Analisis Unjuk Kerja Pemantauan Jaringan OpenNMS (*Open Network Monitoring System*) pada Jaringan TCP/IP

Yohanes Hendra Nugroho[1] Nyoman Putra Sastra[2] Dewa Made Wiharta[3]
Program Studi Teknik Elektro, Fakultas Teknik - Universitas Udayana
Email: yohaneshendranugroho@gmail.com[1] putra.sastra@unud.ac.id[2] wiharta@unud.ac.id[3]

Abstrak

Dalam membangun jaringan pada suatu perusahaan atau universitas yang besar diperlukan sebuah manajemen jaringan untuk memudahkan dalam pengaturan dan pengawasan jaringan tersebut. Semakin kompleks jaringan multimedia yang dibangun, maka semakin sulit bagi administrator untuk melakukan pemantauan terhadap jaringan tersebut. Untuk itu, perlu dibangun sebuah sistem monitoring yang bertujuan mempermudah dalam hal melakukan pemantauan dan juga pemeliharaan dan peningkatan kualitas jaringan. Dalam penelitian ini, dibangun sebuah sistem monitoring OpenNMS (Open Network Monitoring System) dengan tujuan untuk bisa memberi gambaran bagaimana OpenNMS bekerja memonitor jaringan yang ada. Pada penelitian ini, dianalisa kinerja dari perangkat-perangkat yang ada, berupa response time pada setiap service yang berjalan serta service SNMP untuk memonitor kualitas dari perangkat yang ada. Dibangun pula Bot yang berfungsi memberikan informasi yang dibutuhkan melalui aplikasi Telegram. Bot tersebut terintegrasi dengan OpenNMS dan dapat otomatis mengirim informasi alarm yang terpantau dari OpenNMS jika perangkat ataupun jaringan sedang terjadi masalah. Sehingga dari hasil penelitian ini, diharapkan memberikan kemudahan bagi administrator untuk memantau kinerja pada jaringan dan penggunaan alarm Bot Telegram untuk selalu memberikan informasi terbaru jika perangkat atau jaringan sedang mengalami down.

Kata kunci: *Bot Telegram, Jaringan Multimedia, Manajemen Jaringan, OpenNMS, SNMP*

1. PENDAHULUAN

Secara umum, manajemen jaringan merupakan sebuah metode pengawasan terhadap unjuk kerja jaringan dan mengendalikan trafik agar diperoleh kapasitas jaringan dengan pengoperasian yang maksimal pada berbagai situasi [1]. Manajemen jaringan juga berfungsi untuk merencanakan, analisa, evaluasi, desain, administrasi dan mengembangkan jaringan telekomunikasi sehingga dari data tersebut diperoleh kualitas pelayanan yang baik dengan biaya proporsional dan juga kapasitas yang optimal. Manajemen jaringan juga memiliki kemampuan untuk memonitor dan mengontrol jaringan komputer dari sebuah lokasi.

Dalam memanajemen jaringan, jaringan komputer merupakan suatu himpunan interkoneksi sejumlah komputer yang mencakup berbagai perangkat, untuk perangkat keras yaitu komputer, hub, switch, router, PBX, Sentral Telepon, dan lain-lain. Sedangkan untuk perangkat lunak berupa Sistem Operasi (OS)/*Network Operating System*, aplikasi, dan sebagainya. Saat ini banyak *platform* yang dapat digunakan untuk memonitor suatu jaringan seperti *Ground works, Zabbix, Zenoss*, OpenNMS, *Nagios, Hyperic-hq* [1]. Pada penelitian ini penulis menggunakan OpenNMS sebagai *platform* monitoring jaringan. OpenNMS (*Network Management System*) adalah salah satu cara sistem manajemen *monitoring* jaringan yang bebas dan bersifat *open source.* OpenNMS dikembangkan dan didukung oleh komunitas pengguna dan pengembang serta oleh *Group* OpenNMS yang menawarkan layanan komersial dan pelatihan.

OpenNMS yang digunakan pada penelitian ini akan memerlukan plugin tambahan dengan tujuan untuk memperluas jangkauan monitoring dengan *monitoring* yang terdistribusi ataupun terintegrasi dengan *software open source* lain.

2. TINJAUAN PUSTAKA

Pada penelitian yang dilakukan oleh Harry Li telah ditinjau perbedaan antara jaringan nirkabel dengan jaringan kabel konvensional dan membahas tentang

analisis, desain dan implementasi teknologi manajemen WLAN berbasis NMS yang didasarkan kerangka kerja OpenNMS [2].

Sementara itu, penelitian yang dilakukan oleh Basem Shihada ditujukan untuk visualisasi arsitektur perangkat lunak OpenNMS dan menganalisis dari aspek konseptual dan arsitektur dasar [3].

Pada penelitian lain yang dilakukan oleh Shane O'Donnell dibahas tentang perbandingan manajemen jaringan menggunakan Protokol SNMP antara lain *Ground works, Zabbix, Zenoss*, OpenNMS, *Nagios, Hyperic-hq* [1]. Secara umum protokol SNMP banyak digunakan untuk mengembangkan pemantauan manajemen jaringan dikarenakan dapat digunakan pada sebagian besar sistem operasi.

