**FLOWNET: A WEB INTERFACE FOR SOFTWARE-DEFINED NETWORK CONSTRUCTION AND PACKET FLOW TRACING IN MININET**

A MAJOR PROJECT REPORT

**DECLARATION**

**ABSTRACT**

In this research, we present FlowNet, a web interface that gives a platform for understanding the use of Mininet and Software-Defined Networking and packet flows happening in it. Mininet is a well-known network emulator in academics and research. Mininet can simulate both traditional and software-defined networks but does not provide an interface for understanding the packets that pass over the constructed network architecture. This interface contains the components needed to launch, visualize, and alter Mininet network flows in real time and comprehend packet matching in SDN. FlowNET provides a real-time online interface for network configuration via its tools. This introduces FlowNET as a tool for studying network protocols as well as teaching, learning, and comprehending network ideas. In this project, We are proposing to establish an efficient web interface that shows a variety of use cases and examples of how to use FlowNET to establish networks and see the resultant network flows with customizable preferences built with FlowNET.

**KEYWORDS:** mininet, POX, packet flow, flask, OVswitch

**LIST OF FIGURES**

**Figure Name Page No.**

1. Architecture of FlowNET -------------------------------------------------------------------- 12
2. Design of FlowNET --------------------------------------------------------------------------- 13
3. Web Interface of FlowNET ------------------------------------------------------------------ 14
4. Web Interface with Pinging Output --------------------------------------------------------- 14

**LIST OF ABBREVIATIONS**

|  |  |
| --- | --- |
| **Abbreviation** | **Expansion** |
| **SDN** | **Software-Defined Network** |
| **VND** | **Virtual Network Distribution** |
| **ISL** | **Inter-Satellite Connections** |
| **GNN** | **Graph Neural Network** |
| **POX** | **Python-Based Open Source-Openflow/SDN controller** |

**INDEX**

**Topic Page. No**

Abstract------------------------------------------------------------------------------------------- 4

Acknowledgment-------------------------------------------------------------------------------- 5

1. Chapter 1-Introduction--------------------------------------------------------------------- 7
2. Chapter 2- Literature Survey-------------------------------------------------------------- 8
3. Chapter 3-Proposed Work ---------------------------------------------------------------- 10
   1. Objective of the work ----------------------------------------------------------------- 10
   2. Architecture --------------------------------------------------------------------------- 10
   3. Design overview ---------------------------------------------------------------------- 11
4. Chapter 4-Inference and Result----------------------------------------------------------- 12
5. Chapter 5-Conclusion --------------------------------------------------------------------- 13
6. Chapter 6 - References--------------------------------------------------------------------- 14

**CHAPTER 1**

**INTRODUCTION**

Network simulation and emulation tools like Mininet are frequently used in academia to learn and comprehend network concepts and validate new network protocols. Simulators and emulators face a number of difficulties when trying to monitor the state of the network across a large number of network entities and analyze the intricate and frequent message exchanges between these entities.Simulators produce packet traces with a lot of detail and static outputs, making it difficult for the user to comprehend the data. In a similar vein, statistics in emulated networks are typically gathered in their raw form or at the conclusion of the emulation, concealing the dynamics of the protocol's behavior. These issues are addressed by animation and visualization tools for networks.Users can quickly collect a large amount of traffic details using a network animator visualize communication patterns, and gain a deeper comprehension of interaction and causality.

By separating the data plane and the control plane, software-defined networking (SDN) technology aims to centralize network intelligence. While the data plane is managed by a set of switches, the control plane is organized by a set of controllers. As a result, these devices are only responsible for forwarding packets.A centralized control plane known as an SDN controller can be either physical or virtual, or even both.

One of the most widely used network emulators is Mininet.Compared to simulators like NS2, it creates a virtual network with fully functional nodes and allows interaction between these nodes, providing a more complete and realistic picture of a network. Support for OpenFlow and Software-Defined Network (SDN) systems is one of Mininet's key features. Mininet can make it simple to set up SDNs by issuing a single command. Mininet provides some visual tools, such as MiniEdit and Virtual Network Description (VND)[1], that make the creation of a network topology even simpler. However, in contrast to NS2, Mininet does not provide a tool that can be used to visualize the traffic and packets that are moving over the network topology that has been created. In the next chapter we will see different models and applications that several authors have done in this field of work.

**CHAPTER 2**

**LITERATURE SURVEY**

Ahmed Khalid, Jason J. Quinlan, and Cormac J. Sreenan proposed a MiniNAM which is a program that shows networks created by the mininet simulator. Mininet doesn’t have a visualization tool to see the packet flows between the networks. All the components required to start, monitor, and alter Mininet network flows in real-time are included in the tool. A user has the ability to color-code packets according to packet type and source node while viewing certain flows. Three different network protocol samples including an interactive real-time UDP/TCP demonstration using iperf are used to illustrate MiniNAM. They demonstrated how MiniNAM may be used to see a fully functional network and the packet flows associated with it. The research also shows how MiniNAM may be helpful when the network controller is unable to operate the network effectively because a protocol, node, or flow was implemented incorrectly. The three protocol examples are NAT, load balancing, and re-routing.[1]

The platform that is suggested in the cited study is primarily made up of the four components shown in the picture, which include an SDN controller platform, a network simulation platform, astronautically physics simulators, and a graphical user interface platform. In the SDN controller platform, ONOS is the controller. It builds an end-to-end communication link using fundamental network service functions and interfaces to obtain the current topology. A network simulation tool for SDN networks is called Mininet. It includes interfaces with the SDN controller and can mimic network components including hosts, switches, and connections in the network. Based on data simulated by STK(Sattelite tool kit) in the Mininet creates a dynamic network architecture. They employed STK as an astronautic physics simulator, which can be used for mission planning and communications as well as simulating location and orbit data and analyzing ground coverage. A platform for displaying satellite performance and network topology is the graphical user interface. They considered building a more specialized resource model in subsequent work to enhance the performance of other constellations and algorithms. [2]

The paper referred to has developed the future work of the before research i.e the SDN controller platform with ONOS controller, Mininet, and STK the simulate the SDN. They created a three-layer Walker constellation in GEO, MEO, or LEO, a topology creation technique based on ISL (inter-satellite connections) availability periods, and a routing algorithm taking both processing delay and transmission delay into account. The time slice division method and the time slice topology algorithm were used to split the topology creation process into two separate algorithms. The initialization and update calculations, which are applied to time slices in accordance with the following principles, make up the time slice topology algorithm. Only the first-time slice uses the initialization calculation; all subsequent time slices utilize the update computation. One of the most crucial elements that must be taken into account in the network is route latency. The routing delay for a service traveling across the network is made up of two components: processing delay and transmission delay. This platform supports the simulation of ground networks with good performance in solving the issues caused by satellite networks.[3]

In the study, they introduced RouteNet, a novel network model based on Graph Neural Network (GNN) that can comprehend the intricate interaction between input traffic, topology, and routing in order to give precise estimates of the per-source/destination per packet delay distribution and loss. This model is able to generalize over different topologies, routing protocols, and varying traffic intensity since GNNs are designed to enable relationship reasoning and combinatorial generalization over information organized as graphs. They used RouteNet to discover the ideal link location in a network design scenario and QoS-aware routing optimization based on the requirements for the delay, jitter, and packet loss.[4]

In order to construct a programmable network architecture that can be deployed progressively in production networks and allows for the potential of isolating the control mechanisms of the various traffic flows, one option is to use the OpenFlow architecture. The OpenFlow interface is implemented by the SDN network simulator Mininet in a network simulation scenario consisting of two parts, one OpenFlow switch, and three nodes that make up a controller POX. The major goal was to gauge bandwidth and communication between nodes. A single OpenFlow switch (S1) that is linked to three hosts (h2, h3, and h4) as well as a controller POX make up the simulation scenario. Hub and Switch are two newly built components that make up this controller. Open two terminals, one for Mininet and the other for the POX, in order to set up the simulation scenario. It utilized commands in the MiniNet terminal to create a topology. They considered using OpenFlow and Software Defined Networks to implement energy-saving measures in cloud computing in the future.[5]

