SIMULASI JARINGAN *VIRTUAL LOCAL AREA NETWORK* (VLAN) MENGGUNAKAN *POX CONTROLLER*

Muhamad Fahri[1], Andrew Fiade[2], Hendra Bayu Suseno[3]

[1,2,3]Program Studi Teknik Infomatika, UIN Syarif Hidayatullah Jakarta.
Jl.Ir.H.Juanda No.95 Ciputat 15412 Jakarta-Indonesia
mfahri_mf@mhs.uinjkt.ac.id

ABSTRAK

Keterbatasan LAN melahirkan sebuah teknologi VLAN yang memungkinkan adanya konfigurasi dari suatu jaringan komputer secara virtual (virtualisasi). Proses mencocokkan fleksibilitas virtualisasi *server* sulit dilakukan dengan *switch* tradisional, sebab logika kontrol untuk setiap *switch* terletak dalam logika *switching* yang sama. *Software Defined Network* (SDN) memisahkan *control plane* dari *forwarding hardware*. Migrasi *logic control* yang digunakan pada perangkat yang terintegrasi (misalnya *switch ethernet*) menjadi mudah diakses dan secara logis jaringan menjadi terpusat dalam hal pengendalian. Pada penelitian ini dilakukan simulasi jaringan VLAN menggunakan *Pox controller* sehingga dapat mengetahui hasil evaluasi jaringan VLAN menggunakan *pox controller*. Berdasarkan fase *simulation*, konfigurasi jaringan VLAN lebih ditekankan pada *controller*. Berdasarkan hasil pengujian nilai rata-rata *jitter*, pada jaringan VLAN menggunakan 2 buah *switch* nilai rata-rata *Jitter* sebesar 0,009 ms. Nilai rata-rata *jitter* tersebut lebih kecil dari nilai rata-rata *jitter* pada jaringan VLAN yang menggunakan 3 buah *switch* yaitu sebesar 0,027 ms. Sedangkan hasil pengujian nilai rata-rata *packet loss* memiliki nilai yang sama, yaitu 0%. Nilai rata-rata *packet loss* tersebut menunjukkan bahwa kedua skenario tersebut tidak terjadi kehilangan paket.

Kata Kunci: *VLAN, Software Defined Network, Control Plane, Forwarding Hardware, Pox Controller, Jitter, Packet Loss*

ABSTRACT

Limitations of the LAN creates a VLAN technology that allows the configuration of a virtual computer network (virtualization). The process of matching server virtualization flexibility is difficult with traditional switches, since the control logic for each switch lies in the same switching logic. Software Defined Network (SDN) separates the control plane from hardware forwarding. The migration logic controls used on integrated devices (e.g. ethernet switches) are easily accessible and logically the network becomes centralized in terms of control. In this research, VLAN network simulation was conducted using Pox controller so that we can know the result of VLAN network evaluation using pox controller. Based on the simulation phase, VLAN network configuration is more emphasized on the controller. Based on the results of testing the average value of Jitter, on the VLAN network using 2 pieces of the Jitter average value is 0.009 ms. That average value of Jitter is smaller than the average value of Jitter on a VLAN network using 3 switches that is 0.027 ms. While the test results of Packet Loss average value has the same value, i.e. 0%. The average value of Packet Loss indicates that both scenarios did not occur Packet Loss.

Keywords: *VLAN, Software Defined Network, Control Plane, Hardware Forwarding, Pox Controller, Jitter, Packet Loss*

DOI: 10.15408/jti.v10i1.6821

I. PENDAHULUAN

Perkembangan teknologi, terutama pada perangkat lunak komputer (dalam bentuk sistem operasi dan aplikasi), memungkinkan adanya konfigurasi dari suatu jaringan komputer secara virtual (virtualisasi). VLAN merupakan salah satu solusi yang diberikan untuk hal tersebut. [4]

Permasalahan yang ada pada VLAN adalah dalam mengkonfigurasi VLAN. Untuk mencocokkan fleksibilitas virtualisasi server, pengelola jaringan harus mampu untuk secara dinamis menambahkan, drop dan mengubah jaringan. Proses ini sulit dilakukan dengan *switch* tradisional, sebab logika kontrol untuk setiap *switch* terletak dalam logika *switch*ing yang sama. Proses pengelolaan jaringan, tingkat QoS (*Quality of Service*) dan tingkat keamanan dapat sangat memakan waktu jika jaringan perusahaan besar yang menggunakan perangkat jaringan dari beberapa vendor, sebab administrator jaringan harus mengkonfigurasi peralatan masing-masing vendor secara terpisah dan menyesuaikan kinerja serta parameter-parameternya. [5]

