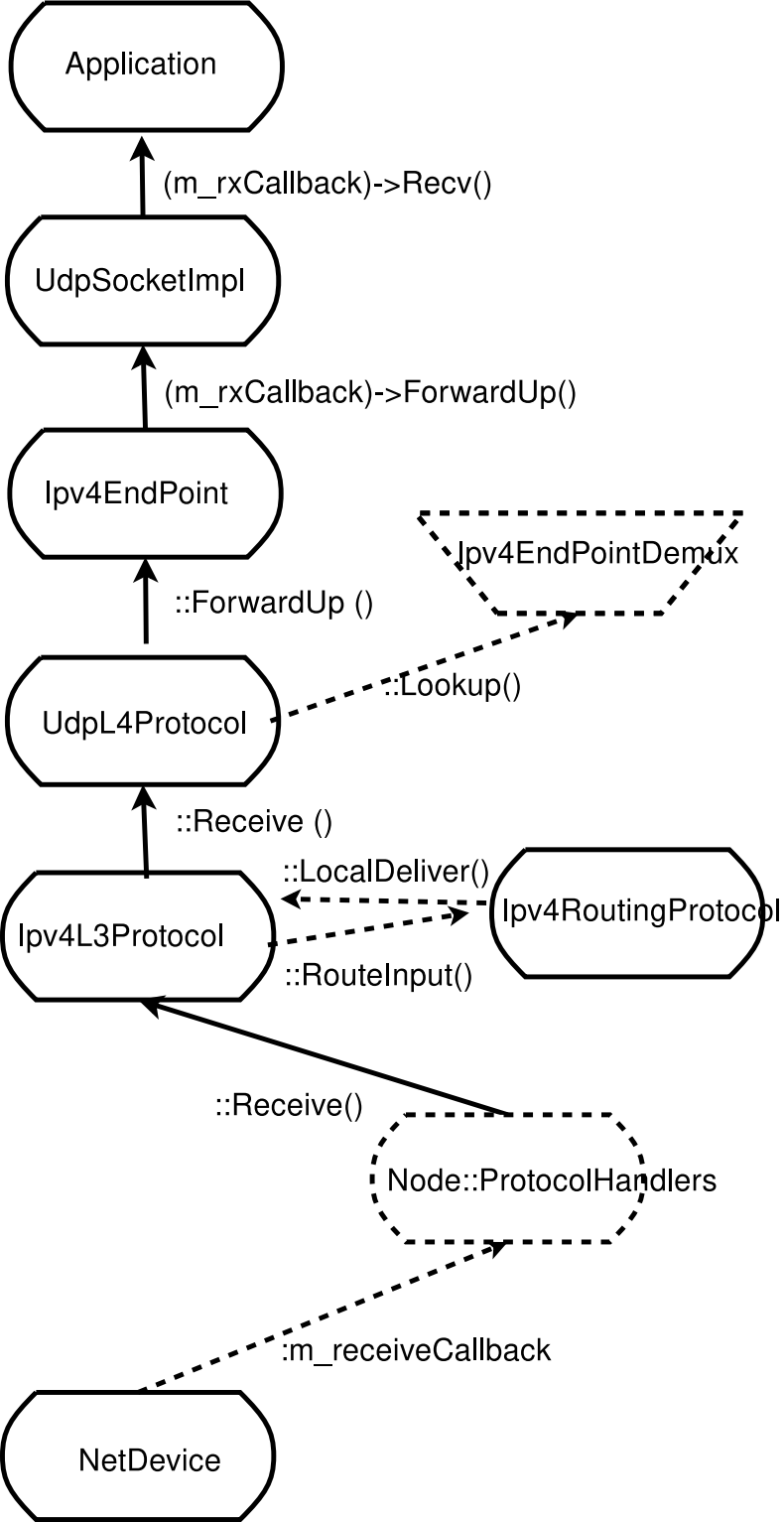

Step in packet receiving process

7. UdpSocketImpl itself calls the Recv() callback set by the Application when data is ready to be read. The application can then call the socket Recv() or RecvFrom() methods to read data (or dummy data) from the socket.

6. Ipv4EndPoint has a callback where a Socket object is able to register a receive method. Here, this callback calls to UdpSocketImpl::ForwardUp()

5. UdpL4Protocol is where the socket-independent protocol logic for UDP is implemented. The Receive() method removes the UDP header and looks up the per-flow context state, which is one or more Ipv4EndPoint objects stored in an Ipv4EndPointDemux (indexed by src addr, src port, dest addr, dest port). It then calls Ipv4EndPoint::ForwardUp() when done.

4. Ipv4L3Protocol removes the IP header, checks checksum (if implemented), and passes the packet to the Ipv4RoutingProtocol registered with Ipv4L3Protocol. The routing protocol in this case decides the packet is for the local host, so it calls back to Ipv4L3Protocol::LocalDeliver(). This function looks up the protocol (in this case UDP) and calls the Receive () method for that protocol.

3. Node::ReceiveFromDevice stores a set of callbacks (protocol handlers) that are looked up based on protocol number and device. In this case, the lookup will result in an Ipv4L3Protocol::Receive() being called.

2. This is typically the Node::ReceiveFromDevice() function

1. NetDevice calls the function registered at Node::m_receiveCallback