

Florida International University
School of Computing and Information Sciences

Software Engineering Focus

Final Deliverable

AmLight - Learning OpenFlow/SDN Network Research and
Experimentation

Team Members: Cindy Perez, Jose Mercardo, Kenia Chang He, and Yuyang Leng

Product Owner: Vasilka Chergarova

Mentor: Jeronimo Bezerra

Instructor: Masoud Sadjadi

The MIT License (MIT)
Copyright (c) 2020 *Florida International University*

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Abstract

OpenFlow networking allows for the management and trafficking of packets using switches. Using the capabilities of different packet types, OpenFlow allows for masking and the matching of these different packet types. IPv6 is one of the packet types, which allows for faster sending and receiving on the networking side; however, IPv6 packets tend to be larger than IPv4 packets.

Table of Contents

Introduction.....	6
Current System.....	6
Purpose of New System	6
User Stories.....	7
Implemented User Stories.....	7
Pending User Stories.....	17
Project Plan	18
Hardware and Software Resources	18
Sprints Plan	20
Sprint 1.....	20
Sprint 2.....	20
Sprint 3.....	22
Sprint 4.....	24
Sprint 5.....	25
Sprint 6.....	26
System Design	27
Architectural Patterns.....	27
System and Subsystem Decomposition	27
Deployment Diagram.....	28
Design Patterns	28
System Validation.....	29
Match Fields Test Cases	29
IPv6 Packet Unit Test	30
Glossary	31
Appendix.....	32
Appendix A - UML Diagrams	32

Use Case Diagrams	32
Class Diagrams	36
Sequence Diagrams	40
Appendix B - User Interface Design	44
Appendix C - Sprint Review Reports	46
Sprint 1	46
Sprint 2	47
Sprint 3	49
Sprint 4	52
Sprint 5	54
Sprint 6	55
Appendix D - User Manual, Installation Guide, and Wishlist	57
User Manual	57
Installation Guide	62
Wishlist	66
References	67

INTRODUCTION

This project is an iteration of an already existing system called Kytos OpenFlow. The purpose of Kytos OpenFlow is to allow users to create network maps using the OpenFlow Logical Switch as a software designed network controller. Our project sought to expand on the capabilities of the current system into a new version, Kytos OpenFlow version 1.3. This expansion would allow users to use many more packet types, including the masking of those types, to create a more realistic network.

Current System

Kytos OpenFlow is a logical switch controller for Kytos. It allows users to use different types of packets to create a realistic network map. An OpenFlow logical switch contains flow tables and group tables which match the user's packet type. This switch communicates with a controller and the controller communicates with the switch by accessing the OpenFlow switch protocol.

Purpose of New System

For the new system, our product owner desired the following functionality:

- The ability for users to have a more extensive packet type to match from.
- The ability for users to use masking with their packets.
- The ability for users to send IPv6 packets.

USER STORIES

This section covers both the implemented and pending user stories.

Implemented User Stories

- **User Story 1**
 - As a developer, I want to set up the environment for Kytos, so that I can start testing and developing applications with Kytos.
 - Acceptance criteria:
 - ☐ Complete “Setting up your dev environment” on kytos.io
- **User Story 2**
 - As a developer, I want to go over the Kytos tutorials, so that I have the knowledge for Kytos.
 - Acceptance criteria:
 - ☐ Complete “How to create your NApp” series on kytos.io
- **User Story 3**
 - As a developer, I want to read over the OpenFlow documentation given to us by the Product Owner, so that I can become familiar with the software we are working with.
 - Acceptance criteria:
 - ☐ Complete reading section 7 on the OpenFlow Switch Specification version 1.3.5 document
- **User Story 4**
 - As a developer, I want to review the source code given to us by the Product Owner, so I can find where the issue is occurring and come up with a solution.
 - Acceptance criteria:
 - ☐ Complete reading the flow.py and v0x04/match_fields.py files on the kytos/of_core repository
- **User Story 5**
 - As a developer, I want to start working on a solution for the issue #75 that was given to us by the Product Owner, so I may implement the solution on the software given to us.
 - Acceptance criteria:
 - ☐ Create pseudo code with the team on masking
- **User Story 6**
 - As a user, I want to be able to use the mask function for the Datalink VLAN ID matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow

- v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 7**
 - As a user, I want to be able to use the mask function for the VLAN Priority Code Point matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 8**
 - As a user, I want to be able to use the mask function for the Datalink Source matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 9**
 - As a user, I want to be able to use the mask function for the Datalink Destination matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 10**
 - As a user, I want to be able to use the mask function for the Datalink Type matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase

- ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 11**
 - As a user, I want to be able to use the mask function for the IPv4 Source matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 12**
 - As a user, I want to be able to use the mask function for the IPv4 Destination matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 13**
 - As a user, I want to be able to use the mask function for the IP Protocol matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 14**
 - As a user, I want to be able to use the mask function for the Input Port matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 15**

- As a user, I want to be able to use the mask function for the TCP Source matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 16**
 - As a user, I want to be able to use the mask function for the TCP Destination matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 17**
 - As a developer, I want to test the implementation for issue #75, so that I can make sure it is working for the users.
 - Acceptance criteria:
 - ☐ Test using the User Manual for each different match field
- **User Story 18**
 - As a user, I want to use the physical input port as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 19**
 - As a user, I want to use the IP DSCP as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 20**

- As a user, I want to use the IP ECN as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 21**
 - As a user, I want to use the UDP Source as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 22**
 - As a user, I want to use the UDP Destination as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 23**
 - As a user, I want to use the ICMPV4 Type as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 24**
 - As a user, I want to use the ICMPV4 Code as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 25**

- As a user, I want to use the IPV6 Source as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- Acceptance criteria:
 - ☐ Implement the IPV6 address class
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 26**
 - As a user, I want to use the IPV6 Destination as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Implement the IPV6 address class
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 27**
 - As a user, I want to use the IPV6 FLabel as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 28**
 - As a user, I want to use the ICMPV6 Type as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 29**
 - As a user, I want to use the ICMPV6 Code as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method

- ☐ Implement the match field's from_of_tlv method
- **User Story 30**
 - As a user, I want to use the IPV6 ND Target as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 31**
 - As a user, I want to use the IPV6 ND SLL as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 32**
 - As a user, I want to use the IPV6 ND TLL as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 33**
 - As a user, I want to use the IPV6 EXTHDR as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 34**
 - As a developer, I want to test the implementation for the first half of issue #77, so that I can make sure it is working for the users.
 - Acceptance criteria:
 - ☐ Test using the User Manual for each different match field
- **User Story 35**

- As a user, I want to use the Metadata as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 36**
 - As a user, I want to use the SCTP Source as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 37**
 - As a user, I want to use the SCTP Destination as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 38**
 - As a user, I want to use the ARP OP as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 39**
 - As a user, I want to use the ARP SPA as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 40**

- As a user, I want to use the ARP TPA as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 41**
 - As a user, I want to use the ARP SHA as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 42**
 - As a user, I want to use the ARP THA as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 43**
 - As a user, I want to use the MPLS Label as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 44**
 - As a user, I want to use the MPLS TC as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 45**

- As a user, I want to use the MPLS BOS as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 46**
 - As a user, I want to use the PBB ISID as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 47**
 - As a user, I want to use the Tunnel ID as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
 - Acceptance criteria:
 - ☐ Add the match field to the constructor of MatchBase
 - ☐ Implement the match field's as_of_tlv method
 - ☐ Implement the match field's from_of_tlv method
- **User Story 48**
 - As a developer, I want to test the implementation for the second half of issue #77, so that I can make sure it is working for the users.
 - Acceptance criteria:
 - ☐ Test using the User Manual for each different match field
- **User Story 49**
 - As a developer, I want to implement and test the IPv6 packet structure, so that the IPv6 packets can be used.
 - Acceptance criteria:
 - ☐ Implement the Pack functionality
 - ☐ Implement the Unpack functionality
 - ☐ Initialize the fields of the IPv6 headers
 - ☐ Create a unit test to test the IPv6 packet structure functionality
- **User Story 50**
 - As a user, I want to have a manual and a PowerPoint to read and understand the

implementations, so that I know what they are used for.

- Acceptance criteria:
 - ☐ Finish the manual and PowerPoint and place them in the BitBucket repository
- **User Story 51**
 - As a user, I want to watch a demo video about the match fields and the IPv6 packet implemented, so that I can use these implementations knowing their functionalities.
 - Acceptance criteria:
 - ☐ Finish all demo videos and place them in the Youtube account

Pending User Stories

There were no user stories that were incomplete by the end of this project.

PROJECT PLAN

In this section, figure 1 illustrates a Gantt chart that shows a summarized agenda of the meetings and tasks, along with their respective duration.

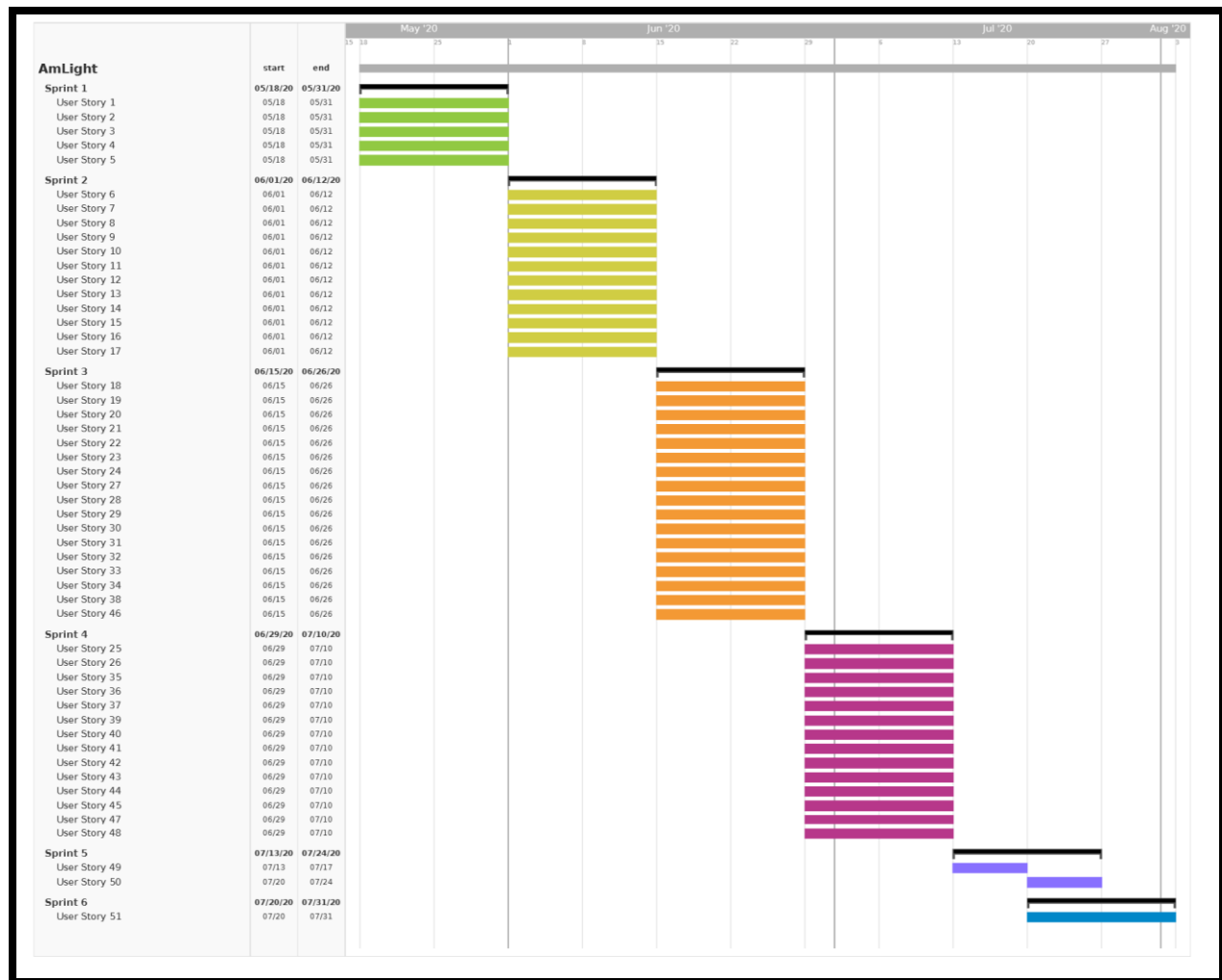


Figure 1- Gantt Chart

Hardware and Software Resources

The following is a list of of all hardware and software resources that were used for this project:

- **Kytos**, an SDN platform, to create and deploy network applications (NApps).
- **Mininet**, a network emulator, to send and receive topology information, such as hosts, switches, and the links among them.
- **Github**, a version control site, to maintain changes to our project.
- **Visual Studio IDE**, with a Python extension installed to develop python scripts (Pycharms or Visual Studio Code could also be used).
- **Oracle VM VirtualBox**, with an Ubuntu Virtual Machine iso installed (any Ubuntu version higher than version 16 will work, but we suggest the latest Ubuntu version available).
- **Trello**, a calendar board site, helped the team understand what was expected of them and by when (with Power-Ups in Trello you can add user story points and have a Gantt chart for all team members to see).

Sprints Plan

This section contains a summary of the work done for each sprint.

Sprint 1

During Sprint 1 we:

- Focused on getting familiar with Kytos and OpenFlow
- Forked the original Kytos project
- Worked on setting up the project goals and documents
- Investigated methods to masking the match fields

The product owner requested the following user stories to be completed this sprint. They are ordered by priority:

- **User Story 1**
 - As a developer, I want to set up the environment for Kytos, so that I can start testing and developing applications with Kytos.
- **User Story 2**
 - As a developer, I want to go over the Kytos tutorials, so that I have the knowledge for Kytos.
- **User Story 3**
 - As a developer, I want to read over the OpenFlow documentation given to us by the Product Owner, so that I can become familiar with the software we are working with.
- **User Story 4**
 - As a developer, I want to review the source code given to us by the Product Owner, so I can find where the issue is occurring and come up with a solution.
- **User Story 5**
 - As a developer, I want to start working on a solution for the issue #75 that was given to us by the Product Owner, so I may implement the solution on the software given to us.

Sprint 2

During Sprint 2 we:

- Worked on masking all the match fields that were in the current Kytos system
- Discovered how to test the match fields in a clearer way, so as to see that the mask was accepted by Kytos in the switch controller

- Started experimenting with masking different packet types (IP addresses versus bit masking)

The product owner requested the following user stories to be completed this sprint. They are ordered by priority:

- **User Story 6**
 - As a user, I want to be able to use the mask function for the Datalink VLAN ID matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 7**
 - As a user, I want to be able to use the mask function for the VLAN Priority Code Point matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 8**
 - As a user, I want to be able to use the mask function for the Datalink Source matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 9**
 - As a user, I want to be able to use the mask function for the Datalink Destination matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 10**
 - As a user, I want to be able to use the mask function for the Datalink Type matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 11**
 - As a user, I want to be able to use the mask function for the IPv4 Source matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 12**
 - As a user, I want to be able to use the mask function for the IPv4 Destination matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 13**
 - As a user, I want to be able to use the mask function for the IP Protocol matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.