Software Defined Networking (SDN) sering disebut sebagai ide baru yang revolusioner dalam jaringan komputer untuk menyederhanakan kontrol jaringan, manajemen dan memungkinkan inovasi melalui programabilitas jaringan. Jaringan komputer biasanya dibangun dari sejumlah besar perangkat jaringan (seperti *switch*, *router*, firewall dan sebagainya) dengan banyak protokol yang kompleks (*software*), yang diimplementasikan dan tertanam. Network engineer bertanggung jawab untuk melakukan manajemen jaringan yang cukup menantang dan rawan kesalahan. [1]

Karakteristik mendasar dari SDN adalah pemisahan *forwarding* dan *control plane*. Fungsi *forwarding* meliputi logika untuk menangani paket yang masuk berdasarkan karakteristik seperti MAC *address*, IP dan VLAN ID. Protokol, logika dan algoritma yang digunakan untuk *forwarding* plane berada pada *control plane. Control plane* menentukan tabel *forwarding* dan logika yang harus diprogram atau dikonfigurasi. Hasil yang paling dasar sinkronisasi ini adalah pencegahan *loop*. [2]

Adanya penerapan QoS pada protokol OpenFlow menjadi daya tarik penulis untuk melakukan penelitian terhadap jaringan VLAN. Pada jaringan SDN terdapat sebuah *controller* yang mengatur jaringan. Adapun *controller* yang digunakan penulis dalam penelitian ini adalah Pox *controller*. Pox *controller* merupakan salah satu *controller* open source dan proses instalasi yang sederhana karena include dalam mininet serta menggunakan bahasa python sama halnya dengan mininet. Pox *controller* memiliki banyak fitur-fitur yang memudahkan para peneliti untuk melakukan penelitian mengenai jaringan dan Pox *controller* juga memiliki banyak fitur khususnya dalam VLAN sehingga memudahkan para peneliti melakukan penelitian mengenai SDN pada jaringan VLAN. Berdasarkan uraian permasalahan yang telah dijelaskan, judul penelitian penulis adalah "Simulasi Jaringan *Virtual Local Area Network* (VLAN) Menggunakan Pox *Controller*". Artikel ini merupakan hasil orisinil dari penelitian yang dilakukan oleh penulis.

II. METODOLOGI PENELITIAN

II.1 *Studi Pustaka*

Dalam penelitian sebelumnya yang berjudul "Simulasi *Virtual Local Area Network* (VLAN) Berbasis *Software* Defined Network (SDN) Menggunakan POX *Controller*" dilakukan simulasi simulasi setup time dengan skenario perubahan jumlah VLAN. Topologi yang diuji mula-mula, terdapat empat buah host dengan dua *switch*, kemudian jumlah host ditambahkan dengan VLAN ID yang bertambah pula. Pengujian *setup time* ini bertujuan untuk mengetahui seberapa lama waktu setup paket openflow mod yang dikirimkan dari *controller* ke *switch* sampai kondisi *switch* stabil yaitu kondisi ketika *switch* tidak lagi meminta paket action terhadap suatu paket.

Dari pengujian yang dilakukan bahwa setup time untuk jumlah *host* yang semakin bertambah maka setup time akan semakin besar. Hal ini dikarenakan *controller* menggunakan mode proactive sehingga semakin banyak *host* maka rule yang dibutuhkan juga semakin banyak sehingga kerja dari *controller* untuk memberikan informasi kepada *switch* semakin banyak.

