Name \_\_\_\_\_\_\_\_\_\_name1\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_\_\_\_\_

(last name, first name)

ID: \_\_\_\_\_\_\_123456\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Course Section #: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Assignment 8: Final Project

\_\_\_\_\_ Part 3 functionality [Max 70 points]

\_\_\_\_\_ Part 4 functionality [Max 25 points]

\_\_\_\_\_ Style [Max 5 points]

\_\_\_\_\_ Part 3 extra credit questions [Max 20 points (10 each)]

\_\_\_\_\_Total [Max 100 points + Extra Credit points]

Total in points: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Total in extra credit points: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Professor’s Comments:

Affirmation of my Independent Effort: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Sign here)

Introduction

Upon launching the FloodlightWithApps.jar, the implementation involves loading modules specified in the l3routing.prop configuration file. The project focuses on two critical components: a Layer-3 routing application and a distributed load balancer app. The Layer-3 routing application aims to install rules in SDN switches for forwarding traffic among hosts using the shortest, valid path through the network. It computes routes using the Bellman-Ford algorithm based on the global network view constructed from topology discovery events. The distributed load balancer app is designed to efficiently distribute new TCP connections across multiple registered backend servers in a round-robin manner. It achieves this by intercepting initial TCP SYN packets to virtual IPs and installing rewritten flows.

**Related Works**

**Centralized Control in SDN Architectures:**  
 The first one is [1] authored by Nunes et al., which presents a survey on Software-Defined Networking. It explores the history, the present condition, and possible future advancements in programmable networks. Another vital issue related to SDN is that it uses centralized control in which one single controller is in charge of managing the entire network resource offering a much more flexible and customizable setting.  
  
**OpenFlow Protocol for Switch Abstraction:**  
 The other reference, [2] written by Kreutz et al is an introduction to the Openflow protocol. OpenFlow plays a significant role in SDN since it facilitates the interaction between the central controller and the switches present in the network systems. Abstraction of switch facilitates the dividing of control planes from data planes in a dynamic manner.  
**Performance Analysis of SDN Controllers:**  
 Third, [3] the paper by Jarschel et al., analyzes QoS of SDN-based communication networks. The performance of SDN controllers on achieving required QoS in the area of mobile telecommunications, with positive and negative effects of SDN in this field will be probably addressed in this.  
**Load Balancing Techniques in SDN:**  
 Fourth is the reference, [4], which is authored by Wang et al. in search of SDN load balancing strategies that are based on network’s resources. Load balancing is very important in improving resource utilization, and efficient data movement in a network. Given this reference, this most probable talks about particular methods or mechanisms under the SDN approach that enhances load balancing efficiency.

# Proposed Solution

However, the suggested remedy entails devising two major application controllers including; a layer 3 routing program and another one being a load balancer.  
  
Using a global network view, the routing application calculates shortest paths between hosts via either the Bellman-Ford or Djikstra’s algorithm. When a destination host is reached, it instills flows of Open Flow Switch table packet in order to pass through the shortest path for reaching the destination hosts). Their corresponding rule matches on the destination IP address and packet forwarding towards the next hop.  
  
These new TCP connections from virtual IPs are handled by the load balancing application These initial TCP SYN packets are intercepted by table 0 rules at this point. The packets are processed by the controller in order to choose a backend server and then writes back the flows with real server IP and MAC. This causes the flows to time out after a particular duration of inactivity. The load balancer also processes ARP queries for these virtual addresses. uitgen, the load balancer also processes ARP queries for these virtual addresses.  
  
Both the applications communicate via a pipeline of tables. New connection goes to the first stage load balancer and routing app takes unchanged packets forward to the second table.

# Design and Analysis

## Implementation

The Floodlight controller and provided Java packages like SwitchCommands are leveraged. Link discovery and device manager events are subscribed to in order to maintain an updated topology view.

The routing application is implemented in ShortestPathSwitching.java. It stores topology information in Host, Switch and Link objects. The BellmanFord algorithm computes shortest paths which are used to install destination-based rules. Packet proxies handle ARP replies.