- **User Story 14**
 - As a user, I want to be able to use the mask function for the Input Port matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 15**
 - As a user, I want to be able to use the mask function for the TCP Source matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 16**
 - As a user, I want to be able to use the mask function for the TCP Destination matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 17**
 - As a developer, I want to test the implementation for issue #75, so that I can make sure it is working for the users.

Sprint 3

During Sprint 3 we:

- Created match field functionality for packet types that were not implemented already into Kytos (managed to finish over half of all packet types that are reasonable to assume a user would use)
- Tested the match fields using the method we found in sprint 2
- Discovered that some packets were not implemented into the Kytos switch controller
- Experimented with testing the physical input port

The product owner requested the following user stories to be completed this sprint. They are ordered by priority:

- **User Story 18**
 - As a user, I want to use the physical input port as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 19**
 - As a user, I want to use the IP DSCP as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 20**
 - As a user, I want to use the IP ECN as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.

- **User Story 21**
 - As a user, I want to use the UDP Source as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 22**
 - As a user, I want to use the UDP Destination as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 23**
 - As a user, I want to use the ICMPV4 Type as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 24**
 - As a user, I want to use the ICMPV4 Code as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 27**
 - As a user, I want to use the IPV6 FLabel as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 28**
 - As a user, I want to use the ICMPV6 Type as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 29**
 - As a user, I want to use the ICMPV6 Code as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 30**
 - As a user, I want to use the IPV6 ND Target as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 31**
 - As a user, I want to use the IPV6 ND SLL as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 32**
 - As a user, I want to use the IPV6 ND TLL as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 33**
 - As a user, I want to use the IPV6 EXTHDR as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 34**
 - As a developer, I want to test the implementation for the first half of issue #77, so that I can make sure it is working for the users.
- **User Story 38**

- As a user, I want to use the ARP OP as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 46**
 - As a user, I want to use the PBB ISID as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.

Sprint 4

During Sprint 4 we:

- Created match field functionality for packet types that were not implemented already into Kytos (managed to finish all packet types that are reasonable to assume a user would use)
- Tested the match fields using the method we found in sprint 2
- Discovered that some packets were not implemented into the Kytos switch controller
- Experimented with testing the IPv6 packets

The product owner requested the following user stories to be completed this sprint. They are ordered by priority:

- **User Story 25**
 - As a user, I want to use the IPV6 Source as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 26**
 - As a user, I want to use the IPV6 Destination as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 35**
 - As a user, I want to use the Metadata as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 36**
 - As a user, I want to use the SCTP Source as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 37**
 - As a user, I want to use the SCTP Destination as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 38**
 - As a user, I want to use the ARP OP as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 39**
 - As a user, I want to use the ARP SPA as one of the match fields for my flow, so

that I can utilize the OpenFlow v1.3 more realistically.

- **User Story 40**
 - As a user, I want to use the ARP TPA as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 41**
 - As a user, I want to use the ARP SHA as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 42**
 - As a user, I want to use the ARP THA as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 43**
 - As a user, I want to use the MPLS Label as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 44**
 - As a user, I want to use the MPLS TC as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 45**
 - As a user, I want to use the MPLS BOS as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 47**
 - As a user, I want to use the Tunnel ID as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- **User Story 48**
 - As a developer, I want to test the implementation for the second half of issue #77, so that I can make sure it is working for the users.

Sprint 5

During Sprint 5 we:

- Discovered how to implement the IPv6 packet structure into the Kytos switch controller
- Discovered how to create a unit test to test our IPv6 packet structure
- Began more extensive documentation and creating a presentation of our final product

The product owner requested the following user stories to be completed this sprint. They are ordered by priority:

- **User Story 49**
 - As a developer, I want to implement and test the IPv6 packet structure, so that the

IPv6 packets can be used.

- **User Story 50**

- As a user, I want to have a manual and a PowerPoint to read and understand the implementations, so that I know what they are used for.

Sprint 6

During Sprint 6 we:

- Finalized all documentations and presentations
- Created demonstrational and informational videos on the final project

The product owner requested the following user stories to be completed this sprint. They are ordered by priority:

- **User Story 51**

- As a user, I want to watch a demo video about the match fields and the IPv6 packet implemented, so that I can use these implementations knowing their functionalities.

SYSTEM DESIGN

This section goes over the design of the system.

Architectural Patterns

Kytos of_core uses a Client Server architecture. The client being the web UI, or anything else that may access the system using the REST API. The server is the of_core module hosted by Kytos. This can be pictured in figure 2.

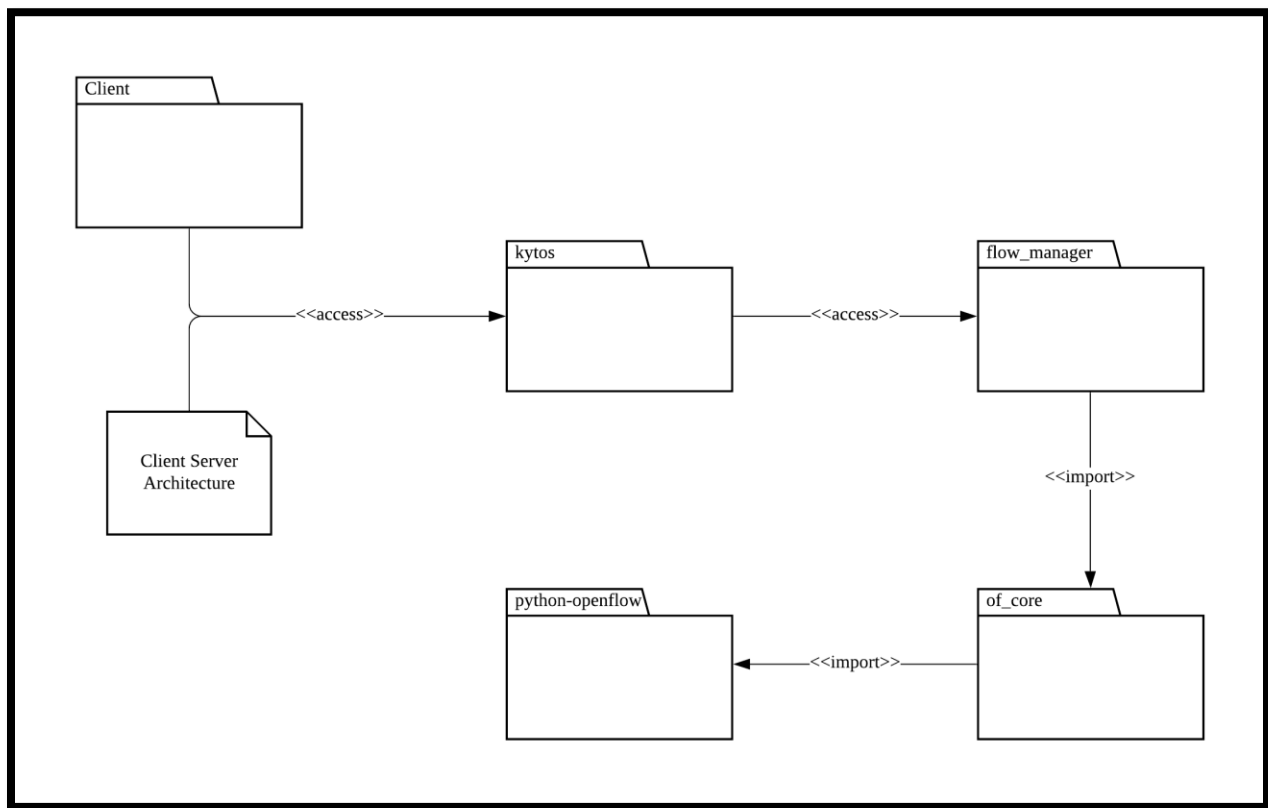


Figure 2- Client Server Architecture of Kytos

System and Subsystem Decomposition

The two major subsystems of the system are Kytos of_core, which is the core logic, and the Web UI, which provides an interface for regular users to interact with the business logic.

Deployment Diagram

The system is deployed as a client web UI, with Kytos being hosted on a server that has all of the Kytos NApps, including Kytos of_core. This can be seen in figure 3.

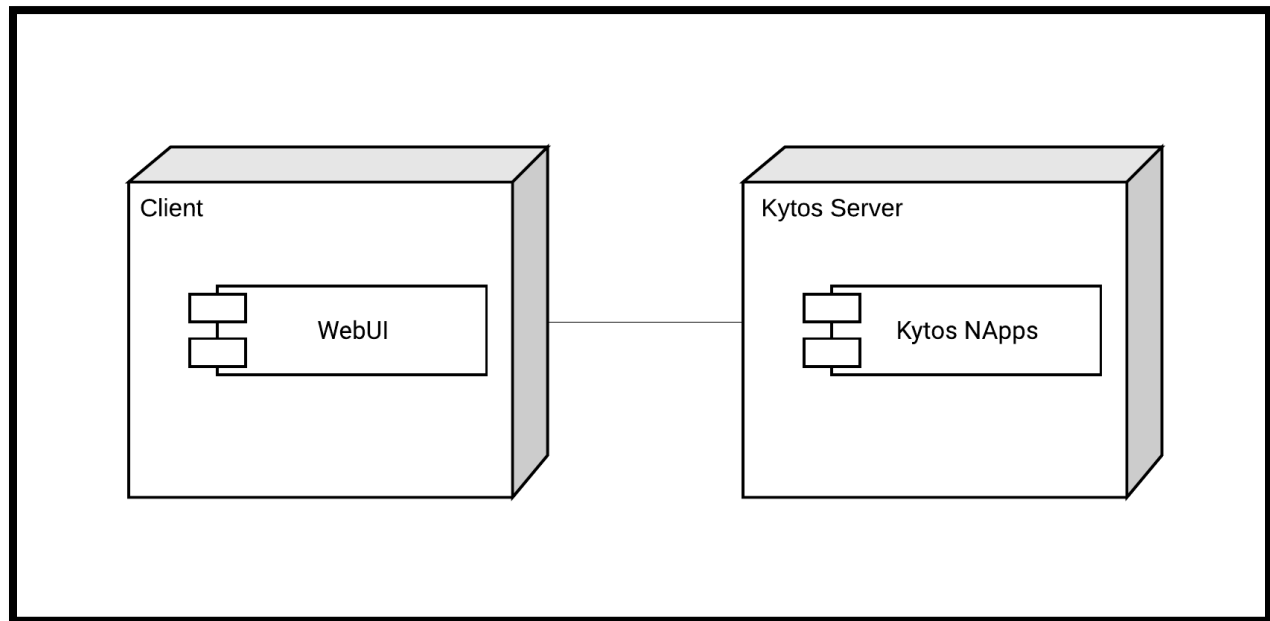


Figure 3- Deployment Diagram of Kytos

Design Patterns

For our system we applied the strategy, facade, and singleton patterns. The main class implements the singleton and facade patterns. As a facade, it exposes the functionality of the system as REST Endpoints, and provides event listeners for Kytos. It is a singleton, because it hosts the logic of the system. The filter class implements the strategy pattern. At run time the algorithm used by the filter is defined. Strategy is used by the Filter class, where we can define a filter function at runtime. Facade is used by the Main class. It exposes the functionality of the system as a set of REST endpoints.

SYSTEM VALIDATION

In this section, the Match Field test cases and the IPv6 Packet unit test results are presented. These tests focus on the TestCase and flows class.

Match Fields Test Cases

This section presents the test cases done for the match field class in detail, such as its purpose, setup, input, and expected output. In addition, it presents the result given, namely whether the actual result is a pass or a fail.

- Purpose
 - To test the match fields function in the match_field class
- Setup
 - Kytos development environment is activated
 - Currently in the flow_manager directory
- Input
 - A flow is needed as a parameter. The priority, idle_timeout, hard_timeout, cookies, match, and actions fields must stay the same as denoted in figure 4. The name of the match field, with a source and destination if needed, and any prerequisites must be the only thing changed in the flow field
- Expected Output
 - FlowMod message sent
- Status
 - Pass

```

kenia@kali:~$ sudo mn --controller=remote
[sudo] password for kenia:
*** Creating network
*** Adding controller
kenia@kali:~$ curl --location --request POST 'http://localhost:8181/api/kytos/flow_manager/v2/flows/00:00:00:00:00:00:01' --header 'Content-Type: application/json' --data-raw '{
  "flows": [
    {
      "priority": 10,
      "idle_timeout": 360,
      "hard_timeout": 1200,
      "cookie": 13057375916202815382,
      "match": {
        "in_port": 2,
        "dl_type": 2048,
        "nw_src": "10.0.3.0/24"
      },
      "actions": [
        {
          "action_type": "output",
          "port": 1
        }
      ]
    }
  ]
}'
kenia@kali:~$
  
```

Figure 4- Match Field Test Case

IPv6 Packet Unit Test

This section presents the unit test done for the IPv6 packet in detail, such as its purpose, setup, input, and expected output. In addition, it presents the result given by Python's unittest unit testing framework, namely whether the actual result is a pass or a fail.

unittest_TestIPv6

- Purpose
 - To test the IPv6 pack and unpack function in the IPv6 class
- Setup
 - Kytos development environment is activated
 - Currently in the python-openflow directory
 - Root user or using the sudo command
- Input
 - No input needed, but if change is needed, the change must be done to the data field in the IPv6 constructor as shown in figure 5
- Expected Output
 - test_IPv6_pack ... ok
 - test_IPv6_unpack ... ok
- Status
 - Pass

```
class TestIPv6(unittest.TestCase):
    """Test IPv6 packets."""

    def test_IPv6_pack(self):
        """Test pack/unpack of IPv6 class."""
        packet = IPv6(next_header=6, hop_limit=64, source="::1",
                      destination="::2", data=b'testdata')
        packed = packet.pack()
        expected = b'\x00\x00\x00\x00\x00\x00\x06@ \x00\x00'
        expected += b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'
        expected += b'\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'
        expected += b'\x00\x00\x00\x02testdata'
        self.assertEqual(packed, expected)

    def test_IPv6_unpack(self):
        """Test unpack of IPv6 binary packet."""
        raw = b'\x00\x00\x00\x00\x00\x00\x0c \x00\x00'
        raw += b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'
        raw += b'\x02\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'
        raw += b'\x00\x00\x00\x01somedata'
        expected = IPv6(next_header=6, hop_limit=64, source="::2",
                        destination="::1", data=b'somedata')
        expected.pack()
        unpacked = IPv6()
        unpacked.unpack(raw)
        self.assertEqual(unpacked, expected)
```

Figure 5- IPv6 Packet Unit Test

GLOSSARY

The following are terms used throughout the final project documentation.

Kytos A software designed network controller that uses OpenFlow protocols

Mask A special value that acts as a data filter, while revealing some parts of digital information, and concealing other parts of the digital information

Match Field A part of data from a packet header that is “matched” or compared to other parts of data from a sender

Network Multiple devices that communicate with one another usually through communication protocols

Network Application (NApp) Optional components for Kytos, which are used to extend functionality for Kytos

OpenFlow Communication protocol for accessing and modifying the forwarding plane of a switch or router

Packet A unit of data flowing from from a sender to a receiver within the switch network

Software Defined Network (SDN) Network where routing is defined programmatically

APPENDIX

Appendix A - UML Diagrams

This section contains the UML diagrams generated for this system.

Use Case Diagrams

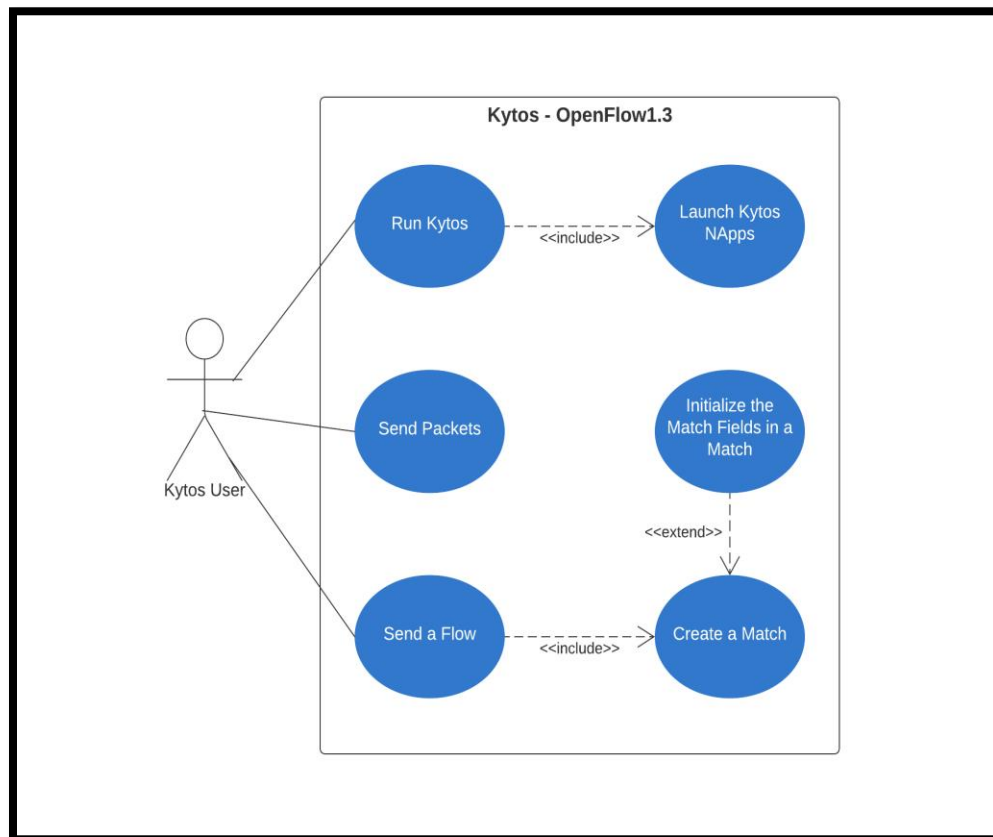


Figure 6- Match Fields Use Case Diagram by Kenia Chang He

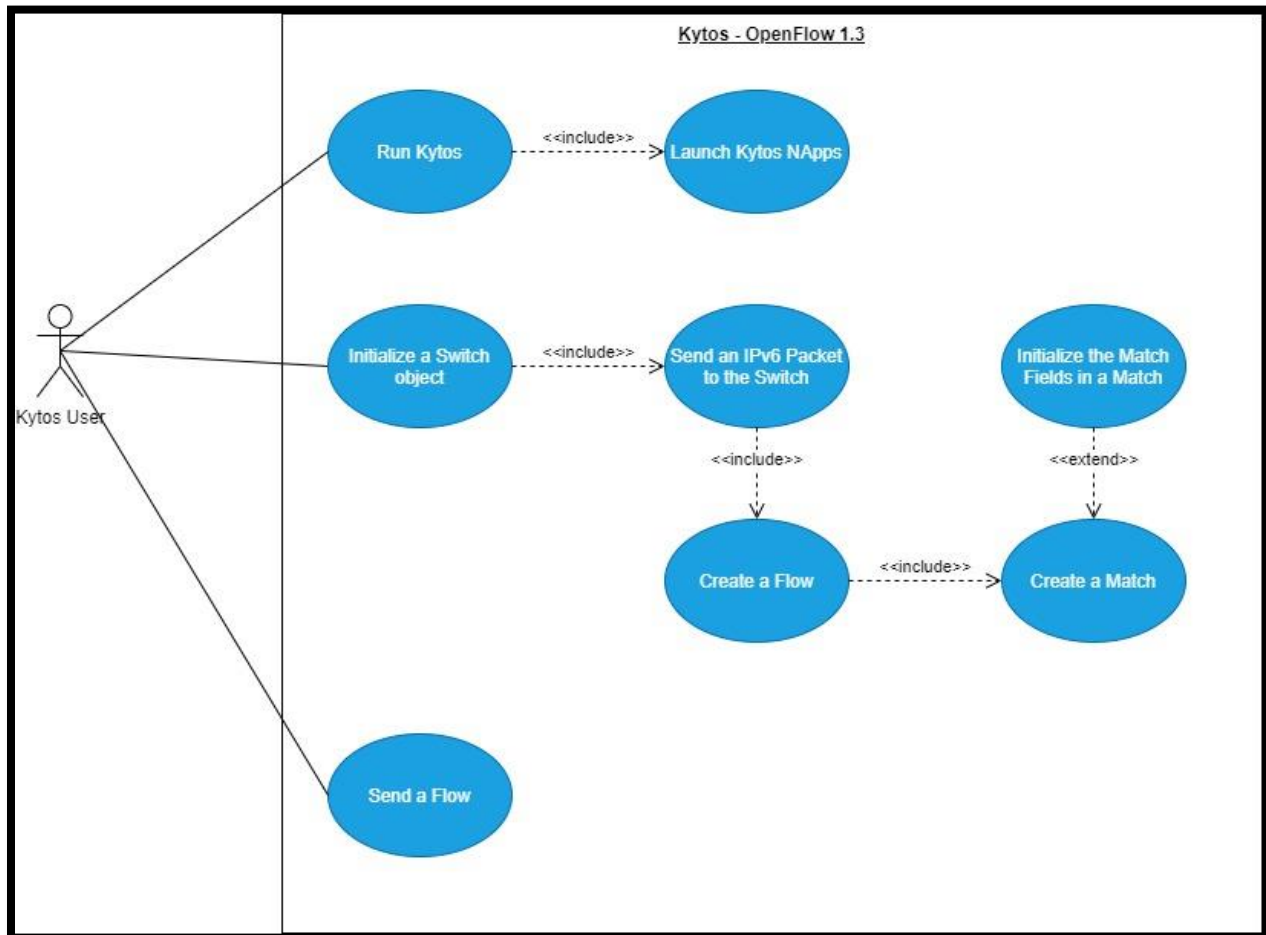


Figure 7- IPv6 Packet Use Case Diagram by Cindy Perez

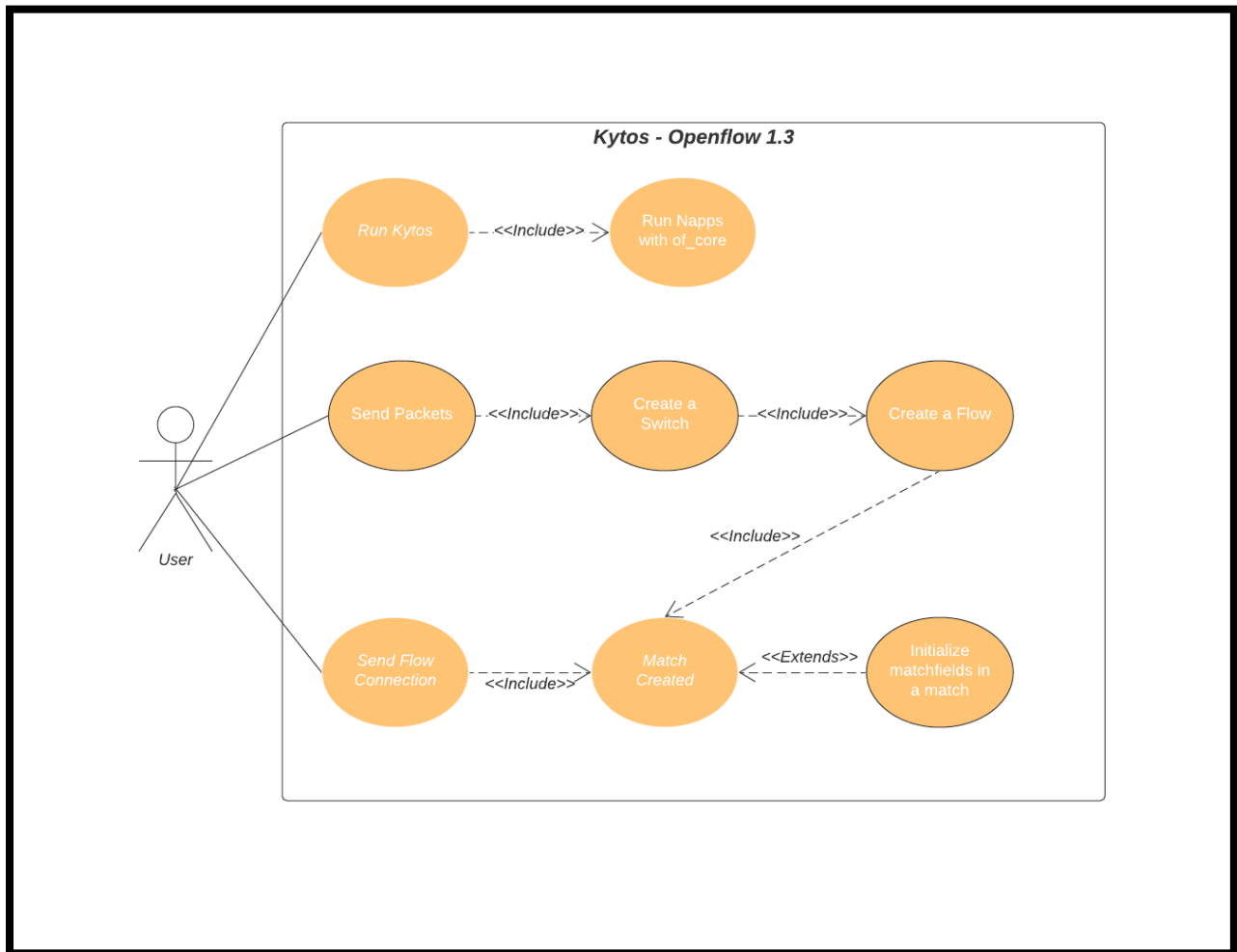


Figure 8- IPv4 Packet Use Case Diagram by Jose Mercardo

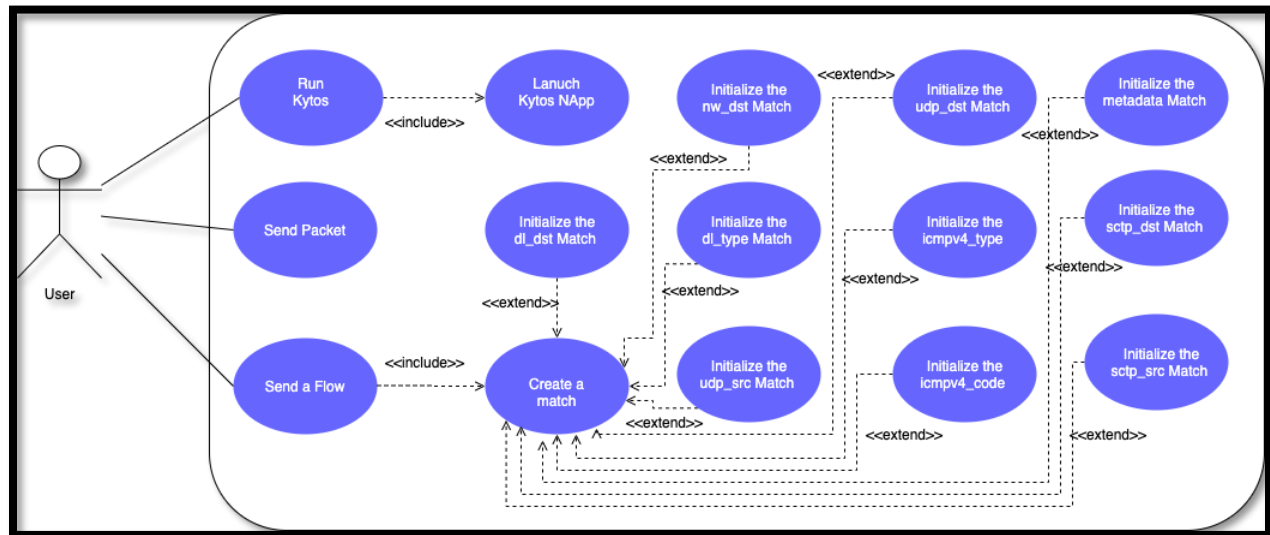


Figure 9- Match Fields Use Case Diagram by Yuyang Leng

Class Diagrams

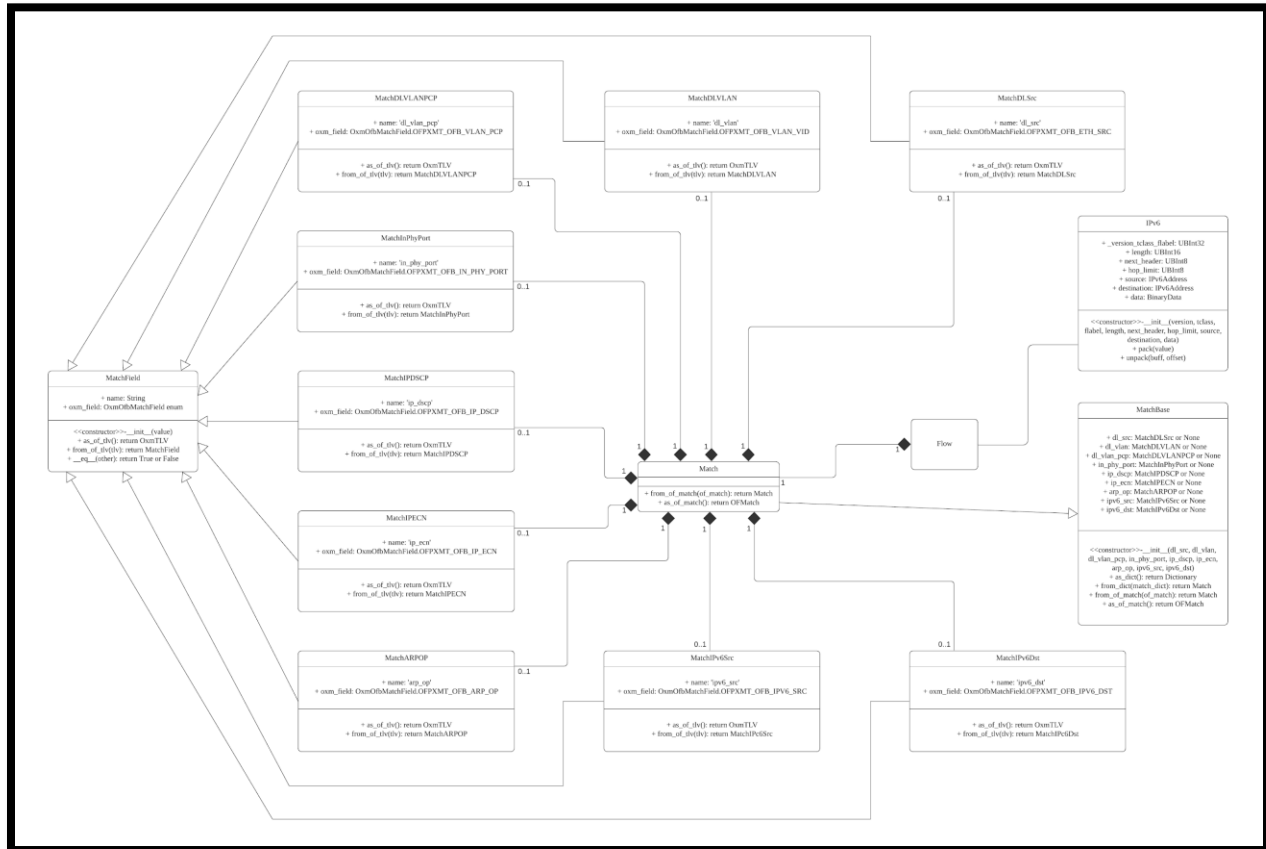


Figure 10- Class Diagram by Kenia Chang He

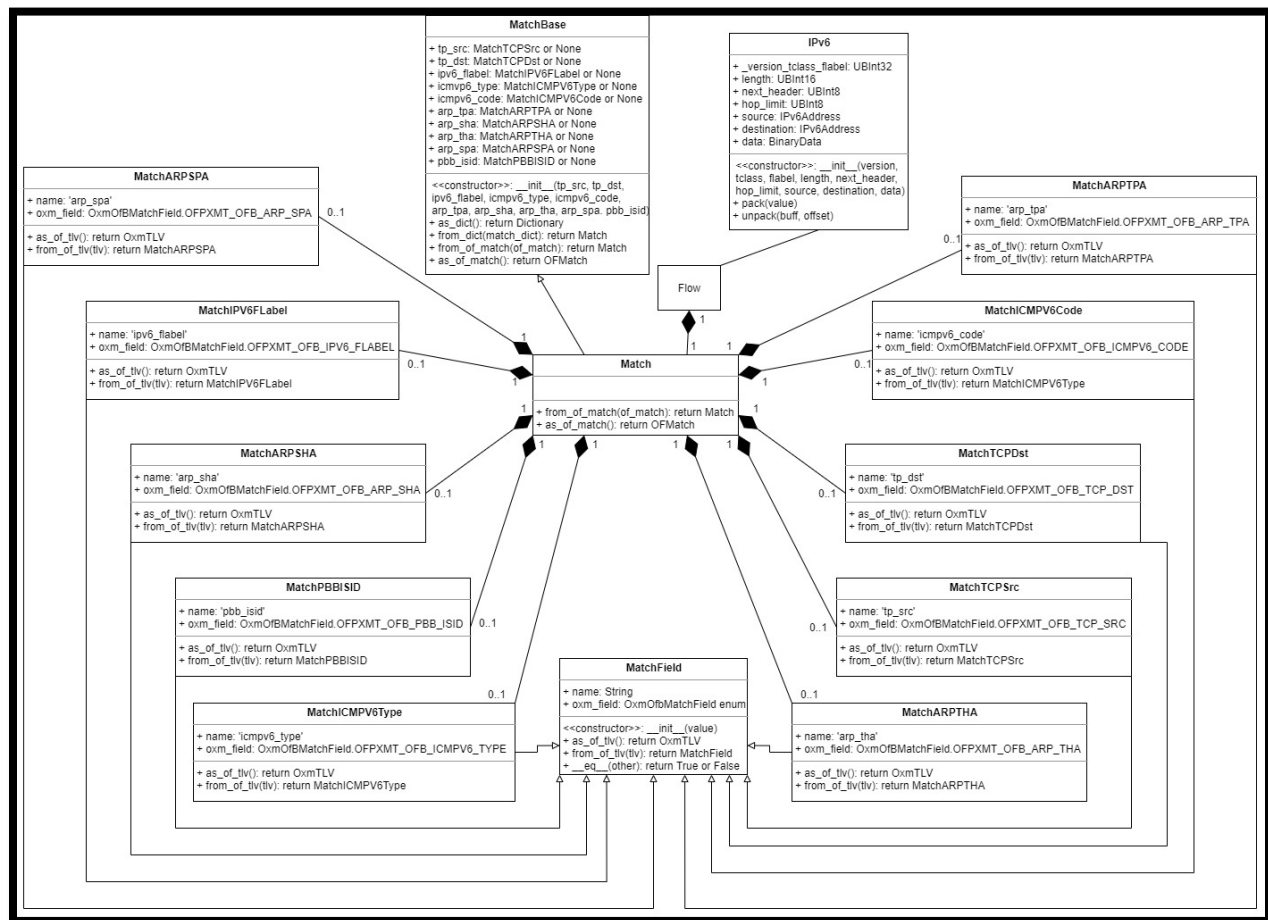


Figure 11- Class Diagram by Cindy Perez

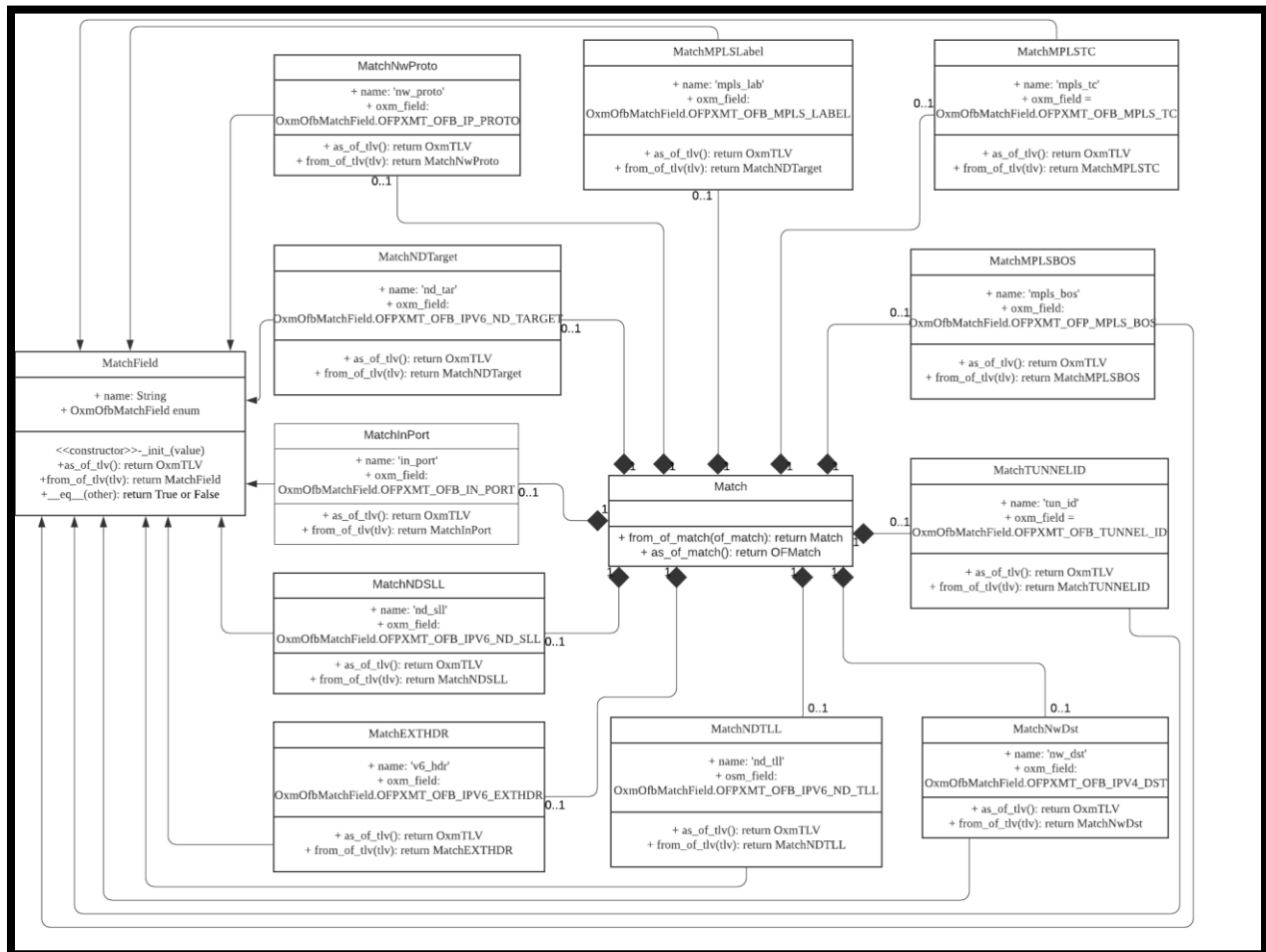


Figure 12- Class Diagram by Jose Mercardo

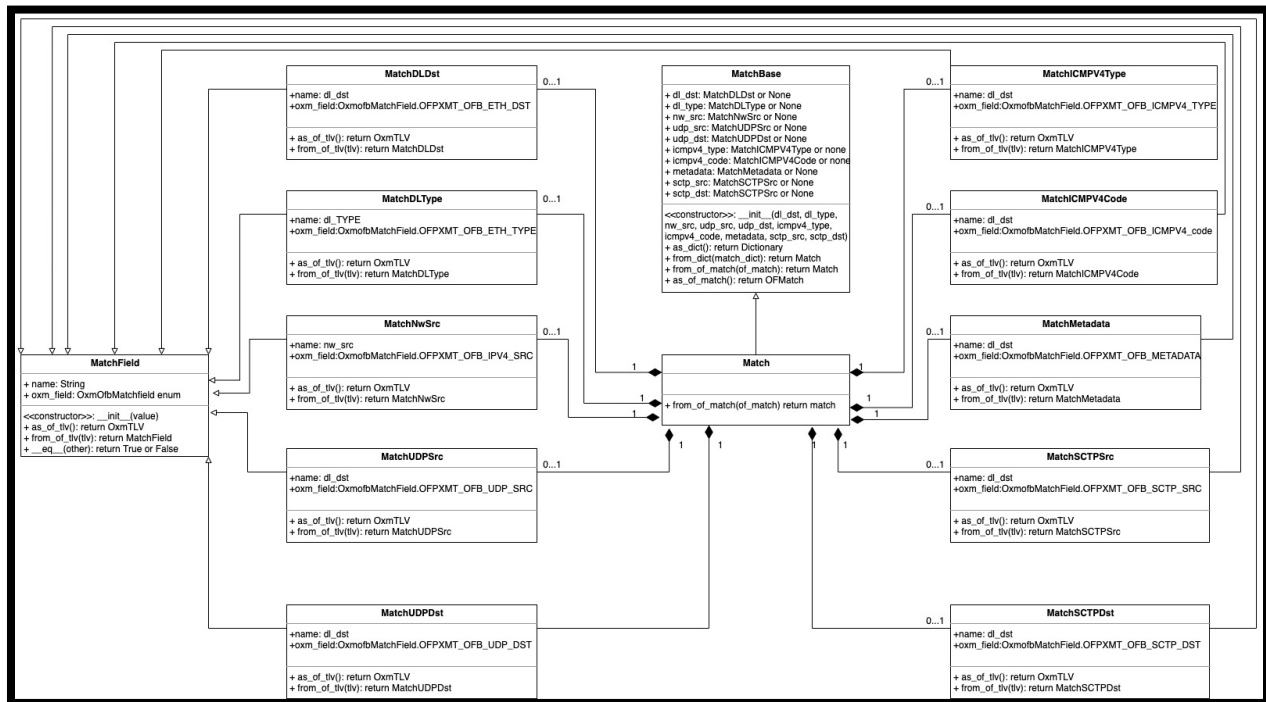


Figure 13- Class Diagram by Yuyang Leng

Sequence Diagrams

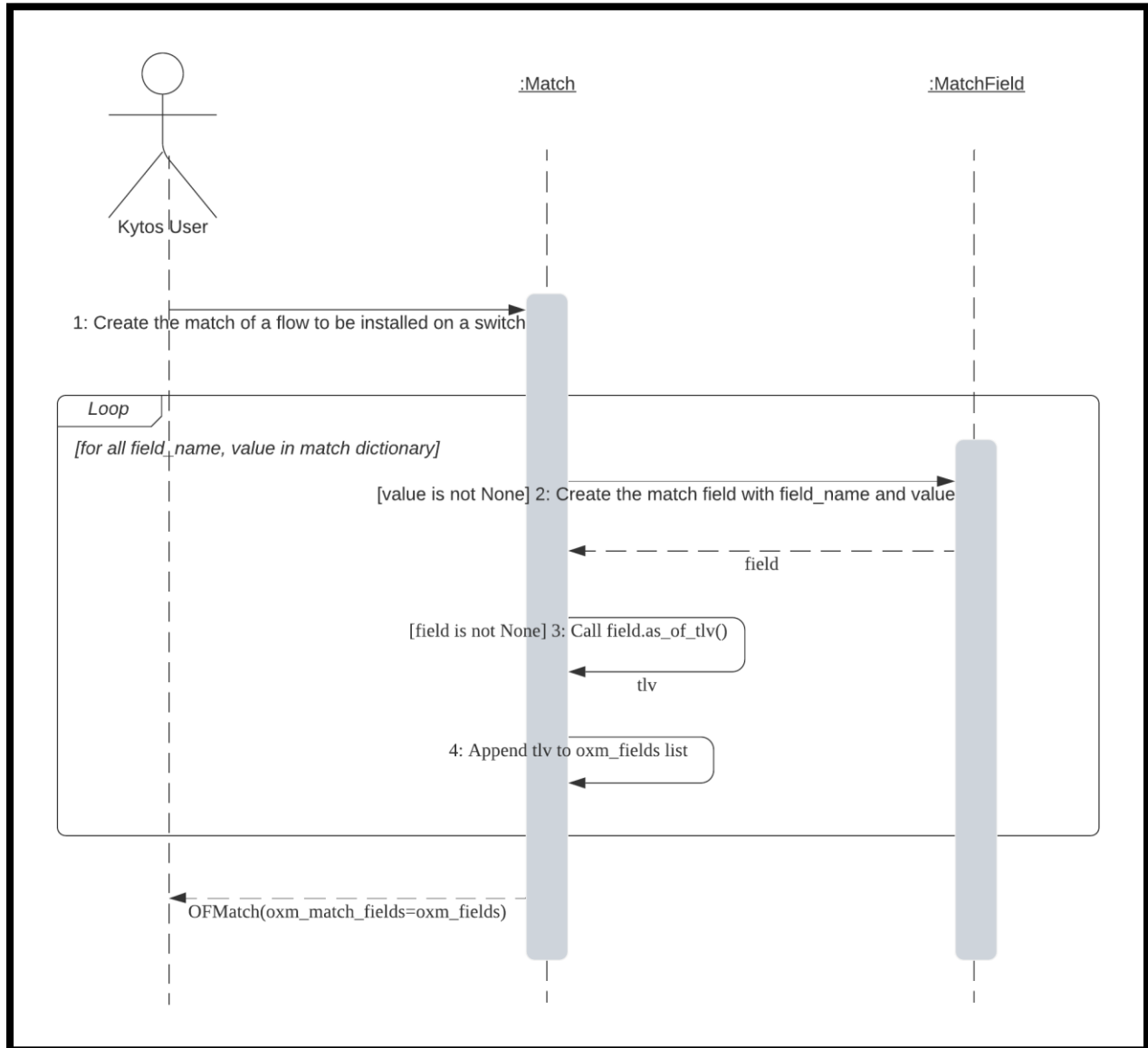