Dalam penelitian ini, penulis membangun sebuah monitoring jaringan OpenNMS yang terintegrasi dengan Bot Telegram, untuk dapat melakukan konfigurasi dan integrasi dengan OpenNMS pada server dan jaringan di Universitas Udayana. Tujuan dari penelitian ini adalah untuk dapat mengetahui kinerja dari tiap server apakah sudah berjalan dengan baik dan dapat memonitor trafik jaringan yang masuk dan keluar serta port-port yang berjalan pada jaringan TCP/IP.

2.1 *Network Managament*

Manajemen jaringan adalah kemampuan untuk mengontrol dan memantau sebuah jaringan komputer pada suatu tempat. Pada OSI (*Open System Interconnection*), standar *network manajemen* harus memenuhi FCAPS (*Fault, Configuration, Accounting, Performance* dan *Security)* [4]. FCAPS sendiri merupakan model dan *framework* dari ISO (*International Organization for Standardization*) *Telecommunications Management Network*.

Parameter dari FCAPS yaitu: 1) *Fault Management*, 2) *Configuration Management*, 3) *Accounting Managemen*t, 4) *Performance Management*, dan 5) *Security Management.*

Fault Management berfungsi untuk menyediakan fasilitas untuk administrator agar dapat memantau jika terjadi kesalahan pada perangkat jaringan yang dikelola sehingga dapat segera ditindaklanjuti (pebaikan). Manajemen konfigurasi berfungsi untuk memonitor infomasi konfigurasi jaringan sehingga memiliki dampak pada perangkat keras dan lunak agar dapat dikelola dengan baik. Tujuan pengelolaan konfigurasi meliputi:

Manajemen Akunting berfungsi mengukur utilisasi (penggunaan) jaringan dari pengguna atau kelompok tertentu untuk melacak informasi penggunaaan jaringan sehingga pengguna individu, institusi maupun unit bisnis dapat ditagih atau dikenakan biaya untuk tujuan akunting. Manajemen kinerja difokuskan untuk memastikan bahwa kinerja jaringan tetap pada tingkat yang dapat diterima. Kinerja jaringan menangani *throughput*, waktu respon jaringan, tingkat *packet loss*, utilisasi *link*, persentase utilisasi, tingkat kesalahan dan sebagainya.

Manajemen keamanan adalah proses pengendalian akses terhadap aset dalam jaringan. Fungsi manajemen keamanan meliputi pengelolaan otentikasi, otorisasi, dan audit jaringan, sehingga pengguna internal dan eksternal hanya memiliki akses ke sumber daya jaringan yang sesuai. Tugas umum lainnya meliputi konfigurasi dan pengelolaan *firewall* jaringan, sistem deteksi intruksi, dan kebijakan keamanan.

2.2 *Simple Network Management Protocol* (SNMP)

SNMP merupakan protokol yang digunakan untuk memonitor dan mengelola berbagai perangkat yang terhubung dalam jaringan [5].

2.2.1 Komponen Utama SNMP

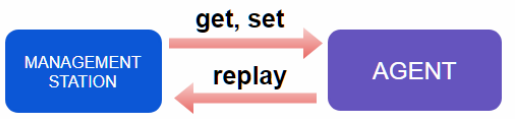
Terdapat tiga komponen utama dari SNMP, seperti pada Gambar 1, yaitu *Management Information Base* (MIB), Agen, dan *Manager*.

Gambar 1. Struktur SNMP

MIB merupakan struktur basis data variabel dari sebuah elemen jaringan yang dimonitor. Agen merupakan *software* yang dijalankan pada setiap perangkat jaringan yang dimonitor. Agen ini bertugas mengumpulkan seluruh informasi yang telah ditentukan oleh MIB. *Manager* adalah *software* yang berjalan pada jaringan yang bertugas meminta informasi ke agen.

2.2.2 Fungsi SNMP

Gambar 2 dan Gambar 3 merupakan ilustrasi Fungsi SNMP. Fungsi *Get* digunakan oleh manager untuk mengambil suatu informasi dari agen MIB. Sedangkan *Set* digunakan oleh manager untuk mengatur atau mengisi nilai variabel pada agen MIB. *Trap* digunakan oleh agen untuk mengirim peringatan kepada *manager. Inform*: digunakan oleh *manager* untuk mengirim peringatan kepada *manager* lainnya.



Gambar 2. SNMP *query*

Gambar 3. SNMP TRAP

2.3 Struktur Manajemen Informasi

Structure of Management Information (SMI) memiliki fungsi mendefinisikan objek yang telah teratur dari sifat mereka. Agen yang terdapat pada perangkat memiliki daftar kejadian objek yang telah terekam. Contohnya status dari perangkat router (apakah nyala atau mati).

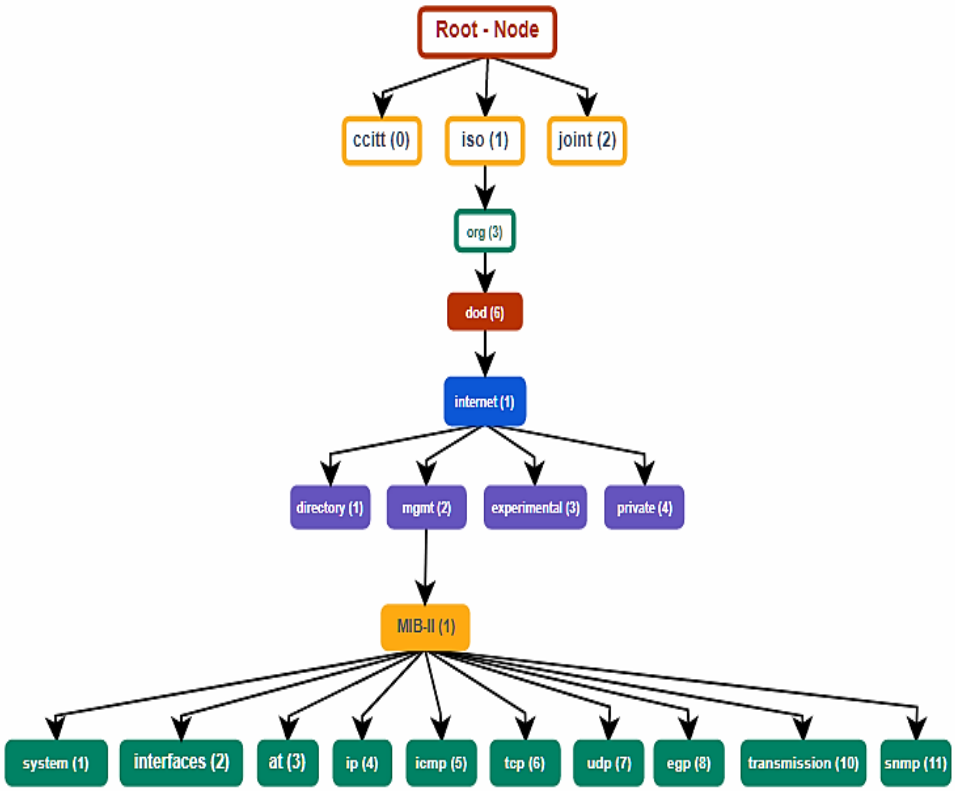
2.3.1 MIB-II

MIB-II adalah kelompok manajemen yang sangat penting dikarenakan setiap perangkat yang mendukung SNMP harus mendukung MIB-II.

Dari Gambar 4, Seksi dari RFC 1213- MIB yang mendefinisikan dasar OIDs untuk *subtree* akan terlihat seperti dibawah ini [6]:

1. mib-2OBJECT IDENTIFIER::= {mgmt 1}
2. sistemOBJECT IDENTIFIER::= {mib-2 1}
3. inferfaceOBJECT IDENTIFIER::= {mib-2 2}
4. atOBJECT IDENTIFIER::= {mib-2 3}
5. ipOBJECT IDENTIFIER::= {mib-2 4}
6. icmp OBJECT IDENTIFIER::= {mib-2 5}
7. tcp OBJECT IDENTIFIER::= {mib-2 6}
8. udp OBJECT IDENTIFIER::= {mib-2 7}
9. egp OBJECT IDENTIFIER::= {mib-2 8}
10. transmission OBJECT IDENTIFIER::= {mib-2 10}
11. snmpOBJECT IDENTIFIER::= {mib-2 11}

Deskripsi lebih jelas ditunjukkan pada Tabel 1



Gambar 4. MIB Tree

Tabel 1 Deskripsi dari group MIB-II [6]

Subtree Name	**OID**	***Description***
Sistem	1.3.6.1.2.1.1	Menyatakan sebuah daftar objek yang berhubungan dengan sistem operasi, seperti *uptime* sistem, kontak sistem, dan nama sistem.
Interfaces	1.3.6.1.2.1.2	Menjaga jalur dari status tiap *interfaces* pada *managed entity*. Grup *interfaces* memantau *interfaces* yang hidup dan mati dan mencatat beberapa hal seperti jumlah oktet yang terkirim dan diterima, *error* dan *disCards*, dan lain sebagainya.
at	1.3.6.1.2.1.3	Grup *Address translation* (at) dihilangkan dan disediakan hanya untuk kompatibilitas dengan versi sebelumnya.
ip	1.3.6.1.2.1.4	Menjaga jalur dari beberapa aspek dari IP, termasuk IP *routing*.
icmp	1.3.6.1.2.1.5	Memeriksa *error* SNMP, *disCards*, dan lain sebagainya
tcp	1.3.6.1.2.1.6	Mencatat,antara beberapa hal lain, keadaan koneksi TCP (misalnya, *closed, listen, synSent*, dan lain sebagainya).
udp	1.3.6.1.2.1.7	Mencatat statistik UDP, datagram yang masuk dan keluar, dan lain sebagainya.
egp	1.3.6.1.2.1.8	Mencatat statistik tentang EGP dan memelihara tabel EGP tetangga.
transmission	1.3.6.1.2.1.10	Belum terdapat objek yang didefinisikan pada grup ini, tetapi media spesifik lain dinyatakan dengan menggunakan *subtree* ini.
snmp	1.3.6.1.2.1.11	Mengukur kehandalan dari implementasi SNMP seperti *managed entity* dan mencatat paket SNMP terkirim dan diterima.

