Student: Siddharth Bidwalkar
Advisor: Dr. Peizhao Hu

Overview
- Performing efficient in-network processing among large scale heterogeneous devices requires lightweight design of processing unit.
- Open vSwitch (OvS) is a virtual switch responsible for packet forwarding among network devices as per forwarding rules given by SDN controller.
- Due to varying processing capabilities and resource constraints, not all IoT devices are equipped to handle in-network processing provided by OvS.
- This project presents a lightweight version of OvS with all basic functionalities intact and few insignificant functionalities removed to make it compatible with all IoT devices.

Software Defined Network
- Software Defined Network (SDN) is a network architecture which decouples control plane and data plane.
- Control logic of the network is implemented in Controller while network switches (like OvS) become simple data forwarding devices in Data plane.
- A programming interface (like Openflow API) is used for communication between Controller and switches in data plane.

Open vSwitch
- Open vSwitch (OvS) is a virtual switch with three basic components: ovs-vswitchd, ovsdb-server and Kernel datapath.
- Ovs-vswitchd is OVS userspace daemon which receives Openflow tables from controller.
- Kernel datapath stores flow cache of packets which it receives from ovs-vswitchd.
- The ovsdb-server provides RPC interfaces to one or more Open vSwitch databases. It allows adding and removing ports, configure QoS queues, associate Openflow controller and switches, enable and disable Spanning Tree Protocol.

Insignificant Functionalities
- OvS Pcap: It reads pcap file from command line and prints each packet's contents as sequence of hex digits.
- Port Mirroring: It creates virtual port which receives copy of all network packets seen on another port.
- Interface Rate Limiting: It ensures packets arrive to or depart from a port at specified rate. It helps in determining QoS for a packet.
- VTEP: It helps OvS emulate a hardware VXLAN Tunnel Endpoint.
- Flow Statistics: It includes lookups, hits and misses for each packet matched at kernel datapath.

Results
- All three versions of OvS were tested on Ubuntu 64-bit Operating system by creating virtual network and passing packets using Mininet.
- Maximum memory required by OvS to run decreased by 9%
- Size of OvS decreased by 0.5%
- Average %CPU used by OvS in Ubuntu 64-bit Operating System decreased from 1.6% to 1%

Challenges
- Although detailed documentation about design and working of main functionalities of OvS is readily available, documentation of insignificant functionalities is hard to find and time consuming.
- As the source code of OvS which is written in C language is not well documented, debugging the code to find functions and its dependencies takes time

References
- “The Design and Implementation of Open vSwitch”, Operational Systems Track