Abstract

Software Defined Network (SDN) is a new type of network that can be programmed by a software controller according to various needs and purposes without actually modifying the hardware [1]. Mininet tool allows creation of realistic virtual network, running real Linux kernel and application code, on a single machine within seconds and hence plays a prominent role as a development platform for SDN [2]. The tool emulates hosts, network switches and controllers within a single virtual machine. The performance of the virtual network created by Mininet depends on the CPU speed. This is because Mininet uses OS level virtualization and there is no guarantee that an emulated host in Mininet that is ready to send a packet will be scheduled promptly by operating system to send a packet. This also results in low fidelity to reproduce an experiment [3]. The proposed solution involves running a simulation engine which acts as an interface between the network entities created by Mininet. All packets are transferred between entities via the simulation engine and hence it models and abstracts the key events related to the packet transfer. Thus these events can be run at much slower or faster speed than real world scenario, without affecting the fidelity of the experiment. With a simulation engine, test results can be consistently reproduced.
1. Introduction

Software-Defined Networking is a new approach to networking that aims to make data networks more flexible, easier to operate and manage, and better able to respond to the changing demands of applications and network conditions. SDN is a modification to the current network architecture and it is believed that SDN can help overcome a number of challenges by improving network utilization and efficiency, increasing automation of common tasks, lowering costs and improving security. SDN will overturn the role that switches and other network devices play by turning them into inexpensive machines for forwarding packets. The network intelligence will now be a separate entity residing in a centralized controller which is based on Open-flow protocol [4].

SDN application development and experimentation is gaining importance and Mininet tool turns out to be the best to the serve the purpose. Unlike experimental testing or simulating networks Mininet uses emulation in a virtual environment. Experimental testing incur huge cost owing to the large experimental test bed required [4]. The simulation approach has the disadvantage that if the modelling of real devices is not accurate enough, the simulation results may differ from the experimental results. Mininet uses virtualization approach to emulate hosts. It uses Open-flow switches (using Open v-switch software) to connect hosts on a physical server. Because emulated host in Mininet is like a virtual machine, real application can readily run on it without any modification. A real Open-flow controller, which is also a real application, can also run on an emulated host. With this approach, emulated hosts and software Open-flow switches can be connected together to form a desired network topology and be controlled by a real Open-flow controller[4]. Since Mininet makes use of OS level virtualization its performance and fidelity is affected by the CPU (machine on which the virtual network is set up) speed. Synchronizing the packet transfer for every host by running another application (simulation engine) can eliminate the bottlenecks. Simulation engine makes use of the notion of a software clock to achieve synchronism. [5]

2. Design Methodology

To build a prototype so that Mininet can be used for time related network and application performance. The idea is to build a simulation engine so that all packets are transferred between entities (switch, hosts) via the simulation engine. This simulation engine can schedule the packets on consistent basis between various entities [6]. The final proposal is to use Mininet for spawning various entities which include host, switches, controllers, as is the case today. However instead of using virtual Ethernet interface to transfer packets between various entities, a simulation engine is used. Simulation engine intercepts the traffic through each and every link. All packets are transferred between entities via simulation engine as shown in the figure 1.
The possible links are a host-switch vet pair, a host-host vet pair or a switch-switch vet pair. Each of these links is broken into two and the free end terminals are connected to the simulation engine. Such simulation engine would be able to model discrete events such as transmit of packets by an entity, transfer of packet between two entities, receive of packet by an entity. The above Figure depicts the brief overview of simulation engine inserted into the Mininet existing model.

3. System Architecture

Simulation engine is added between host and switch entity. Now the traffic between various entities passes through simulation engine. Data send from client hosts passes through simulation engine before reaching switches and server host. Simulation engine consists of buffers or queues such that every data send by each client hosts are stored inside the buffers in simulation engine [7]. Now this simulation engine is controlled by a tick from the master host. Data stored in the buffers from each client hosts are released as soon as the tick is received from master host. Simulation engine again releases data from buffers only after receiving the next tick from master. Thus simulation engine schedules the packets on a consistent basis between various entities which is shown in the figure 2. Mininet makes use of the addHost () function in the Mininet class for the creation of hosts. The same function was put to use for the creation of Simhost which acts as the simulation engine from the design. In the addHost () function basic parameters associated with a typical network node like IP address, MAC address etc. are initialized just before calling the Host class object[8]. Thus along with the hosts created as per the topology demand, an additional host named Simhost is also created. Simhost requires multiple interfaces unlike a single interface in the case of hosts. Thus no default IP or MAC address were specified while Simhost is created using the addHost () function.

![Figure 2: Network created after the addition of simhost](image-url)
3.1. Simhost Workflow

The Simhost has virtual interfaces as many as twice the number of hosts in the network topology. Each interface receives the packet based on the tick and stores it into a buffer and sends the packet from the buffer in another tick as shown in the above fig 3.
3.2. Master Workflow

The master is writing a tick value into a file which is read by the client to send the packet based on the tick. Here the Master is generating the traffic, Client is the traffic source (i.e. sending packets based on tick) and Server is the traffic sink (i.e. which receives the packets sent by the Clients).