II.2 Metode Simulasi

Pada metode simulasi ini meliputi beberapa langkah yang akan dilakukan yaitu: [3]

1. Problem Formulation
2. Conceptual Model
3. Input Output Data
4. Modeling
5. Simulation
6. Verfication and Validation

7. Experimentation
8. Output Analysis

III. HASIL DAN PEMBAHASAN

III.1 Problem Formulation

Pada jaringan *Virtual Local Area Network* (VLAN) menggunakan *switch* tradisional, administrator jaringan harus melakukan konfigurasi pada setiap *hardware* jaringan yang terhubung pada jaringan tersebut. Konfigurasi harus mengikuti cara konfigurasi dari setiap *hardware* tersebut. Setiap *hardware* jaringan memiliki cara konfigurasi yang berbeda-beda menyesuaikan vendor *hardware*. Hal ini merupakan suatu permasalahan dalam hal membuat sebuah jaringan menggunakan *switch* tradisional yang mengakibatkan administrator jaringan harus mengetahui cara konfigurasi setiap *hardware* yang terhubung. [5]

Dari permasalahan tersebut dapat disimpulkan bahwa dibutuhan sebuah solusi yang dapat membantu administrator jaringan dalam hal melakukan konfigurasi. Solusi yang dapat diterapkan untuk mengatasi permasalahan tersebut adalah dengan menerapkan *Software* Defined Network (SDN). SDN bersifat terbuka dan memisahkan *control plane* dengan data plane sehingga jaringan SDN memudahkan administrator dalam hal berinovasi jaringan dan menjadikan infrastruktur yang ada sebagai entitas logis.

III.2 *Conceptual Model*

Pada skenario 2 simulasi dilakukan untuk Conceptual model dalam penelitian ini adalah membuat konsep simulasi dengan membuat topologi 2 model skenario jaringan VLAN menggunakan simulasi Virtual Network Description (VND) melalui website http://www.ramonfontes.com/vnd, kemudian dijalankan dengan menggunakan mininet dan pox *controller*.

Adapun ketentuan skenario tersebut yaitu:
1. Jumlah *switch* yang digunakan adalah 2 buah *switch* dan 3 buah *switch*
2. Pada masing-masing skenario terdapat 6 buah PC dan 1 buah pox *controller*

III.3 *Input Output Data*

Adapun *input* yang digunakan adalah konfigurasi VLAN. Pada 2 *switch* terdapat 3 buah *PC* yang merupakan 3 buah VLAN yang berbeda, yaitu VLAN 10, VLAN 20 dan VLAN 30. Berikut ini merupakan *input* untuk setiap *PC* yang terhubung:

Tabel 1 Data Input PC

PC	*IP Address*	*Subnet Mask*	VLAN
PC 0	10.0.0.1	255.255.255.0	10
PC 1	10.0.0.2	255.255.255.0	20
PC 2	10.0.0.3	255.255.255.0	30
PC 3	10.0.0.4	255.255.255.0	10
PC 4	10.0.0.5	255.255.255.0	20
PC 5	10.0.0.6	255.255.255.0	30

Variabel yang digunakan untuk mendapatkan output pada simulasi ini yaitu :
1. *Jitter*

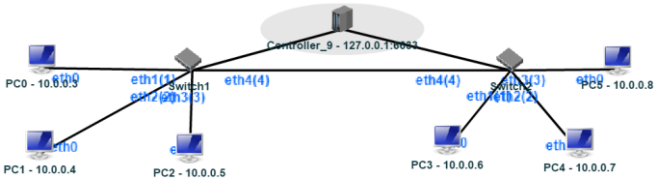
Output ini menunjukkan waktu yang dibutuhkan suatu paket data terkirim dari node pengirim ke node tujuan.
2. *Packet Loss*

Output ini untuk mengukur presentase jumlah data yang dikirim dan data yang diterima

III.4 *Modeling*

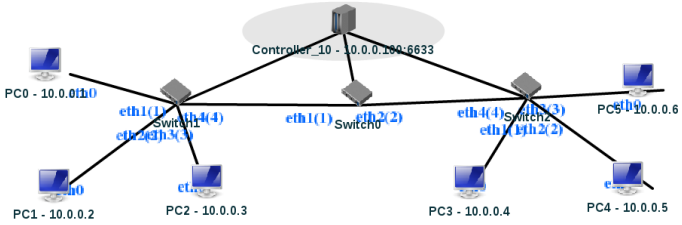
Pada tahapan *modeling* penulis membuat ilustrasi sebuah model jaringan VLAN berdasarkan tahapan sebelumnya. Terdapat 2 model jaringan VLAN yang penulis buat dengan penjelasan sebagai berikut:

1. Skenario 1 Menggunakan 2 Buah *Switch:*



Gambar 1. Skenario 1 menggunakan 2 buah *switch*

2. Skenario 2 Menggunakan 3 Buah *Switch:*



Gambar 2. Skenario 2 menggunakan 3 buah *switch*

III.5 Simulasi

Simulasi yang dilakukan pada jaringan VLAN menggunakan aplikasi mininet dan sistem operasi Linux Ubuntu LTS 14.04.5 yang berjalan pada virtual box. Sebelum simulasi dilakukan, ada beberapa hal yang harus disiapkan dan diinstal, yaitu virtual box, Linux Ubuntu LTS 14.04.5, mininet dan Pox *Controller*. Proses instalasi Linux Ubuntu LTS 14.04.5 dilakukan dengan menggunakan virtual box. Untuk mengetahui kinerja jaringan, penulis menggunakan aplikasi Iperf versi 2.0.5 yang berjalan di dalam PC.

III.6 *Verification and Validation*

Verifikasi dan validasi data dilakukan dengan memeriksa jaringan yang dibuat sudah sesuai atau tidak. Masing-masing skenario akan dilakukan percobaan pada tahapan ini untuk mengetahui apakah simulasi jaringan yang telah dirancang pada tahapan sebelumnya sudah sesuai dengan ketentuan yang ditetapkan pada tahapan sebelumnya. Jika terjadi kesalahan pada percobaan yang dilakukan pada tahapan ini, maka akan dilakukan koreksi atau perbaikan pada tahapan simulasi. Jika tidak terjadi kesalahan, maka akan dilanjutkan ke tahapan selanjutnya yaitu experimental dan output analysis.

III.7 *Experimentation*

Ada 2 skenario yang dilakukan. Setiap skenario dilakukan percobaan sebagai berikut:

1. Pengujian konektivitas jaringan yang telah dibuat pada mininet.
2. Mengirimkan paket UDP. Waktu yang digunakan adalah 20, 40 dan 60 detik dengan masing-masing waktu dilakukan tiga kali percobaan. Nilai yang keluar pada akhir percobaan adalah *Jitter* dan *Packet Loss*.

III.8 Pengujian Konektivitas Jaringan

Pada fase atau tahapan ini penulis melakukan pengujian terhadap koneksi antarVLAN. Pengujian ini dilakukan setelah menjalankan *controller* yang telah dibuat sebelumnya. Pengujian yang penulis lakukan adalah menggunakan perintah pingall. Pingall dilakukan setelah *controller* dijalankan. Berikut adalah gambar ketika melakukan pingall setelah *controller* dijalankan:

```
mininet> pingall
*** Ping: testing ping reachability
pc0 -> X X pc3 X X
pc1 -> X X X pc4 X
pc2 -> X X X X pc5
pc3 -> pc0 X X X X
pc4 -> X pc1 X X X
pc5 -> X X pc2 X X
*** Results: 80% dropped (6/30 received)
```

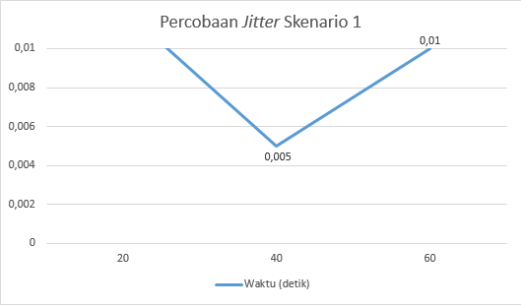
Gambar 3. *Pingall* setelah *controller* dijalankan

Gambar 3 menggambarkan tentang pengujian terhadap konektivitas antarpc. Pemeriksaan konektivitas pada gambar 3 dilakukan setelah *controller* dijalankan dan membutuhkan waktu selama 1 menit 20 detik. Pemeriksaan tersebut membutuhkan waktu yang lama karena melakukan pemeriksaan terhadap semua pc yang terhubung.

Hasil pengujian tersebut menunjukkan bahwa setiap pc hanya terkoneksi dengan 1 buah pc lainnya. Pc0 terkoneksi dengan pc3, pc1 terkoneksi dengan pc4, pc2 terkoneksi dengan pc5 dan sebaliknya pc3 terkoneksi dengan pc0, pc4 terkoneksi dengan pc1 dan pc5 terkoneksi dengan pc2. Pingall juga menunjukkan bahwa paket yang dikirim sebanyak 30 hanya dapat terkirim sebanyak 6. Hal tersebut ditunjukkan pada bagian 6/30 received.