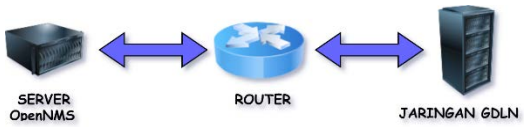
2.4 *Open Network Monitoring System* (OpenNMS)

OpenNMS merupakan suatu pemantauan jaringan pada kampus, perusahaan, ataupun tempat lainnya dan platform manajemen jaringan yang dikembangkan dalam model *open source*. Fitur–fitur yang terdapat pada OpenNMS yaitu: 1) Service Polling: berfungsi untuk menentukan ketersediaan layanan dan latensi, termasuk pengukuran ketersediaan distribusi dan latensi serta pelaporan hasil. 2) Pengumpulan data: berfungsi mengumpulkan, menyimpan, dan melaporkan hasil data yang telah dikumpulkan dari node protokol termasuk SNMP, HTTP, dan NS client. 3) *Thresholding* berfungsi untuk mengevaluasi data *latency* yang disurvei atau data kinerja yang dikumpulkan. 4) *Event management* berfungsi menerima kejadian, baik internal maupun eksternal, termasuk melalui perangkat SNMP, 5) Alarm dan otomatisasi berfungsi mengurangi kejadian menurut *scripting* tindakan otomatis berpusat di sekitar alarm, dan 6) Pemberitahuan bertugas untuk memberitahu tentang kejadian penting dengan mengirimkan melalui *e-mail*, XMPP, atau cara lain

3. METODE PENELITIAN

3.1 Rancangan Penelitian

Gambaran umum dari rancangan penelitian dapat dilihat pada Gambar 5, yaitu terdiri dari sebuah server monitoring OpenNMS yang nantinya dapat memantau secara keseluruhan perangkat jaringan yang ada dan terhubung melalui router.



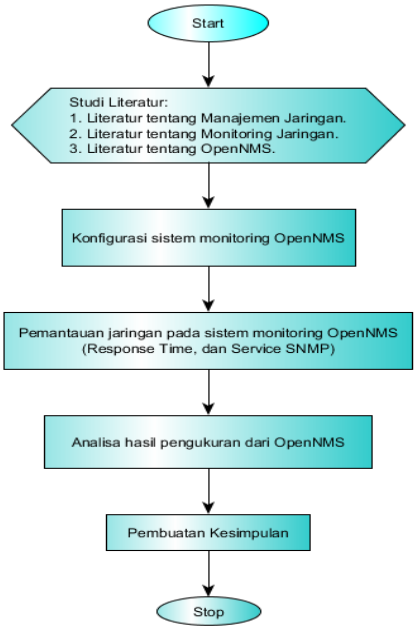
Gambar 5. Gambaran Umum *Monitoring* Jaringan OpenNMS

3.2 Alur Analisis

3.2.1 Alur Analisis Umum

Setelah cukup mendapatkan studi literatur yang dibutuhkan, tahap selanjutnya adalah melakukan konfigurasi OpenNMS dengan jaringan yang ada di Universitas Udayana. Setelah tahap konfigurasi selesai, selanjutnya dilakukan pemantauan jaringan berupa *response time* dari setiap *service* yang ada

dijaringan, yaitu ICMP *response time*, SSH *response time* dan juga service SNMP. Kemudian setelah data didapat, dilakukan analisa terhadap jaringan dan selanjutnya pembuatan kesimpulan dari hasil analisa tersebut. Untuk lebih jelas, dapat dilihat pada flowchat Gambar 6.



Gambar 6. Alur Analisis Umum

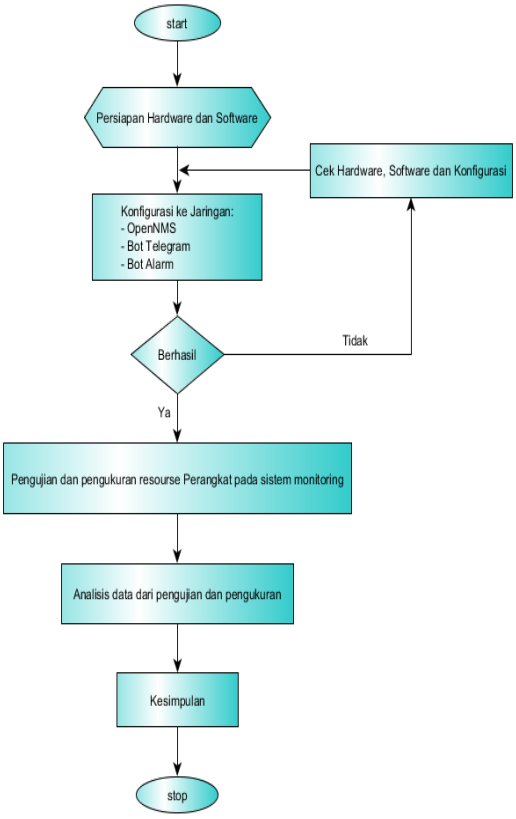
3.2.2 Alur Konfigurasi Sistem *Monitoring* OpenNMS