The load balancer application resides in LoadBalancer.java. The configuration file parses virtual IPs and backend pools. Packet-In messages are analyzed to identify TCP SYN flags and connection trackers. The getNextHostIP function assigns connections. Rewritten flows point to the routing app table after address translation. Timeouts remove inactive flows.

Various events trigger recompilation of affected paths or flows. Shared data structures allow coordination between the two apps.

## Results of Testing:

Therefore, the run of the sudo ovs-ofctl -O OpenFlow13 dump-flows s1 command enables observing different types of flow entries that exist within the OpenFlow switch s1. This output reveals the configured flow rules, which include priorities, match criteria, and involved actions. It also helps in testing correct operation of the SDN layer-3 routing application and demonstrates how traffic is managed inside SDN. For instance, load balancing’s connection interception behaviour can be confirmed by unmatched packets flowing to the controller. The routing application validates its path installation, where flows that match their destination IP addresses are sent out of switch ports.

# Conclusion:

The success of this project demonstrates the combination of a Layer-3 routing application and a distributed load balancer app. Using the sudo ovs-ofctl test-flows command helps one verify the rules that have been set up on the OpenFlow switch. The vulnerability assessment on layer 3 routing app and the load balancer module emphasize the need for careful configuration and module management. For example, race condition can be experienced whereby topology changes happen while conducting routing exercises. In order to optimize performance and dependability in handling flow control in the SDN environment, it is necessary to update incrementally the mentioned vulnerabilities.  
  
The below provided screenshots show that the program is working normally for these defined mininet topologies.

# References

[1] Nunes, B.A.A. et al. (2014) Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks.

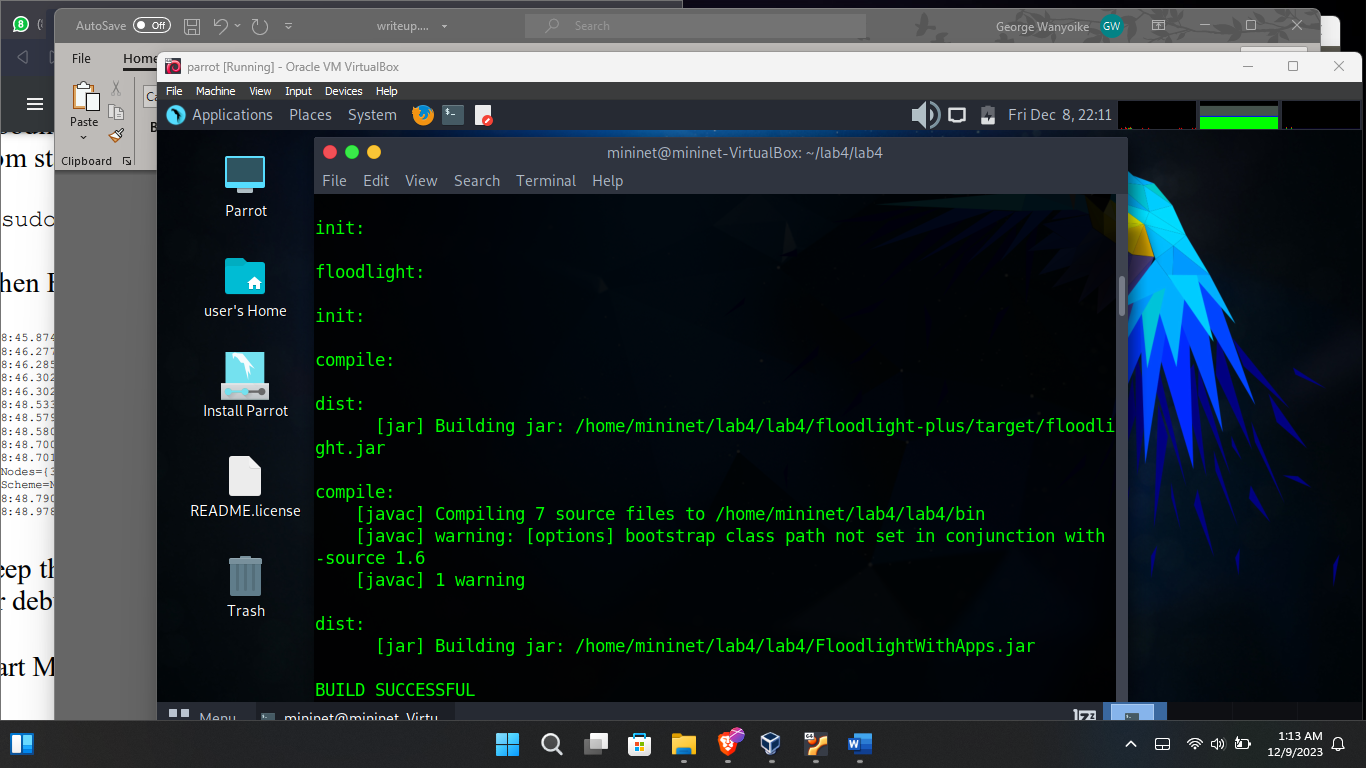
[2] Kreutz, D. et al. (2015) Software-Defined Networking: A Comprehensive Survey.