Figure 4: Flow chart of Master program
Later a separate host named SIMHOST (simulation engine) is created which is added between the links of a pair of nodes created in the topology. It is created for load balancing.

3.3. USER_CMD workflow

![Flow chart of USR_CMD program.](image)

The USR_CMD host controls the master and simhost through commands. If the command received is c/s/h/f it sends the control to the master else to the simhost as shown in the above Fig 5.

4. Testing the Module with Single Switch Topology

In single switch topology any numbers of hosts are communicated with a single switch. So every host to host communication occurs only through a single switch. Here this topology is tested for 100 hosts and a single switch. When the command `sudo mn --topo single, 3` is executed, for the enhanced Mininet we are able to create the network topology given below. Here a Simhost or Simulation engine comes in between hosts and switch. Each host is connected to switch via Simhost. So all the packet transfer or data transfer occurs through Simhost or simulation engine.
When the command `sudo mn --topo single, 3` is executed, server program will be running in host1, client program in host 2 which has to send packet to server and master program will be running in host 3. The output is shown below:

```
mnet> @Mininet-v2.3.0:~$ sudo mn --topo linear,3 --mac
*** Creating network
*** Adding controller
*** Adding hosts: h1 h2 h3
*** Adding switches: s1 s2 s3
*** Adding links: (h1, s1) (h2, s2) (h3, s3) (s1, s2) (s2, s3)
*** Configuring hosts h1 h2 h3 M4
*** Starting controller
*** Starting 3 switches s1 s2 s3
Total number of hosts : 5
Number of Interfaces : 10
executing 2 th time h2
executing 3 th time h3
*** Starting CLI:
```

Figure 6: Single switch Topology with Simhost

Figure 7: Single switch topology with 3 hosts, Simhost and master.
Figure 8: server is receiving the packets sent from the clients through simhost interfaces.

5. Application
1. It is used as a learning tool for the users who are new to networking concepts.
2. Can be used to debug different networks.

6. Conclusion And Future Scope Of Work
Software Defined Networking(SDN) which allows users to manage the computer networking and Mininet tool which is used to create a realistic virtual network by creating switches, controller and application codes using single command and in single machine. The tool emulates hosts, Openflow switches, and Open flow controller within a single virtual machine. Mininet tool consists of several limitations due to its unpredictable packet forwarding rate and enhanced version made it suitable for time related applications and also for network debugging.

Now in the enhanced version of Mininet in which all packets are transferred between entities via the simulation engine and hence it models and abstracts the key events related to the packet transfer. Thus these events can be run at much slower or faster speed than real world scenario, without affecting the fidelity of the experiment. With a simulation engine, test results can be consistently reproduced. The main packet events for all entities, their durations are controlled consistently by simulation engine. This engine can be used to simulate high speed links. Traffic can be analyzed using network analyzer. As all the packet transfer occurs through Simhost these data can be analyzed using this tool.

New network features with entirely new architecture can be implemented by the user using the enhanced tool. Test module can be used on very large topologies with different types of application traffic. Exact code and tests scripts can be used in real production network after testing with the enhanced tool.

References
Maria Krupa A, Nikitha N, Rajani Reddy R, Vidya Shree R, Rajeswari :: Mininet Tool Enhancement For Experimentation And As A Development Platform For SDN Applications By Creating An Emulated Open-Flow Network In A Virtual Environment

Biographies

Ms. Maria Krupa A, an Under-graduation (B.E) student, Dept. Of Information science & engineering, New Horizon College of Engineering Bengaluru, Karnataka affiliated to visvesvaraya University Belgaum, Karnataka. Areas of interest are networking, Data mining.

Ms. Nikitha N, an Under-graduation (B.E) student, Dept. Of Information science & engineering, New Horizon College of Engineering Bengaluru, Karnataka affiliated to visvesvaraya University Belgaum, Karnataka. Areas of interest are networking, Web Designing.

Ms. Rajani Reddy R, an Under-graduation (B.E) student, Dept. Of Information science & engineering, New Horizon College of Engineering Bengaluru, Karnataka affiliated to visvesvaraya University Belgaum, Karnataka. Areas of interest are networking, robotics.

Ms. Vidya Shree R, an Under-graduation (B.E) student, Dept. Of Information science & engineering, New Horizon College of Engineering Bengaluru, Karnataka, affiliated to visvesvaraya University Belgaum, Karnataka. Areas of interest are mobile robotics, networking.

Mrs. Rajeswari, Assistant Professor, Department of Information Science & Engineering, New Horizon College of Engineering, Karnataka, affiliated to visvesvaraya University Belgaum, Karnataka, India. She has done her Mtech in Computer Science and Engineering. She has teaching experience of 11 years. Her research interests include Digital Image Processing. She has published 6 papers till date in national and international journals.

References:


