Jurnal Jaringan Komputer dan Keamanan
JJKK, Vol. 01, No. 01, Februari 2020: 1-13
Received: 5 Des 2019; Revised: 2 Jan 2019; Accepted: 3 Feb 2020

Arsitektur Software Defined Network: Implementasi Pada Small Network

Intraswadaya Hidayat, Bima Abdi Perdana

Program Magister Teknik Informatika
Universitas Bina Darma
email : intraswadaya@student.binadarma.ac.id,
bimaabdiperdana@student.binadarma.ac.id
Jl. A. Yani No. 12, Palembang 30624, Indonesia

Abstract

Software Defined Network and Openflow protocols are currently growing rapidly. But its use in small networks and home offices is still very rare. This article discusses the use of Software Defined Network architecture in small and home office networks so that it can be developed so that the SDN architecture can be an alternative in network development that requires complex network management but is limited by the cost of infrastructure development.

Kata kunci: *Software Define Network, Arsitektur, Small Network, Protocol, Framework*

Abstrak

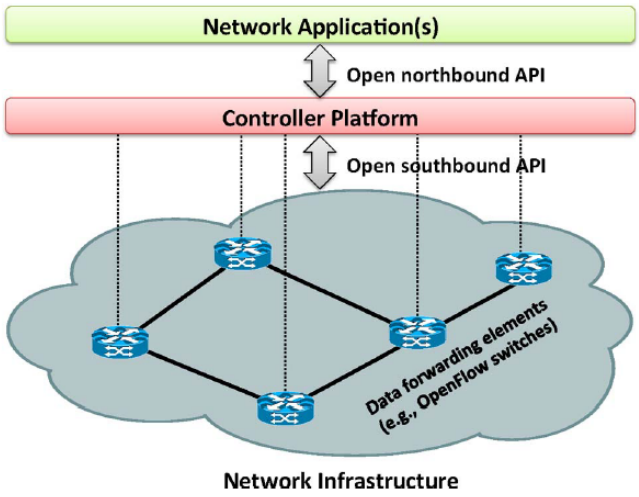
Software Defined Network dan protokol Openflow pada saat ini masih terus berkembang dengan pesat. Namun pemanfaatannya pada jaringan small dan home office masih sangat jarang. Pada artikel ini dibahas tentang penggunaan arsitektur Software Defined Network pada jaringan small dan home office agar dapat dikembangkan sehingga arsitektur SDN dapat menjadi alternatif dalam pengembangan jaringan yang membutuhkan manajemen jaringan yang kompleks namun terbatasi oleh biaya pengembangan infrastruktur.

Kata kunci: *Software Define Network, Arsitektur, Small Network, Protocol, Framework*

1 PENDAHULUAN

Sebuah jaringan komputer biasanya dibangun dari beberapa perangkat jaringan seperti, router, switch dan perangkat lainnya yang mempunyai berbagai macam protokol yang tertanam pada perangkat tersebut. Perangkat tersebut biasanya dikonfigurasi oleh admin jaringan

dengan berbagai kebijakan yang disesuaikan untuk pengguna, (Negara, E.S., 2019). Teknologi jaringan pada saat ini memiliki beberapa keterbatasan diantaranya kebijakan (Network Policy) yang selalu berubah-ubah, sulit untuk diukur, kebatasan terhadap vendor-vendor tertentu, (Negara, E.S. and Andryani, R., 2014). Untuk mengatasi beberapa permasalahan tersebut, maka industri jaringan menciptakan sebuah arsitektur baru yaitu arsitektur yang disebut dengan nama Software Defined Network. Software Defined Network (SDN) adalah sebuah konsep baru dalam mendesain, mengelola dan mengimplementasikan jaringan, terutama. Konsep ini memungkinkan kita untuk berinovasi dalam penelitian dan pengembangan jaringan sehingga kita bisa memenuhi kebutuhan jaringan yang semakin lama semakin kompleks. Berikut ini adalah gambar dari arsitektur SDN.



Gambar 1: Arsitektur software defined network

Dari Gambar 1 Controler Platform melakukan kontrol langsung atas keadaan pada Data Forwarding Elements melalui API (Application Programming Interface). Pada kasus ini Data Forwarding Elements yang dipakai adalah Openflow Swiches. Openflow adalah protokol yang tergolong baru yang dirancang di stanford university pada tahun 2008. Protokol ini bertujuan untuk mengontrol data plane switch yang telah dipisahkan secara fisik dari control plane menggunakan sebuah controller pada sebuah server. Control plane berkomunikasi dengan data plane melalui protokol openflow.

2 TINJAUAN PUSTAKA

2.1 Software Defined Network

Software-Defined Network (SDN) adalah sebuah paradigma arsitektur baru dalam bidang jaringankomputer, yang memiliki karakteristik dinamis, manageable, cost-effective, dan adaptable, sehingga sangat ideal untuk kebutuhan aplikasi saat ini yang bersifat dinamis. Arsitektur ini memisahkan antara network control dan fungsi forwarding, sehingga network control

tersebut menjadi directly programmable (dapat diprogram secara langsung), sedangkan infrastruktur yang mendasarinya dapat diabstraksikan untuk layer aplikasi dan network services, (Ummah, Izzatul, Desianto A, 2016).

Dalam arsitektur SDN terdapat dua komponen utama, yaitu Control Plane dan Data Plane.Control Plane adalah komponen pada jaringan yang berfungsi untuk mengontrol jaringan, yaitu konfigurasi sistem, manajemen jaringan, menentukan informasi routing table dan forwarding table. Data Plane merupakan komponen yang bertanggungjawab meneruskan paket, menguraikan header paket, mengatur QoS, dan enkapsulasi paket. SDN mempunyai potensi yang besar untuk mengubah cara sebuah jaringan beroperasi, dan OpenFlow telah dipuji sebagai "akar dari ide baru dalam dunia networking", (McKeown et-al, 2008). Tujuan utama dari SDN adalah untuk mencapai pengelolaan jaringan yang lebih baik dengan tingkatan dan kompleksitas yang besar serta memastikan bahwa semua keputusan dari sistem kontrol dibuat dari titik pusat (controller). SDN memperkenalkan suatu metode untuk meningkatkan tingkat abstraksi pada konfigurasi jaringan, menyediakan mekanisme yang secara otomatis bereaksi terhadap perubahan yang sering terjadi dan terus-menerus untuk jaringan, (Hyojoon, kim, Nick Feamster, 2013). Tujuan lainnya adalah mengatasi keterbatasan terhadap vendor dan kebutuhan akan hardware jaringan yang bisa dibilang memerlukan biaya yang tidak murah. Pada jurnal yang berjudul "Perancangan dan Simulasi Arsitektur Software-Defined Networking Berbasis OpenFlow dan OpenDaylight Controller Studi Kasus: STMIK AMIKOM Yogyakarta" yang ditulis oleh Tulungan (2015) disebutkan bahwa topologi jaringan tradisional dapat digantikan dengan arsitektur SDN apabila switch yang digunakan pada arsitektur tersebut telah mendukung protokol openflow sehingga arsitektur SDN dapat menjadi alternatif bagi small office dan home office untuk mengurangi biaya penggunaan hardware jaringan.

Mateo, M. (2009) mengatakan, Switch OpenFlow terdiri dari dua jenis, yang pertama adalah hardware-base switch, yang telah dijual secara komersial oleh beberapa vendor. Switch jenis ini telah memodifikasi hardware-nya, menggunakan TCAM dan menggunakan OS khusus untuk mengimplementasikan Flow-Table dan OpenFlow protokol. Jenis yang kedua adalah software-base switch yang menggunakan sistem UNIX / Linux untuk mengimplementasikan seluruh fungsi OpenFlow switch, (Mateo,P. Manuel, 2013).

Menurut McKeown, dkk., (2008) bahwa switch OpenFlow paling tidak memiliki 3 bagian yaitu (1) Sebuah tabel flow yang terhubung dengan sebuah aksi untuk setiap flow yang masuk. (2) Sebuah Secure channel yang mengkoneksikan antar switch dengan remote/eksternal proses (yang disebut kontroler). (3) Protokol OpenFlow, yang melakukan komunikasi antara switch dengan kontroler, (McKeown et-al, 2013).

Terdapat beberapa perbedaan mendasar dari switch komersial (switch biasa) dengan switch OpenFlow. Perbedaan tersebut menurut Chung Yik,EE., (2012), terlihat seperti pada Tabel 1, (Chung Yik,2012).