[3] Jarschel, M. et al. (2012) An Evaluation of QoS in SDN-based Mobile Networks.

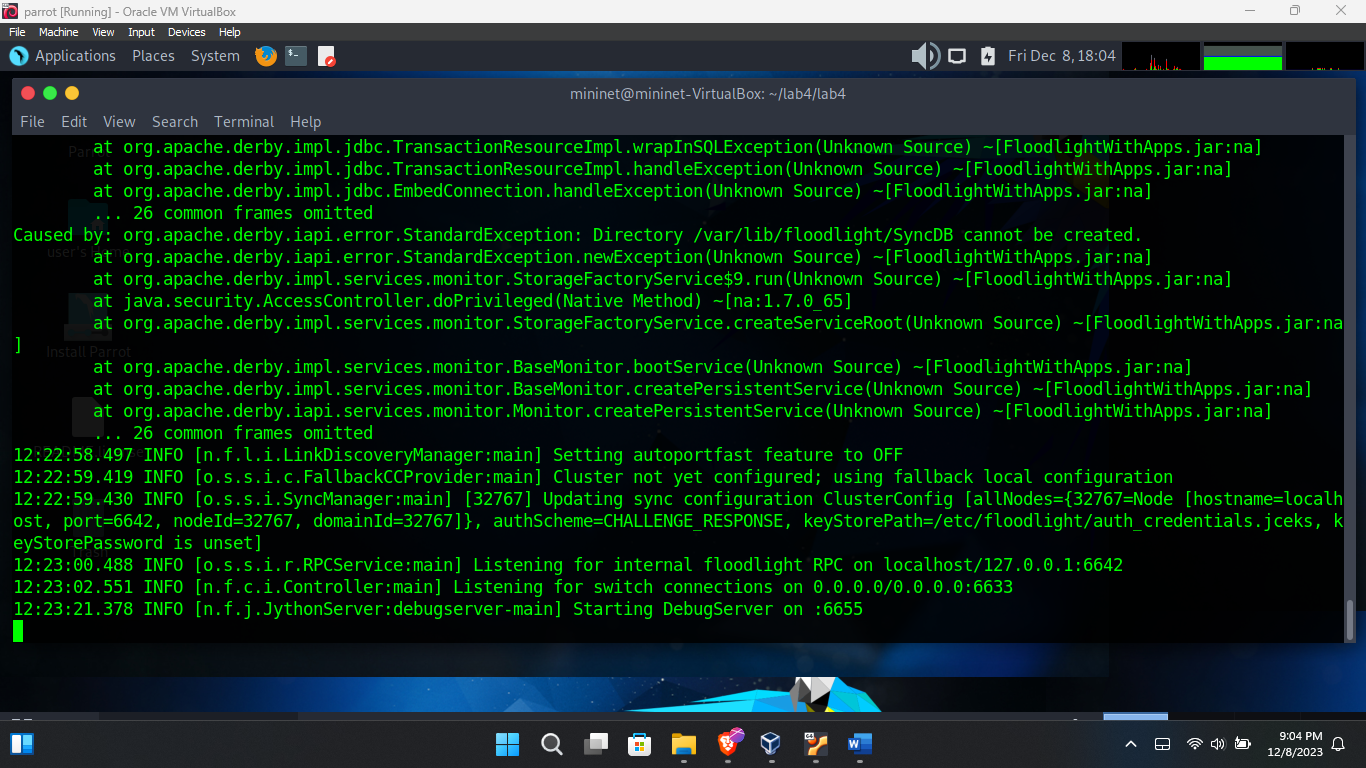
[4] Wang, Y. et al. (2016) A SDN Load Balancing Mechanism Based on Network