Figure 14- Match Fields Sequence Diagram by Kenia Chang He

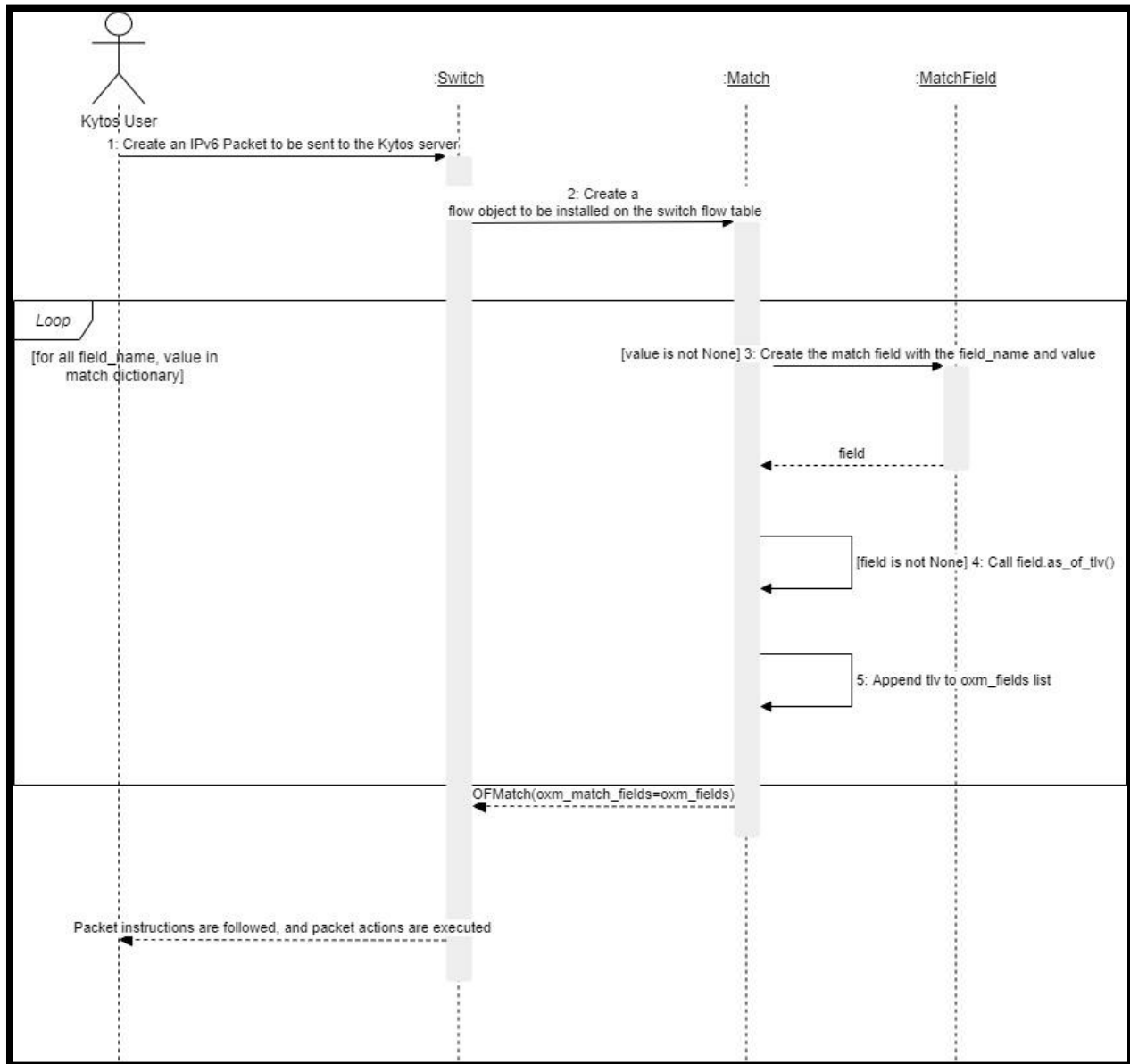


Figure 15- Ipv6 Packet Sequence Diagram by Cindy Perez

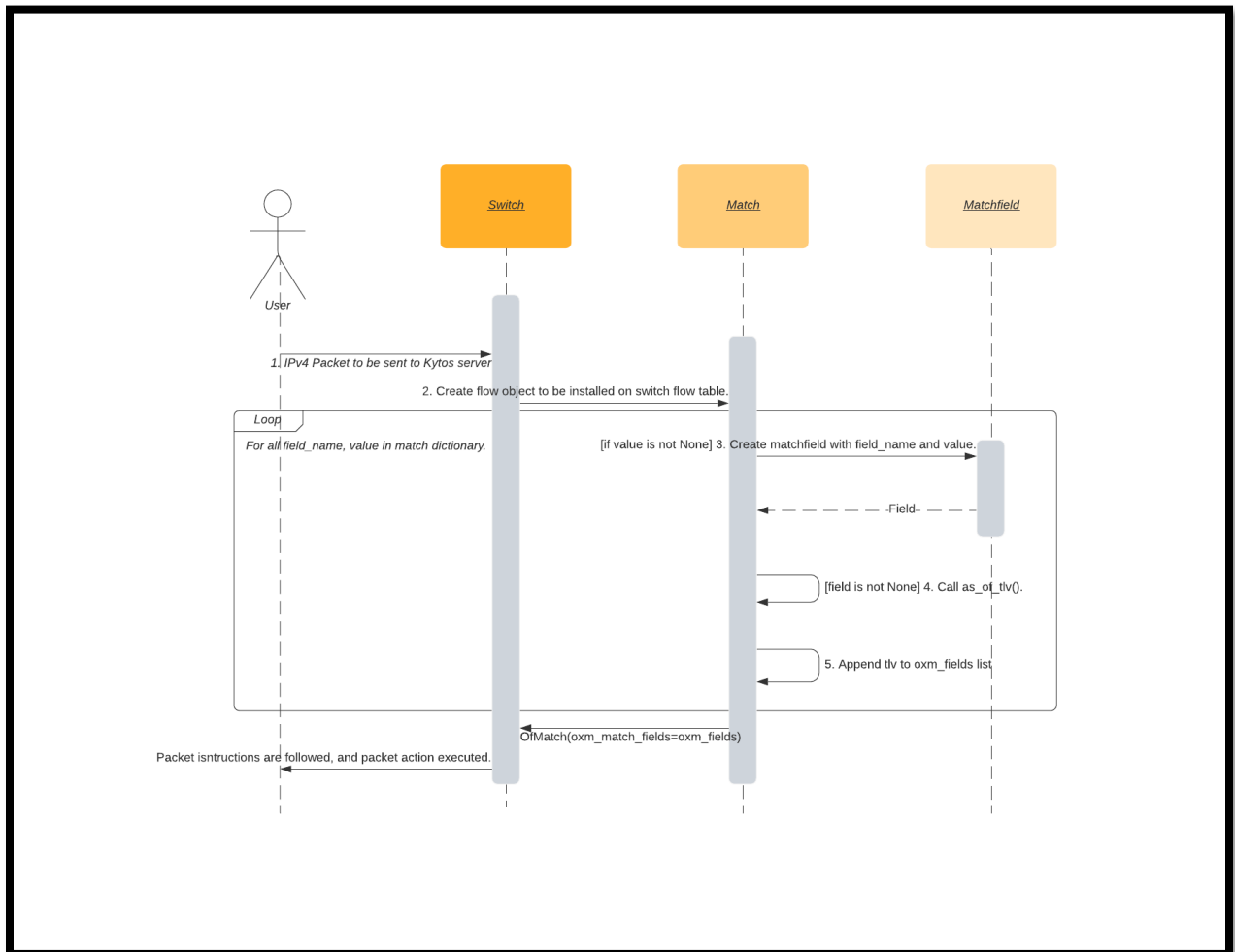


Figure 16- IPv4 Packet Sequence Diagram by Jose Mercardo

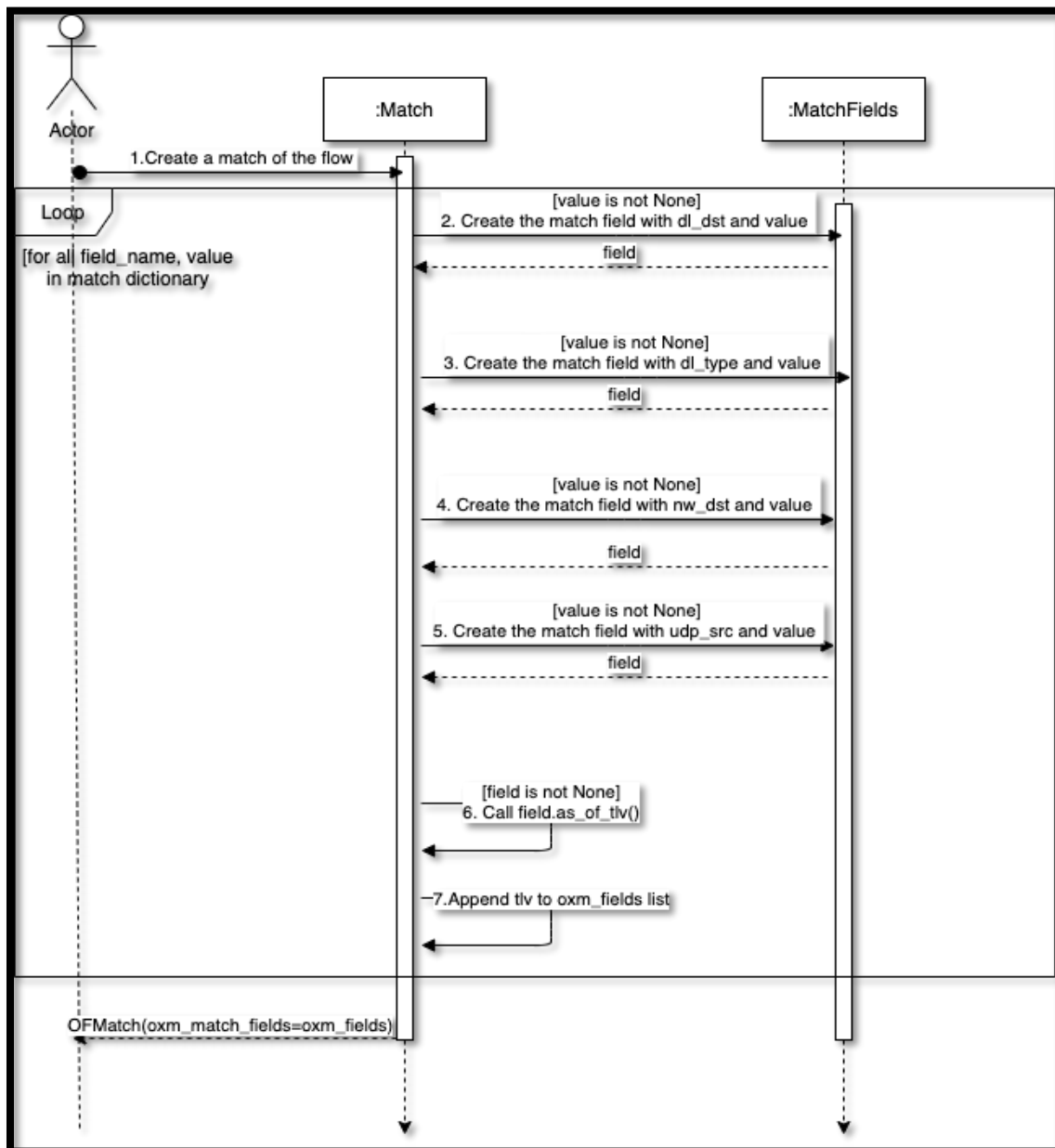


Figure 17- Match Fields Sequence Diagram by Yuyang Leng

Appendix B - User Interface Design

This section contains reference images for the user interface.

