A Study and Performance Evaluation of OpenDaylight Controller in Software Defined Networks (SDN)

Dr. Radwan Dandah[*]
Dr. Talal Al Aateky[**]
Ranim Sino[***]

(Received 29 / 12 / 2019. Accepted 17 / 5 / 2020)

□ ABSTRACT □

Software defined networks SDN is one of the most influential types of networks in information and communication technology compared to all traditional network technologies where there are many challenges, SDN is one of the most promising solutions for the Internet in the future and provides a strong network infrastructure with high specifications and low cost, and represents the future of the next generation of network engineering due to the easy division of networks, programming, monitoring, control and management through a central control, and the separation of control plane from the data-forwarding plane in SDN networks facilitates the process of managing and developing networks, as this technology is currently used in data centers and wireless networks, and is a solution to many of the problems faced by traditional networks.

SDN networks are characterized by sufficient dynamism to deal with the different conditions of the network, and the controller is one of its most important components and is considered the smartest component in the network, and given the importance of choosing the appropriate controller according to the different parameters and conditions of the SDN network,in this research we conducted an analysis of the characteristics of Software defined Networks, The analysis relied on comparing SDN networks with the presence of a single OpenDaylight controller (ODL) and the presence of several controllers ODL, OpenDaylight controller was chosen as one of the famous controllers and it is distinguished from others as an open source and contains a distributed datastore and is designed to suit the data center environment, Taking into account the OpenFlow protocol supported on the southern interface by this controller, the research includes simulations of software defined networks topologies using the Mininet emulator, and various scenarios and parameters such as data rate, packet delay, and throughput were analyzed by the D-ITG tool.

Keywords: Software Defined Networks, OpenDaylight Controller, OpenFlow Protocol, Throughput, Mininet ,D-ITG.

[*]**Professor, Department of Computer Networks & System, Faculty of Information Technology, University of Tishreen, Lattakia, Syria.**
[**]**Assistant Professor, Department of Computer Networks & System, Faculty of Information Technology, University of Tishreen, Lattakia, Syria.**
[***]**Postgraduate Student (Ph.D.), Department of Computer Networks & System, Faculty of Information Technology, University of Tishreen, Lattakia, Syria.**

دراسة و تقييم أداء المتحكم OpenDaylight في الشبكات المعرفة برمجيا" SDN

د.رضوان دنده*

د.طلال العاتكي**

رنيم سينو***

(تاريخ الإيداع 29 / 12 / 2019. قُبِل للنشر في 17/ 5 / 2020)

□ ملخّص □

تعد الشبكات المعرفة برمجيا" SDN واحدة من أكثر أنواع الشبكات تأثيرا" في تكنولوجيا المعلومات والاتصالات مقارنة بجميع تكنولوجيات الشبكات التقليدية التي يوجد فيها الكثير من التحديات، كما تعدّ شبكات SDN من أكثر الحلول الواعدة للإنترنت في المستقبل وتوفر بنية تحتية قوية للشبكات بمواصفات عالية وتكلفة منخفضة، وتمثل مستقبل الجيل القادم لهندسة الشبكات بسبب سهولة تقسيم الشبكات وبرمجتها ومراقبتها، والتحكم فيها وإدارتها من خلال وحدة تحكم مركزية، كما إنّ فصل مسارات التحكم عن مسارات تمرير البيانات في شبكات SDN، يسهّل عملية إدارة وتطوير الشبكات، حيث تستخدم هذه التقنية حاليا" في مراكز البيانات والشبكات اللاسلكية، كما تعد حلا" للكثير من المشاكل التي تواجهها الشبكات التقليدية.

تتميز شبكات SDN بالديناميكية الكافية للتعامل مع الظروف المختلفة للشبكة، ويعدّ المتحكم أحد أهم مكوناتها ويعتبر المكون الأذكى في الشبكة، ونظرا" لأهمية اختيار المتحكم المناسب حسب البارامترات والظروف المختلفة لشبكة SDN، قمنا في هذا البحث بإجراء تحليل لخصائص الشبكات المعرفة برمجيا"، و اعتمد التحليل على مقارنة شبكات SDN بوجود متحكم وحيد OpenDaylight (ODL) مع عدة متحكمات ODL، تم اختيار المتحكم ODL كونه يعدّ من المتحكمات الشهيرة ويتميز عن غيره بأنه مفتوح المصدر ويحتوي على datastore موزعة ومصمم ليناسب بيئة مراكز البيانات، مع الأخذ بعين الاعتبار بروتوكول OpenFlow المدعوم على الواجهة الجنوبية من قبل هذا المتحكم، ويتضمن البحث محاكاة لطبولوجيا الشبكات المعرفة برمجيا" باستخدام المحاكي Mininet، كما تم تحليل السيناريوهات والمعاملات المختلفة مثل معدل البيانات وتأخير الرزم والإنتاجية من خلال الأداة D-ITG.

الكلمات المفتاحية: الشبكات المعرفة برمجيا" ،المتحكم OpenDaylight، بروتوكول OpenFlow، الإنتاجية، D-ITG، Mininet.

* **أستاذ، قسم النظم والشبكات الحاسوبية، كلية الهندسة المعلوماتية ، جامعة تشرين- اللاذقية- سورية.**

** **مدرس، قسم النظم والشبكات الحاسوبية، كلية الهندسة المعلوماتية ، جامعة تشرين – اللاذقية- سورية.**

*** **طالبة دراسات عليا (دكتوراه)، قسم النظم والشبكات الحاسوبية ، كلية الهندسة المعلوماتية ، جامعة تشرين- اللاذقية-سورية.**

مقدمة:

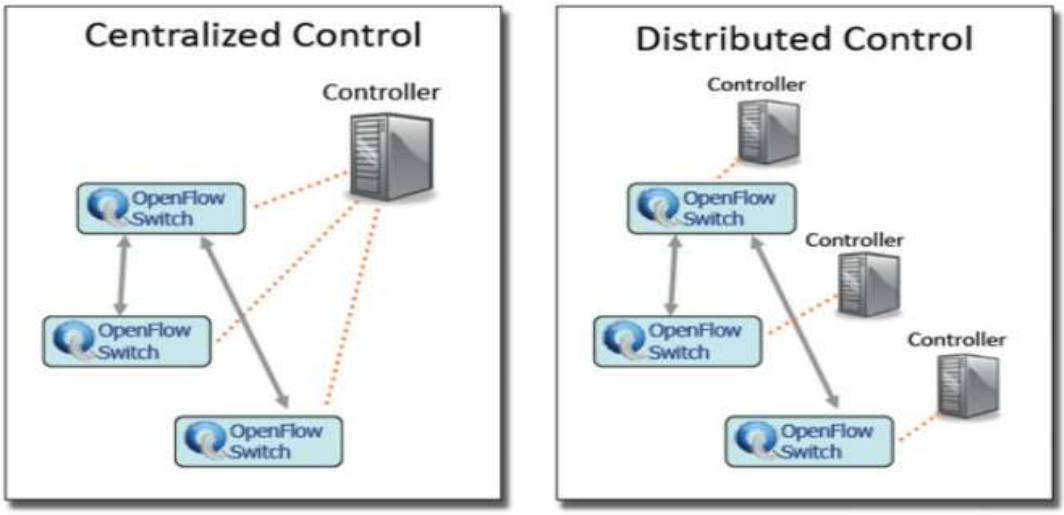
تعد الشبكات المعرفة برمجيا" SDN مجالا" حديثا" يهتم به الكثير من الباحثين في مختلف المجالات، وخاصة في مجال هندسة الشبكات، كونها قدّمت حلا" للعديد من التحديات التي تواجه الشبكات التقليدية، حيث يعتمد هذا الحل على مركزة ذكاء الشبكة بعيدا"عن أجهزة توجيه البيانات، يعني ذلك تحقيق نظام تشغيل شبكي يتم فيه إنجاز مهام الشبكة بدون الحاجة إلى برمجيات إضافية لكل عنصر في الشبكة، وذلك نظرا" لتعامل المتحكم بصورة تجريدية وافتراضية مع المبدلات المتصلة به واحتفاظه بنظرة عامة وشاملة للشبكة بكاملها.

لعبت الشبكات المعرفة برمجيا" SDN دورًا مهمًا في التغلب على القيود المرتبطة بالشبكات التقليدية، كما جذبت اهتمام الكثير من الباحثين نظرا" للمرونة التي توفرها في عملية إدارة وتطوير الشبكات، من خلال فصل ارتباط التطوير بالبنية التحتية، وتعتمد شبكات SDN على فصل مسارات البيانات عن مسارات التحكم لجعل التحكم بالشبكة مركزيا"، وبالتالي تقليل الأخطاء التي من الممكن أن تتعرض لها الشبكات، لذلك نهتم في هذا البحث بدراسة وتحليل أداء المتحكم (**O**pen**D**ay**l**ight(ODL**)**، وتم اختياره في ظل عدد كبير من المتحكمات في الشبكات المعرفة برمجيا"، كونه يعدّ مفتوح المصدر ومكتوب بلغة الجافا ويتميز بتقديمه واجهة رسومية GUI للمستخدم، ويحتوي على datastore موزعة ويتفوق عن غيره من المتحكمات كونه متعدد النياسب multi-threaded ، ونسعى من خلال هذا البحث إلى تسهيل إدارة الشبكة وتحسين زمن الاستجابة، وتخفيف الحمل على المتحكم ODL من خلال توزيع الحمل على عدة متحكمات ODL باستخدام خوارزمية موازنة الحمل التي قدمها الباحثون في [19] والمستخدمة من قبل الأداة D-ITG، وذلك بهدف تحليل وتقييم أداء شبكة SDN المدروسة بالاعتماد على المحاكي Mininet ودراسة النتائج، ويستخدم البروتوكول OpenFlow للاتصال بين طبقة التحكم وطبقة تمرير البيانات، حيث يعد مناسبا" لبيئة مراكز البيانات، وذلك للتعامل مع الازدياد الكبير في تدفق البيانات، وفي شبكات الحرم الجامعي حيث يستخدم لعزل تدفق البيانات التابع للباحثين عن بقية التدفقات في الشبكة، وتم توضيح الرسائل المتبادلة في كل تدفق بين المبدل ومتحكمات ODL عن طريق OpenFlow.

