CS4226 Project

Student Info

Name	Muhammad Niaaz Wahab
Matric number	A0200161E

Setup

1. Put the `mininetTopo.py` in a separate directory
    a. Put the `topology.in` in this directory
2. Put the `controller.py` in a folder with pox.py
    a. Put the `policy.in` in this directory

Design decisions

Task 1

⇒ *modified* `mininetTopo.py`

Here, I only modified mininetTopo.py

I had to retrieve the information from text file

I parsed the information then

    created arrays to store hosts, switches and links

I then used `addSwitch`, `addHost` and `addLink` to add to generate the topology

Task 2

⇒ *modified* `controller.py`

Save packets to dictionary

I chose to use dictionary to have fast retrieval, compared to O(n) for a python array list

Task 3

⇒ *modified* `controller.py`

For this task, I simply stored TTL in another dictionary alongside the previous one which stored packet info

For each time that a packet comes in, I first update entries in both dictionaries

In thie section, I chose to only update the entries in the tables first only, without checking TTL yet

I did this because any entry that came in would update the table anyway, and override the entries in the dictionaries, so there would be no point in deleting the entries at that point.

At the end of processing I then check TTL. Packets who existed beyond TTL would be removed from both dictionaries.

This way, the information would only affect the next packet's navigation in the topo map.

Task 4

⇒ *modified* `controller.py`

I took the input when doing hte previous task, so that made doing this task slightly easier

My first approach was to take the 2 nodes offline. However, I quickly realised that would break the connection for other routes, not simply that link

I later followed the documentation to send the policies to each switch

Task 5

⇒ *modified* `mininetTopo.py` + `controller.py`

To decide which connection should be considered "premium", I could have chosen from the following

1. From premium source ⇒ premium
2. To premium destination ⇒ premium
3. Both source and destination premium ⇒ premium
4. Either source OR destination premium ⇒ premium

I chose to do #4, where so long as either of source of destination are premium, the connection would be premium

I did this because when a customer pays for premium service, they expect premium service regardless of whoever they are communicating with.

There could be a potential free-rider problem with the other non-paying for premium party, but the priority is to give premium-paying customers what they expect