**CHAPTER 3**

**PROPOSED WORK**

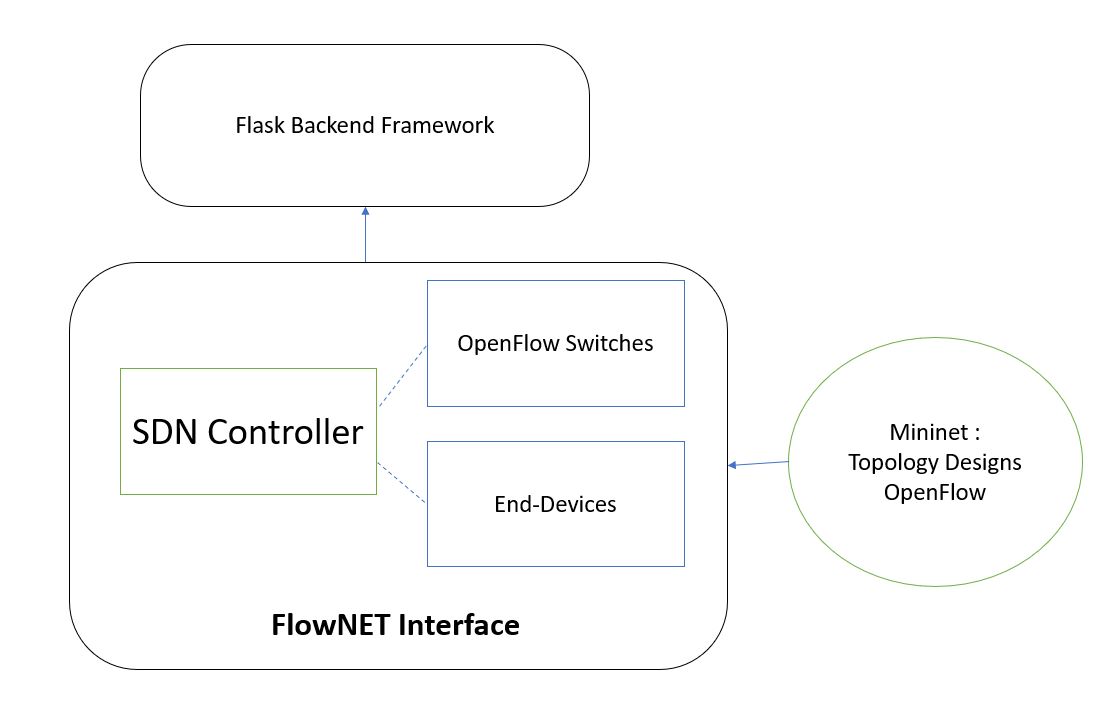
**3.1 OBJECTIVES:**

• To Establish a web interface that can design a topology using a mininet framework for Software Defined Networks.

• Enabling the Interface to exhibit the packet flow through OpenFlow protocols under the supervision of the SDN controller.

• Making a well-defined learning base through animating and providing utilities of mininet as an interface helps researchers and scholars to understand the working of Software-Defined Networks.

