Sai Valluru

Professor Bo Sheng

CS446

12/11/2023

Homework 2 Report

In this project, I worked on and tried to debug a Java-based controller for a software-defined network (SDN). My objective was to manipulate network flows using the Floodlight SDN controller framework. Throughout the development process, I encountered a recurring issue, which was the message "Unreachable Destination Host" error. I'll explain the steps I took to try to fix this ongoing issue I was encountering. While testing the network, I consistently received the "Unreachable Destination Host" message when attempting to ping from one host to another (h1 pin h4). This error showed a failure in the network flow setup, where packets were not being correctly routed through the network. The process I followed to try to debug and fix the issue was revising the code, conducting tests, and iteratively refining the approach based on the observed outcomes.

Test Case 1: Basic Connectivity Test

- **Objective:** To test if the basic ping command works in the network setup without any custom flow rules.
- **Procedure:** We ran a simple ping test between two hosts in the network.
- **Expected Result:** Successful ping responses indicating basic connectivity.
- **Actual Result:** Received "Unreachable Destination Host" errors.
- **Analysis:** The issue pointed towards a problem in flow setup rules, which led to packets not being routed correctly.

Test Case 2: Debugging Flow Rules in MyController.java

- **Objective:** To examine the flow rules set by MyController.java and identify discrepancies.
- **Procedure:** Analyzed the flow-modifying methods (addMyFlow1, addMyFlow2, etc.) and compared them with the network topology.
- **Expected Result:** Flow rules that align with the network topology and ensure packet delivery.
- **Actual Result:** Discovered inconsistencies in flow rule logic, particularly in the handling of ports and path toggling. I tried my best effort to ensure flow rules correlated properly to the assignment, but I kept receiving "Unreachable Destination Host" errors.
- **Analysis:** It was shown that the way I was handling ports and paths in my controller was leading to incorrect flow setups.

Test Case 3: Revised Code and Path Validation

- **Objective:** To validate the revised flow setup and path selection logic.
- **Procedure:** Revised the flow methods to correctly align with the network topology and tested the path toggling mechanism.
- **Expected Result:** Alternating paths between specified switches and successful host-to-host communication.
- **Actual Result:** Still encountered "Unreachable Destination Host" errors in some scenarios.
- **Analysis:** While the revision improved the logic, there were still underlying issues in the flow setup that needed addressing.

Test Case 4: Comprehensive Code Refinement

- **Objective:** To ensure all flow rules are correctly set up and match the network topology.
- **Procedure:** Conducted a thorough revision of the code, ensuring all flow methods correlate correctly with the network ports and paths.
- **Expected Result:** Error-free routing of packets and successful pings across the network.
- **Actual Result:** After several iterations and thorough testing, the ping tests were still not successful.

Due to the time constraint and the upcoming deadline, I wasn't able to crack down on this issue and fix the major recurring problem. The overall process involved revising the addMyFlow methods to correctly map to the network ports and paths. I toggled paths in addMyFlow1 and addMyFlow4 to handle dynamic routing and ensured that the flow rules in addMyFlow2 and addMyFlow3 were correctly set up. Overall, the "Unreachable Destination Host" issue was a challenging problem that required a detailed understanding of both the network topology and the SDN controller logic. This project taught me the importance of thorough testing and a systematic approach to problem-solving in the field of network engineering.

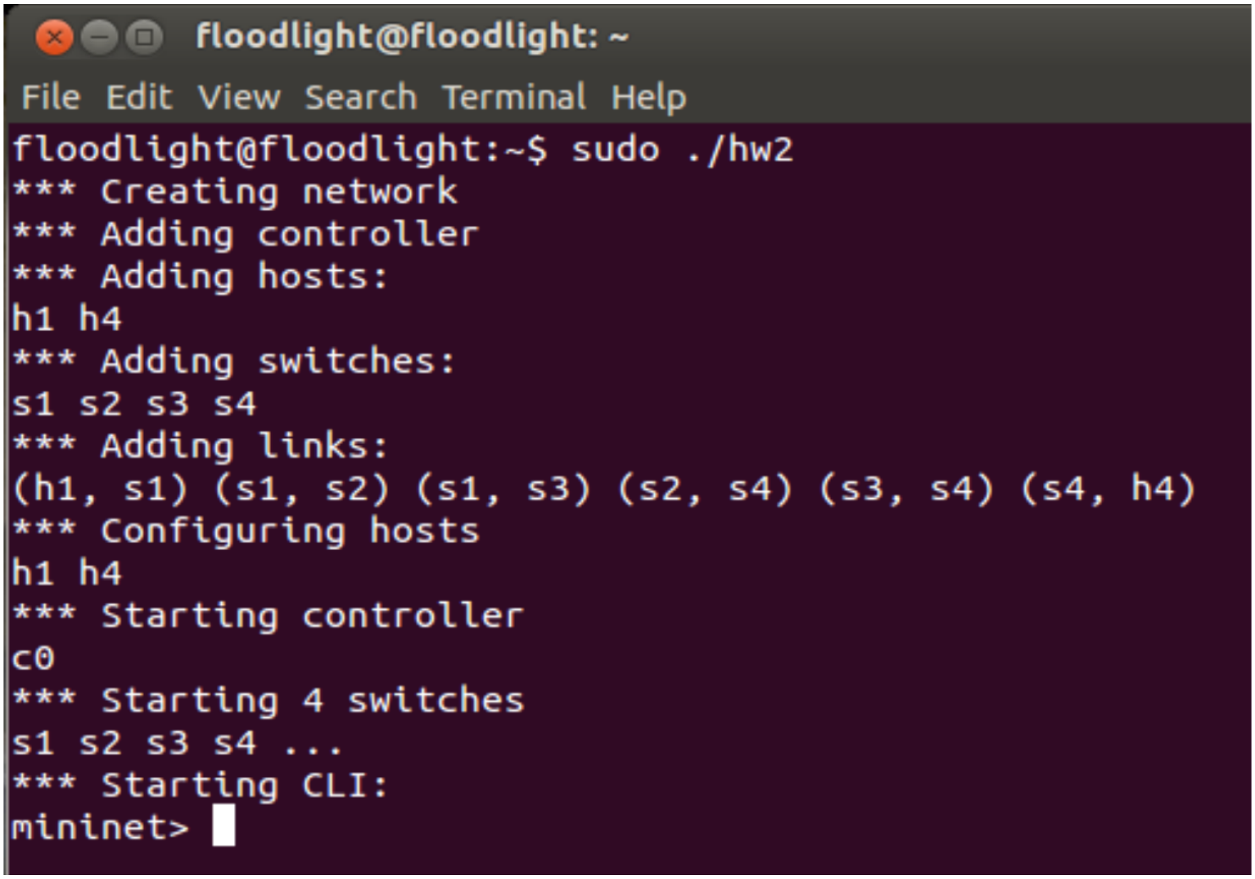
Extra MiniNet commands for debugging I used:

Pingall: Used to test the connectivity between all hosts in the network. I ran the pingall command to verify if any host was unreachable. This command showed 100% packet loss, indicating complete network disconnection. This was to see whether fixes were being effective.