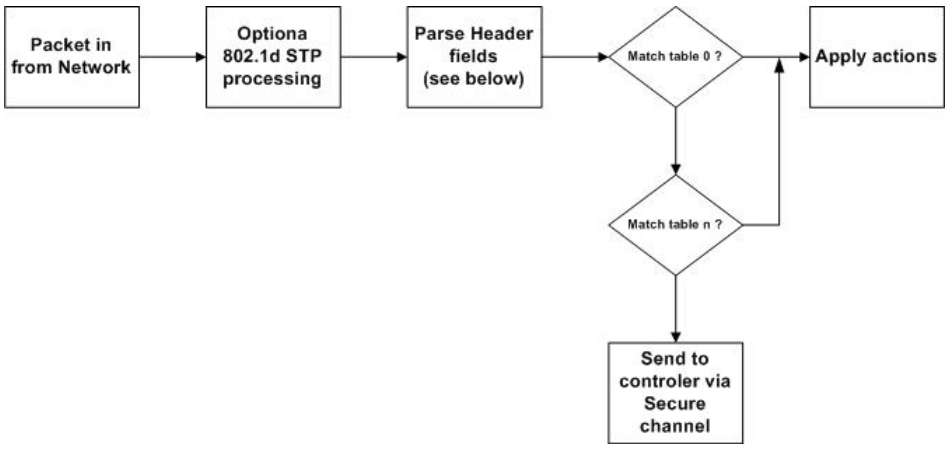
ONF (2009) menggambarkan proses pencocokan paket yang masuk kedalam switch seperti langkahpada gambar 3 dibawah ini.

Paket data yang masuk kedalam switch akan dicekfield header nya dan dicocokkan dengan tabel flow, bila cocok, paket tersebut akan diteruskan sesuai dengan action atau instruksi apa yang harus dilakukan dengan paket tersebut, namun bila tidak cocok pada tabel, akan dicocokkan lagi pada baris berikutnya hingga ke-n baris, bila hingga akhir dari tabel tidak ada kecocokan, maka paket akan dikirim ke kontroler menggunkan secure channel dan membiarkan

Table 1: Perbedaan switch OpenFlow dengan switch komersial(switch biasa) sumber: (Chung Yik, EE., 2012)

Switch OpenFlow	**Switch Komersial(Switch Biasa)**
Terpisahnya control path dan data path	Control Patah dan Data Path terletak pada perangkat yang tidak sama/ terpisah)
Memungkinkan terjadi invasi dalam jaringan	membatasi inovasi dalam jaringan
Menyediakan platform yang dapat diteliti dan diujicobakan pada jaringan sesungguhnya.	Tetap dan sulit untuk diujicobakan (dibuat tetap oleh vendor)
Fungsi yang dapat didefiniskan oleh user (Dapat di program ulang)	Arsitektur tertutup sehingga tidak dapat (Tidak dapat di program ulang)
Setiap keputusan untuk melakukan pengiriman dilakukan oleh kontroler	Mengirimkan semua paket yang diterima keluar dari switch

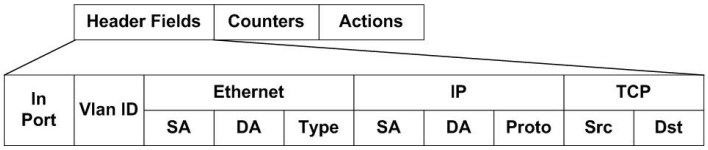
kontroler mengolah paket tersebut. Pencocokan paket akan dilakukan berdasarkan prioritas dari paket tersebut, paket yang tidak memiliki wildcat (biasanya di beri notasi '*') atau terdefinisi secara spesifik akan diprioritaskan, (Anonim, 2012).



Gambar 2: Pencocokan paket (sumber: ONF, 2009)

McKeown, N., dkk (2008) menyatakan bahwa Protokol OpenFlow adalah sebuah standard terbuka yang memungkinkan peneliti melakukan kontrol langsung pada jalannya paket

data yang akan di routing kan pada jaringan. OpenFlow melakukan sentralisasi terhadap kerumitan dari jaringan kedalam sebuah software kontroler, sehingga seorang administrator dapat mengaturnya dengan mudah, hanya dengan mengatur kontroler tersebut. McKeown, N., memperkenalkan konsep OpenFlow ini dengan ide awal adalah menjadikan sebuah network dapat di program/ dikontrol,(McKeown et-al, 2013).



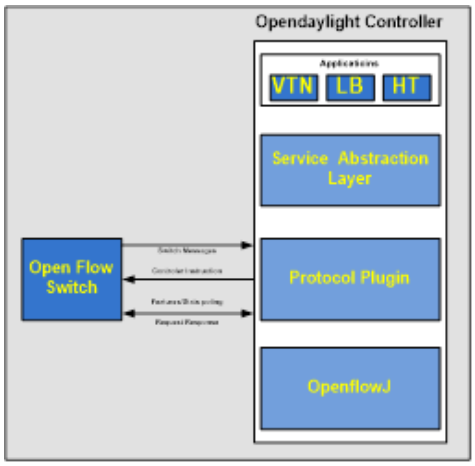
Gambar 3: Open Flow Table fields (sumber : Mateo M. P., 2009)

Pada Gambar 3, terlihat bahwa protokol OpenFlow terdiri dari 3 fields yaitu Header Fields, Counter dan Action. Mateo, M. P., (2009) menjelaskan, header fields adalah sebuah packet header yang mendefinisikan flow, fields nya terdiri dari enkapsulasi seperti enkapsulasi segmen pada protokol VLAN ethernet standard, Counter adalah sebuah fields yang menjaga jejak jumlah paket dan byte untuk setiap flow, dan waktu jejak paket terakhir cocok dengan flow (untuk membantu dalam membuang atau me-nonaktif-kan flow), dan Action adalah fields yang mendefinisikan bagaimana paket data akan diproses, (Mateo,P. Manuel, 2013).

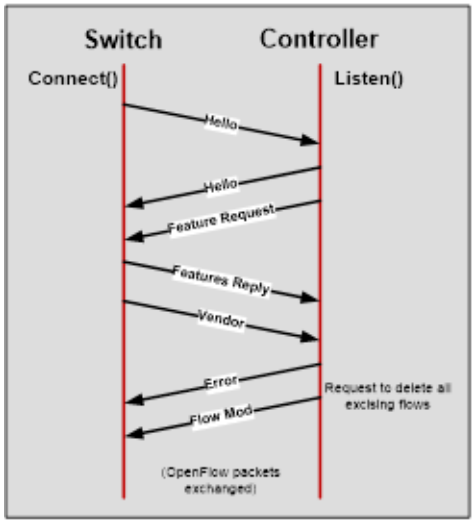
Menurut McKeown, N. (2008) Sebuah kontroler OpenFlow bertanggung jawab untuk menambahkan atau menghilangkan isi dari flow dari OpenFlow table yang ada didalam perangkat OpenFlow itu sendiri. Ada 2 tipe dari kontroler yaitu kontroler statis, dapat berupa perangkat yang dapat menambahkan atau menghilangkan flow dari flow table secara statis. Kontroler dinamis, secara dinamis memanipulasi isi dari flow sehingga cocok untuk beberapa konfigurasi. Tanggung jawab kontroler menurut (Vishnoi, A., et-al 2013), dan tergambarkan pada Gambar 4 adalah:

1. Menyediakan mekanisme untuk koneksi dan interaksi dengan underlyingplatform (dalam hal ini terhadap OpenFlow switch).
2. Menginterpretasikan pesan yang dikirim olehswitch OpenFlow.
3. Menyediakan semua instruksi pada setiap spesifikasi (baik spesifikasi yang dibutuhkan, tambahan atau keduanya) untuk dapat diprogram kedalam switch.
4. Memberikan mekanisme kepada switch untuk mendapatkan informasi/pendapat secara umum hingga ke yang detail dan harus dapat menginterpretasikan respon yang tepat, (Vinshoi, Anilkumar., Khumbhare, Abhijit, 2013).

Appleman, M., et-al (2012), ketika switch terhubung dengan sebuah kontroler, yang pertama dilakukan kontroler adalah mengirimkan Hello packet, kemudian switch mengirimkan Hello packetKembali, selanjutnya kontroler akan mengirimkan sebuah Features Request packet, switch harus membalas dengan sebuah Features Reply. Kemudian switch mengirimkan sebuah pesan Vendor, jika pesan Vendor tidak dimengerti oleh kontroler maka kontroler akan mengirimkan sebuah Error packet ke switch. Kontroler juga mengirimkan sebuah Flow Mod packet untuk menghapus semua flow dari switch. Proses lengkap dapat dilihat pada Gambar 5, (Applemen,M., De Boer,M 2013).



Gambar 4: Tanggung jawab kontroler (sumber: Vishnoi,A., et-al, 2013)



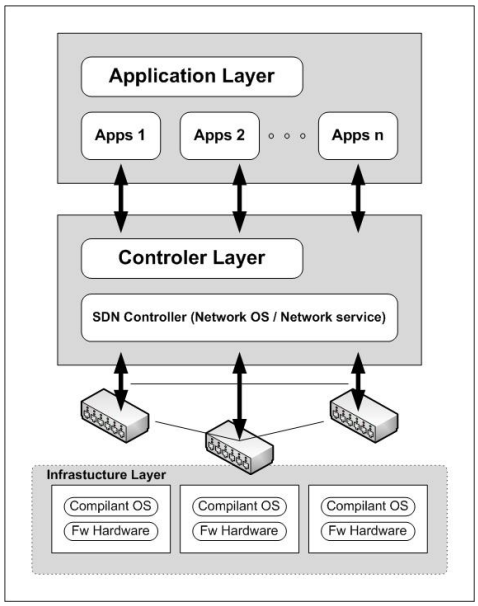
Gambar 5: Paket flow antara switch dan kontroler (sumber : Appleman, M., et-al, 2012)

2.2 Arsitektur Jaringan Tradisional dan Arsitektur SDN

Model arsitektur jaringan yang ada saat ini (traditional network) masih rumit karena masing-masing perangkat mempunyai konfigurasi yang berbeda yaitu control plane dan data forwarding plane tertanam pada perangkat masing-masing. Pada perangkat jaringan yang ada saat ini konfigurasi routing, firewall, DNS, NAT, dan IP Public nya berada di dalam perangkat tersebut.

Arsitektur SDN mempunyai perbedaan dengan arsitektur tradisional, dimana control plane dan forwarding plane nya dipisah dalam suatu perangkat yg berbeda seperti yang ditunjukkan pada Gambar 1. Dalam artian semua perangkat jaringan yang terhubung nantinya akan dikendalikan oleh application plane yang berkolaborasi dengan control plane untuk men-

ginstruksikan kemana suatu traffic atau paket data diarahkan (pada forwarding plane).



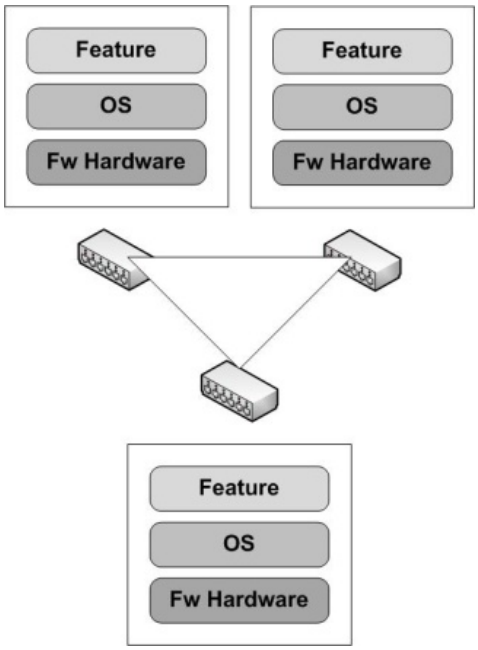
Gambar 6: Arsitektur SDN

Pada arsitektur tradisional setiap perangkat jaringan memiliki control plane-nya sendiri sehingga dapat membuat keputusan berdasarkan informasi yang dimiliki perangkat, sedangkan pada arsitektur SDN perangkat jaringan hanya memiliki kemampuan terbatas dalam membuat keputusan, dan dibutuhkan controller untuk membuat keputusan yang lebih kompleks. Arsitektrus tradisional dapat dilihat pada Gambar 7.

2.3 Implemtentasi Software Defined Network Dan Simulation Tools

SDN telah diusulkan untuk memfasilitasi evolusi dan inovasi dalam jaringan dengan memungkinkan pengembangan dari protokol dan layanan secara pesat. Berikut ini adalah beberapa tools untuk melakukan simulasi pada arsitektur Software Defined Network. Mininet adalah sebuah tools untuk memungkinkan seluruh jaringan Openflow diemulasikan dalam satu mesin, mempermudah dalam melakukan proses pengembangan, (Bob Lantz, 2010). Layanan baru, aplikasi, dan protokal dapat diuji coba dan dikembangan pada emulator sebelum diimplementasikan kedalam hardware. Mininet telah mendukung Openflow v.1.0. tapi dapat juga dimodifikasi agar mendukung versi terbaru. NS-3[11] Network Simulator juga dapat digunakan untuk melakukan simulasi yang mendukung protokol Openflow, tetapi versi sekarang hanya mendukung Openflow v0.89.

Beberapa penelitian telah dilakukan untuk meniliti bagaimana arsitektur Software defined network dapat digunakan pada sebuah jaringan pada perusahaan-perusahaan kecil. Pada saat ini perusahaan-perusahaan tersebut telah membutuhkan kebijakan jaringan yang semakin kompleks namun memiliki keterbatasan kepada hardware jaringan. Karena lingkungan perusahaan tersebut maka dibutuhkanlah sebuah perangkat jaringan yang memiliki biaya yang rendah. Sebuah jaringan yang mempunyai tingkat keamanan rendah kemungkinan men-



Gambar 7: Arsitektur Jaringan Tradisional

jadi sasaran atau host dari sebuah malware, sehingga menyebabkan sebuah gangguan pada jaringan yang membuat transaksi bisnis perusahaan tersebut dapat terganggu. Sedangkan pada home office akan sangat sulit apabila harus mempekerjakan seorang network administrator.