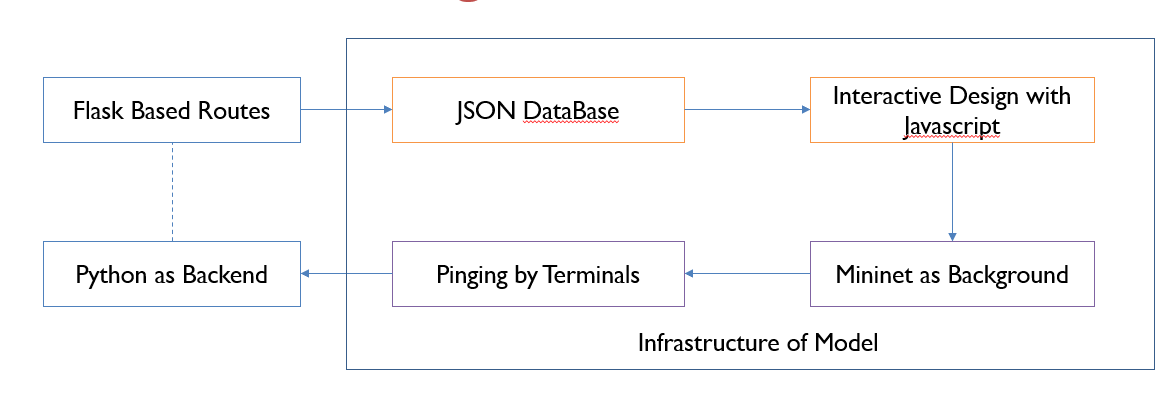
**3.2 ARCHITECTURE**



*Fig 1 Architecture of FlowNET*

**3.3 DESIGN OVERVIEW**

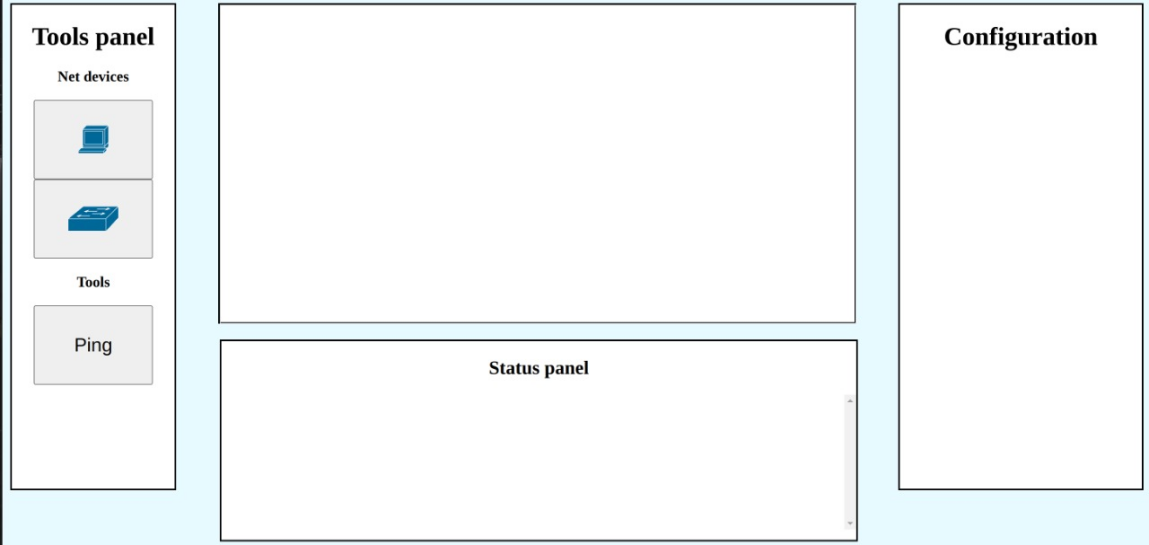
Initially a flask based interface has been created which allow us to work with different routes and pages. These routes helps to post and get data whenever it is required for us using forms as a database. For Database, we utilized the concept of a JSON file where we store all information about a switch, and host that might include IP address, x-y coordinate position on screen, selected or not , and so. When a form iis posted to some route , it can be retrieved from same route later using Javascript. It provides us with all interactive implemention like canvas for topology , links, button click functions and so. Now After all this has been done , python packages has been written as backend to say what every component in our topology has to do. Terminal Commands and Pinging has been done using these files where ip address will be posted to a route when ping is clicked and it will be retrieved later by a function and pinged using ‘ping’ command. As SDN controller, POX has been used and switch in this interface is considered as openVswitch as these modules provide such functionality. Finally integrating all these modules and packages we have created a interface which allow us to ping and send packets in a SDN Network.