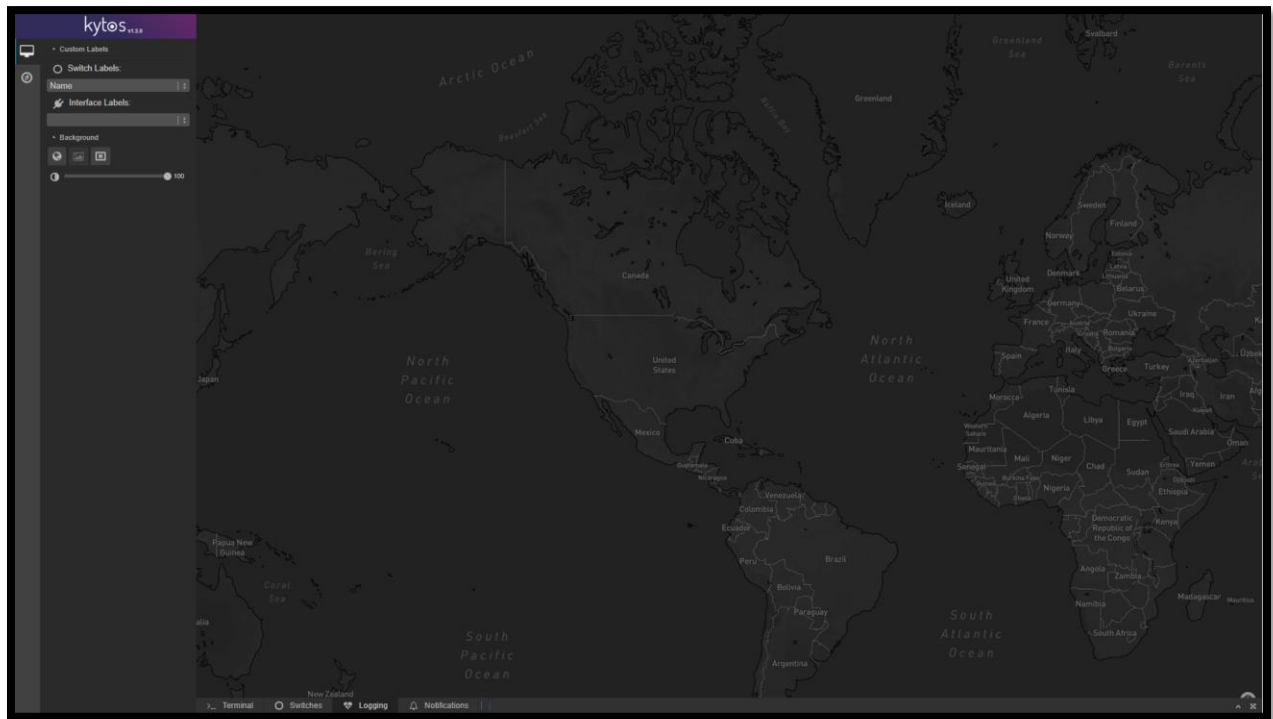


Figure 18- User Interface for the Kytos Website

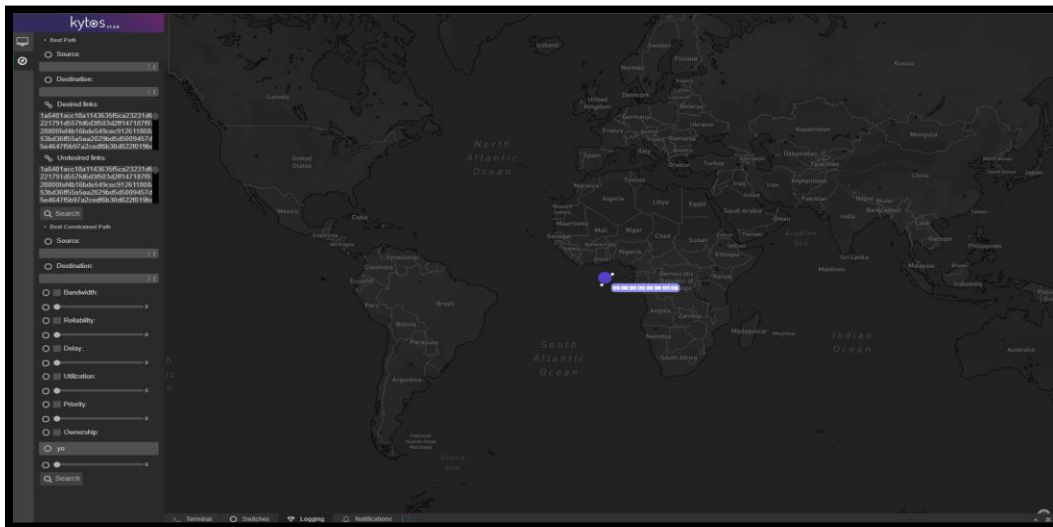


Figure 19- Kytos Website with User Input

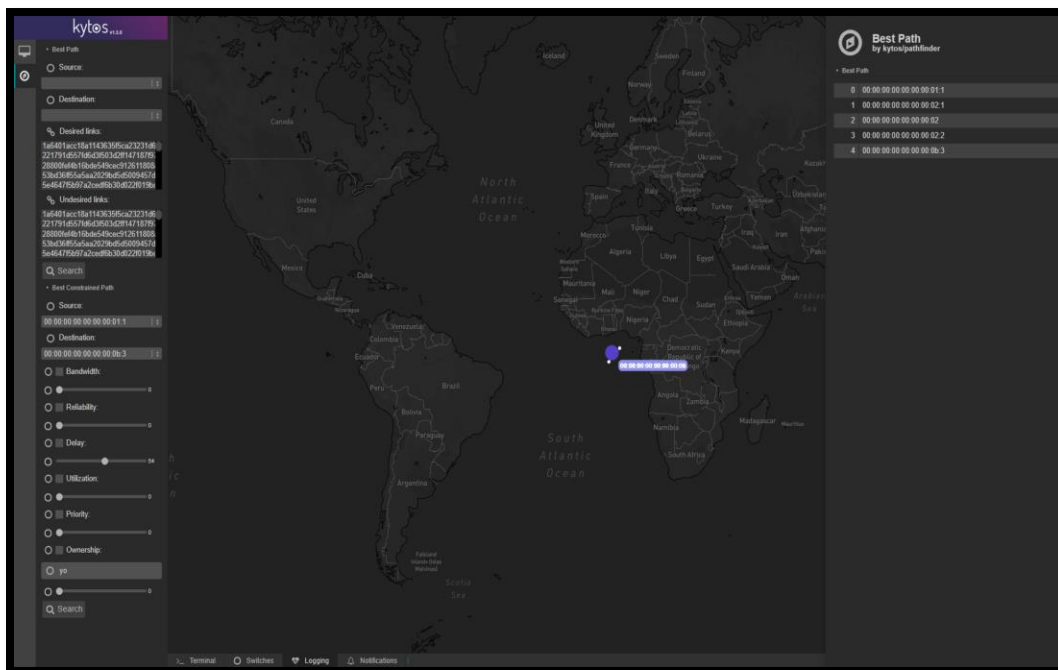


Figure 20- Kytos Website with Output

Appendix C - Sprint Review Reports

This section contains the sprints reviews and retrospectives.

Sprint 1

Review Meeting Minutes:

Date: 5/31/20

Attendees: Cindy Perez, Kenia Chang He, Jose Mercardo, and Yuyang Leng.

Start Time: 3:00PM

End Time: 3:30PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- User Story 1: As a developer, I want to set up the environment for Kytos, so that I can start testing and developing applications with Kytos.
- User Story 2: As a developer, I want to go over the Kytos tutorials, so that I have the knowledge for Kytos.
- User Story 3: As a developer, I want to read over the OpenFlow documentation given to us by the Product Owner, so that I can become familiar with the software we are working with.
- User Story 4: As a developer, I want to review the source code given to us by the Product Owner, so I can find where the issue is occurring and come up with a solution.
- User Story 5: As a developer, I want to start working on a solution for the issue #75 that was given to us by the Product Owner, so I may implement the solution on the software given to us.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting:

- None.

Retrospective Meeting Minutes:

Date: 5/31/20

Attendees: Cindy Perez, Kenia Chang He, Jose Mercardo, and Yuyang Leng.

Start Time: 3:30PM

End Time: 3:41PM

What went wrong?

- Did we do a good job estimating our team's velocity?
 - Yes, our team met the velocity that we estimated at the Planning Meeting for this sprint.
- Did we do a good job estimating the points (time required) for each user story?
 - No, mostly because we got our user story way late into the sprint.
- Did each team member work as scheduled?
 - Yes, each team member did the work that they needed to do, and did not fall behind too much.

What went right?

- The progress and everyone working on the tutorials was ahead of schedule a lot of the time.
- Good communication between the team and everyone knew what the others were doing and what the other team members were working on.

How to address the issues in the next sprint?

- How to improve the process?
 - Improve product owner communication with the team.
 - Scheduling the meetings at better times during the sprint.
- How to improve the product?
 - Implementing our pseudocode for issue #75.
 - Asking the product owner for more issues ahead of time to convert into more user stories.

Sprint 2

Review Meeting Minutes:

Date: 6/12/20

Attendees: Cindy Perez, Kenia Chang He, Jose Mercardo, Yuyang Leng, Antonia (Kytos developer), Rogerio (Kytos developer), and Vasilika Chergarova.

Start Time: 3:00PM

End Time: 3:36PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- User Story 6: As a user, I want to be able to use the mask function for the Datalink VLAN ID matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 7: As a user, I want to be able to use the mask function for the VLAN Priority Code Point matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 8: As a user, I want to be able to use the mask function for the Datalink Source matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 9: As a user, I want to be able to use the mask function for the Datalink Destination matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 10: As a user, I want to be able to use the mask function for the Datalink Type matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 11: As a user, I want to be able to use the mask function for the IPv4 Source matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 12: As a user, I want to be able to use the mask function for the IPv4 Destination matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 13: As a user, I want to be able to use the mask function for the IP Protocol matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 14: As a user, I want to be able to use the mask function for the Input Port matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 15: As a user, I want to be able to use the mask function for the TCP Source matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 16: As a user, I want to be able to use the mask function for the TCP Destination matching field of a flow as of OpenFlow v1.3, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 17: As a developer, I want to test the implementation for issue #75, so that I can make sure it is working for the users.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting:

- None.

Retrospective Meeting Minutes:

Date: 6/12/20

Attendees: Cindy Perez, Kenia Chang He, Jose Mercardo, and Yuyang Leng.

Start Time: 8:40PM

End Time: 8:52PM

What went wrong?

- Did we do a good job estimating our team's velocity?
 - Yes we did.
- Did we do a good job estimating the points (time required) for each user story?
 - Yes, we were able to complete all the user stories for this sprint in a timely manner.
- Did each team member work as scheduled?
 - Yes, each team member worked as scheduled.

What went right?

- Better communication with the Product Owner and the team.
- The team communicated a lot with one another, and were constantly helping each other.
- Communication with the Kytos developer went smoother due to the team having access to the Slack group channel.

How to address the issues in the next sprint?

- How to improve the process?
 - Follow tutorials for testing and read the openflow documentation that were given by the Kytos development team.
- How to improve the product?
 - Next sprint we will be working on fixing issue #77 for the Kytos project.

Sprint 3

Review Meeting Minutes:

Date: 6/26/20

07-21-2020

Page 49 of 67

Attendees: Cindy Perez, Kenia Chang He, Jose Mercardo, and Yuyang Leng.

Start Time: 3:00PM

End Time: 3:36PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- User Story 19: As a user, I want to use the IP DSCP as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 20: As a user, I want to use the IP ECN as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 21: As a user, I want to use the UDP Source as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 22: As a user, I want to use the UDP Destination as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 23: As a user, I want to use the ICMPV4 Type as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 24: As a user, I want to use the ICMPV4 Code as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 27: As a user, I want to use the IPV6 FLabel as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 28: As a user, I want to use the ICMPV6 Type as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 29: As a user, I want to use the ICMPV6 Code as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 30: As a user, I want to use the IPV6 ND Target as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 31: As a user, I want to use the IPV6 ND SLL as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 32: As a user, I want to use the IPV6 ND TLL as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 34: As a developer, I want to test the implementation for the first half of issue #77, so that I can make sure it is working for the users.
- User Story 38: As a user, I want to use the ARP OP as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting:

- User Story 18: As a user, I want to use the physical input port as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- Reason of rejection: Bad Field Error occurs when testing.
- User Story 33: As a user, I want to use the IPV6 EXTHDR as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- Reason of rejection: Bad Field Error occurs when testing.
- User Story 46: As a user, I want to use the PBB ISID as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- Reason of rejection: Bad Field Error occurs when testing.

Retrospective Meeting Minutes:

Date: 6/26/20

Attendees: Cindy Perez, Kenia Chang He, Jose Mercardo, and Yuyang Leng.

Start Time: 8:43PM

End Time: 8:48PM

What went wrong?

- Did we do a good job estimating our team's velocity?
 - Yes, we worked the amount of hours we estimated.
- Did we do a good job estimating the points (time required) for each user story?
 - No, some of the match fields are giving us an unexpected Bad Field error that we did not estimate into our user stories; these user stories were: implementing the physical input port, PBB ISID, and IPv6 extension header match fields.
- Did each team member work as scheduled?
 - Each team member completed their user stories as scheduled.

What went right?

- Team communication was amazing this sprint.
- Communication with the Kytos developers was also good.
- Rate of user story completion also went ahead of schedule at times.

How to address the issues in the next sprint?

- How to improve the process?
 - Have different branches of our main repository, so we can make pull requests to the main Kytos repository.
- How to improve the product?
 - Developing the next set of match field functions.

Sprint 4

Review Meeting Minutes:

Date: 7/10/20

Attendees: Cindy Perez, Kenia Chang He, Jose Mercardo, and Yuyang Leng.

Start Time: 3:00PM

End Time: 3:31PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- User Story 25: As a user, I want to use the IPV6 Source as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 26: As a user, I want to use the IPV6 Destination as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 35: As a user, I want to use the Metadata as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 36: As a user, I want to use the SCTP Source as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 37: As a user, I want to use the SCTP Destination as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 39: As a user, I want to use the ARP SPA as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 40: As a user, I want to use the ARP TPA as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 41: As a user, I want to use the ARP SHA as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 42: As a user, I want to use the ARP THA as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 43: As a user, I want to use the MPLS Label as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.

- User Story 44: As a user, I want to use the MPLS TC as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 45: As a user, I want to use the MPLS BOS as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 47: As a user, I want to use the Tunnel ID as one of the match fields for my flow, so that I can utilize the OpenFlow v1.3 more realistically.
- User Story 48: As a developer, I want to test the implementation for the second half of issue #77, so that I can make sure it is working for the users.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting:

- None.

Retrospective Meeting Minutes:

Date: 7/10/20

Attendees: Cindy Perez, Kenia Chang He, Jose Mercardo, and Yuyang Leng.

Start Time: 8:36PM

End Time: 8:41PM

What went wrong?

- Did we do a good job estimating our team's velocity?
 - Yes, we did a good job estimating our velocity.
- Did we do a good job estimating the points (time required) for each user story?
 - Yes, each user story was well estimated.
- Did each team member work as scheduled?
 - Yes, each team member worked as scheduled.

What went right?

- Great team communication between team members.
- Great communication between team members and the Kytos developers.
- Time management was amazing this sprint, and many more tests could be done to double check team members' individual work.

How to address the issues in the next sprint?

- How to improve the process?
 - Consolidate all of the team's code into one Github repository for the pull request

according to the Kytos development standards.

- How to improve the product?
 - Work on implementing the IPv6 packet structure.

Sprint 5

Review Meeting Minutes:

Date: 7/24/20

Attendees: Cindy Perez, Kenia Chang He, Jose Mercardo, and Yuyang Leng.

Start Time: 3:00PM

End Time: 3:32PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- User Story 49: As a developer, I want to implement the IPv6 packet structure, so that the IPv6 packets can be used.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting:

- None.

Retrospective Meeting Minutes:

Date: 7/24/20

Attendees: Cindy Perez, Kenia Chang He, Jose Mercardo, and Yuyang Leng.

Start Time: 8:39PM

End Time: 8:43PM

What went wrong?

- Did we do a good job estimating our team's velocity?
 - Yes, the team followed the velocity very well.
- Did we do a good job estimating the points (time required) for each user story?
 - Yes, the user stories were done on time.
- Did each team member work as scheduled?
 - Every team member worked way ahead of schedule.

What went right?

- Team communication and speed was very good; the team also finished work ahead of schedule.
- Putting up the final deliverables on google docs so that all the team members could come and change things as needed was a good idea.

How to address the issues in the next sprint?

- How to improve the process?
 - Next sprint is dedicated to finishing the final deliverables.
- How to improve the product?
 - Next sprint is dedicated to finishing the final deliverables.

Sprint 6

Review Meeting Minutes:

Date: 7/30/20

Attendees: Cindy Perez, Kenia Chang He, Jose Mercardo, and Yuyang Leng.

Start Time: 8:44PM

End Time: 8:48PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- User Story 51: As a user, I want to watch a demo video about the match fields and the IPv6 packet implemented, so that I can use these implementations knowing their functionalities.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting:

- None.

Retrospective Meeting Minutes:

Project: AmLight

Date: 7/30/20

Attendees: Cindy Perez, Kenia Chang He, Jose Mercardo, and Yuyang Leng.

Start Time: 8:51PM

End Time: 8:57PM

What went wrong?

