DETEKSI ANOMALI UNTUK IDENTIFIKASI BOTNET KRAKEN DAN CONFICKER MENGGUNAKAN PENDEKATAN RULE BASED

Aisyatul Karima
Fakultas Ilmu Komputer, Universitas Dian Nuswantoro, Semarang 50131
E-mail : aisya@research.dinus.ac.id

ABSTRAK

Attack yang tersebar di internet terdiri dari berbagai macam tipe. Salah satu jenis attack yang populer adalah botnet. Botnet bisa menyebar dalam jaringan komputer tanpa diketahui siapa usernya dan di mana lokasinya. Serangan ini akan menyerang kapanpun dan di manapun, sehingga menyebabkan ketidaknyamanan pengguna layanan internet. Oleh karena itu, sangat diperlukan rule based yang mampu mendeteksi jaringan yang terinfeksi botnet maupun yang tidak. Botnet terdiri dari berbagai jenis dengan perilaku masing-masing yang menyebabkan user kesulitan untuk mengklasifikasi tipe botnet yang bisa digunakan dalam deteksi botnet. Peneliti mengusulkan rule based instrusion detection yang baru untuk mendeteksi botnet khususnya untuk deteksi kraken dan conficker menggunakan deteksi anomali. Rule based diperoleh dari trafik jaringan yang nyata melalui teknik observasi. Dengan observasi tersebut, rule set mampu memberikan hasil yang signifikan dibandingkan dengan implementasi Intrusion Detection System, seperti yang tedapat pada Snort.

Kata kunci : Botnet, conficker, kraken, rule based, Intrusion Detection System (IDS).

1. PENDAHULUAN

Kebutuhan jaringan komputer telah tumbuh pesat di berbagai sektor kehidupan. Oleh karena itu orang yang tidak bertanggung jawab akan memanfaatkan perkembangan teknologi ini. Keamanan adalah hal paling penting dan vital dalam jaringan komputer. Sejumlah attack yang menyerang data membuat user merasa tidak nyaman. Attack ini terdiri dari berbagai macam tipe, salah satunya yaitu *botnet*. Dalam sebuah penelitian menunjukkan bahwa sekitar 40% dari 800 juta komputer yang terhubung ke internet terinfeksi *botnet* [1].

Botnet adalah jaringan komputer yang terinfeksi berbagai macam serangan [2]. Komputer yang terinfeksi ini disebut zombie dan mereka akan dikontrol oleh botmasters [3]. Botmaster mampu mengirimkan commands untuk zombie ini dengan berbagai macam jalan dan meluncurkan berbagai macam jenis tindakan kriminal dalam jaringan seperti stealing personal identity, meluncurkan serangan Distributed denial of service (DDoS), mengirim spam email atau scanning aktivitas [4].

Botnet mampu menyebar dalam jaringan komputer tanpa diketahui siapa user dan di mana lokasinya. Hal inilah yang menyebabkan para pengguna layanan internet merasa kurang nyaman, oleh karena itu *rule based* yang mampu mendeteksi jaringan apakah terserang *botnet* atau tidak sangat diperlukan.

Pada saat ini, perilaku dari jaringan komputer dianggap sebagai salah satu indikator untuk mengetahui berbagai jenis bots dalam jaringan, namun sangat jarang ditemukan untuk mendeteksi serangan yang paling berpotensial dari jenis *kraken* dan *conficker*. Investigasi perilaku dari jenis *botnet* terhadap pembangunan *rule based* sangat diperlukan untuk meningkatkan performansi deteksi *botnet* terutama untuk deteksi *kraken* dan *conficker*. Pembangunan *rule based* diperlukan untuk mengetes trafik jaringan yang ada untuk mengukur performansi deteksi terhadap tipe *botnet* yang baru yaitu *kraken* dan *conficker*.

Paper ini terdiri dari lima bagian. Bagian pertama adalah pendahuluan tentang serangan *botnet*. Bagian kedua menjelaskan tentang landasan teori tentang penelitian ini. Bagian ketiga, peneliti mengenalkan metode deteksi *botnet*. Bagian keempat, peneliti menjelaskan tentang analisa *rule based* yang digunakan untuk identifikasi *botnet kraken* dan *conficker*. Bagian kelima adalah kesimpulan dan harapan ke depannya.

2. LANDASAN TEORI

Keamanan adalah hal yang sangat penting dan vital dalam jaringan komputer. Masalah muncul ketika sejumlah attack menyerang jaringan, khususnya *botnet*. Penyebaran aktivitas illegal seperti peluncuran beberapa serangan DDoS terhadap

target, penyebaran malware, pishing and clicking fraud adalah skema yang dibuat oleh *botnet*. Oleh karena itu *botnet* menjadi serangan yang paling serius terhadap kemanan dunia cyber. *Botnet* memiliki banyak tipe dengan perilakunya masing-masing. Hal inilah yang menyebabkan kesulitan dan menjadi tugas yang sangat kompleks untuk mengetahui jaringan mana yang terinfeksi *botnet*. Untuk mendeteksi perilaku *botnet* menggunakan anomaly based detection, dan untuk mengklasifikasikan tipe *botnet* menggunakan signature based detection.