Pada Gambar 7, flowchart ini menjelaskan bahwa tahap pertama yang dilakukan yaitu mempersiapkan perangkat *hardware* berupa PC *server*, perangkat jaringan, Bot Telegram, Bot alarm dan *software* OpenNMS. Setelah semua perangkat siap, tahap berikutnya adalah menginstall dan mengkonfigurasi OpenNMS, Bot Telegram, Bot alarm ke jaringan. Selanjutnya, mengecek *hardware* dan *software* apakah sudah berjalan dengan semestinya, jikalau belum, dilakukan pengecekan kembali terhadap perangkat yang digunakan dan juga konfigurasi yang dilakukan. Dan jika semua perangkat yang dipergunakan sudah berjalan dengan semestinya, masuk ketahap selanjutnya yaitu melakukan pengujian dan pengukuran *resourse* jaringan pada sistem *monitoring* OpenNMS.

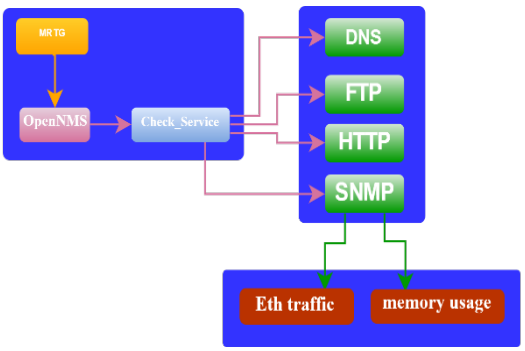
Dan untuk arsitektur yang digunakan pada sistem *monitoring* OpenNMS, dapat dilihat pada Gambar 8 yang dapat dijelaskan :

- Sistem *monitoring* OpenNMS akan memonitor jaringan TCP/IP dengan cara men-*check* service apa saja yang ada dijaringan tersebut.
- Hubungan antara MRTG dengan OpenNMS bersifat independen dan berfungsi sebagai *monitoring* jaringan yang terintergrasi dengan OpenNMS.

Pada service SNMP, OpenNMS dapat memonitor penggunaan beban *memory* dan trafik yang terjadi pada setiap perangkat yang sudah memiliki service tersebut.

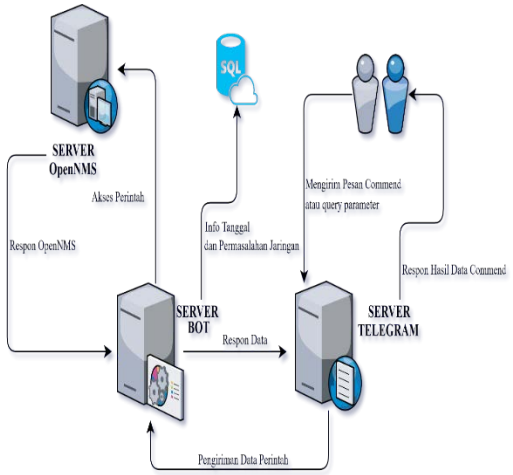


Gambar 7. Alur Konfigurasi Sistem *Monitoring* OpenNMS



Gambar 8. Arsitektur Sistem *Monitoring* OpenNMS

Untuk Alarm Bot Telegram mekanisme proses kerja secara kontinu. Dimana pada Bot tersebut melakukan permintaan data alarm kepada server OpenNMS setiap 10 detik. Setelah data didapat, Bot akan menyimpan data tersebut ke *database cloud* dan juga mengirim hasil data ke pengguna melalui server Telegram. Pada Bot alarm ini juga diatur hanya mengambil data alarm yang bersifat *severity* Major ke atas. Lebih jelas, dapat dilihat pada Gambar 9.



Gambar 9. Arsitektur Sistem Bot Telegram

4. HASIL DAN PEMBAHASAN

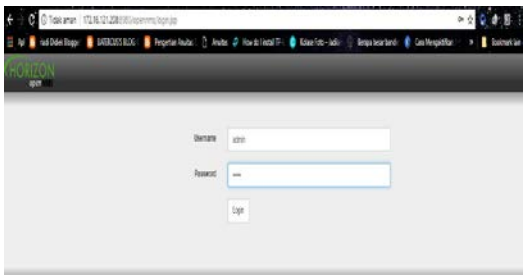
4.1 Hasil Instalasi dan konfigurasi

4.1.1 Instalasi OpenNMS

Pada tahap instalasi OpenNMS, dilakukan beberapa tahapan sebagai berikut :

1. Instalasi *Operating System* ubuntu *server* 14.04 pada komputer *server*.
2. Instalasi perangkat lunak pendukung lainnya agar dapat menjalankan OpenNMS, seperti: JDK (*Java Development Kit*), tomcat dan postgreSQL.
3. Instal snmp dan snmpd
4. Instalasi dan konfigurasi OpenNMS.