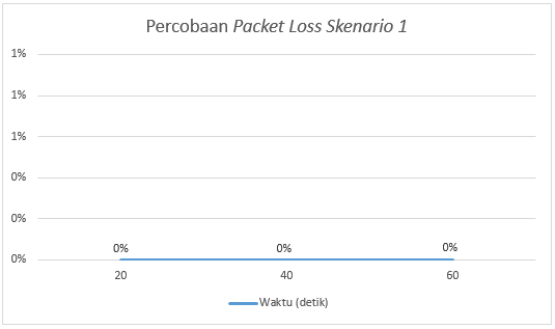
III.9 Pengujian Performa Jaringan Dengan Paket UDP

a. Skenario 1



Gambar 4. Grafik *jitter* skenario 1

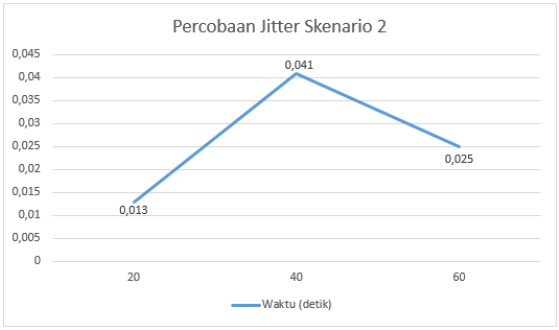
Gambar 4 menunjukkan perubahan rata-rata nilai *jitter* pada setiap percobaan. Nilai *jitter* terbesar ditunjukkan pada percobaan pertama dan ketiga dengan waktu percobaan 20 dan 60 detik. Sedangkan nilai *jitter* terkecil ditunjukkan pada percobaan kedua dengan waktu percobaan 40 detik.



Gambar 5. Grafik *packet loss* skenario 1

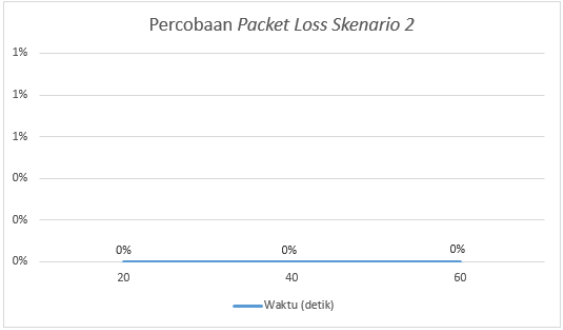
Gambar 5 menunjukkan perubahan nilai *packet loss* pada setiap percobaan. Seluruh percobaan pada skenario ini mendapatkan hasil 0%.

b. Skenario 2



Gambar 6. Grafik *jitter* skenario 2

Gambar 6 menunjukkan perubahan rata-rata nilai *jitter* pada setiap percobaan. Nilai *jitter* terbesar ditunjukkan pada percobaan pertama dan ketiga dengan waktu percobaan 40 detik. Sedangkan nilai *jitter* terkecil ditunjukkan pada percobaan kedua dengan waktu percobaan 20 detik.

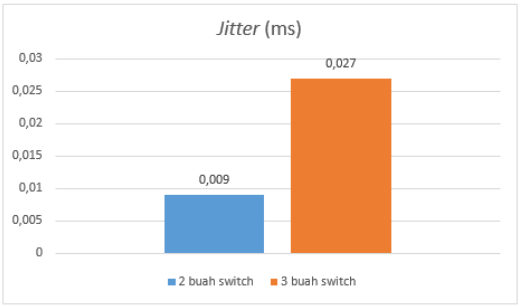


Gambar 7. Grafik *Packet Loss* Skenario 2

Gambar 7 menunjukkan perubahan nilai *packet loss* pada setiap percobaan. Seluruh percobaan pada skenario ini mendapatkan hasil 0%.