Karena mereka melakukan beberapa aktivitas yang membahayakan untuk memperoleh keuntungan [5], mereka membangun teknik deteksi *botnet* secara kontinyu. Untuk menghindari aktivitas yang membahayakan ini, mesin yang terinfeksi *botnet* dalam jaringan harus segera ditemukan untuk dihilangkan. Sebuah organisasi atau instansi yang telah terinfeksi *botnet* dalam jaringan mereka akan kehilangan reputasinya dan akan menjadi issue terbesar dalam dunia [4].

Intrusion Detection System (IDS) digunakan untuk mendeteksi *botnet* dan untuk mencegah attack. Tujuan dari IDS adalah untuk mengidentifikasi peristiwa dari pelanggaran keamanan yang sesuai dengan kesepakatan kesatuan sumber daya atau layanan [6]. Kata kunci yang digunakan dalam IDS adalah resiko, kerentanan, attack, penetrasi, intrusion dan intrusion detection.

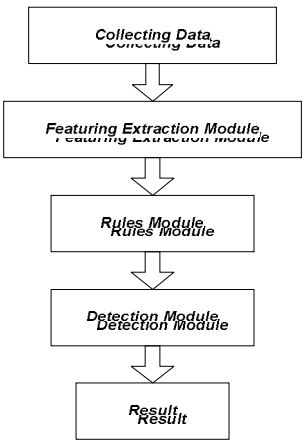
Untuk menganalisa jaringan dari infeksi *botnet*, pengujian jaringan harus dilakukan. Sebuah piranti juga diperlukan untuk implementasi deteksi *botnet* berdasarkan *rule*, yang diperoleh dengan mengamati perilaku dan klasifikasi tipe *botnet*. Dengan menggunakan *rule* based untuk deteksi *botnet* mampu membantu peneliti untuk memonitor jaringan terhadap serangan dari luar.

Penelitian sebelumnya [7] menggunakan syistem yang sudah expert yang memanfaatkan Artificial Intelligence (AI) dalam IDS. Sistem tersebut menggunakan *rule based* intrusion detection dan model pencegahan untuk sistem biometric. Perilaku metode *rule based* intrusion detection yang telah digunakan untuk menganalisa korelasi dari perilaku komunikasi dengan *rule*s yang telah disebutkan [8]. Metode yang digunakan dalam penelitian sebelumnya mampu untuk mendeteksi trafik jaringan yang sudah terenkripsi dari software P2P, aktivitas malware yang tidak diketahui dan infeksi malware melalui web browser, namun tidak dapat membedakan tipe *botnet*. Berdasarkan pada [9], untuk mengidentifikasi trafik anomali dengan keamanan jaringan sebagai kunci area aplikasinya, paket klasifikasi *rule based* adalah salah satu dari metode yang ampuh, dalam penelitian sebelumnya hanya fokus untuk mengidentifikasi trafik anomali tanpa ada identifikasi tipe dari *botnet* itu sendiri.

Pendekatan *rule based* yang digunakan dalam penelitian ini berbeda dengan penelitian sebelumnya. Penelitian ini tidak hanya mendeteksi *botnet* secara global, namun juga mendeteksi tipe *botnet* yang lebih spesifik yaitu *kraken* dan *conficker*. *Rule based* yang dihasilkan dari trafik network yang nyata melalui teknik observasi terlebih dahulu. Dengan observasi trafik network yang nyata tersebut, *rule* set yang dihasilkan mampu memberikan hasil yang signifikan dibandingkan dengan implementasi IDS yang sudah ada, seperti Snort.

3. METODE PENELITIAN

Metode penelitian yang peneliti usulkan adalah berdasarkan trafik jaringan monitoring pasif. Berdasarkan gambar 1 ditunjukkan bahwa arsitektur dari *rule based* yang diusulkan untuk metode deteksi *botnet*.



Gambar 1: Metode Penelitian

3.1 Collecting Data

Untuk pertama kalinya, peneliti mengumpulkan data dari trafik jaringan. Monitoring trafik bertugas untuk mendeteksi group host yang memiliki perilaku yang sama dan pola komunikasi dengan mengamati trafik jaringan. Dalam penelitian ini, peneliti memberikan dua versi dari trafik jaringan yaitu trafik normal dan trafik abnormal.

Peneliti mengambil data besar dalam format file tcpdump, dan membaginya ke dalam lima bagian. Masing-masing bagian terdiri dari 100.000 flow trafik abnormal. Bagian-bagian tersebut masing-masing memiliki ukuran 60.000 KB. Tiga bagian dari lima bagian tersebut digunakan untuk menganalisa proses dan yang lainnya digunakan untuk proses perbandingan. Untuk memecah file menggunakan libcap library dan menggunakan wireshark untuk membaca format file pcap. Selain trafik abnormal, peneliti juga menggunakan tiga file normal untuk membedakan antara perilaku abnormal trafik dan normal trafik. File Trafik normal yang sudah dipecah tersebut masing-masing memiliki ukuran 10.000 KB.