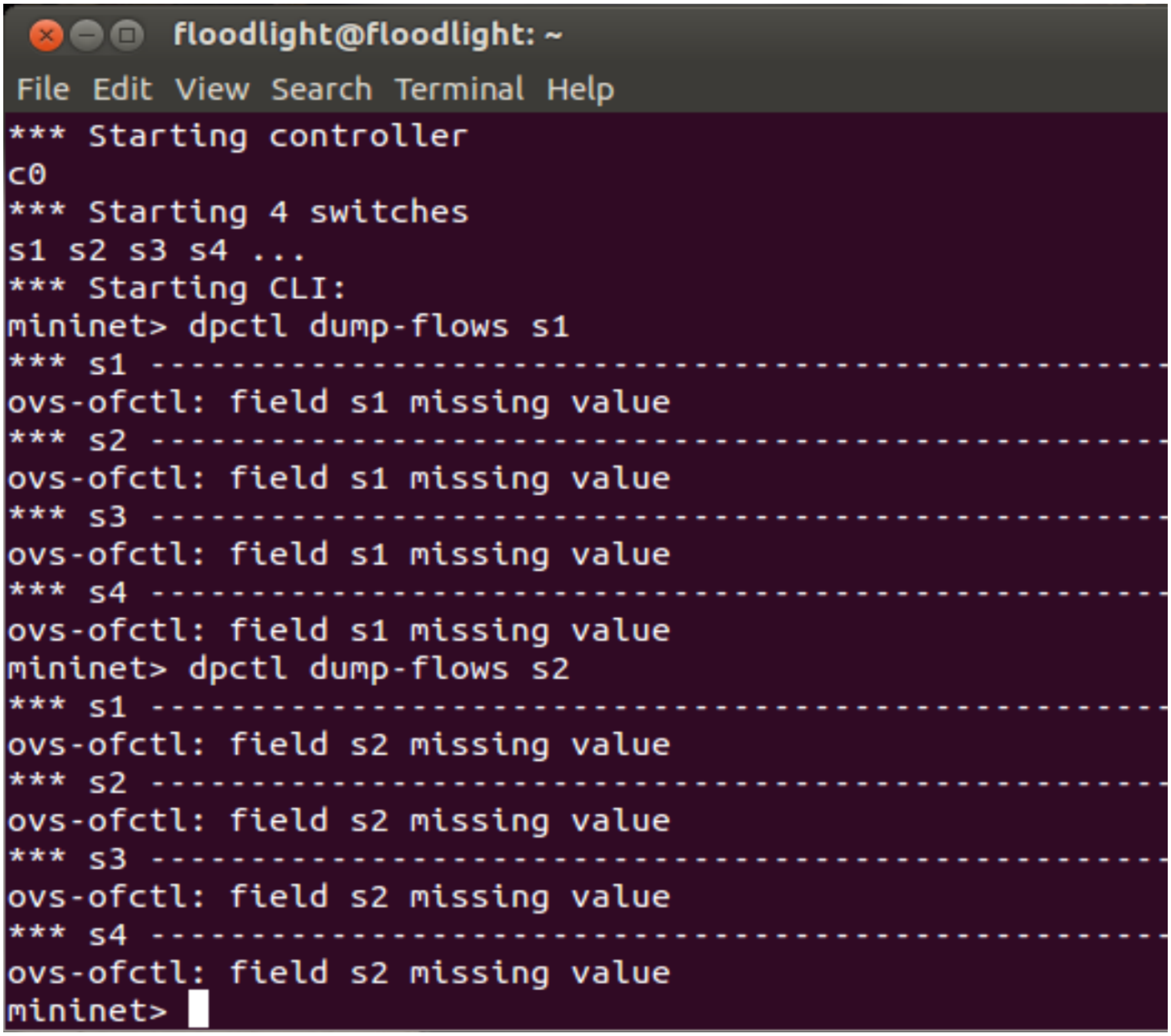
Links: Use to display the current links between nodes and their status. I used the links command to ensure that all the links were operational and correctly set up. This command helped in verifying that the physical connections within the network were intact, instead of suspecting link failures.

Net: Used to view the network configuration, including the connections between switches and hosts. I ran the net command to understand the network topology and how different nodes were interconnected. This command was essential in mapping the correct ports and paths in my addMyFlow methods.