(Calvert et-al, 2011) menegaskan bahwa langkah pertama dalam mengatur jaringan rumahan adalah dengan cara mengetahui apa yang sebenarnya terjadi pada jaringan tersebut. Mereka menyarankan agar kontroler berfungsi sebagai "Home Network Data Recorder" untuk membuat logs yang akan digunakan untuk melakukan troubleshooting atau fungsi lainnya.

Feamster (2013), menyarankan agar jaringan harus beroperasi dengan cara "plug in and forget", yaitu dengan melakukan outsourcing untuk pengaturan jaringan kepada pihak ketiga sebagai tenaga ahli yang melakukan manajemen secara remote terhadap programmable switches dan melakukan monitoring terhadap jaringan dan algoritma yang digunakan untuk mendeteksi kemungkinan terjadinya masalah keamanan pada jaringan.

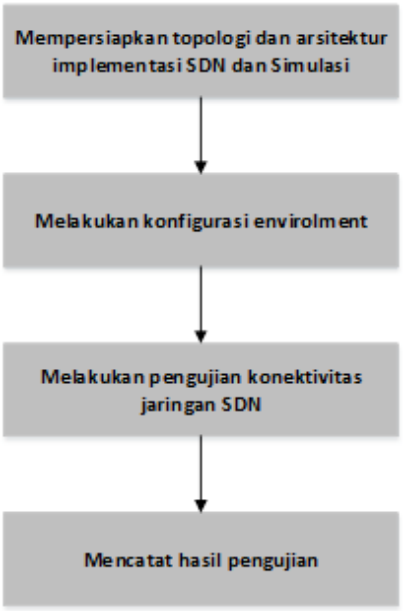
Mortier et-al (2012), mempunyai pendapat yang berbeda, mereka percaya bahwa para pengguna menginginkan untuk memahami dan mengontrol keadaan jaringan mereka sendiri daripada mengikuti traditional policies. Sebuah jaringan rumahan mungkin akan lebih teratur apabila pengguna benar-benar memahami kebutuhan mereka akan jaringan tersebut. Untuk mencapai tujuan ini mereka membuat prototype jaringan dengan menggunakan arsitektur SDN untuk menyediakan pengguna sebuah "jendela" agar dapat melihat bagaimana jaringan mereka dimanfaatkan dengan menggunakan sebuah controller.

Mehdi et-al (2011), berpendapat bahwa sebuah Anomaly Detection System (ADS) yang diimplementasikan didalam sebuah programmable home network menyediakan informasi yang lebih akurat untuk mengidentifikasi adanya sebuah aktifitas berbahaya (dalam hal ini berupa

virus) daripada ADS yang dijalankan pada ISP. Selain itu implementasinya akan dapat beroperasi tanpa adanya penurunan pada performa dan pada saat yang sama juga dapat mengurangi beban pada ISP sehingga ISP tidak perlu melakukan pengawasan kepada sebagian besar jaringan. Algoritma ADS juga dapat beroperasi bersama dengan services lainnya pada controller seperti Home OS yang dapat bereaksi kepada aktifitas mencurigakan dan membuat laporan akan adanya ketidaknormalan pada ISP atau admin jaringan lokal.

3 METODOLOGI PENELITIAN

Metode penelitian yang digunakan dalam penelitian ini menggunakan metode penelitan tindakan dan experiment, adapun tahapan penelitian yang merupakan siklus dari experiment ini ditunjukkan pada Gambar 8.



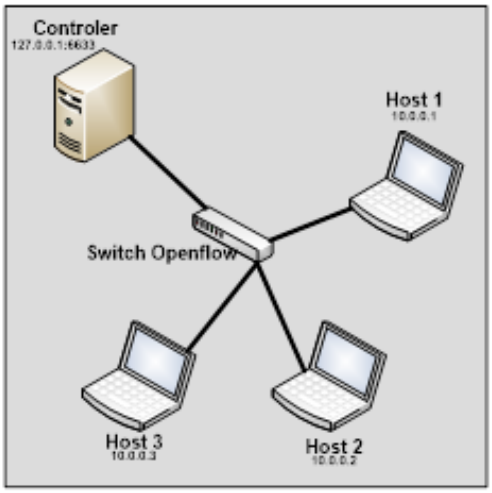
Gambar 8: Tahapan penelitian

4 PENGUJIAN DAN HASIL

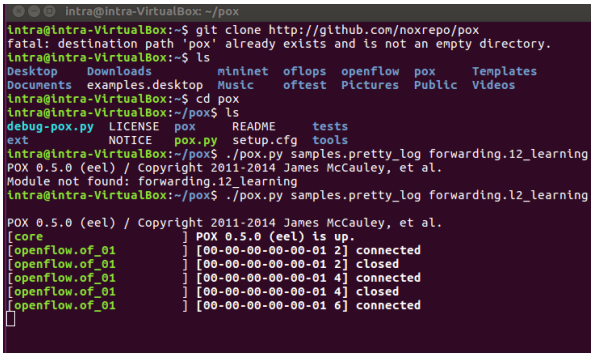
Kami melakukan simulasi jaringan SDN untuk Untuk Small/Home office pada system operasi linux Ubuntu versi 16.04.5 dengan menggunakan mininet dan pox sebagai controller. Berikut ini adalah topologi dari jaringan Small/home office yang kami lakukan.