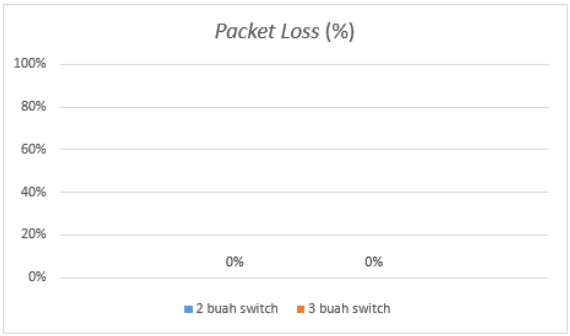
III.10 Output *Analysis*

Hasil dari pemaparan hasil percobaan yang penulis dapatkan pada masing-masing skenario simulasi, maka kemudian diambil nilai rata-rata dari parameter *jitter* dan *packet loss*. Nilai rata-rata dari kedua parameter dari masing-masing skenario dibandingkan satu sama lain untuk mendapatkan skenario mana yang memiliki nilai terbaik untuk masing-masing parameter. Hasil keseluruhan simulasi ditampilkan dalam bentuk grafik.



Gambar 8. Grafik *jitter*

Gambar 8 menunjukkan perbandingan nilai rata-rata *jitter* untuk masing-masing skenario. Suatu jaringan dapat dikatakan bagus apabila memiliki nilai *jitter* yang kecil. Semakin kecil nilai *jitter* maka semakin lancar proses pengiriman data. Hasil simulasi yang penulis lakukan menunjukkan bahwa semakin banyak *switch* yang digunakan, maka akan mempengaruhi nilai *jitter*.



Gambar 9. Grafik *packet loss*

Gambar 9 menunjukkan nilai rata-rata *packet loss* untuk masing-masing skenario hasil simulasi. Suatu jaringan dapat dikatakan bagus apabila memiliki nilai *packet loss* yang kecil. Semakin kecil nilai *packet loss*, maka data yang hilang saat pengiriman data juga semakin sedikit.Kedua skenario simulasi yang penulis lakukan memiliki nilai rata-rata *packet loss* terbaik karena nilai *packet loss* pada kedua skenario tersebut adalah 0%. Nilai *packet loss*

tersebut menunjukkan bahwa tidak ada paket yang hilang dalam proses pengiriman.

IV. PENUTUP

4.1 Kesimpulan

Berdasarkan hasil pengujian jaringan VLAN dilakukan dengan menggunakan 2 skenario, yaitu pembuatan jaringan VLAN dengan menggunakan 2 buah *switch* dan 3 buah *switch*. Berdasarkan hasil pengujian nilai rata-rata *Jitter* yang dijelaskan, nilai rata-rata *Jitter* dapat dipengaruhi oleh jumlah *device* yang digunakan. Pada jaringan VLAN menggunakan 2 buah *switch* nilai rata-rata *Jitter* sebesar 0,009 ms. Nilai rata-rata *Jitter* tersebut lebih kecil dari nilai rata-rata *Jitter* pada jaringan VLAN yang menggunakan 3 buah *switch* yaitu sebesar 0,027 ms. Sedangkan hasil pengujian nilai rata-rata *Packet Loss* yang dijelaskan memiliki nilai yang sama, yaitu 0%. Nilai rata-rata *Packet Loss* tersebut menunjukkan bahwa kedua skenario tersebut tidak terjadi kehilangan paket.

4.2 Saran

Berdasarkan penelitian ini saran dari penulis adalah bahwa untuk penelitian selanjutnya menggunakan *controller* lainnya agar dapat mengetahui lebih dalam perbedaan proses setiap *controller*. Penulis menyarankan untuk mengimplementasikan jaringan VLAN yang menerapkan SDN pada jaringan saat ini supaya dapat melihat perbedaan performansi jaringan saat ini dengan jaringan yang menerapkan SDN. Penulis menyarankan untuk menambahkan proses intervlan *routing* agar semua VLAN dapat saling terkoneksi.

DAFTAR PUSTAKA

[1] Azodolmolky, Siamak. 2013. *Software Defined Networking with OpenFlow*. Birmingham-Mumbai: Packt Publishing.

[2] Goransson, Paul & Black, Chuck. 2014. *Software Defined Networks A* Comprehensive *Approach*. USA: Morgan Kaufmann.

[3] Madani, S.A., Kazmi, J & Mahlkecht, S. 2010. *Modeling and Simulation.*

[4] Pratama, I Putu Agus Eka. 2015. *Handbook Jaringan Komputer*. Bandung: Informatika

[5] Stallings, William. 2013. *Software Defined Networks and OpenFlow*. The Internet Protocol Journal.