أهمية البحث وأهدافه:

توجهت الأبحاث إلى إيجاد طريقة مثالية تعمل على استبدال البنية الحالية مع زيادة في المرونة والأداء وخفض تكاليف بناء وإدارة الأجهزة الشبكية، فظهرت تقنية الشبكات المعرفة برمجيا" SDN التي تركزت على دمج عالم الشبكات بعالم البرمجيات، بغية إحداث تغيير جذري للبنية التحتية بما يتناسب مع التطور التقني، وتقوم شبكات SDN بفصل مستوى تمرير البيانات عن مستوى التحكم وتسمح بتحكم مركزي للشبكة، ومن هنا تأتي أهمية هذا البحث في إيجاد طريقة لإدارة الشبكة والتخفيف من تعقيدها، كما أن فصل أجهزة التوجيه عن منطق التحكم يسمح بنشر أسهل للبروتوكولات والتطبيقات الجديدة وتصور وإدارة الشبكة بشكل واضح، فبدلا" من فرض السياسات وتنفيذ البروتوكولات على الأجهزة، يتم اختصار الشبكة إلى أجهزة توجيه بسيطة ووحدة تحكم أو وحدات تحكم بالشبكة هي من تقوم بصنع القرار، ومن هنا نسعى إلى مقارنة وتقييم أداء الشبكات المعرفة برمجيا" التي تحوي متحكم وحيد OpenDaylight ، مع حالة شبكات SDN تحوي عدة متحكمات OpenDaylight تتقاسم الحمل فيما بينها كما يوضحه الشكل (1)، وملاحظة ازدياد أداء

شبكة SDN مع ازدياد عدد متحكمات ODL ، بعد أن تم تخفيف الحمل على المتحكم المركزي الوحيد وبالتالي على شبكة SDN ككل.



الشكل(1) (شبكة SDN تحوي عدة متحكمات ODL controllers) و (شبكة SDN تحوي متحكم وحيد ODL controller).

طرائق البحث ومواده:

يتضمن هذا البحث تحسين أداء الشبكات المعرفة برمجيا" وخاصة بالنسبة للشبكات الكبيرة، ونسعى إلى زيادة عدد متحكمات ODL الموجودة في الشبكة المدروسة باستخدام Mininet ، بهدف تشارك الحمل بشكل عادل ومتساو بين المتحكمات، وتخفيض زمن الاستجابة، كونه في حالة وجود متحكم وحيد ODL يحوي كامل معلومات الشبكة المعرفة برمجيا"، هذا يؤدي للعديد من التحديات منها قدرة تحمل وزمن استجابة أكبر، وأيضا" في حالة فشل المتحكم يؤدي إلى تعطل عمل كامل الشبكة، بينما في حالة وجود عدة متحكمات ODL فإن توقف أحدها عن العمل لن يؤثر على عمل الشبكة، ويعتبر محاكي الشبكات المعرفة برمجيا" Mininet منصة تعتمد على لغة Python [18]، وتعمل على أنظمة Linux، بالإضافة إلى أنها تدعم برنامج Wireshark الذي يتم تضمينه مع الحزمة الخاصة بالمنصة، بالتالي إعطاء إمكانية لتحليل أداء الشبكة، ويتضمن Mininet واجهة سطر أوامر مدركة للشبكة (CLI)، وتوفر واجهة مستخدم بسيطة يوفرها الملف النصي Miniedit، ويعمل Mininet على تشغيل وإدارة مجموعة من الأجهزة المضيفة والمبدلات ووحدات التحكم، باستخدام المحاكاة الإفتراضية لجعل نظام واحد يبدو كأنه شبكة كاملة، كما سنستخدم في هذا البحث الأداة D–ITG(Distributed Internet Traffic Generator) [20]، وهي عبارة عن منصة عمل قادرة على توليد المرور (Traffic) وقياس أداء الشبكة مثل إنتاجية الشبكة والتأخير وفقدان الرزم.

قام العاملون في هذا البحث [4] بوضع معايير لفحص أداء المتحمات في الشبكات المعرفة برمجيا"، تسمح بمقارنة المتحكمات بشكل عادل تبعا" لفئاتها المختلفة، بينما قام الباحثون في هذه الدراسة [5] بإجراء مقارنة بين أشهر المتحكمات في الشبكات المعرفة برمجيا"، من ناحية الأداء ومن ناحية الخصائص، حيث عرضت مزايا كل متحكم ومشاكله وجرى تقييم أداء المتحكمات بناءا" على التأخير في المتحكم ومعدل الإنتاجية throughput له، وذلك بتغيير عدد عناصر الشبكة وفق عتاديات محددة.

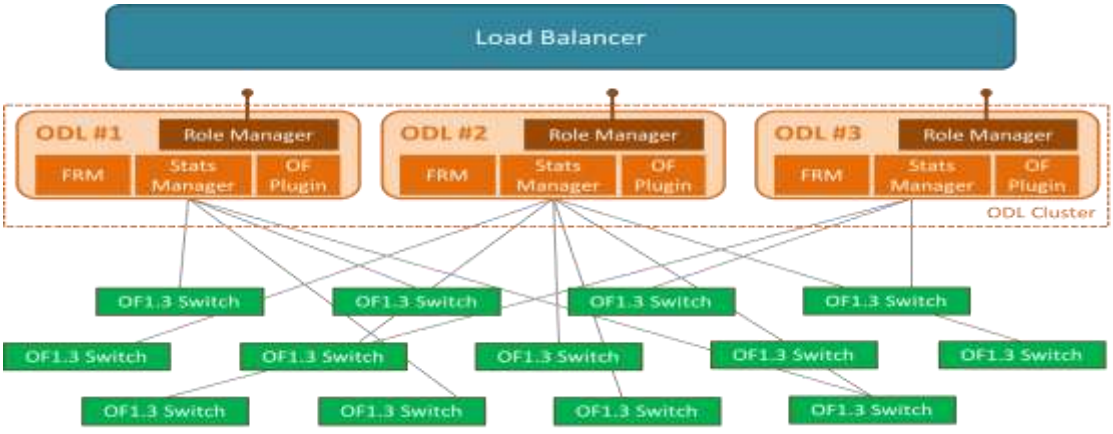
قام الباحثون في [6] بدراسة نظرية للمتحكم من نوع OpenDaylight ، تتضمن الدراسة بنيانه وواجهاته الشمالية مع المكونات ذات المستوى الأعلى وواجهاته الجنوبية مع المكونات ذات المستوى الأدنى، وأكد على ضرورة دعم المتحكم لعدة واجهات جنوبية، وليس فقط البروتوكول OpenFlow، بينما قام الباحثون في هذه الدراسة [7] بتطوير نموذج ماركوفي تحليلي، وأجروا عدة قياسات لأداء المتحكم في الشبكات المعرفة برمجيا".

تناول الباحثون في [8] الأخطاء على مستوى خطوط توجيه البيانات، أي تعطل الوصلات والعقد، حيث تم تقديم إطار العمل CORONET، الذي اعتمد على تعريف عدد من الخطوط الاحتياطية في حال تعطل الأساسية، وتعمل الخوارزمية الرئيسية على مبدأ معرفة التحديثات المستمرة لطبولوجيا الشبكة، حيث تستطيع إدارة عملية الأخطاء بعدد قليل من رسائل التحكم.

قام الباحثون في هذه الدراسة [9] بإجراء تحليل لخصائص الشبكات المعرفة برمجيا"، اعتمد التحليل على مقارنة الشبكات التقليدية والشبكات المعرفة برمجيا" مع عرض الاختلافات الهامة، يتضمن البحث محاكاة لطبولوجيا الشبكة التقليدية باستخدام محاكي GNS3 ، ومحاكاة لطبولوجيا الشبكات المعرفة برمجيا" باستخدام محاكي GNS3 ومحاكي mininet ، بينما قام الباحثون في هذه الدراسة [10] بالاعتماد على المتحكم POX الذي يستخدم منصة JAVA وبالاعتماد على المحاكي Mininet والتركيز على آلية توزيع الحمل في الشبكة وقياس الأداء.

في هذه الدراسة [11] تم تقديم شرح تفصيلي وشامل لبروتوكول التدفق المفتوح OpenFlow بنسخه المختلفة، للحصول على إحصائيات المرور من أجهزة الشبكة، ومن ثم استخدام هذه المعلومات لحساب عرض النطاق الترددي على كل وصلة في الشبكة، كما تم استخدام Mininet لإنشاء طبولوجيا الاختبار وfloODLight كوحدة تحكم لشبكة openflow.

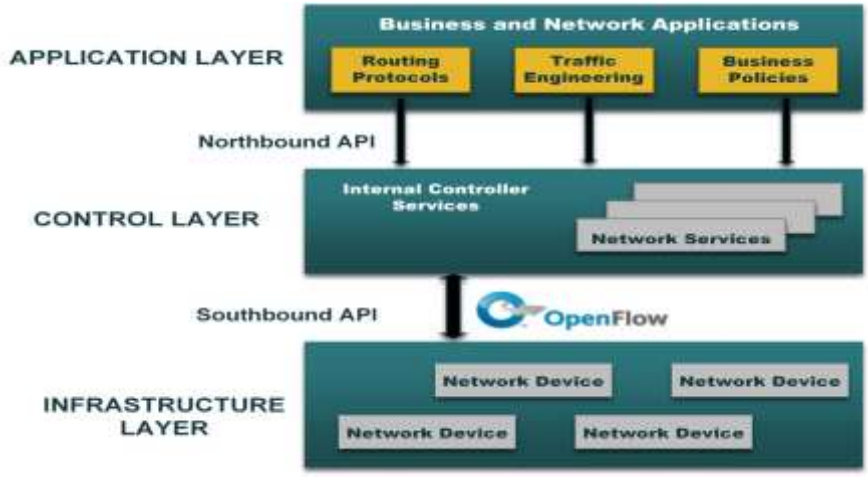
جميع الدراسات السابقة اعتمدت على وجود متحكم مركزي وحيد أدى إلى زيادة الحمل بشكل كبير على المتحكم، وبالتالي انخفاض في أداء وإنتاجية الشبكة، لذلك اعتمدنا في دراستنا هذه إلى تقييم أداء شبكة SDN، ودراسة وتحليل النتائج لعدة سيناريوهات تحوي متحكم وحيد ODL ، ثم نبدأ بزيادة عدد المتحكمات كما يبينه الشكل(2)، فنجد زيادة أداء الشبكة بشكل ملحوظ وزمن الاستجابة أقل بكثير من حالة وجود متحكم وحيد ODL.



الشكل (2) شبكة SDN تحوي عدة متحكمات ODL.

1- بنية الشبكات المعرفة برمجيا" (Software Defined Networks) SDN:

الشبكات المعرفة برمجيا" SDN هي واحدة من التقنيات المهمة التي يمكن استغلالها في هندسة الشبكات، حيث تعطي مرونة وديناميكية للشبكة، كما تجعلها أكثر فاعلية وقابلة للتكيّف وتحسّن من أداء الشبكة، يتم فيها فصل وظائف التحكم عن وظائف نقل البيانات، كما تجذب شبكات SDN اهتمام الكثير من الباحثين نظرا" للمرونة التي توفرها في عملية إدارة وتطوير الشبكات، [12] من خلال فصل ارتباط التطوير بالبنية التحتية، وتم تقسيم البنية المعمارية، كما هو موضح في الشكل (3) إلى 3 طبقات :



الشكل (3) البنية المعمارية للشبكات المعرفة برمجيا" SDN Architecture .

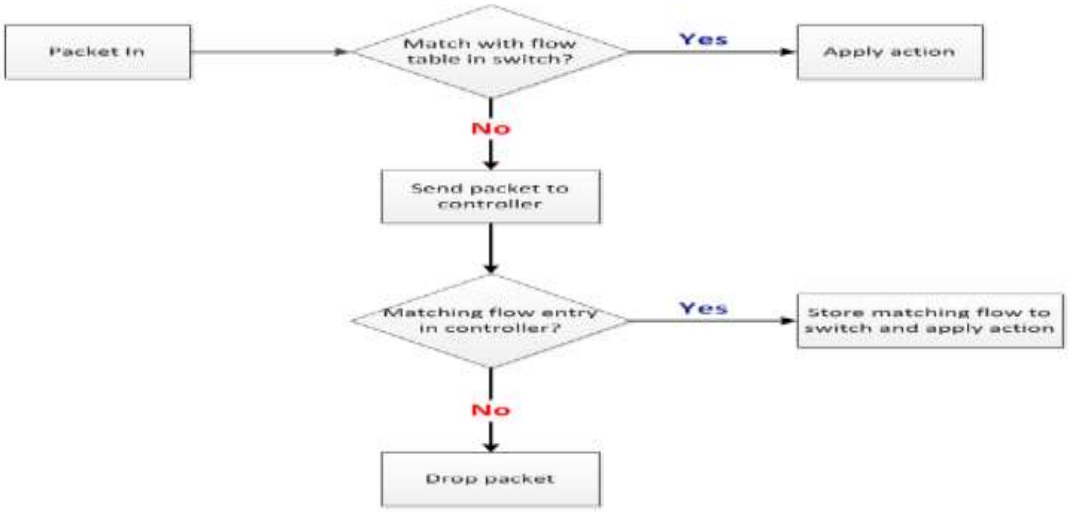
1. طبقة التطبيقات (Application Layer) : تتكون من التطبيقات والخدمات التي تقدمها الشبكة للمستخدم.

2.طبقة التحكم (Control Layer): هذه الطبقة تحتوي على المتحكم الذي يعتبر أساس هذه التقنية، فهو المسؤول عن قرارات التوجيه والإدارة والتحكم بكامل وظائف الشبكة وتتصل مع طبقة التطبيقات من خلال واجهة برمجة التطبيقات (API).

3. طبقة البنية التحتية (Infrastructure Layer) :والتي تتكون من أجهزة الشبكة (سويتشات،راوترات،...) تتلقى الأوامر من طبقة التحكم، وتقوم بتنفيذها حيث تتصل هذه الطبقة مع طبقة التحكم عن طريق البروتوكول OpenFlow وهو بروتوكول خاص بتقنية SDN ينظم التواصل بين المتحكم والبنية التحتية [13].

2- بروتوكول التدفق المفتوح OpenFlow (OF) Protocol :

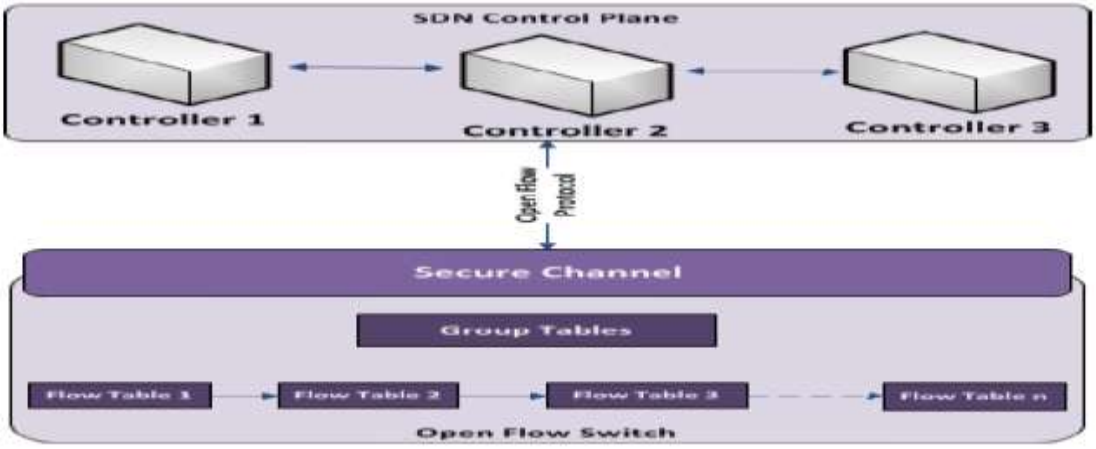
هو بروتوكول معياري مفتوح المصدر يمكّن من التحكم بتجهيزات الشبكة بشكل برمجي، ومصمم بشكل رئيسي للباحثين في مجال الشبكات ليقوموا باختبار تجاربهم على شبكات حقيقية وشبكات الحرم الجامعي، ويستخدم بروتوكول OpenFlow لإدارة الواجهة الجنوبية للبنية العامة لشبكة SDN، كما يزوّد بروتوكول التدفق المفتوح النفاذ إلى جداول التدفق التي توجه المبدلات لكي تقوم بمعالجة الرزم، حيث يمكن للمبرمج باستخدام هذه الجداول أن يغيّر بشكل سريع وسهل تدفق البيانات والسياسة العامة في الشبكة، من خلال تعديل القواعد الخاصة بتوجيه الرزم وإرسالها إلى المبدلات [3]، بدلا" من أن يقوم بتعديل كل مبدل على حدى أو عتاديا"، كما يوفر البروتوكول سهولة إدارة الشبكة والتعامل مع تغيير طبولوجيتها واستخدام قواعد فلترة الرزم، حيث يقوم البروتوكول OpenFlow بمعالجة التدفقات الواردة إلى مبدل OF، كما هو مبين في الشكل(4)، وذلك من خلال مطابقة التدفق مع القواعد الموجودة في الجدول، فإذا لم توجد قاعدة مطابقة يقوم المبدل بتغليف الرسالة وإرسالها إلى المتحكم الذي يقرر إرسال قاعدة مناسبة أو سحب الرزمة.



الشكل (4) معالجة التدفقات في بروتوكول Open Flow

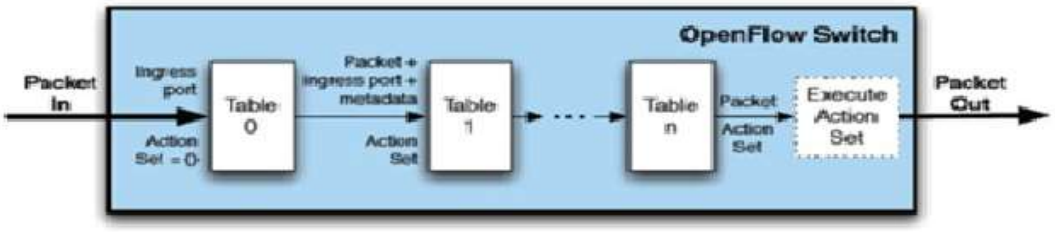
3- البنية التحتية لشبكة OpenFlow :

تتألف البنية التحتية للشبكات التي تستخدم بروتوكول OpenFlow من عدد من التجهيزات أو المبدلات التي تعمل وفق قيادة المتحكم كما يوضحه الشكل (5):



الشكل (5) البنية التحتية لشبكة OpenFlow.

من العناصر الرئيسية التي تتألف منها البنية التحتية لشبكة OpenFlow هو مبدل التدفق المفتوح Open Flow Switch، الذي يتألف من واحد أو أكثر من جداول التدفق، حيث تقوم هذه المبدلات بإنجاز عمليات البحث والتوجيه [2]، ويتم إدارة هذه الجداول من قبل المتحكم الذي يتصل بقناة آمنة مع المبدل، يتألف كل جدول تدفق من عدد من المداخل، الذي يعرف القواعد التي تتألف كل قاعدة منها من ترويسات الرأس ، العداد، ومجموعة من الأفعال المترافقة مع كل فعل جديد في الشبكة كما يبينه الشكل (6).



الشكل (6) مداخل مبدل التدفق المفتوح.

ويتضمن بروتوكول التدفق المفتوح الأنواع التالية من الرسائل كما يوضحه الشكل (7):

1. الرسائل من المتحكم إلى المبدل (Controller To Switch Message): يتم تجهيز هذا النوع من الرسائل وتهيئته عند المتحكم وإرسالها إلى المبدل، وقد لاتتطلب معظم هذه الرسائل استجابة من المبدل، فيما يلي الأنواع المختلفة من هذه الرسائل:

-OFPT_FEATURES_REQUEST: عندما يتصل المبدل مع المتحكم، يقوم المتحكم بالاستعلام عن حالة المبدل ومميزاته، يستجيب المبدل برسالة الاستجابة التي تتضمن خصائص معينة.

-OFPT_BARRIER_REQUEST: يقوم المتحكم بإرسال رسالة Barrier Request ليتأكد من أن الرسالة التي أرسلت قبل هذا الطلب قد تم معالجتها لدى المبدل.

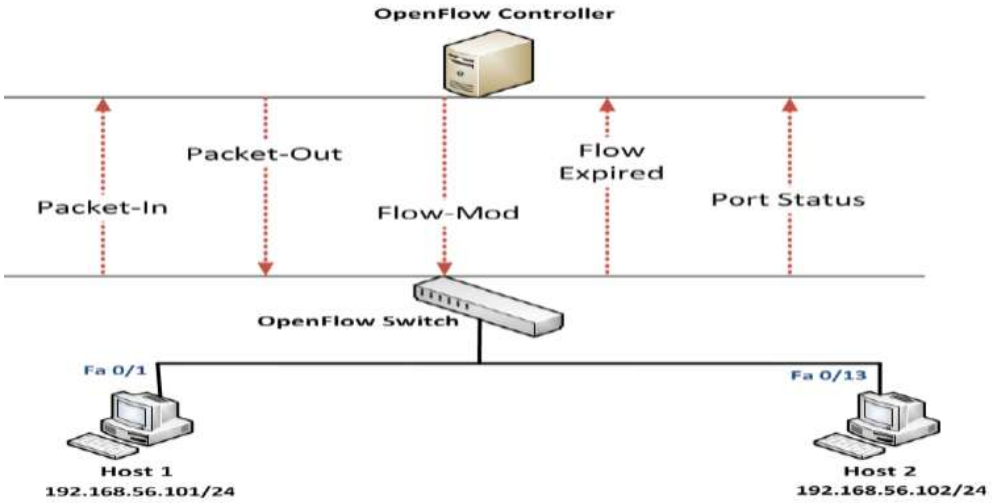
2. الرسائل من المبدل إلى المتحكم (Asynchronous Messages): يتم إرسال هذا النوع من الرسائل من المبدل إلى المتحكم لإعلامه بالأحداث، نبين فيما يلي بعض أنواع هذه الرسائل حيث يظهر الشكل التالي(7) الرسائل المتبادلة بين المبدل والمتحكم:

- OFPT_PACKET_IN: إذا تم استقبال رزمة على منفذ معين من المبدل ولم تطابق الرزمة أي من القواعد المخزنة ضمن جدول التدفق، عندئذ يقوم المبدل بتغليف الرزمة وإرسالها إلى المتحكم من أجل اتخاذ قرار بشأنها.

- OFPT_FLOW_REMOVED: يتم إضافة القواعد الخاصة بالتوجيه في جداول التدفق وفق قيمة زمنية تدل على مدة صلاحية تطبيق هذه القاعدة.

- OFPT_PORT_STATUS: عندما يغير منفذ في المبدل حالته، على سبيل المثال تم تعطيل المنفذ لسبب ما فإن المبدل يقوم بإرسال هذه الرسالة Port-Status إلى المتحكم لكي يقوم بإعلامه بحالته.

- OFPT_ERROR: عندما يحدث خطأ في المبدل يتم إرسال رساله إلى المتحكم لإعلامه بذلك [14].



الشكل (7) الرسائل المتبادلة في بروتوكول OpenFlow.

3. الرسائل المتبادلة بين المتحكم والمبدل بكلا الاتجاهين (Symmetric Messages): يتم تبادل مثل هذا النوع من الرسائل في الاتجاهين أي بين المبدل والمتحكم، كما يوضحه الشكل (8) تتضمن الأنواع التالية من الرسائل مثل:

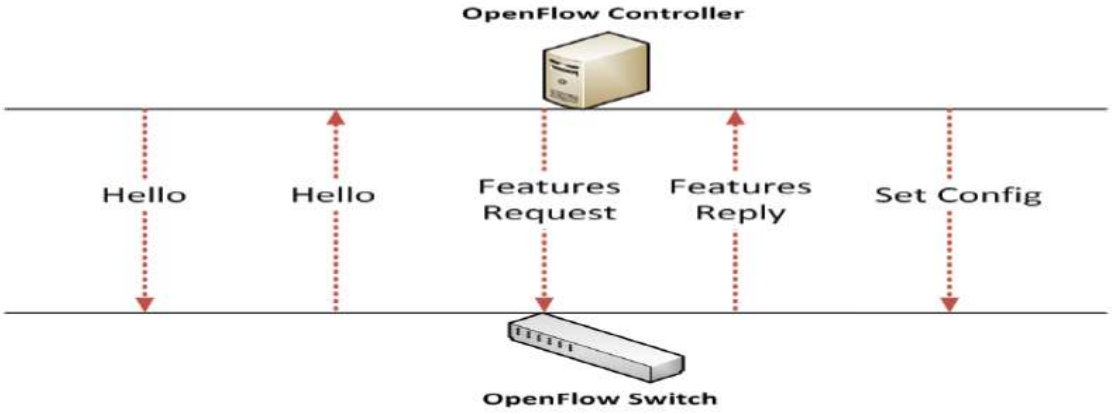
Hello : من المتحكم إلى المبدل يقوم المتحكم بإرسال رقم النسخة إلى المبدل.

Hello : من المبدل إلى المتحكم يقوم بإرساله رقم نسخته.

Feature Request: يسأل المتحكم عن أي المنافذ يستخدمها المبدل.

Features Reply: يستجيب المبدل بقائمة من منافذ وسرعات المنفذ والجداول المدعومة والأفعال.

Set Config : يطلب المتحكم من المبدل أن يرسل تاريخ انتهاء الصلاحية للقواعد المخزنة في جداوله.



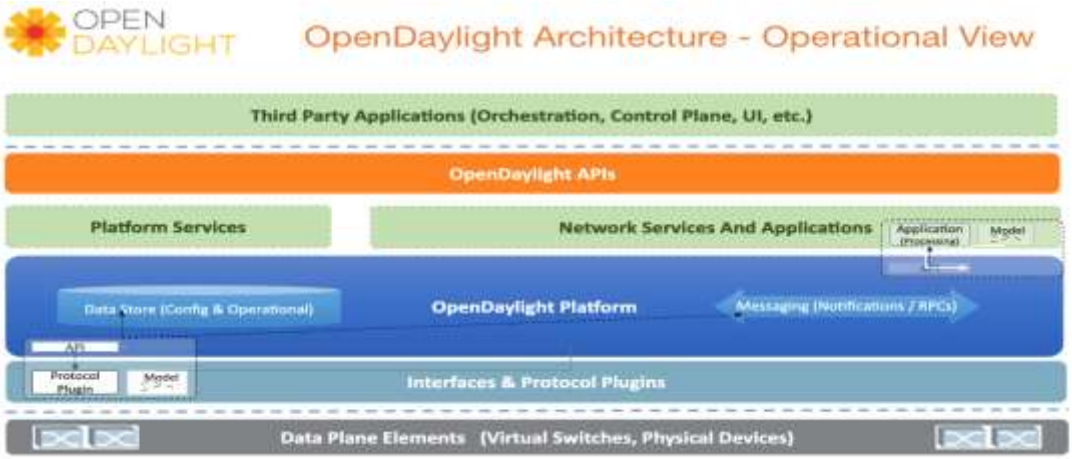
الشكل (8) نموذج إنشاء الاتصال بين المتحكم والمبدل وفق بروتوكول OpenFlow.

5. المتحكم في شبكة SDN التي تستخدم بروتوكول التدفق المفتوح OpenFlow:

يعتبر المتحكم العقل الرئيسي المدبر في شبكة SDN، حيث يعتبر مسؤولا" عن اتخاذ القرارات فيما يتعلق بتوجيه الرزم، التي لا يوجد لها مداخل مطابقة في جداول التدفق الموجودة لدى المبدلات، كما يعتبر مسؤولا" عن كل عمليات إدارة الشبكة وتوزيع التدفقات والتجاوب مع الأحداث في الشبكة، مثل خروج عقد واتصال أو معالجة التزاحم وغيرها من

الأحداث في الشبكة، ويمكن للمبدل أن يقوم بإنشاء الاتصال مع متحكم وحيد أو مع عدة متحكمات [1]، ويعتبر وجود عدة متحكمات ضمن الشبكة أمرا" يزيد من وثوقيتها، حتى لو تعطلت إحدى المتحكمات فإن المبدل يستطيع الاستمرار بالعمل مع باقي المتحكم بنمط OF، وعندما يتم تهيئة OF فإن المبدل ينشيء الاتصال مع كل المتحكمات بداية ويعلمهم بتعريفاته، قد ترسل العديد من المتحكمات أوامر من المتحكم إلى المبدل ، لكن الاستجابة أو الأخطاء المتعلقة بهذه الأوامر يجب أن ترسل فقط على الوصلة مع المتحكم المتعلق بهذا الأمر [2].

5-1 بنية المتحكم OpenDaylight: يعد المتحكم OpenDaylight العقل المدبر لشبكة SDN، حيث يقوم بمراقبة الشبكة وإدارتها وتوليد قواعد التوجيه المناسبة لكل حالة من حالات الشبكة لكي يرسلها إلى المبدلات المناسبة، والتي تقوم بتخزينها ضمن جداول التدفق الخاصة به، [15] وبناء على الاستعلامات التي يطلبها من المبدلات يقوم بتعديل قواعد النفاذ أو تغيير جداول التدفق، حيث تختلف عملية إدارة الشبكة في SDN عن الشبكات التقليدية، بأن شبكات SDN تجعل الإدارة مركزية في المتحكم، وبالتالي فإن المبدلات لاتعرف شيئا" عن بقية الشبكة وتعريفاتها، فقط تتلقى الأوامر من المتحكم، لذلك عند تغيير الطبوبوجيا أو عند وجود أخطاء سيكون هناك تأخيرا إضافيا" في معالجة الخطأ، لأن المبدلات ستنتظر أن يقوم المتحكم بحساب الطبولوجيا الجديدة، ومن ثم اختبار الخصائص، لذلك نسعى في هذه الدراسة إلى تقليل زمن معالجة الطلب وتخفيف الحمل على المتحكمODL، حيث أنّ سرعة المعالجة تكون أكبر مع زيادة عدد متحكمات ODL في شبكة SDN كما أنّ فشل أحد متحكمات ODL لايؤدي إلى توقف عمل الشبكة، حيث يمكن لمتحكم آخر ODL أن يقوم بدوره وبالتالي زيادة التوافرية، مع العلم أن المتحكم OpenDaylight يعتمد على طبقة الخدمات، والتي تقوم بوصل إضافات البروتوكولات الموجودة في الواجهة الجنوبية مع الخدمات في الواجهة الشمالية [16]، تتواصل طبقة التطبيقات مع الأجهزة باستخدام وحدات (modules) مختلفة كما يبينه الشكل (9) :

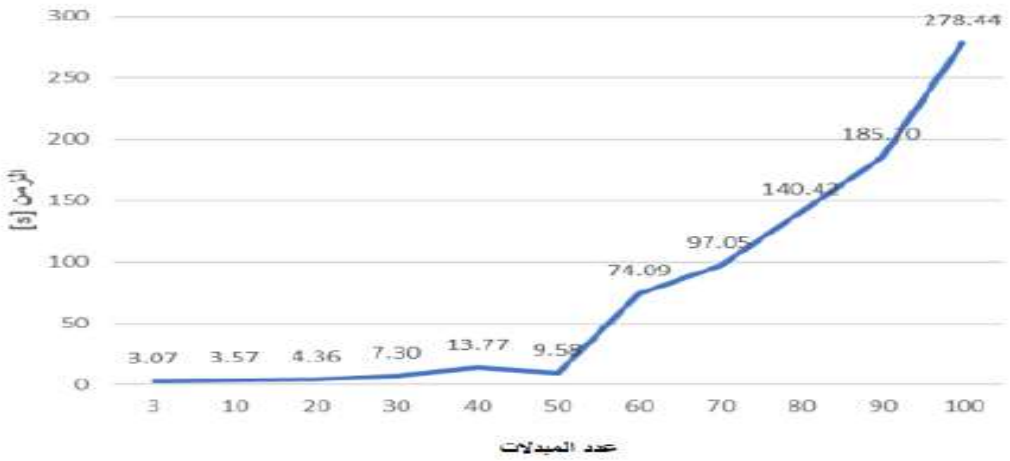


الشكل (9) بنية المتحكم OpenDaylight.

6- كشف طبولوجيا شبكة SDN من قبل المتحكم OpenDaylight :

اكتشاف عناصر شبكة SDN يجري بتبادل مجموعة من رسائل التدفق المفتوح Hello بين المتحكم OpenDaylight وكل عنصر، بينما يجري اكتشاف الوصلات بوساطة تبادل رسائل البروتوكول LLDP [16]، يقوم المتحكم OpenDaylight بإرسال رسائل تدفق مفتوح من نوع رزم صادرة Packet out لكل منفذ Port على المبدل، حيث

تكون التعليمات في هذه الرسالة هي إرسال LLDP على كل منفذ من منافذ المبدل، ويقوم كل مبدل بإرسال رسائل تدفق مفتوح من نوع رزم واردة Packet in إلى المتحكم، وبناء عليه يقوم المتحكم OpenDaylight بمعرفة طبولوجيا الشبكة كما يبينه الشكل(10)، ونلاحظ أن الزمن اللازم لكشف شكل الشبكة من قبل المتحكم OpenDaylight، يزداد بازدياد عناصر الشبكة ووصلاتها بشكل بطيء عند عدد مبدلات 50 بعد هذه العتبة ثم يزداد بصورة سريعة.



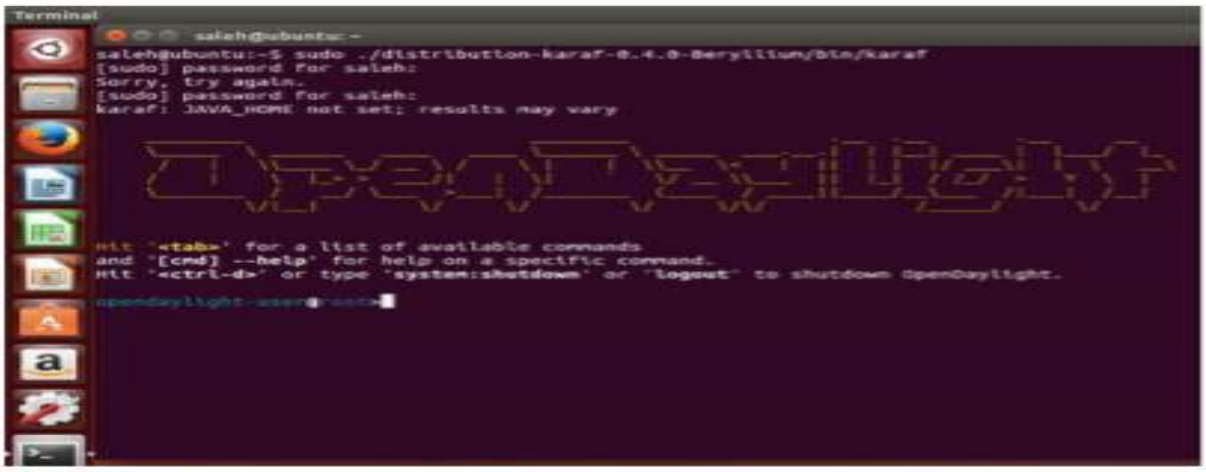
الشكل (10) زمن كشف طبولوجيا شبكة SDN من قبل OpenDaylight.

النتائج والمناقشة:

لقد لاحظنا أن جميع الدراسات السابقة فيما يخص قياس معاملات الأداء، لم تأخذ بعين الاعتبار سوى وجود وحدة تحكم واحدة في طبولوجيا الشبكة SDN، تقوم عقدة تحكم مركزية واحدة OpenDaylight بإرسال معلومات مثل معلومات التوجيه وغير ذلك إلى كافة أجهزة الشبكة، لذلك نسعى في هذا البحث إلى تحسين أداء الشبكة، وتقليل زمن معالجة الخطأ الذي يقوم به المتحكم المركزي الوحيد ODL، الذي يعني وجود نقطة واحدة للإدارة ويؤدي إلى العديد من التحديات، منها قدرة تحمل وزمن استجابة أكبر، بالإضافة إلى تحديات جوهرية في المرونة وخاصة بالنسبة للشبكات الكبيرة، حيث أن وضع كل الوظائف ضمن متحكم واحد يتطلب طاقة حسابية ومساحة تخزينية كبيرة ، لذلك نعتمد في هذا البحث على دراسة أداء متحكم ODL وحيد وأداء عدة متحكمات ODL، وتهيئتها لتقوم بتشارك الحمل بالتساوي، وذلك لإثبات أن زيادة عدد المتحكمات سيقلل من التأخير في الشبكة، ومن عدد الرزم التي يتم إسقاطها، وبالتالي سيحسن بشكل كبير من جودة الخدمة المقدمة ومن أداء الشبكة ككل، ولإجراء المحاكاة العملية قمنا بتنزيل نسخة من Virtual machine (virtual box) ، وإنشاء آلتان افتراضيتان منفصلتان من Ubuntu 14.4 باستخدام VMware [17]، الأولى اسمها OpenDaylight ونقوم بتحميل أحدث نسخة من OpenDaylight فيها من الموقع www.opendaylight.org/downloads.،أما الآلة الإفتراضية الثانية Mininet-VM يمكن تحميلها من الموقع www.mininet.org.لإجراء محاكاة لطبولوجيا الشبكة باستخدام المحاكي Mininet كما يبينه الشكل(11):

الشكل (11) إنشاء آلتان إفتراضيتان باستخدام virtual box .

وكما نعلم أن المتحكم OpenDaylight مكتوب بلغة الجافا Java، لذلك يجب تنزيل بيئة java run-time من خلال التعليمتان التاليتان:sudo apt-get install openjdk-7-jdk ، sudo apt-get install openjdk-7- jre ، ثم نقوم بكتابة التعليمات التالية في بيئة Ubuntu Terminal لتشغيل المتحكم OpenDaylight كما يبينه الشكل التالي (12) : 1)*cd ODL/bin* ، 2) *-/ODL/bin$ ./karaf-of13*:



الشكل (12) تشغيل المتحكم OpenDaylight SDN Controller.

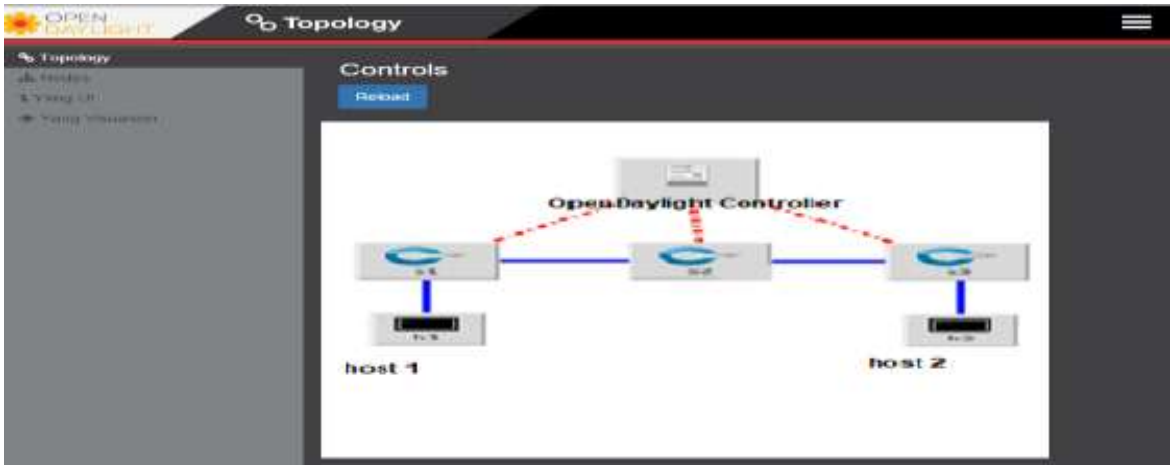
قبل أن نقوم بفتح المتحكم ODL، نقوم بإضافة بعض الميزات له مع العلم أنه بالوضع الإفتراضي لايحوي أية ميزات يجيب أن نقوم بإضافتها له من خلال التعليمات التالية 1) *>feature* ،2) *>feature:install ODL-l2switch-switch-ui* ، 3) *>feature:install ODL-dlux-all*

كما نقوم بإضافة الحزمة bw إلى المسار ODL/ext، حيث تقوم هذه الحزمة بحساب عرض النطاق الترددي المستخدم لكل وصلة كل ثانية، كما نقوم بإضافة الحزمة bellman إلى المسار ODL/ext، حيث تقوم بأعمال القياس والمراقبة، وبعدها نقوم باستدعاء وحدة التحكم ODL من خلال المتصفح Firefox حيث يدعم المتحكم ODL واجهة

web GUI، ونقوم بتشغيل المتحكم عن طريق إدخال الرابط التالي في المتصفح http://192.168.42.137:8181/index.html#/topology كما يبينه الشكل(13)،ومن ثم وضع اسم للمستخدم username وكلمة المرور password هما admin لكل منها :

الشكل (13) واجهة الدخول إلى المتحكم OpenDaylight SDN Controller

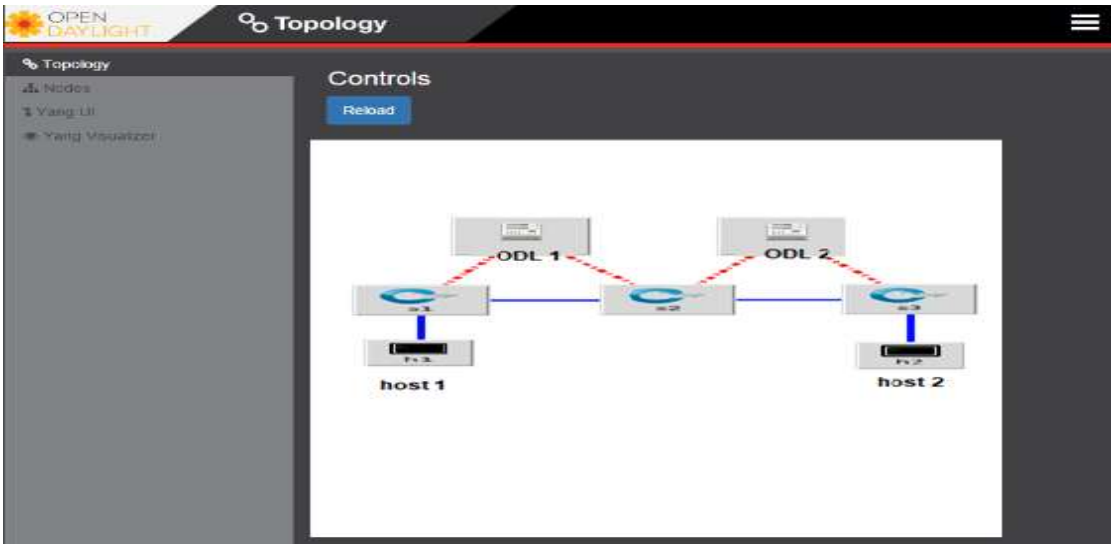
ثم نقوم أولا" بتشغيل الآلة الإفتراضية Mininet-VM التي تم إنشاؤها في الخطوات السابقة، عن طريق إدخال كلمة السر واسم للمستخدم هما mininet، ونقوم ثانيا" بإنشاء طبولوجيا الشبكة المدروسة، باستخدام المحاكي Mininet على الآلة الإفتراضية Mininet-VM ،مع العلم أن طبولوجيا الشبكة للسيناريو الأول كما يبينه الشكل(14)، تتألف من ثلاثة مبدلات OpenFlow متصلة إلى وحدة تحكم مركزية واحدة ODL .



الشكل (14) طبولوجيا السيناريو الأول للشبكة المدروسة باستخدام Mininet

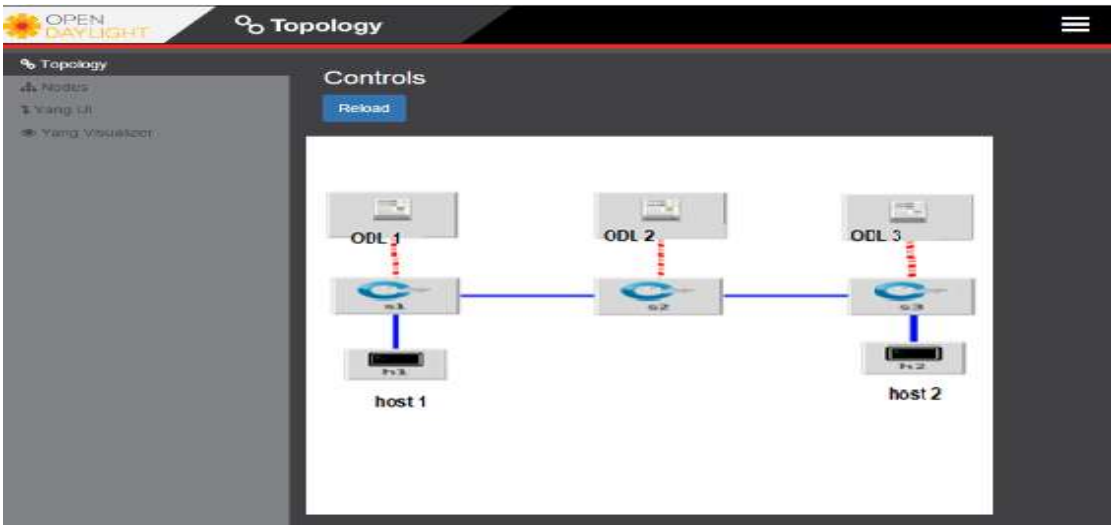
يملك المتحكم ODL العنوان 192.168.56.101، مع رقم منفذ port=6633 ومضيفان host1(h1:10.0.0.1) و host2(h2:10.0.0.2)، بينما طبولوجيا الشبكة للسيناريو الثاني تتألف من ثلاثة مبدلات Openflow ، ومتحكمين OpenDaylight(ODL1 & ODL2)، ومضيفان host1(h1:10.0.01) و host2(h2:10.0.0.2)، كما يبينه الشكل (15) ، حيث يتصل المبدل الأول والثاني مع المتحكم الأول ODL ذو

العنوان 192.168.56.101 مع رقم منفذ port=6633 ، ويتصل المبدل الثاني والثالث مع المتحكم ODL الثاني ذو العنوان 192.168.56.102 مع رقم منفذ port=6634 .



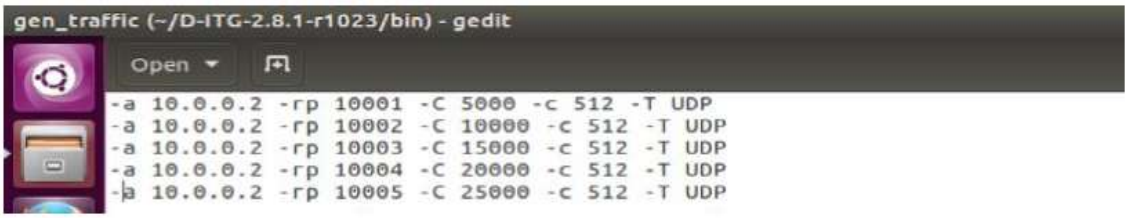
الشكل (15) طبولوجيا السيناريو الثاني للشبكة المدروسة باستخدام Mininet

أما طبولوجيا الشبكة للسيناريو الثالث تتألف من ثلاثة مبدلات Openflow وثلاثة متحكمات OpenDaylight ومضيفان host1(h1:10.0.0.01) و host2(h2:10.0.0.2)، حيث يتصل المبدل الأول مع المتحكم ODL الأول ذو العنوان 192.168.56.101 مع رقم منفذ port=6633 ، والمبدل الثاني يتصل مع المتحكم ODL الثاني ذو العنوان 192.168.56.102 مع رقم منفذ port=6634 ، وكما يبينه الشكل (16) المبدل الثالث يتصل مع المتحكم الثالث ODL ذو العنوان 192.168.56.103 مع رقم منفذ port=6635 .



