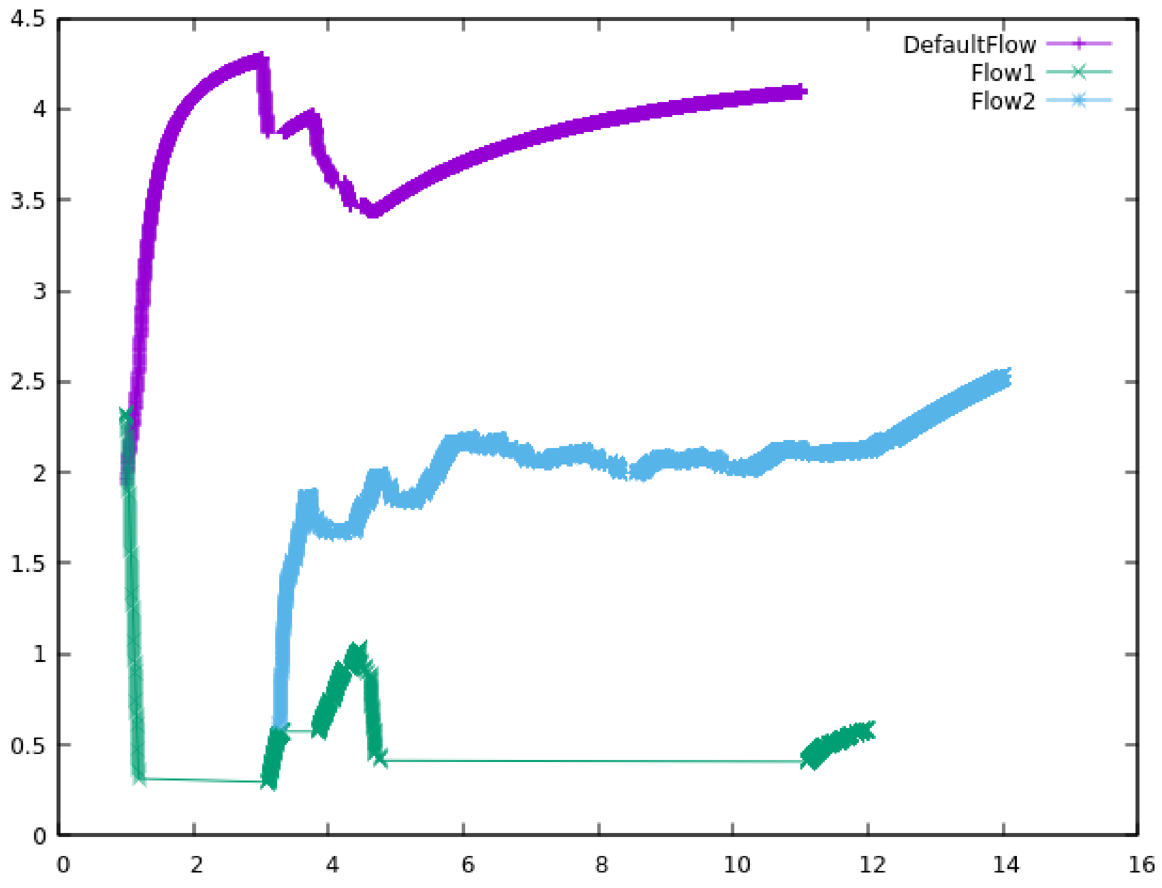
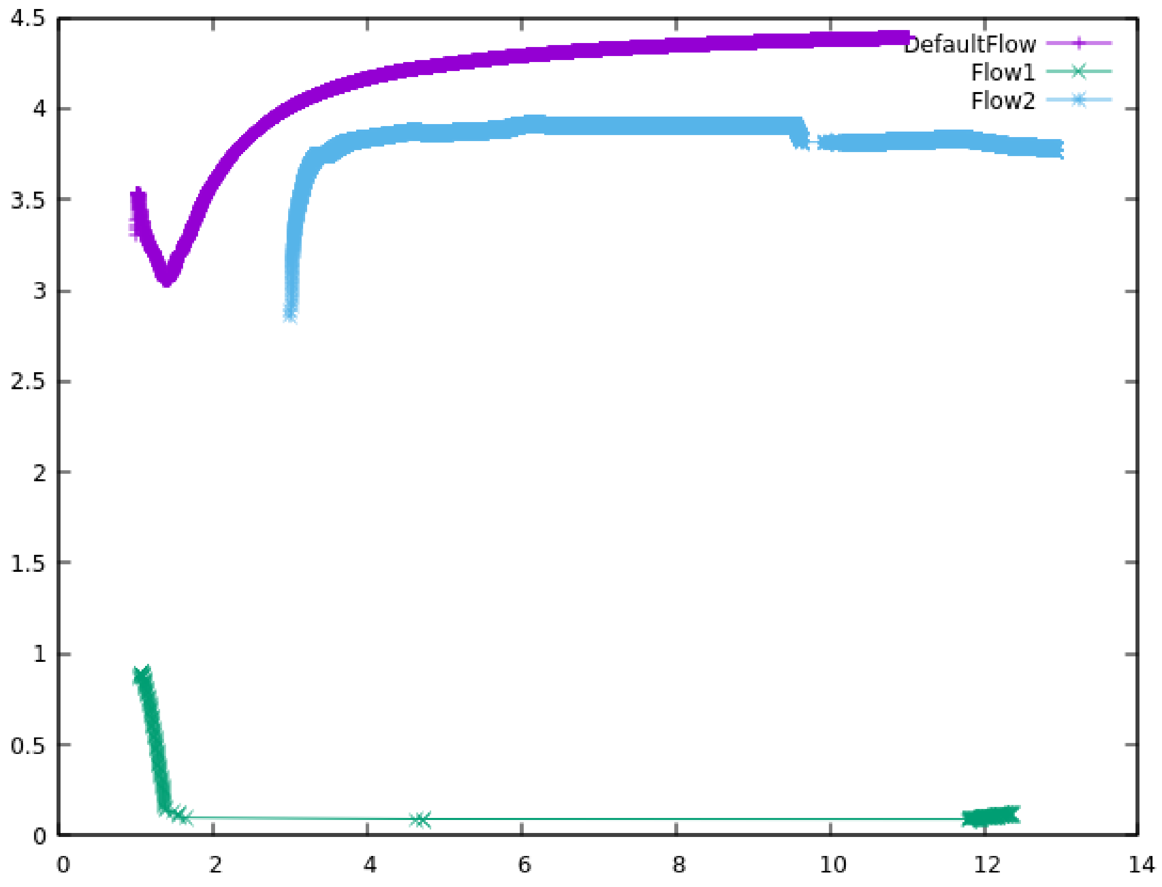
2016312761 여혁수 as1 report