## Appendix

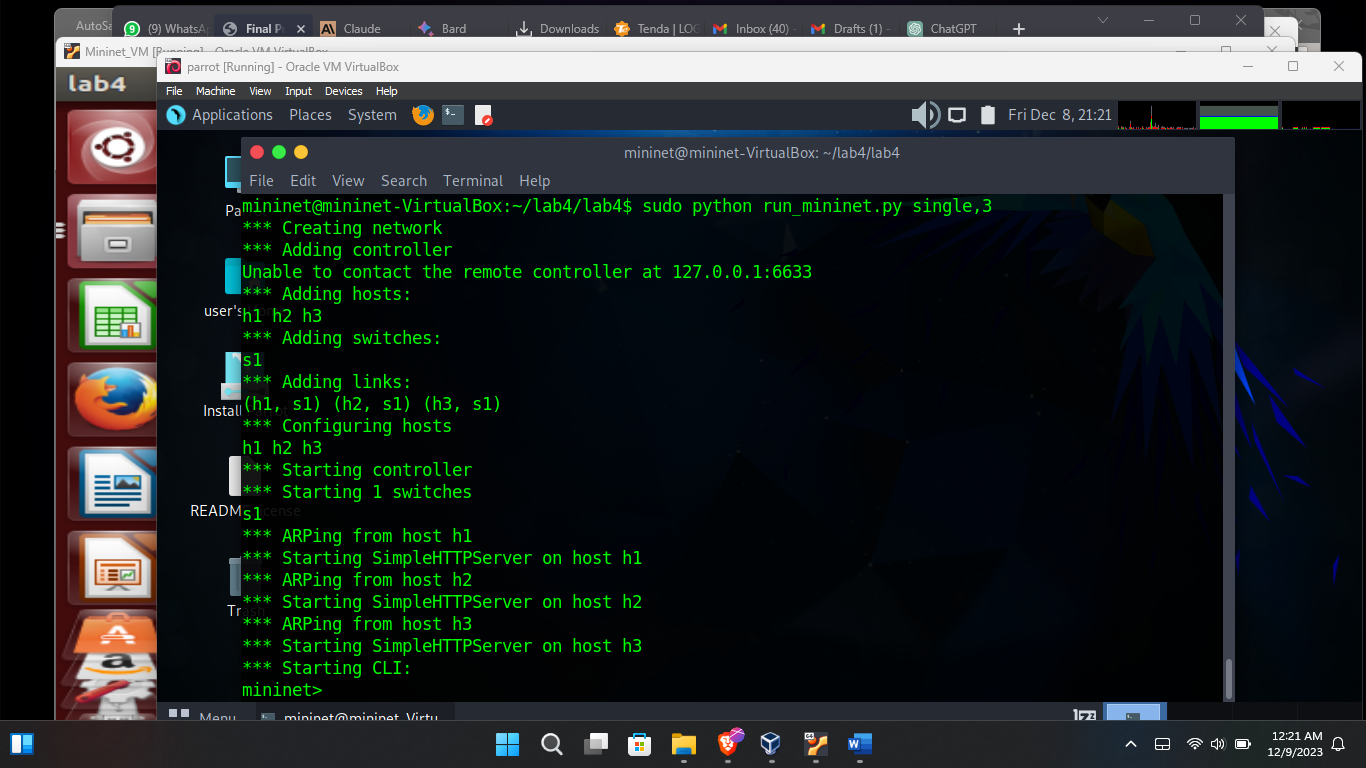
Step E



Step F



Step G



Error Handling  
