1.2 Sample Output . . . 14
- 6.2 Established Method . . . 14
  - 6.2.1 Sample Invocation . . . 14

6.2.2 Sample Output 14
6.3 Ghost Traceroute 15
6.3.1 Sample Invocation 15
6.3.2 Sample Output 15

7 Wrap-Up 15

1 Introduction

Traceroute is one of the key tools used for network troubleshooting and scouting. It has been available since networking's early days. Because traceroute is based on TTL header modification, it crafts its own network packets.

The goal of this project is to re-implement a traceroute with cutting edge techniques designed to improve its scouting capabilities and bypass SPI firewalls. In this project you will learn about packet injection and sniffing. The project also teaches about more subtle topics such as network latency, packet filtering, and Q.O.S.

1.1 Instructions

This project has to be completed sequentially as each part depends on the previous ones. You are required to work in groups of at most 2 people (one-person teams, while allowed, are discouraged). The required coding language is C. You should consider using libnet [1] and libpcap [2], however using raw sockets is permitted as well.

Note that the exercises become progressively more difficult, with Exercise 5 likely to consume the greatest amount of effort. Please plan your work accordingly: there is no penalty for early submission.

1.2 Submission and Grading

The project is due in two parts. The first part includes exercises 1 and 2, and the second part should be fully functional, including exercises 1, 2, 3, 4, and 5. The first part will be worth 40% of the project grade, which means that exercises 1 and 2 will be graded entirely based on your part 1 submission. In all, exercises will have equal weight, about 20% of the project grade each.

Submission is by email to **cs155ta@cs.stanford.edu**, and is due by 11:59pm on **May 4th** for Part 1 and by 11:59pm on **May 11th** for Part 2. Provide a tgz archive that contains your code. When extracted, you project has to be in a directory named project2_name1[_name2] where name1 (and name2 when present) is(are) the last name(s) of the student(s) it the team. The subject line of the email needs to be "CS155 project2". Please include in the body of the e-mail the name of each member of the team, as well as their email addresses.