Q1.



Q2.



Q3. Explain the difference of flows when Flow1 and Flow2 are set to TCP and UDP.

When Flow1 and Flow2 are set to UDP, Flow1 quite smoothly send many bytes. And when application of Flow2 starts, Flow1 disturbs communication of rest two flows (default, flow2) for several seconds by sending some packets. However, when Flow1 and Flow2 are set to TCP, Flow1 can't send packets almost at all except right after the application starts and right before the application stops. Because two flows which are relatively fast blocks the communication of Flow1. As a result, speed of DefaultFlow and Flow2 are decreased by intervening of Flow1 when Flow1 and Flow2 are set to UDP. While showing consistent high speed of DefaultFlow and Flow2 if Flow1 and Flow2 are set to TCP. Thereafter, speed of Flow1 is higher when Flow1 and Flow2 are set to UDP than TCP.

Q4.

(sending)If connection of socket is succeeded, ScheduleStartEvent() is called in StartApplication() of onoff-application.cc. Then this call flows to StartSending() -> ScheduleNextTx(). SchedulNextTx() function schedules timing of sending packets, and call SendPacket(). In SendPacket(),

packet is created and call Send() with packet, then logging the sending info.

When Send() is called, This function is in udp-socket-impl.cc, and this function returns the result of DoSend(packet). DoSend() function again returns the result of DoSendTo (packet). In DoSendTo function, PacketTag is added to each packet. Then the copy of packet goes to UdpL4Protocol::Send function in udp-l4-protocol.cc again. Header is added to each packet in this function. Then m_downTarget callback function is called. This function callback to Ipv4L3Protocol::Send() function. This function call outInterface->Send function with packet. So go to Ipv4Interface::Send in ipv4-interface.cc, and this function call again m_device->Send() function with packet. So LoopbackNetDevice::Send() function is executed and Simulator::ScheduleWithContext() function is called with packet.

After scheduling and return of SendPacket(), recall ScheduleNextTx(). Then repeat the scheduling and this cycle is repeating until sending bytes amount of value of variable named m_maxBytes defined in ScheduleNextTx() function. Then ScheduleNextTx() calls StopApplication().

(receiving)If NetDevice calls the function m_receiveCallback(), it goes to Node::ReceiveFromDevice() function. And this function results in calling Ipv4L3Protocol::Receive() function. Ipv4L3Protocol removes the IP header. Then pass the packet to Ipv4RoutingProtocal. After that, looks up the protocol and again call Receive() function. Then in Receive() function, UDP header is removed and calls Ipv4EndPoint::ForwardUp() function. Ipv4EndPoint has callback function, and goes to UdpSocketImpl::ForwardUp() function. In this function, Recv() callback function is called and finally application can read data from the socket. Per received packet, the function logged some information about packet, and get name of socket. After that, m_rxTrace function which has packet as parameter is called. So packet goes to rx function if it is exists, and print some info or something.