الشكل (16) طبولوجيا السيناريو الثالث للشبكة المدروسة باستخدام Mininet.

بعد أن تم تأسيس الاتصال بين المبدلات والمتحكمات في السيناريوهات الثلاثة المبنية باستخدام المحاكي Mininet، يتم الاستعانة بالأداة D-ITG لتوليد عدد من التدفقات قمنا بتوليد 5 تدفقات مختلفة لمدة 15sec كما يبينه الشكل(17)، كل تدفق ينتج عدد من الرزم التي تم توليدها وهي : 5000 ،10000، 15000 ، 20000، 25000 رزمة لكل تدفق على التوالي:



الشكل (17) ملف التدفقات المولدة باستخدام الأداة D-ITG

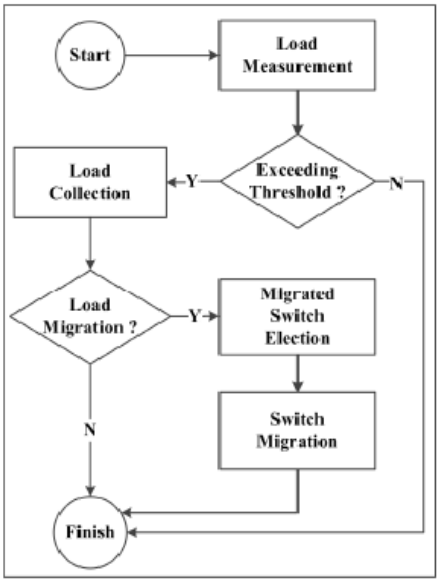
ولتقييم أداء الشبكة من خلال دراسة عدد من المحددات، التي تعتبر من المقاييس الرئيسية التي تؤثر على أداء الشبكة، مثل محدد التأخير الذي يمثل الفرق بين زمن الإرسال وزمن الاستقبال لكل رزمة مولدة، وأيضا" محدد الإنتاجية وعدد الرزم التي تم إسقاطها، وسنقوم بمقارنة السيناريوهات الثلاثة في حال وجود متحكم OpenDaylight وحيد وفي حال وجود عدة متحكماتODL وتحليل النتائج في كل سيناريو، ولدراسة التدفقات التي تم توليدها باستخدام الأداة D-ITG، نقوم بقياس معامل تحليل الأداء متوسط قيم التأخير لكل من السيناريوهات الثلاثة وتسجيها في الجدول(1):

Packets	Average delay(s) 1 Open Daylight controller	Average delay(s) 2 Open Daylight controllers	Average delay(s) 3 Open Daylight controllers
5000	7.638484	0.288012	**0.141019**
10000	6.582452	3.571308	**1.504948**
15000	5.536829	4.449001	**1.341814**
20000	5.568706	3.030572	**1.159171**
25000	6.135001	0.120633	**0.008946**

جدول (1) متوسط قيم التأخير لكل التدفقات من أجل السيناريوهات الثلاثة.

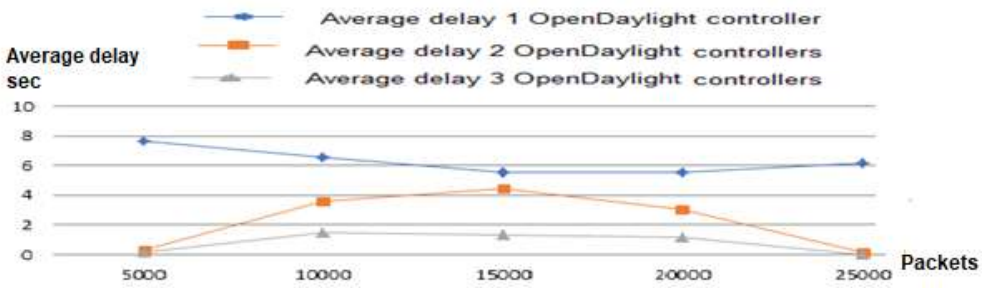
نلاحظ من القيم الموجودة في الجدول السابق أن متوسط قيم التأخير تتناقص مع ازدياد عدد المتحكماتODL، حيث نلاحظ في السيناريو الثاني من أجل التدفق الأول الذي يولد 5000 رزمة، كان متوسط قيم التأخير 0.288012، بينما في السيناريو الأول من أجل نفس التدفق كانت متوسط قيم التأخير 7.638484، نلاحظ أن قيمة التأخير من أجل السيناريو الثاني الذي يحوي متحكمين ODL أقل منها من أجل السيناريو الأول في حالة وجود متحكم ODL وحيد، بينما من أجل السيناريو الثالث الذي يحوي ثلاثة متحكمات ODL نلاحظ أن قيمة التأخير قلت بشكل كبير وملحوظ أصبحت 0.141019، وهذا مايوضحه المخطط البياني(20)، ولاحظنا استجابة أسرع في معالجة البيانات وبالتالي تأخير أقل، لانه يتم تقسيم الحمل بين المتحكمات ODL التي تحويها الشبكة، بسبب خوارزمية موازنة الحمل التي قام بها العاملون في البحث [19] والتي تستخدمها الأداة D-ITG في توليد المرور والتدفقات في الشبكة [20]، ويوضح تابع موازنة الحمل مبدأ عمل الخوارزمية والمخطط التدفقي كما في الشكل (18) :

```
# Load Balancer Function
def loadbalance():
        linkURL = "http://localhost:8080/wm/topology/links/json"
        fetchResponse(linkURL,"Switchlinkinfo")
        computeRoute()
        url=('http://localhost:8080/wm/core/switch/all/flow/json')
        fetchResponse(url,"getswitchlatency")
        getlinkLatency()
        fetchLinkCost()
        addFlow()
```



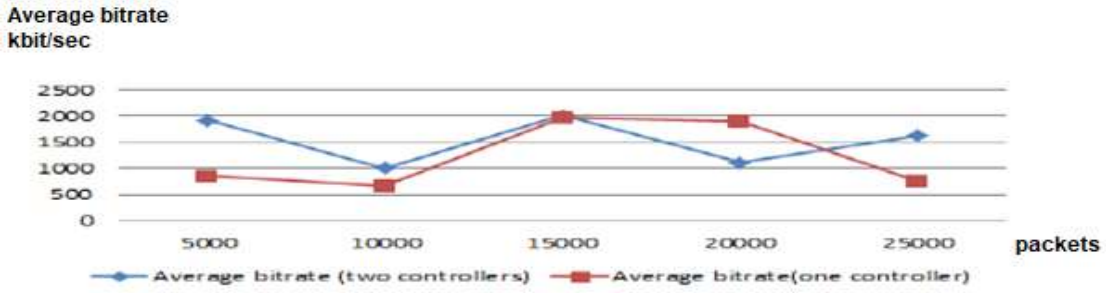
الشكل(18) تابع موازنة الحمل مع المخطط التدفقي للخوارزمية المستخدمة من قبل الأداة D-ITG [20] [19] .

تعمل هذه الخوارزمية [19] التي درسها الباحث Yuanhao مع مجموعته بشكل فعال عندما تحوي شبكة SDN المدروسة متحكمين فما فوق، وتتعاون المتحكمات مع بعضها البعض للحفاظ على توازن الحمل بينها، في البداية عندما يتم تنفيذ المحاكاة يتم تشغيل الخوارزمية لتأخذ معلومات التحكم عن الحمل في كل متحكم، وتتحقق فيما إذا كان الحمل يتجاوز العتبة، فإذا كان الحمل يتجاوز العتبة، سوف يطلب المتحكم ODL معلومات عن حالة الحمل لدى جميع المتحكمات ODL الموجودة معه في الشبكة المدروسة، عن طريق إرسال رسائل تحكم في الخلفية إلى المبدل، بعد تبادل معلومات عن جميع حالات الحمل لدى جميع المتحكمات ODL من قبل المتحكم ذو الحمل الأعلى في الشبكة، ويتم ترتيب المتحكمات حسب الحمل الأعلى تنازليا"، يقوم المتحكم ذو الحمل الأعلى بإرسال رسالة إلى الـ switch ليتم اتخاذ القرار من قبله بإرسال رسالة إلى المتحكم ODL ذو الحمل الأقل في الشبكة، ثم يتم ضبط العتبة بقيمة جديدة مع العلم أن قيمة العتبة غير ثابتة، ويتم ضبطها بشكل دوري وأتوماتيكي وكل مرة تأخذ قيمة جديدة بناء على طبولوجيا الشبكة المدروسة وعدد المتحكمات ODL الموجودة فيها والحمل المطبق عليها، ونستنتج أنه كلما ازداد عدد المتحكمات كلما قل التأخير وتحسن أداء الشبكة ووجودة الخدمة المقدمة، كما يبينه المخطط البياني (19)، حيث يمثل المحور الأفقي عدد الرزم المولدة من كل تدفق كل ثانية، ويمثل المحور العمودي قيم متوسط التأخير مقدرة بالsec.



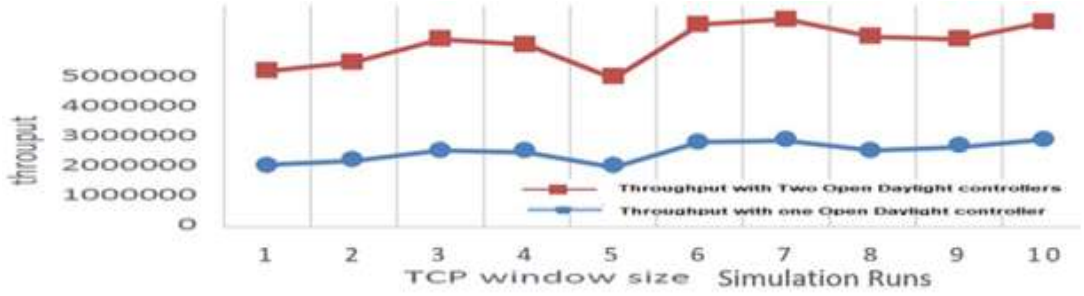
الشكل (19) متوسط قيم التأخير من أجل السيناريوهات الثلاثة.