- Did we do a good job estimating our team's velocity?
 - Yes, we did a good job estimating our team's velocity.
- Did we do a good job estimating the points (time required) for each user story?
 - Yes, the user story was estimated correctly.
- Did each team member work as scheduled?
 - Yes, each member worked as scheduled.

What went right?

- Team communication went well, and it was smart to record the audio and video separately in order to edit them in an easier way.

How to address the issues in the next sprint?

- How to improve the process?
 - This is the last sprint for the development of this project.
- How to improve the product?
 - This is the last sprint for the development of this project.

Appendix D - User Manual, Installation Guide, and Wishlist

User Manual

This section contains all the necessary steps in order to successfully test the match fields with masking and the IPv6 packet structure implemented for Kytos.

Match Fields

Step 1: Enter the command in the terminal to run Kytos in foreground mode.

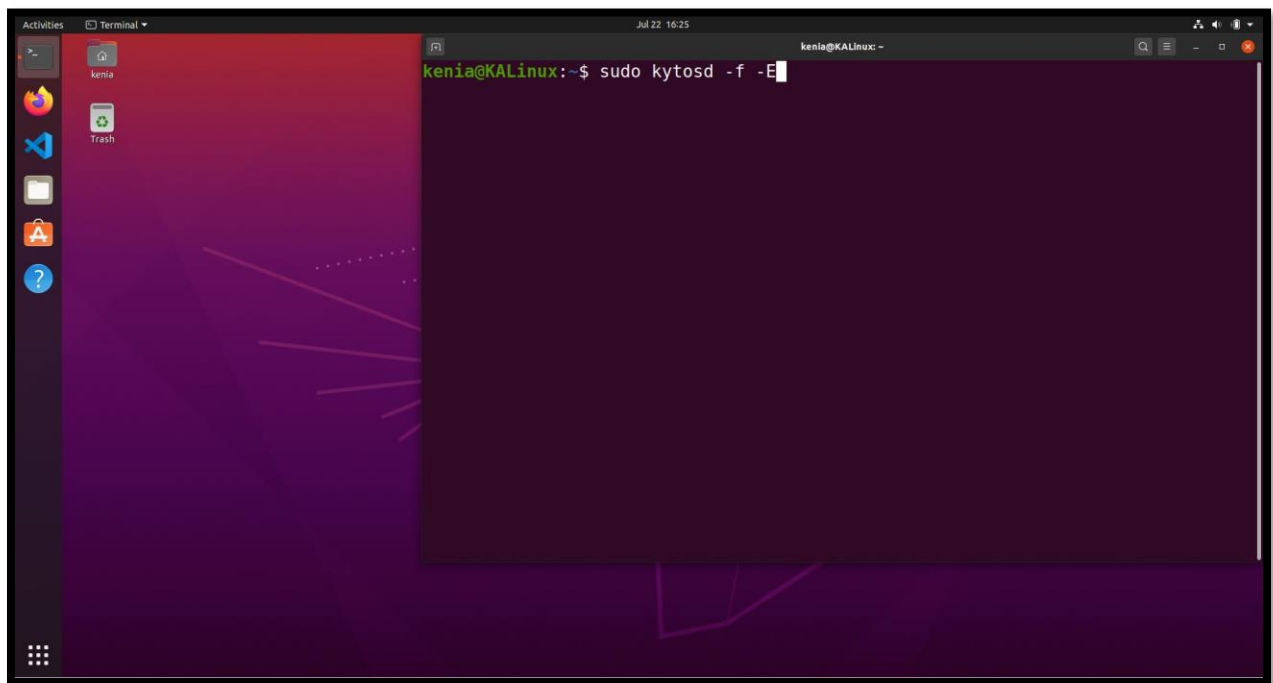


Figure 21- sudo kytosd -f -E command

Step 2: Enter the command in the terminal to run Mininet.

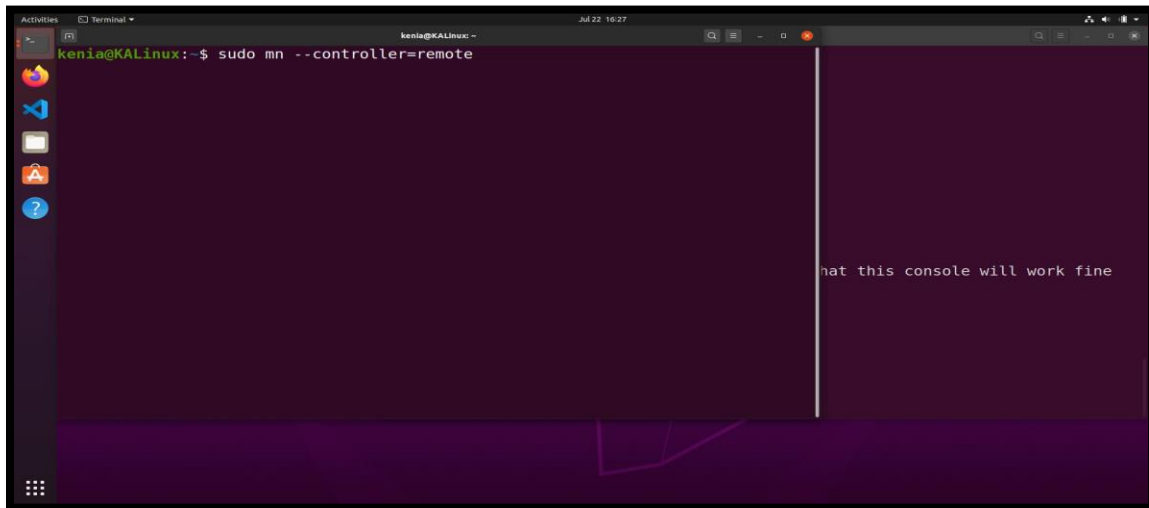


Figure 22- `sudo mn --controller=remote` command

Step 3: Enter the command in the terminal to send the flow to the Kytos server and it is processed by the flow_manager NApp.

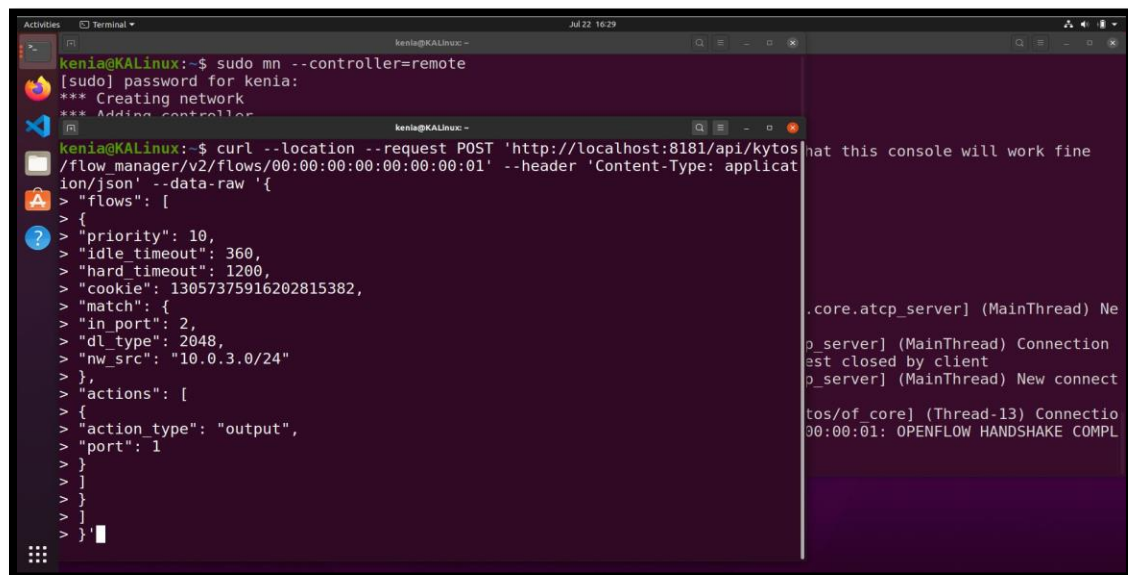
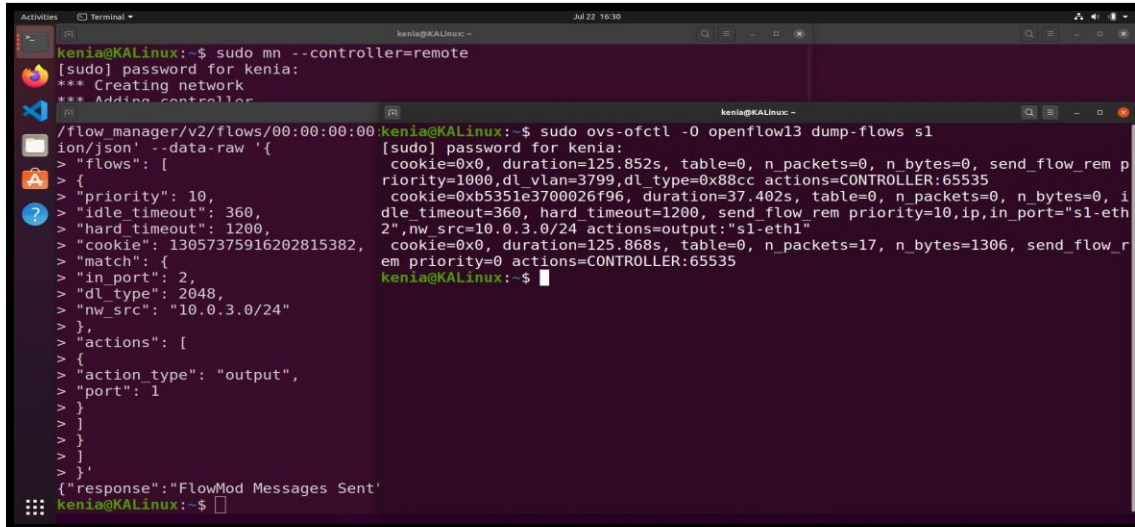


Figure 23- flow for the flow_manager NApp command

Step 4: Enter the command in the terminal to check the flow installed on switch 1.



```

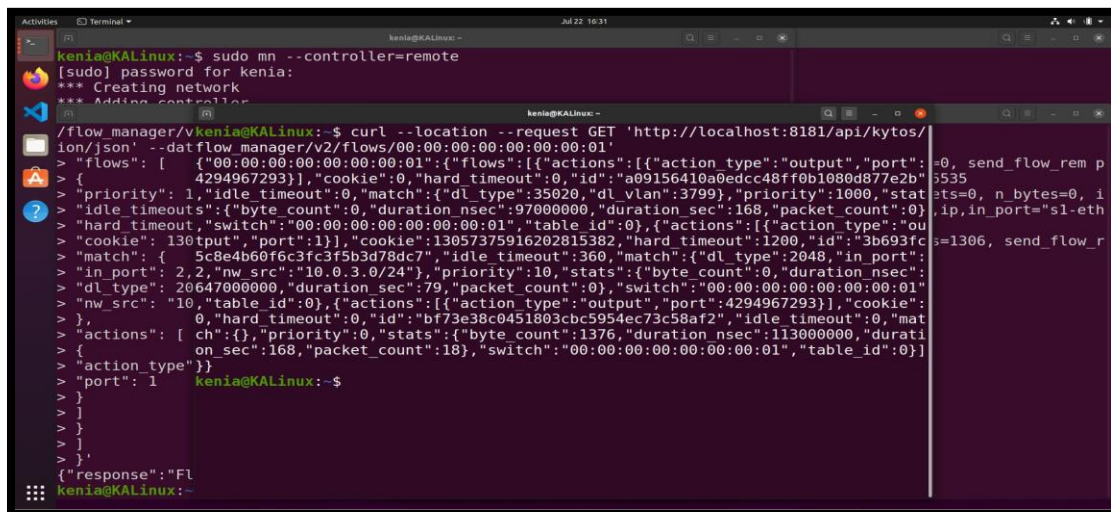
kenia@KALinux:~$ sudo mn --controller=remote
[sudo] password for kenia:
*** Creating network
*** Adding controller

/flow_manager/v2/flows/00:00:00:00:kenia@KALinux:~$ sudo ovs-ofctl -O openflow13 dump-flows s1
[sudo] password for kenia:
cookie=0x0, duration=125.852s, table=0, n_packets=0, n_bytes=0, send_flow_re p
riority=1000,dl_vlan=3799,dl_type=0x88cc actions=CONTROLLER:65535
cookie=0xb5351e3700026f96, duration=37.402s, table=0, n_packets=0, n_bytes=0, i
dle timeout=360, hard timeout=1200, send_flow_re priority=10,ip,in_port="s1-eth
2",nw_src=10.0.3.0/24 actions=output:"s1-eth1"
cookie=0x0, duration=125.868s, table=0, n_packets=17, n_bytes=1306, send_flow_r
em priority=0 actions=CONTROLLER:65535
kenia@KALinux:~$
{"response":"FlowMod Messages Sent"}
kenia@KALinux:~$

```

Figure 24- `sudo ovs-ofctl -O openflow13 dump-flows s1` command

Step 5: Enter the command in the terminal to check the flow processed by the flow_manager NApp.



```

kenia@KALinux:~$ sudo mn --controller=remote
[sudo] password for kenia:
*** Creating network
*** Adding controller

/flow_manager/vkenia@KALinux:~$ curl --location --request GET 'http://localhost:8181/api/kytos/
ion/json' --data-raw '{
  "flows": [
    {
      "priority": 1, "idle timeout": 0, "match": { "dl_type": 35020, "dl_vlan": 3799 }, "priority": 1000, "stat
      "idle timeout": { "byte count": 0, "duration nsec": 97000000, "duration sec": 168, "packet count": 0 },
      "hard timeout": { "switch": "00:00:00:00:00:00:01", "table id": 0 }, { "actions": [ { "action type": "ou
      "cookie": 1301put", "port": 1 }, { "cookie": 13057375916202815382, "hard timeout": 1200, "id": "3b693fc
      "match": { "5c8e4b60f6c3fc3f5b3d78dc7", "idle timeout": 360, "match": { "dl_type": 2048, "in port":
      "in port": 2, 2, "nw_src": "10.0.3.0/24" }, "priority": 10, "stats": { "byte count": 0, "duration nsec":
      "dl_type": 2064700000, "duration sec": 79, "packet count": 0 }, "switch": "00:00:00:00:00:00:01"
      "nw_src": "10", "table id": 0 }, { "actions": [ { "action type": "output", "port": 4294967293 }, { "cookie":
      }, { "hard timeout": 0, "id": "bf73e38c0451803cbc5954ec73c58af2", "idle timeout": 0, "mat
      "actions": [ { "ch": {}, "priority": 0, "stats": { "byte count": 1376, "duration nsec": 113000000, "durati
      "on sec": 168, "packet count": 18 }, "switch": "00:00:00:00:00:00:01", "table id": 0 } ]
      "action type": "
      "port": 1
    }
  ]
}'
kenia@KALinux:~$
{"response":"Fl
kenia@KALinux:~$

```

Figure 25- check the flow in the flow_manager NApp

IPv6 Packet

Step 1: Go to where the python-openflow repository is on the machine.

