CESE4045 - HDPN - Assignment 1 (SDN)

Student: Rafayel Gardishyan (Student ID: 5686547)

Exercise 1.1

File: `parkinglot.py`
Topo: `{'parkinglottopo': ParkingLotTopo}`
Command: `sudo mn --custom parkinglot.py --topo parkinglottopo,10`

Ping results (between `h1` and `h20`): on average around 105 ms.
This is to be expected, as the packet has to travel from host to the first switch, then through 9 links and from the last switch to the host again, resulting in $2 \cdot 9 \cdot 5 + a$ ms, where $a = 2 \times$ default delay on links (we only specify the delay on the switch links).

Exercise 1.2

File: `parkinglot_extended.py`
Topo: `{'extendedparkinglottopo': ExtendedParkingLotTopo}`
Command: `sudo mn --custom parkinglot_extended.py --topo extendedparkinglottopo,10`

Ping results (between `h1` and `h20`): DESTINATION HOST UNREACHABLE

Problem: Because there is more than 1 (bidirectional)path available between the source and destination hosts, the ARP packet gets stuck in a loop.

Observations:

- Host sends ARP request (Who is `h20`? Please tell `h1`)
- This packet gets flooded over the network (switch sends it to all the ports but the one it got it from)
- The packet traverses through the network and comes back to the switch connected to `h1`
- Switch is confused: the packet came from a different port than the original packet, thus the host must have switched a port
- In the meantime, `h20` sends an ARP reply (I am `h20`), but this never gets to `h1`, because the switch connected to h1 doesn't know which port h1 is connected to.

Solution: Using a switch which makes use of `Spanning Tree Protocol`. For this I made use of the OVSBridge class in Mininet, with the `stp` argument set to `True` (Spanning Tree Protocol).

Ping results (between `h1` and `h10`): on average around 12 ms. (expected)

Exercise 1.3

File: `aggregationtopology.py`
Topo: `{'aggtopo': AggTopo}`
Command: `sudo mn --custom aggregationtopology.py --topo aggtopo,10,4`

Ping results (between `h1` and `h10`): on average around 34 ms.
This is expected: $2 \cdot 3 \cdot 5 + a$ ms, where $a = 2 \times$ default delay on links.

Exercise 1.4

File: `parkinglot_extended_nofix.py`
Topo: `{'extendedparkinglottopo': ExtendedParkingLotTopo}`
Commands:
(1) `sudo mn --custom parkinglot_extended_nofix.py --topo extendedparkinglottopo,10`
(2) `./bin/ryu_manager ryu/app/simple_switch_stp_13.py`

Solution: The SDN controller now uses the `Spanning Tree Protocol` and writes the correct routing rules to the (dumb) switches, fixing the switching loop problem.

Note for Exercise 1.6: Sometimes the bandwidth measurement overshoots.This is due to VM performance. I tried running the code on another computer, and it doesn't happen there, so should be my VM