Untuk hasil dari instalasi OpenNMS dapat dilihat pada Gambar 10.



Gambar 10. Hasil Instalasi OpenNMS

4.1.2 Konfigurasi *Node* dengan OpenNMS

OpenNMS tidak dapat langsung menampilkan *node-node* yang terdapat pada jaringan yang akan dimonitor. Perlu dilakukan beberapa konfigurasi lanjutan agar dapat melakukan proses pemantauan jaringan.

1. Konfigurasi Discovery
   Pada halaman OpenNMS masuk pada menu *Configure Discovery*, untuk jangkauan *discovery* masukan alamat IP yang akan dimonitor.
2. Konfigurasi Map
   Setelah instalasi OpenNMS fitur *map* tidak akan diaktifkan, sehingga *taskbar* tidak akan terlihat link untuk *map* dari jaringan.

sebelumnya dilakukan konfigurasi untuk menampilkan *map*, yaitu:

1. Semua operasi pendukung OpenNMS harus di stop.
2. Pada terminal ketikan perintah “cp /etc/opennms/map.disable /etc/opennms/map.enable”
3. Jalankan kembali operasi yang mendukung OpenNMS.

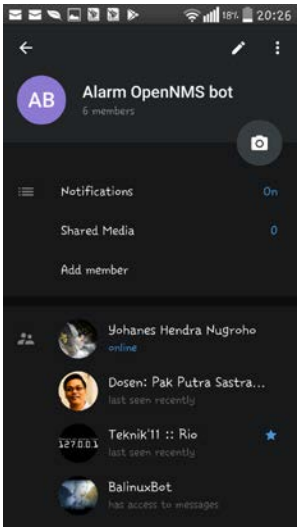
Untuk hasil konfigurasi node dengan OpenNMS dapat dilihat pada Gambar 11.

Gambar 11. Hasil Konfigurasi *Node* dengan OpenNMS

4.1.3 Pengaturan Alarm Bot Telegram

Tujuan dari dibangun Bot Telegram yaitu untuk memberikan informasi *alarm* yang terjadi pada jaringan Universitas Udayana yang terpantau melalui OpenNMS. *Alarm* OpenNMS berisi *node-node* yang mengalami permasalahan pada jaringan. Pada Bot Telegram ini bertugas mengirimkan informasi *alarm* hanya pada status “Major” dan “Critical”, dikarenakan pada status ini jaringan sudah *down* dan harus segera ditindaklanjuti. Untuk Bot Telegram sendiri dibangun menggunakan *framework javascript* yaitu *Node*JS. Dan media penyimpanan data menggunakan *service* dari google yaitu firebase firestore.

Untuk hasil dari pengaturan Bot Telegram, dapat dilihat pada Gambar 12.



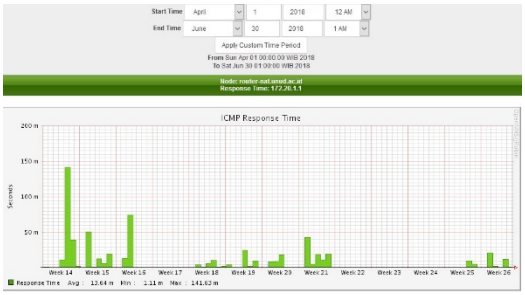
Gambar 12. Hasil Pengaturan Alarm Bot Telegram

4.2 Pembahasan

4.2.1 Fungsionalitas Monitoring

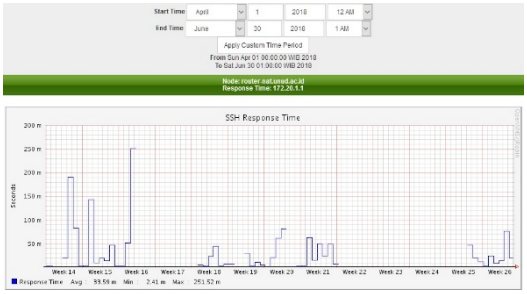
Pada hasil ini akan melihat kinerja dari pemantauan jaringan OpenNMS berupa ICMP *response time,* SSH *response time* dalam bentuk *resourse graph.*

Gambar 13 adalah hasil *monitoring* OpenNMS untuk *response time* pada *service* ICMP dari *node* router-nat.unud.ac.id. Data hasil *monitoring* tersebut ditampilkan dalam bentuk *Resourse Graph* yang merupakan data *monitoring* dari tanggal 1 April 2018 sampai dengan 30 Juni 2018. Dari hasil *monitoring* tersebut, dapat dilihat bahwa service ICMP untuk rata-rata *response time* pada *node* router-nat.unud.ac.id adalah 13,64 ms dengan maksimal *response time* yang terjadi adalah 141,63 ms dan untuk minimal *response time* pada *node* router-nat.unud.ac.id adalah 1,11 ms.