*Fig 2 Design of FlowNET*

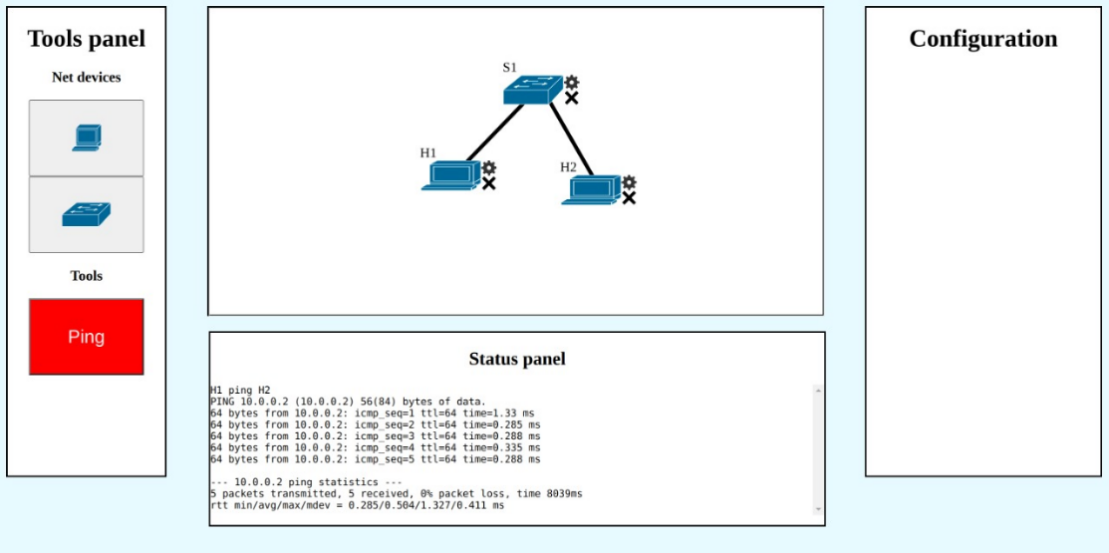
**CHAPTER 4**

**RESULTS ANALYSIS AND DISCUSSION**

****

*Fig 3 Web Interface of FlowNET*

From the Design exhibited in above Figure 3, We have accomplished a SDN network using Mininet as Core Emulator. This interface currently provides with the access to create a topology of switches, routers and host devices. On behind, there runs a SDN controller to manage all data flow between the devices and in our case as mention earlier POX controller has been Utilized for this purpose. For this interface to run mininet has to be ran as core in background terminal as it controls all action based on commands from the interface when it is pinged as we do in terminal version of Mininet. We can configure our IP addresses, Masks, gateways everything using a configuration panel in the sidebar. Below Figure 4 , is a output example of pinging between 2 hosts in single topology where we can see the packets has been received by the destination host.

****

*Fig 4 Web Interface with Pinging Output*

**CHAPTER 5**

**CONCLUSION AND FUTURE WORK**

Stack used for designing the user interface: Flask, backend framework which utilizes python libraries, Mininet network Emulator Utilities, and assistance Web Developing Tools such as HTML5, CSS, and so. In this briefing, we exhibit our application FlowNET via a web interface that aids in packet flow understanding over OpenFlow protocols. In our real-time integrated platform, we use iperf to generate traffic and FlowNET utilities to adapt and change the traffic flows. We show how FlowNET may be used to see a fully functional network and its related packet flows. Furthermore, and more crucially, we demonstrate the efficacy of FlowNET via use cases and how an SDN controller controls as a single logical switch. While the examples cover a wide variety of use cases, the interactive presentation shows how to visualize and configure various FlowNET settings. We also begin Mininet network topologies, modify settings, create traffic throughout the network in this presentation. In our Future work we would like a present a visualization of packetflow , flowtable and how a flow has been delivered by the controller. There is no option for choosing controller and that has to be provided as an add-on.

**CHAPTER 6**

**REFERENCES**

1. Khalid, J. J. Quinlan, and C. J. Sreenan, "MiniNAM: A network animator for visualizing real-time packet flows in Mininet," 2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN), 2017, pp. 229-231, DOI: 10.1109/ICIN.2017.7899417.
2. X. Guo, B. Guo, K. Li, C. Fan, H. Yang and S. Huang, "A SDN-enabled Integrated Space-Ground Information Network Simulation Platform," 2019 18th International Conference on Optical Communications and Networks (ICOCN), 2019, pp. 1-3, doi: 10.1109/ICOCN.2019.8934091.
3. Yu Zhang, Bo Wang, Bingli Guo, Yabo Yuan, Tao Dong, Jie Yin, Kexin Li, Xiang Guo, Shanguo Huang,A Research on Integrated Space-Ground Information Network Simulation Platform Based on SDN, Computer Networks, Volume 188,2021,107821, ISSN 1389-1286,https://doi.org/10.1016/j.comnet.2021.107821.
4. K. Rusek, J. Suárez-Varela, P. Almasan, P. Barlet-Ros, and A. Cabellos-Aparicio, "RouteNet: Leveraging Graph Neural Networks for Network Modeling and Optimization in SDN," in IEEE Journal on Selected Areas in Communications, vol. 38, no. 10, pp. 2260-2270, Oct. 2020, DOI: 10.1109/JSAC.2020.3000405.
5. Ligia Rodrigues Prete, A. A. Shinoda, C. M. Schweitzer, and R. L. S. de Oliveira, "Simulation in an SDN network scenario using the POX Controller," 2014 IEEE Colombian Conference on Communications and Computing (COLCOM), 2014, pp. 1-6, DOI: 10.1109/ColComCon.2014.6860403.