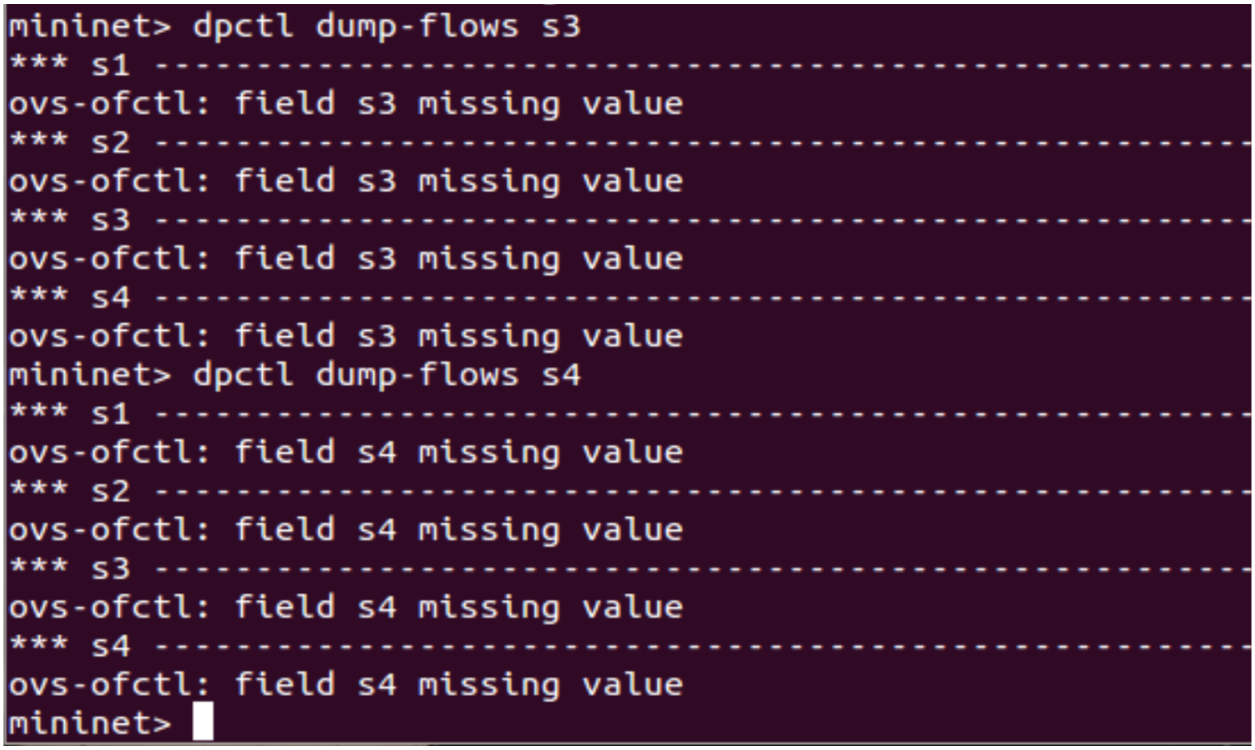
Screen Snapshots:



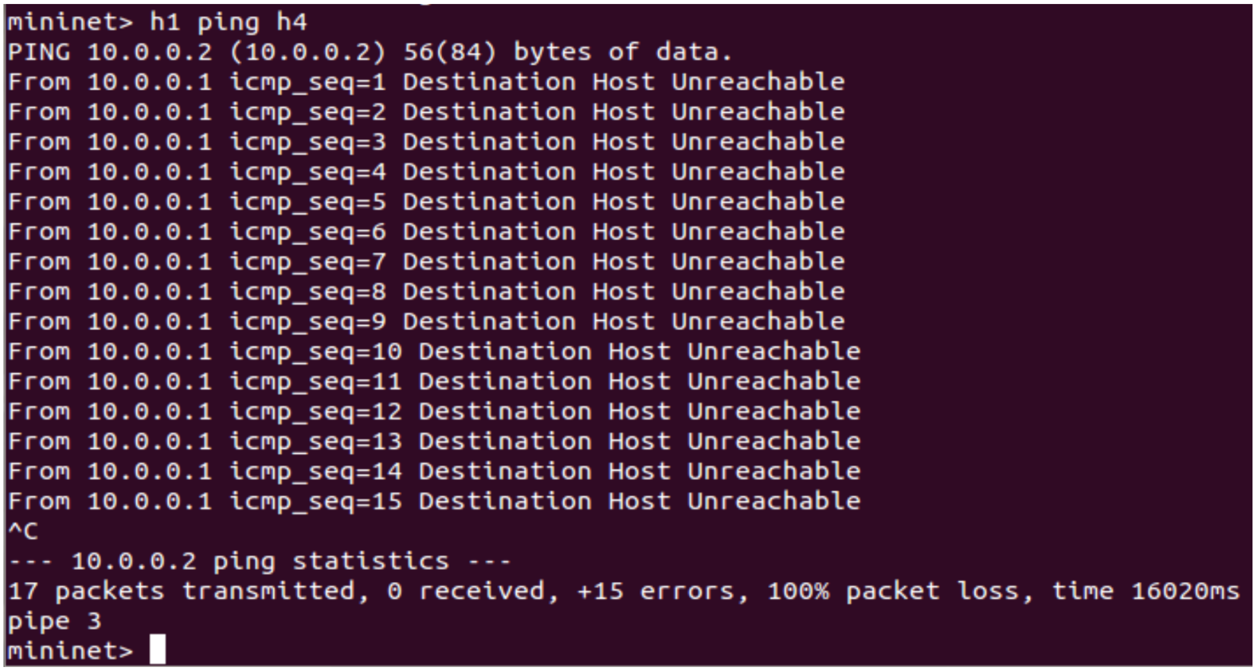
```
floodlight@floodlight:~$ sudo ./hw2
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h4
*** Adding switches:
s1 s2 s3 s4
*** Adding links:
(h1, s1) (s1, s2) (s1, s3) (s2, s4) (s3, s4) (s4, h4)
*** Configuring hosts
h1 h4
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
*** Starting CLI:
mininet> 
```



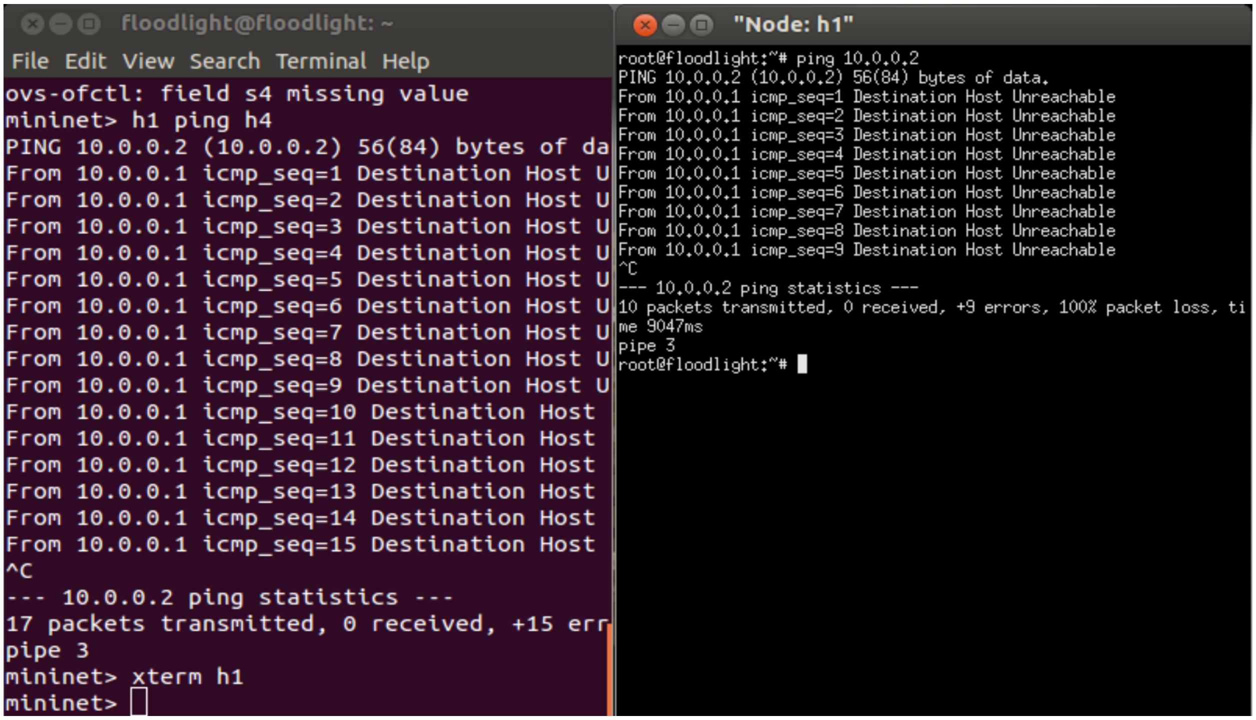
```
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
*** Starting CLI:
mininet> dpctl dump-flows s1
*** s1 ----------------------------------------------------
ovs-ofctl: field s1 missing value
*** s2 ----------------------------------------------------
ovs-ofctl: field s1 missing value
*** s3 ----------------------------------------------------
ovs-ofctl: field s1 missing value
*** s4 ----------------------------------------------------
ovs-ofctl: field s1 missing value
mininet> dpctl dump-flows s2
*** s1 ----------------------------------------------------
ovs-ofctl: field s2 missing value
*** s2 ----------------------------------------------------
ovs-ofctl: field s2 missing value
*** s3 ----------------------------------------------------
ovs-ofctl: field s2 missing value
*** s4 ----------------------------------------------------
ovs-ofctl: field s2 missing value
mininet> 
```



```
mininet> dpctl dump-flows s3
*** s1 ---------------------------------------------------
ovs-ofctl: field s3 missing value
*** s2 ---------------------------------------------------
ovs-ofctl: field s3 missing value
*** s3 ---------------------------------------------------
ovs-ofctl: field s3 missing value
*** s4 ---------------------------------------------------
ovs-ofctl: field s3 missing value
mininet> dpctl dump-flows s4
*** s1 ---------------------------------------------------
ovs-ofctl: field s4 missing value
*** s2 ---------------------------------------------------
ovs-ofctl: field s4 missing value
*** s3 ---------------------------------------------------
ovs-ofctl: field s4 missing value
*** s4 ---------------------------------------------------
ovs-ofctl: field s4 missing value
mininet> 
```



```
mininet> h1 ping h4
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable
From 10.0.0.1 icmp_seq=2 Destination Host Unreachable
From 10.0.0.1 icmp_seq=3 Destination Host Unreachable
From 10.0.0.1 icmp_seq=4 Destination Host Unreachable
From 10.0.0.1 icmp_seq=5 Destination Host Unreachable
From 10.0.0.1 icmp_seq=6 Destination Host Unreachable
From 10.0.0.1 icmp_seq=7 Destination Host Unreachable
From 10.0.0.1 icmp_seq=8 Destination Host Unreachable
From 10.0.0.1 icmp_seq=9 Destination Host Unreachable
From 10.0.0.1 icmp_seq=10 Destination Host Unreachable
From 10.0.0.1 icmp_seq=11 Destination Host Unreachable
From 10.0.0.1 icmp_seq=12 Destination Host Unreachable
From 10.0.0.1 icmp_seq=13 Destination Host Unreachable
From 10.0.0.1 icmp_seq=14 Destination Host Unreachable
From 10.0.0.1 icmp_seq=15 Destination Host Unreachable
^C
--- 10.0.0.2 ping statistics ---
17 packets transmitted, 0 received, +15 errors, 100% packet loss, time 16020ms
pipe 3
mininet> 
```



```
ovs-ofctl: field s4 missing value
mininet> h1 ping h4
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of da
From 10.0.0.1 icmp_seq=1 Destination Host U
From 10.0.0.1 icmp_seq=2 Destination Host U
From 10.0.0.1 icmp_seq=3 Destination Host U
From 10.0.0.1 icmp_seq=4 Destination Host U
From 10.0.0.1 icmp_seq=5 Destination Host U
From 10.0.0.1 icmp_seq=6 Destination Host U
From 10.0.0.1 icmp_seq=7 Destination Host U
From 10.0.0.1 icmp_seq=8 Destination Host U
From 10.0.0.1 icmp_seq=9 Destination Host U
From 10.0.0.1 icmp_seq=10 Destination Host
From 10.0.0.1 icmp_seq=11 Destination Host
From 10.0.0.1 icmp_seq=12 Destination Host
From 10.0.0.1 icmp_seq=13 Destination Host
From 10.0.0.1 icmp_seq=14 Destination Host
From 10.0.0.1 icmp_seq=15 Destination Host
^C
--- 10.0.0.2 ping statistics ---
17 packets transmitted, 0 received, +15 err
pipe 3
mininet> xterm h1
mininet> 
```

```
root@floodlight:~# ping 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable
From 10.0.0.1 icmp_seq=2 Destination Host Unreachable
From 10.0.0.1 icmp_seq=3 Destination Host Unreachable
From 10.0.0.1 icmp_seq=4 Destination Host Unreachable
From 10.0.0.1 icmp_seq=5 Destination Host Unreachable
From 10.0.0.1 icmp_seq=6 Destination Host Unreachable
From 10.0.0.1 icmp_seq=7 Destination Host Unreachable
From 10.0.0.1 icmp_seq=8 Destination Host Unreachable
From 10.0.0.1 icmp_seq=9 Destination Host Unreachable
^C
--- 10.0.0.2 ping statistics ---
10 packets transmitted, 0 received, +9 errors, 100% packet loss, ti
me 9047ms
pipe 3
root@floodlight:~# 
```