Dari gambar 8 diatas dapat kita lihat terdapat 3 host, 1 switch dan 1 buah controller, dimana controller tersebut menggunakan pox, ip range dari ketiga host tersebut adalah 10.0.0.1 sampai dengan 10.0.0.3 sedangkan ip yang digunakan oleh controller adalah 127.0.0.1:6633 karena controller tersebut berjalan pada local host.

Langkah pertama dalam melakukan simulasi adalah dengan cara melakukan instalasi pox apabila dalam system operasi yang digunakan belum terinstall pox, setelah pox terinstal langkah selanjutnya adalah mengaktifkan controller pox tersebut.



Gambar 9: Topologi Small/Home office



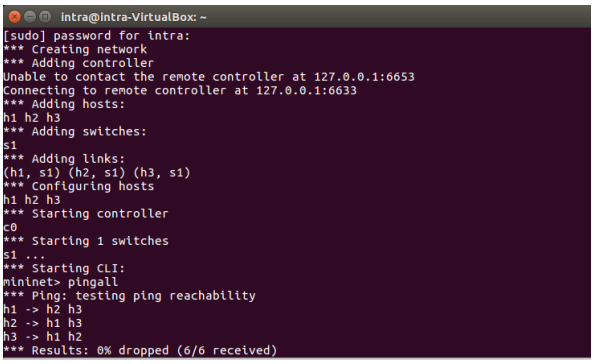
Gambar 10: Mengaktifkan Controller pox

Setalah mengaktifkan controller pox langkah selanjutnya adalah melakukan installasi mininet, dan selanjutnya adalah membuat topologi jaringan yang akan digunakan seperti pada Gambar 8. Topologi ini dibuat dengan script python. Gambar dibawah ini menunjukan topologi yang sudah dibuat dan dijalankan dalam mininet.

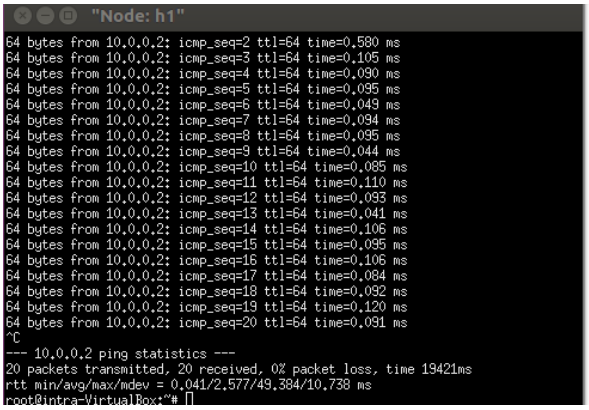
Pada gambar 10 diatas dapat dilihat seluruh host switch dan controller telah terhubung, hal ini dapat dilihat pada saat dilakukan test ping keseluruh host. Setelah melakukan ping test ke 3 host tersebut yaitu host1,2 dan 3 terdapat keterangan result:0% dropped(6/6 received).

Pada Gambar 11 adalah gambar poses ping yang dilakukan pada Host 1(10.0.0.1) pada Host 2 (10.0.0.2), di gambar tersebut terlihat ada tulisan 0% packet loss yang berarti tidak ada data yang hilang atau terdapat koneksi yang terputus pada saat proses ping, hal ini berarti Host 1 dan Host 2 sudah terhubung.

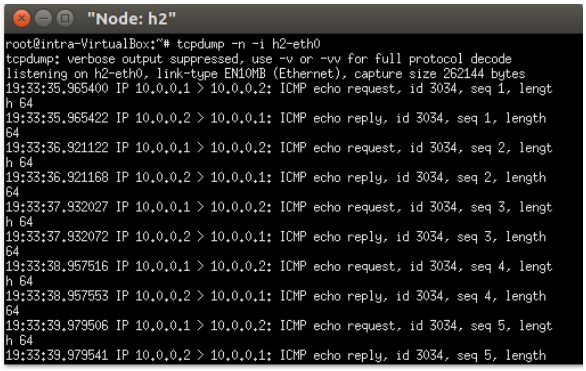
Gambar 12 diatas adalah tampilan pada saat Host 2 Menjalankan TCP DUMP bersamaan pada saat Host 1 melakukan Ping ke Host 2, TCP DUMP sendiri adalah sebuah packet analyzer yang berfungsi untuk menangkap atau memantau traffic jaringan.