وكما نعلم أنه كلما كانت كمية البيانات المنقولة في الشبكة المدروسة كل ثانية أكبر، كلما كان ذلك يعبر عن سلوك أفضل للشبكة، وبالتالي بمقارنة قيم متوسط معدل البتات من أجل السيناريوهات الثلاثة، نجد أن قيم معدل البتات بتقسيم الحمل على متحكمين ODL في الشبكة، هي أكبر بكثير من قيم معدل البتات بوجود متحكم واحدODL، ومن أجل السيناريو الثالث نجد أن قيم معدل البتات أعلى بكثير من حالة وجود متحكمين أو متحكم واحد ODL، وبالتالي كمية الـtraffic في الشبكة تزداد، وهذا يحسن من أداء الشبكة بشكل كبير، بسبب توزيع الحمل على أكثر من متحكم، كما يبينه الشكل(20) يمثل المحور الأفقي عدد الرزم المولدة من كل تدفق كل ثانية، ويمثل المحور العمودي قيم متوسط معدل البتات مقدرة kbit/sec.



الشكل (20) قيم متوسط معدل البتات .

ولتحليل معامل قياس الأداء الإنتاجية، نلاحظ في حالة وجود متحكم وحيد Open daylight تكون الإنتاجية أقل بحوالي 35% بالنسبة للسيناريو الأول، وفي حالة وجود متحكمين تكون 25% بالنسبة للسيناريو الثاني، ويعود ذلك إلى أن معالجة البيانات تتم بتشارك الحمل وتوزيعه على المتحكمين OpenDaylight، وهذا يعطي استجابة ومعالجة أسرع وتأخير أقل، وكذلك الأمر بالنسبة للسيناريو الثالث ، كما يبينه الشكل (21).



الشكل (21) الإنتاجية لحالة وجود متحكم وحيد OpenDaylight، وحالة وجود متحكمين ODL 2 .

وكما نعلم أنه كلما كانت خسارة الرزم أقل، كلما زادت إنتاجية الشبكة وجودة الخدمة المقدمة، لذلك لابد من دراسة محدد عدد الرزم المفقودة، فبعد تنفيذ المحاكاة على السيناريوهات الثلاثة، نجد من خلال النتائج أنه من أجل السيناريو الأول الذي يحوي متحكم Open Daylight وحيد يكون عدد الرزم المفقودة أكبر، وبالتالي كمية البيانات المنقولة والتي تقدر بال Kbit/sec تكون أقل، مقارنة مع السيناريو الثاني الذي يحوي متحكمين ODL، وأيضا" بالمقارنة مع السيناريو الثالث يكون عدد الرزم التي تمت خسارتها شبه معدوم packet dropped 0، وبالتالي كمية الـtraffic في

الشبكة أكبر ويؤدي بدوره إلى زيادة إنتاجية الشبكة وأداء أفضل للشبكة المدروسة، والمخطط البياني(22) الذي يمثل العلاقة بين عدد الرزم التي تم توليدها ويمثله المحور الأفقي، وبين عدد الرزم المفقودة والتي تم إسقاطها ويمثله المحور الرأسي ، يؤكد أنه كلما كان عدد المتحكمات أكبر كلما كان عدد الرزم التي تمت خسارتها شبه معدوم وهذا ينعكس بشكل إيجابي على أداء الشبكة ككل وجودة الخدمة المقدمة .

الشكل(22) الرزم المفقود من أجل السيناريوهات الثلاثة (1 ODL, 2 ODL , 3 ODL).

الاستنتاجات والتوصيات:

تم في هذا البحث دراسة أداء المتحكم OpenDaylight في الشبكات المعرفة برمجيا" SDN ، وكما نعلم عند تصميم الشبكات المعرفة برمجيا" يجب اختيار المتحكم بعناية ، لذلك اخترنا المتحكم OpenDaylight كونه يعتبر من المتحكمات الشهيرة ومصمم ليناسب بيئة مراكز البيانات الموزعة، وتم تقييم وتحليل أداؤه اعتمادا" على عدة بارامترات باستخدام المحاكي Mininet وأداة توليد المرور والتدفقات للرزم D-ITG، بعد أن تم بناء طبولوجيا الشبكة لثلاثة سيناريوهات، ولاحظنا من خلال تحليل النتائج أن أداء الشبكة يزداد مع ازدياد عدد متحكمات OpenDaylight بشكل كبير كما يزداد الtraffic وجود الخدمة المقدمة في شبكة SDN المدروسة.

و يمكن مستقبلا" دراسة الجانب الأمني لتحسين أداء شبكة SDN ، كما يمكن تحسين تصميم وحدة التحكم OpenDaylight من أجل قابلية التوسع والمرونة والتوافرية، كون هذه الوحدة ODL مفتوحة المصدر Open Source وتقبل التعديل في لغتها البرمجية المكتوبة بالجافا، كما يمكن تطبيق خوارزميات الجدولة وخوارزمية Dijkstra وخوارزمية اختيار المسار الأقصر ليتم نقل البيانات عبره للحصول على استجابة سريعة وزمن أقل، وذلك بهدف تحسين أداء الشبكات المعرفة برمجيا" SDN وجودة الخدمة باستخدام المحاكي Mininet وبالاستعانة بالأداة D-ITG .

[1] Syed Abdullah Shah, Jannet Faiz, Maham Farooq, Aamir Shafi, and Syed Akbar Mehdi. "*An architectural evaluation of SDN controllers*." In Communications (ICC), 2013 IEEE International Conference on, pp. 3504-3508. IEEE, 2018.

[2] David Erickson. " *The beacon openflow controller*." In Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking, pp. 13-18. ACM, 2017.

[3] K. Poularakis, G. Iosifidis and L. Tassiulas, "*SDN-Enabled Tactical Networks: Extending Programmable Control to the Edge*," in IEEE Communications Magazine, vol. 56, no. 7, pp. 132-138, July 2018..

[4] D. B. Hoang and M. Pham, "*On software-defined networking and the design of sdn controllers*," in Network of the Future (NOF), 2015 6th International Conference on the. IEEE, 2017, pp. 1–3.
[5] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "*Software-defined networking: A comprehensive survey*," Proceedings of the IEEE, vol. 103, no. 1, pp. 14–76, 2017.
[6] ROUT.S, SAHOO.B, PATRA.S, (2017), *" Performance Evaluation of the Controller in Software Defined Networking". In Computational Intelligence in Data Mining*, Vol. 556, Springer.
[7] HEMID A, *"Software Defined Networking, Proceedings of CSCUBS*", 6 P. https://tools.ietf.org/html/draft-ietf-bmwg-sdn controllerbenchmar kmeth-09-, 2018.
[8] KIM,H.,SANTOS,J,R.,TURNER,Y., CORONET:" *FaultTolerance for Software Defined* Networks",IEEE.PP:1-2,2017.
[9] Ivan Grgurevic, Zvonko Kavran, Anthony Pušeljic(student) Faculty of Transport and Traffic Sciences ,*" Simulation Analysis of Characteristics and Application of Software-Defined Networks*", University of Zagreb D epartment of Information and Communication Traffic Zagreb, Croatia ivan.grgurevic@fpz.hr, Publishing Institution of the University of Zilina,2017. 25-31.
[10] Raeisi, *"Software-based Fast Failure Recovery in Load Balanced SDN-based Datacenter Networks"*. IEEE, 2018.
[11] Diyar Jamal Hamad , Khirota Gorgees Yalda, and Ibrahim Tanner Okumus, *"Getting traffic statistics from network devices in an SDN environment using OpenFlow*",2018.
[12] Bruno NunesAstuto, Marc Mendonça, Xuan Nam Nguyen, Katia Obraczka, ThierryTurletti, "*A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks*" ,Communications Surveys and Tutorials, IEEE Communications Society, Institute of Electrical and Electronics Engineers, 2017, 16 (3), pp.1617 - 1634.
[13] GENG LI. Basic network flows;" *OpenFlow as a datapath programming standard*", 9 October.2017. http://zoo.cs.yale.edu/classes/cs434/cs434-2017spring /lectures/02-prognet-openflow.pdf.
[14] Kotani, D.; Suzuki, K.; Shimonishi, H." *A Design and Implementation of OpenFlow Controller handling IP Multicast with Fast Tree Switching*". In Proceedings of the IEEE/IPSJ International Symposium on Applications and the Internet (SAINT), 16–July 2016 ;pp. 60–67.
[15] OpenDaylight New Project Proposals,"https://wiki.opendaylight.org/view/Project_Proposals:Main#Proposals_Submitted_For_Review.
[16]] OpenDaylight Consortium ,(2019). http://www.opendaylight.org.
[17] "VMWare NSX – the platform for network virtualization, (2019). http://www.vmware.com/files/pdf/products/nsx/VMware-NSXDatasheet.Pdf.
[18] Mininet Overview - Mininet. http://mininet.org/overview/. Accessed 26 Apr 2018.
[19] Yuanhao Z., Limin X., Wenbo D., Deguo L.," *A Load Balancing Strategy for SDN Controller based on Distributed Decision*",". In Proceedings of the IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications,2014.
[20] A. Botta, W. Donato and A. Dainorri, "*D-ITG 2.8.1 Manual*", Computer for Interaction and Communications (COMICS) Group, pp. 3-6, 2018.