Gambar 13. *Resourse Graph* ICMP *Response Time* router-nat.unud.ac.id

Gambar 14 adalah hasil *monitoring* OpenNMS untuk *response time* pada *service* SSH dari *node* router-nat.unud.ac.id. Dari data hasil *monitoring* tersebut, dapat dilihat bahwa *service* SSH untuk rata-rata *response time* pada *node* router-nat.unud.ac.id adalah 33.,59 ms dengan maksimal *response time* yang terjadi adalah 251,52 ms dan untuk minimal *response time* pada *node* router-nat.unud.ac.id adalah 2,41 ms.



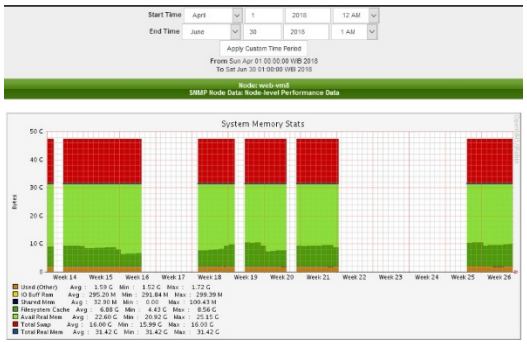
Gambar 14. *Resourse Graph* SSH Response Time Switch router.nat.unud.ac.id

Gambar 15 adalah hasil *monitoring* OpenNMS untuk *timeout* TCP Open Connections dari *node* web-vm8. Berfungsi sebagai informasi berapa waktu yang diperlukan oleh *node* web-vm8 untuk memperbaiki data yang rusak dan mengirimkannya kembali setelah data berhasil diperbaiki. Dari data hasil *monitoring* tersebut, dari data rata-rata *timeout* TCP Open Connections adalah 114.9 ms yang didapat dari selisih *time out* rata-rata sebesar 429,28 ms dan *time in* rata-rata sebesar 314,34 ms. Dan untuk *timeout* maksimal yang terjadi adalah 405 ms, yang dimana didapat dari selisih *time out* maksimal sebesar 1,10 s dan *time in* maksimal sebesar 695 ms.

Gambar 15. *Resourse Graph service* SNMP untuk TCP Open Connections pada web-vm8

Gambar 16 adalah hasil *monitoring* OpenNMS untuk *system memory status*

dari *node* web-vm8. Fungsi dari data *monitoring* ini adalah untuk me-*monitor* beban memori dari web-vm8, sehingga hasil dari data *monitoring* tersebut dapat dijadikan acuan bagi administrator untuk perlu atau tidak meningkatkan kinerja memori pada web-vm8. Dapat dilihat bahwa *node* web-vm8 memiliki memori total sebesar 31.42 GB dengan memori yang sudah terpakai rata-rata sebesar 6,88 GB dan memori terbesar yang digunakan adalah 8,56 GB dalam kurun waktu yang dipantau. Pada grafik tersebut juga dapat dipantau untuk memori yang masih dapat digunakan yaitu sebesar 22.60 GB.



Gambar 16. *Resourse Graph Service* SNMP untuk System Memory status pada web-vm8

4.2.2 Fungsionalitas *Event Management* pada OpenNMS

Pada hasil ini akan melihat kinerja dari pemantauan jaringan OpenNMS berupa *monitoring event management* yang terjadi pada jaringan di Universitas Udayana.

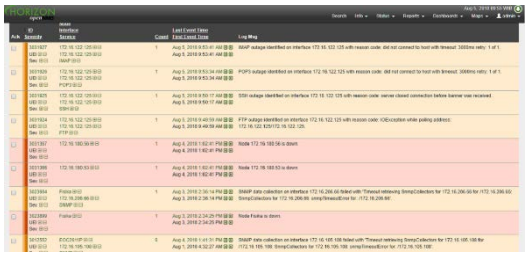
Gambar 17 merupakan *event management* dari hasil *monitoring* OpenNMS terhadap perangkat dan jaringan yang ada di Universitas Udayana. Pada tampilan ini, data hasil *monitoring* OpenNMS berupa ID *event*, status *severity*, *time* saat terjadi *event*, *node*, nomor *interface*, *service* yang berjalan dan keterangan notifikasi singkat dari *node*. Fungsi *event management* dari hasil monitoring OpenNMS adalah agar administrator dapat mengetahui *event-event* apa saja yang terjadi selama termonitor, dengan begitu administrator dapat menjadikan hasil tersebut sebagai bahan pertimbangan apakah jaringan sudah bekerja dengan baik atau perlu dilakukan optimalisasi. Dan dari hasil *monitoring* OpenNMS terhadap jaringan Universitas Udayana, rata-rata berjalan dengan normal dan jika ada *node* yang mengalami *severity minor* maka dengan otomatis akan memperbaiki data dan *node* akan kembali normal.

Gambar 17. Tampilan Hasil *Monitoring Event Management* OpenNMS

4.2.3 Fungsionalitas *Alarm* pada OpenNMS

Pada hasil ini akan melihat kinerja dari pemantauan jaringan OpenNMS berupa informasi *alarm* jika terjadi permasalahan pada jaringan di Universitas Udayana.