Gambar 11: Mengaktifkan Mininet



Gambar 12: Ping Host 1 ke Host 2



Gambar 13: TCP DUMP

5 KESIMPULAN

Software defined network masih terus dikembangkan dan dapat diadopsi dengan berbagai cara. Salah satu cara untuk memanfaatkannya adalah dengan menggunakan arsitektur SDN

sebagai pengganti arsitektur jaringan tradisional. Dengan menggunakan arsitektur SDN, para pengguna jaringan Small office Home office dapat melakukan manajemen terhadap jaringan sehingga memiliki jaringan yang optimal karena disesuaikan dengan kebutuhan pengguna masing-masing. Penggunaan arsitektur SDN juga dapat meningkatkan keamanan dalam jaringan tersebut tanpa harus menggunakan berbagai perangkat keras jaringan lainnya sehingga dapat mengurangi biaya yang ditanggung oleh pengguna jaringan Small office Home office.

Referensi

Applemen,M., De Boer,M. (2013).“Performance Analysis of OpenFlow Hardware ”,University Of Amsterdam 2012,http://staff.science.uva.nl/ delaat/rp/2011-2012/p18/report.pdf

Anonim, (2013). “OpenFlow Switch Specification Version 1.0.0 (Wire Protocol 0x01)“ Open Networking Foundation, 2009, www.opennetworking.org/images/stories/downloads/sdn-resources/onfspecifications/OpenFlow/OpenFlow-spec-v1.0.0.pdf

Bob Lantz, Brandon Heller, and Nick McKeown. (2010). A network in a laptop: rapid prototyping for software-defined networks. In Proc. 9th ACM SIGCOMM Workshop on Hot Topics in Netw.

Chung Yik,EE.(2013).”IMPLEMENTATION OF AN OPEN FLOW SWITCH ON NETFPGA ” Universiti Teknologi Malaysia.

T.R. Henderson, M. Lacage, G.F. Riley, C. Dowell, and JB Kopena. (2008). Network simulations with the ns-3 simulator. SIGCOMM demonstration

Hyojoon, kim, Nick Feamster. (2013). Improving Network Management with software Defined Networking. Jurnal. Georgia Institute of Technology, Georgia.

K.L. Calvert, W.K. Edwards, N. Feamster, R.E. Grinter, Y. Deng, and X. Zhou.(2011). Instrumenting home networks. ACM SIGCOMM Computer Commun. Review, 41(1):84–89.

McKeown,N., Anderson,T., Balakrishnan ,H., Parulkar,G., Peterson,L., Rexford ,J., Shenker ,S., Turner ,J. (2013). “OpenFlow: Enabling Innovation in Campus Networks” , Stanford University 2008,www.OpenFlow.org/documents/OpenFlow-wplatest. pdf

Mateo,P. Manuel, (2013). “OpenFlow Switching Performance ”, POLITECNICO DI TORINO.

Mortier, T. Rodden, T. Lodge, D. McAuley, C. Rotsos, AW Moore, A. Koliousis, and J. Sventek. (2012). Control and understanding: Owning your home network. In 2012 4th Int. Conf. on Commun. Syst. and Netw. (COMSNETS), , pages 1–10. IEEE.

S. Mehdi, J. Khalid, and S. Khayam. (2011). Revisiting traffic anomaly detection using software defined networking. In Recent Advances in Intrusion Detection, pages 161–180. Springer.

Negara, E.S. and Andryani, R., 2014. A Review: Security Framework Information Technology for University Based on Cloud Computing.

Negara, E.S., 2019. Jaringan Komputer Routing dan Switching Essentials.

N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. (2008). "Openflow: enabling innovation in campus networks. ACM SIGCOMM Computer Commun". Review,38(2):69–74.

N. Feamster.(2010). Outsourcing home network security. In Proc. 2010 ACM SIGCOMM workshop on Home networks, pages 37–42. ACM.

Ummah, Izzatul, Desianto A. 2016. Perancangan Simulasi Jaringan Virtual Berbasis Software-Define Networking. Jurnal. Department of Computational Science, Telkom University, Bandung.

Vinshoi, Anilkumar., Khumbhare, Abhijit.(2013). "Open Flow 1.3.1 Support: Controller View ", IBM.