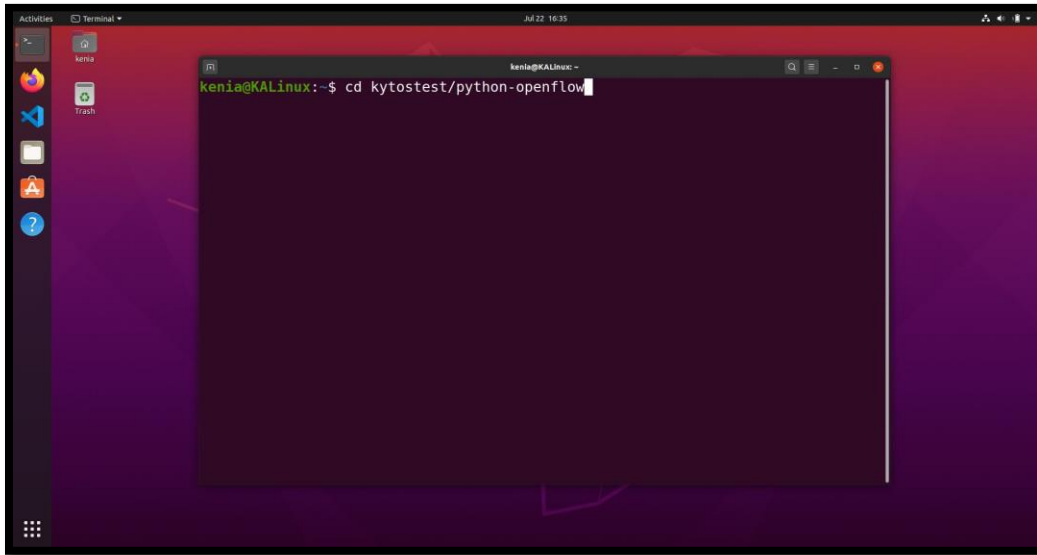


Figure 26- cd kytostest/python-openflow command

Step 2: Enter the command in the terminal to run the unit test for IPv6 packet.

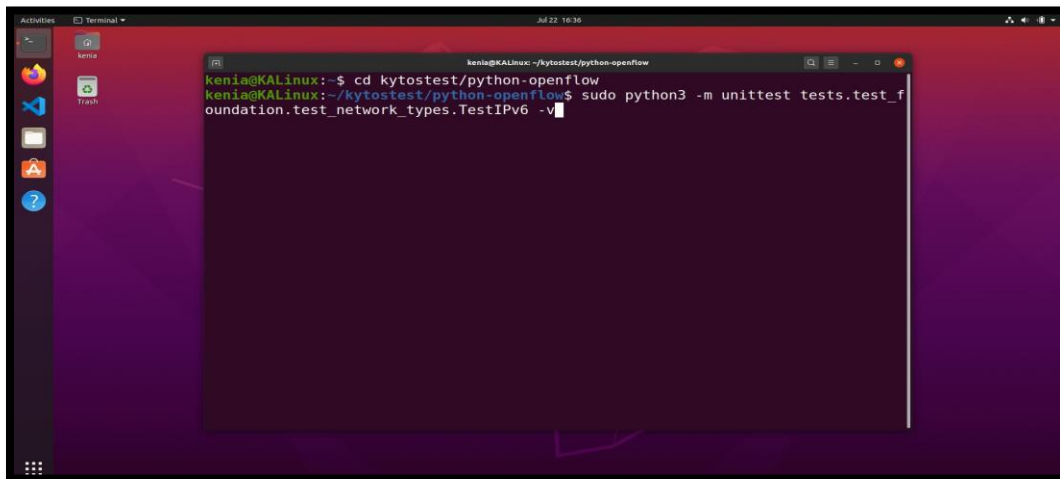
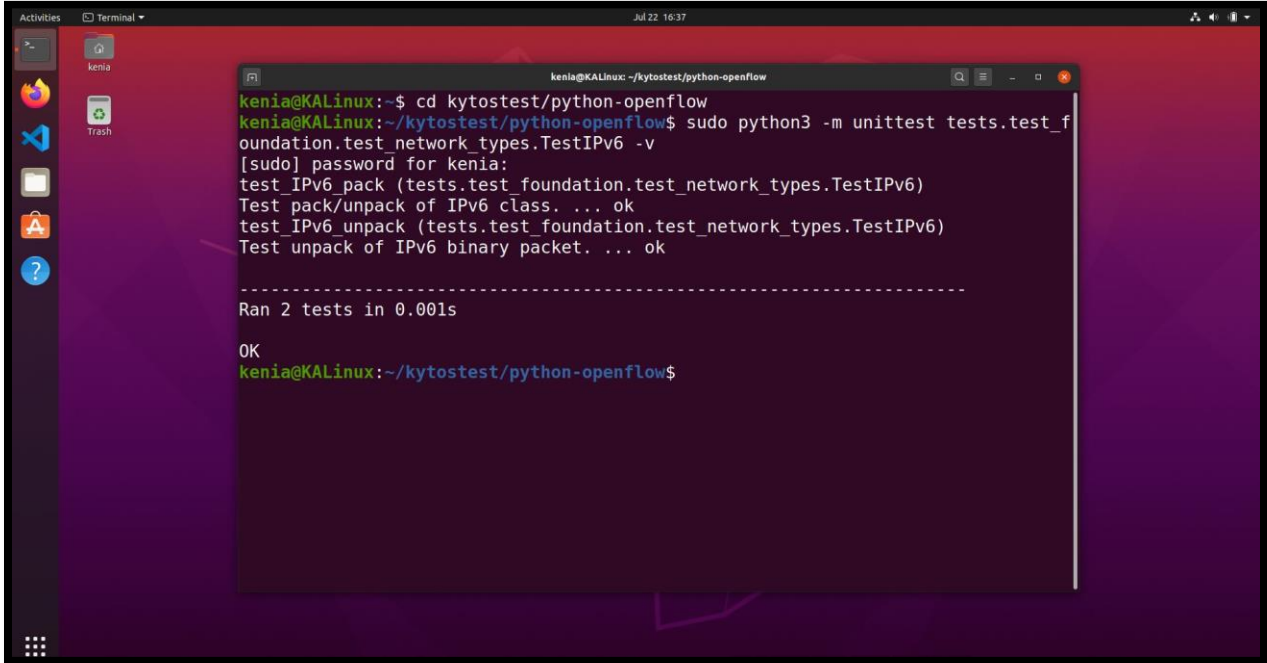


Figure 27- command to test the unit test

Here is the result of testing for the IPv6 packet.



The screenshot shows a terminal window on a Linux desktop. The user is in the directory `~/kystest/python-openflow`. They run the command `sudo python3 -m unittest tests.test_foundation.test_network_types.TestIPv6 -v`. The terminal output shows two tests passing: `test_IPv6_pack` and `test_IPv6_unpack`. The tests are part of `tests.test_foundation.test_network_types.TestIPv6`. The output indicates that the pack/unpack of IPv6 class and the unpack of IPv6 binary packet are both successful. The tests ran in 0.001s.

```
kenia@KALinux: ~/kystest/python-openflow
kenia@KALinux:~/kystest/python-openflow$ sudo python3 -m unittest tests.test_foundation.test_network_types.TestIPv6 -v
[sudo] password for kenia:
test_IPv6_pack (tests.test_foundation.test_network_types.TestIPv6)
Test pack/unpack of IPv6 class. ... ok
test_IPv6_unpack (tests.test_foundation.test_network_types.TestIPv6)
Test unpack of IPv6 binary packet. ... ok
-----
Ran 2 tests in 0.001s

OK
kenia@KALinux:~/kystest/python-openflow$
```

Figure 28- result of testing the unit test

Installation Guide

This section contains all the necessary steps in order to successfully run Kytos for the first time on any clean Ubuntu virtual machine.

Step 1: Enter the command in the terminal to install the latest version of Python which is python3.

```
ubuntu@ip-172-31-42-4:~$ sudo apt-get install python3
```

Step 2: Enter the command in the terminal to download the package lists.

```
ubuntu@ip-172-31-42-4:~$ sudo apt-get update
```

Step 3: Enter the command in the terminal to upgrade the packages from step 2.

```
ubuntu@ip-172-31-42-4:~$ sudo apt-get upgrade
```

Step 4: Enter the command in the terminal to install the dependency package.

```
ubuntu@ip-172-31-42-4:~$ sudo apt-get install libpython3-dev
```

Step 5: Enter the command in the terminal to install the python virtual environment package.

```
ubuntu@ip-172-31-42-4:~$ sudo apt-get install python3-venv
```

Step 6: Enter the command in the terminal to install the curl package.

```
ubuntu@ip-172-31-42-4:~$ sudo apt-get install curl
```

Step 7: Enter the command in the terminal to install the python3 pip tool.

```
ubuntu@ip-172-31-42-4:~$ sudo apt-get install python3-pip
```

Step 8: Enter the command in the terminal to make a directory called kytostest.

```
ubuntu@ip-172-31-42-4:~$ mkdir kytostest
```

Step 9: Enter the command in the terminal to upgrade the necessary python packages.

```
ubuntu@ip-172-31-42-4:~$ pip3 install --upgrade pip setuptools wheel
```

Step 10: Enter the command in the terminal to go to the kytostest directory.

```
ubuntu@ip-172-31-42-4:~$ cd kytostest
```

Step 11: Enter the command in the terminal to download Kytos onto the machine.

```
ubuntu@ip-172-31-42-4:~/kytostest$ for repo in python-openflow kytos-utils kytos; do  
> sudo git clone https://github.com/kytos/${repo}  
> cd ${repo}  
> sudo python3 setup.py develop  
> cd ..  
> done
```

Step 12: Enter the command in the terminal to start running Kytos in background mode.

```
ubuntu@ip-172-31-42-4:~/kytostest$ sudo kytosd
```

Step 13: Enter the command in the terminal to install the necessary Kytos NApps (you may need to run the command again if the installation fails).

```
ubuntu@ip-172-31-42-4:~/kytostest$ sudo kytos napps install kytos/of_core \  
> kytos/flow_manager \  
> kytos/of_12ls \  
> kytos/of_11dp \  
> kytos/topology
```

Step 14: Enter the command in the terminal to check the status of the Kytos NApps installed.

```
ubuntu@ip-172-31-42-4:~/kytostest$ kytos napps list
```

Here are the results that should be given after running the command above.

```
ubuntu@ip-172-31-42-4:~/kytostest$ kytos napps list
```

Status	NApp ID	Description
[ie]	kytos/flow_manager:2.3	Manage switches' flows through a REST API.
[ie]	kytos/of_core:1.4.1	OpenFlow Core of Kytos Controller, responsible for main OpenFlow operations.
[ie]	kytos/of_l2ls:1.1.1	A L2 learning switch application for OpenFlow switches.
[ie]	kytos/of_lldp:0.1.4	Discover network-to-network interfaces (NNIs) using the LLDP protocol.
[ie]	kytos/storehouse:1.3.0	Persistence NApp with support for multiple backends
[ie]	kytos/topology:3.6.1	Manage the network topology.

Status: (i)nstalled, (e)nabled

Step 15: Enter the command in the terminal to install Mininet.

```
ubuntu@ip-172-31-42-4:~/kytostest$ sudo apt-get install mininet
```

Step 16: Enter the command in the terminal to test Mininet.

```
ubuntu@ip-172-31-42-4:~/kytostest$ sudo mn --test pingall
```

Here is the result of testing Mininet.

```
ubuntu@ip-172-31-42-4:~/kytostest$ sudo mn --test pingall
*** No default OpenFlow controller found for default switch!
*** Falling back to OVS Bridge
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
*** Starting 1 switches
s1 ...
*** Waiting for switches to connect
s1
*** Ping: testing ping reachability
h1 -> h2
h2 -> h1
*** Results: 0% dropped (2/2 received)
*** Stopping 0 controllers

*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
completed in 0.236 seconds
```

Step 17: Enter the command in the terminal to check the process running in the machine.

```
ubuntu@ip-172-31-42-4:~/kystotest$ sudo ss -tunlp
```

Here are the results of the currently running processes command. Take note of the pid of “kystosd” as this can change from system to system.

```
ubuntu@ip-172-31-42-4:~/kystotest$ sudo ss -tunlp
Netid State Recv-Q Send-Q Local Address:Port Peer Address:Port users:((("systemd-resolve",pid=716,fd=12))
udp UNCONN 0 0 127.0.0.53%lo:53 0.0.0.0:* users:((("systemd-network",pid=695,fd=15))
udp UNCONN 0 0 172.31.42.4%eth0:68 0.0.0.0:* users:((("kystosd",pid=10158,fd=11))
tcp LISTEN 0 128 0.0.0.0:8181 0.0.0.0:* users:((("systemd-resolve",pid=716,fd=13))
tcp LISTEN 0 128 127.0.0.53%lo:53 0.0.0.0:* users:((("sshd",pid=1008,fd=3))
tcp LISTEN 0 128 0.0.0.0:22 0.0.0.0:* users:((("kystosd",pid=10158,fd=10))
tcp LISTEN 0 100 0.0.0.0:6653 0.0.0.0:* users:((("sshd",pid=1008,fd=4))
tcp LISTEN 0 128 [::]:22 [::]:* users:((("sshd",pid=1008,fd=4))
```

Step 18: Enter the command in the terminal to stop the Kytos process running in the background, replace the number “10158” with the pid of Kytos process observed from step 17.

```
ubuntu@ip-172-31-42-4:~/kystotest$ sudo kill -9 10158
```

Step 19: Enter the command in the terminal to check the currently running processes, and the process “kystosd” should be gone and not show up in the currently running processes list.

```
ubuntu@ip-172-31-42-4:~/kystotest$ sudo ss -tunlp
Netid State Recv-Q Send-Q Local Address:Port Peer Address:Port users:((("systemd-resolve",pid=716,fd=12))
udp UNCONN 0 0 127.0.0.53%lo:53 0.0.0.0:* users:((("systemd-network",pid=695,fd=15))
udp UNCONN 0 0 172.31.42.4%eth0:68 0.0.0.0:* users:((("systemd-resolve",pid=716,fd=13))
tcp LISTEN 0 128 0.0.0.0:22 0.0.0.0:* users:((("sshd",pid=1008,fd=3))
tcp LISTEN 0 128 127.0.0.53%lo:53 0.0.0.0:* users:((("sshd",pid=1008,fd=4))
tcp LISTEN 0 128 [::]:22 [::]:* users:((("sshd",pid=1008,fd=4))
```

Wishlist

The following is a list of items we would like to be added to the next iteration of the product:

- Testing for our masking implementation of the physical input port needs to be done more extensively (there is a special piece of hardware needed in order to test this functionality)
- The packet structure for the PBB ISID needs to be implemented into the Kytos switch controller and the masking functionality needs to be tested
- The packet structure for the IPv6 Extension Header needs to be implemented into the Kytos switch controller and the masking functionality needs to be tested

REFERENCES

Kytos SDN Platform :: Your network, your rules! Retrieved July 21, 2020, from <https://kytos.io/>

Open Networking Foundation. OpenFlow Switch Specification Version 1.3.5 Published March 26, 2015. Retrieved July 21, 2020, from <https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.3.5.pdf>