Gambar 18 merupakan tampilan hasil *monitoring* jaringan yang dimana pada saat jaringan yang dimonitor sedang mengalami permasalahan. Untuk kolom *alarm* ini akan menyimpan data untuk status *minor* ke atas yaitu *minor, mayor* dan *critical.* Data yang disimpan berupa ID *severity*, status *severity, node, interface*, *service count, last and first event time*, *Log Msg*. Data ini akan terus tersimpan sampai *node* kembali normal. Pada kolom *Alarm* terdapat *node* dengan *severity Major*. Pada *severity* ini, mengindikasikan bahwa *node* tersebut sedang mengalami *down* dan harus ditindaklanjuti agar *node* tersebut kembali berjalan normal.

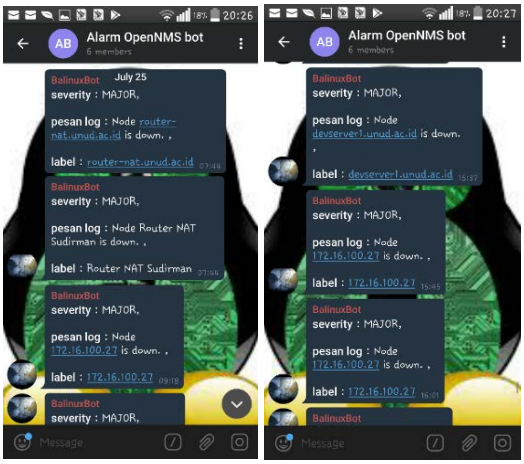


Gambar 18. Tampilan Hasil *Alarm* OpenNMS

4.2.4 Fungsionalitas Monitoring *Alarm* Bot Telegram

Pada hasil ini akan melihat kinerja dari *Alarm* Bot Telegram berupa informasi data alarm yang terpantau oleh OpenNMS terhadap jaringan Universitas Udayana.

Dapat dilihat dari Gambar 19, untuk hasil *monitoring* OpenNMS pada jaringan yang ada di Universitas Udayana, terdapat jaringan yang sedang mengalami *down*. Jaringan yang *down* ini dikumpulkan oleh OpenNMS dalam kolom *Alarm,* sehingga Bot mengambil daftar *alarm* dan mengirimkan ke grup telegram secara otomatis. Informasi *alarm* yang terjadi pada *node* adalah tidak terlalu detail seperti ditampilkan pada Bot, Bot ini hanya menampilan *severity*, pesan *log* dan *label*.



Gambar 19. *Alarm* Bot Telegram

5. KESIMPULAN

Kesimpulan dari analisis unjuk kerja pemantauan jaringan TCP/IP pada Universitas Udayana dengan menggunakan OpenNMS adalah sebagai berikut.

Dari hasil dan pembahasan pada monitoring OpenNMS, untuk resourse graph ICMP response time dan SSH response time pada tiap perangkat sudah dapat terpantau. Dan dari hasil tersebut jika perangkat atau jaringan sedang mengalami down, akan langsung terpantau dan ditampilkan dalam kolom “alarm”.

Alarm Bot Telegram memiliki fungsi untuk mengambil hasil data alarm di OpenNMS. Hasil data alarm akan ditampilkan dalam bentuk chat text dengan rincian informasi severity yang terjadi, pesan log yang berisi status node saat itu dan label yang berisi nama node.

6. DAFTAR PUSTAKA

[1] O’Donnell, Shane. 2000. *Network Management: Open Source Solutions to Proprietary Problems.* Cary, NC: 975 Walnut St., Suite 242.

[2] Li, Harry Ph.D., and Chen, Guangjing. 2005. *Wireless LAN Network Management System.* San Jose: Computer Engineering Department, Collage of Engineering, San Jose State University.

[3] Shihada, Basem. 2002. *Conceptual & Concrete Architectures of Open Network Management System (OpenNMS)*. Ontario, Canada: University of Waterloo, Dept. of Computer Science.

[4] Goyal, P., Mikkilineni, R. and Ganti, M. 2009. FCAPS *in the Business Services Fabric Model*. in Enabling Technologies: Infrastructures for Collaborative Enterprises. WETICE ’09. 18th IEEE International Workshops on pp.45-51.

[5] “SNMP Research International, Inc.” [Online]. Available: www.snmp.com. [Accessed: 22-Feb-2018]

[6] Mauro, D., and Schmidt, K. 2005. *Essential* SNMP. O’Reilly, Sebastopol, CA.

[7] “Tutorials OpenNMS.” [Online]. Available: https://wiki.opennms.org/wiki/Tutorials. [Accessed: 22-Feb-2018]

[8] Wiguna, I Ketut Mustika. 2008. *Analisis Unjuk Kerja Network Monitoring Nagios pada Jaringan TCP/IP.* Bukit Jimbaran, Bali: Jurusan Teknik Elektro, Fakultas Teknik, Universitas Udayana.