Project: DDoS Attack Simulation with Mininet

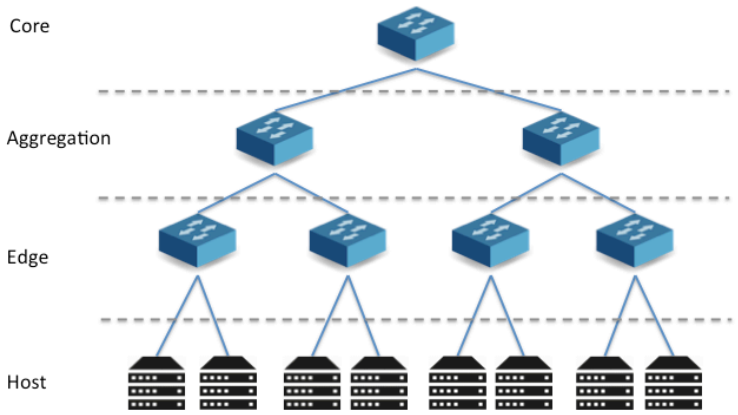
Due date: Monday, November 27th 10:30 AM

Description
In this project, you will simulate a distributed denial of service (DDoS) attack using the Mininet and socket Python APIs. Mininet will be used to create a virtualized network with a data center topology. Various metrics will be configured in order to make the simulation easier (i.e., link bandwidth, CPU resources, etc.). The Python socket API will be used to construct a simple code for DoS. You will use this constructed network and the DoS code to carry out DDoS attacks and measure the varying performance with respect to number of hosts.

Prerequisites:
Mininet: http://mininet.org/download/

Ubuntu users can install via `` `sudo apt-get install mininet` ``. Users with other OS (Windows, OSX, other Linux distribution) should download some virtual machine software (VirtualBox, VMware, etc.) and the VM image.

Objectives
A. Write python code to create a network topology using the Mininet API. Your topology will follow a typical data center network topology. This topology is a tree consisting of four levels: core switch, aggregation switches, edge switches, and hosts. The code should create the described topology with varying number of hosts. The number of switches should not change. This will allow you to measure performance as the number of nodes increases. The below image shows an example tree topology with 8 hosts.



An initial configuration of 8 hosts can be achieved by using the default tree topology in Mininet and setting the `depth=3` and `fanout=2`. However, in order to add more hosts/links and customize their parameters you should create your own topology. Below is an example implementation of a single switch topology from the Mininet documentation.

```
class SingleSwitchTopo(Topo):         "Single switch connected to n hosts."
  def build(self, n=2):
    switch = self.addSwitch('s1')
    for h in range(n):                 # Each host gets 50%/n of system CPU
      host = self.addHost('h%s' % (h + 1), cpu=.5/n)
      #10 Mbps, 5ms delay, 10% loss, 1000 packet queue
      self.addLink(host, switch, bw=10, delay='5ms', loss=10,
        max_queue_size=1000, use_htb=True)
```

B. Write python code to perform DDoS. For the sake of this project, we will consider a DDoS attack to be occurring if one of the following is true: (1) the host cannot handle the number of incoming requests, or (2) any links to the host are saturated. The first can be achieved by continuously connecting to, sending to, and disconnecting from a particular host. The second will more likely occur if the messages sent to the host are larger. You can use either method in developing your code. Use the socket programming learned in class.

C. Simulate a DDoS attack. The easiest way to ensure an attack occurs, even with a small number of nodes, is to limit CPU resources and link bandwidth.

```
self.addHost(name, cpu=f)      #"cpu=f" is the fraction of
CPU resources the virtual host receives

self.addLink(node1, node2, bw=10, delay='5ms',
max_queue_size=1000, loss=1)  #"bw=10" is the number of
Mb/s for link bandwidth
```

Perform these attacks with varying number of hosts (at least 8, 16, 32, 64, 128). Select a specific host to be the target, and execute the DoS attack with all other hosts. You should gather relevant metrics for measuring performance during DDoS, such as throughput, time for host to receive and process message, etc.

D. (**GRADUATE STUDENTS ONLY**) Write python code that implements DDoS protection at the host or switch. The goal of this code is to recognize a DDoS attack, identify the attacking hosts, and stop the attack. You should provide additional performance analysis with respect to the effect of your DDoS protection on the attack.

E. Write a brief project report with results. The report should include:

1. a brief description of your implementation (e.g., how you achieved the DDoS, how you measured results, etc.)
2. the different parameters and testing scenarios used

3. graphs depicting the performance and written explanations of the graphs
4. documentation for running your code

Submit

1. Your code with at least one comment for each function describing the role of the function. It would be beneficial to provide any additional comments in dense or large blocks of code.
2. Your typed project report.

Submit both a hard copy in class and electronic copy to `rplata1@lsu.edu`