3.2 Featuring Extraction Module

Peneliti memilih feature yang akan digunakan untuk mendeteksi *botnet* dengan feature selection. Feature diambil dari KDD 1999 yang menceritakan tentang feature *dataset* dari *intrusion detection.* Berdasarkan beberapa feature dan penjelasan tentang feature sesuai dengan karakteristik *botnet* attack, terdapat 6 feature yang telah dipilih dalam penelitian ini. Keenam feature tersebut adalah *source IP* [9], *destination IP* [9], *source port* [10], *destination port* [10], *protocol* [9], *dan byte size* [9]. Setelah proses *feature selection*, maka dibutuhkan *feature extraction module* untuk mentransformasikan input data ke dalam data set dari feature yang sudah disebutkan di atas.

3.3 Rule Module

Setelah selesai proses *featuring extraction module*, bisa ditentukan *rule* yang bisa digunakan sebagai *rules module* untuk deteksi *botnet*. *Rule-rule* ini merupakan jenis *rule based* dari deteksi *botnet* berdasarkan *featuring extraction module*. Dalam tahap ini, menggunakan pendekatan *rule based* untuk menentukan beberapa *rule* dari deteksi *botnet.*

Pendekatan *rule based* yang digunakan pada penelitian ini berbeda dengan penelitian sebelumnya. Penelitian ini tidak hanya mendeteksi *botnet* secara global, tetapi juga untuk mendeteksi tipe *botnet* yaitu *kraken* dan *conficker*. *rule* ini terdiri dari framework untuk mengidentifikasi *kraken* dan *conficker*. *Rule* ini diperoleh dari hasil analisa perilaku *kraken* dan confciker yang diambil dari beberapa feature untuk mengidentifikasi keduanya. Dalam penelitian ini, penggunaan pendekatan *rule based* untuk mengecek perbedaan log antara trafik normal dan abnormal, serta untuk mengklasifikasikan trafik abnormal ke dalan tipe *botnet*, apakah itu termasuk *kraken* atau *conficker*. Dalam *rule based* ini juga menggunakan standar alarm dan laporan yang akan dieksekusi jika aktivitas normal. Mesin *rule* akan mengecek *rule* untuk mendeteksi poin intrusion dan tipe dari intrusion jika aktivitas abnormal ditemukan.

3.4 Detection Module

Aplikasi akan mendeteksi trafik jaringan, sehingga attack atau serangan yang berbahaya yang melalui jaringan bisa diobservasi. Deteksi bekerja dengan menjalankan trafik jaringan ke dalam aplikasi yang sudah dbangun dengan beberapa *feature selection* yang mempunyai beberapa *rule module*. Untuk membangun aplikasi yang mampu mendeteksi *botnet*, peneliti menggunakan algoritma tentang proses analisa untuk mendapatkan *botnet kraken* dan *conficker*.

3.5 Result

Dari *detection module* yang diperoleh menggunakan *rule based* yang sudah diimplementasikan, peneliti bisa memperoleh hasil akhir. Hasil ini menunjukkan komputer mana yang terdeteksi *botnet* sesuai dengan feature yang sudah disebutkan di atas. Dari hasil penelitian, bisa diketahui tipe *botnet* apa yang telah menyerang, dan hasilnya berupa IP address mana yang terinfeksi *botnet kraken*, dan IP address mana yang terinfeksi *botnet conficker.*

4. HASIL DAN ANALISA

4.1 Analisa Perilaku Dari Kraken dan Conficker

Berdasarkan tujuan dari penelitian ini yang fokus pada deteksi perilaku *botnet* khususnya *kraken* dan *conficker*. Tabel 1 di bawah ini menunjukkan perilaku dari *kraken*. Terdapat beberapa feature yang menjadi indikator utama untuk menentukan jaringan yang terserang *kraken*. Berdasarkan tabel 1, feature yang digunakan untuk mengidentifikasi *kraken* adalah *destination port, byte size dan protocol*. masing-masing feature memiliki parameter sendiri. Untuk mengidentifikasi *kraken*,

destination port yang digunakan adalah port 447, 443 and 137. Selain itu, *byte sizes* yang digunakan adalah 24, 74, 66, 115, 116 and 117 bytes. Protocol yang digunakan adalah TCP dan UDP.

Tabel 1: Perilaku kraken

No	*Feature*	*Item*	*Description*
1.	*Destination port*	*447*	*Port ddm-dfm (Distributed File Management)*
		443	*Https webserver (SSL)*
		137	*Netbios name service (Windows)*
2.	*Byte size*	*24*	*Bytes*
		74	*Bytes*
		66	*Bytes*
		115	*Bytes*
		116	*Bytes*
		117	*Bytes*
3.	*Protocol*	*TCP*	*Transmission Control Protocol*
		UDP	*User Datagram Protocol*

Tabel 2 di bawah menunjukkan perilaku dari *conficker*. Terdapat beberapa feature yang menjadi indikator utama untuk menentukan jaringan yang terserang *conficker*.

Tabel 2: Perilaku conficker

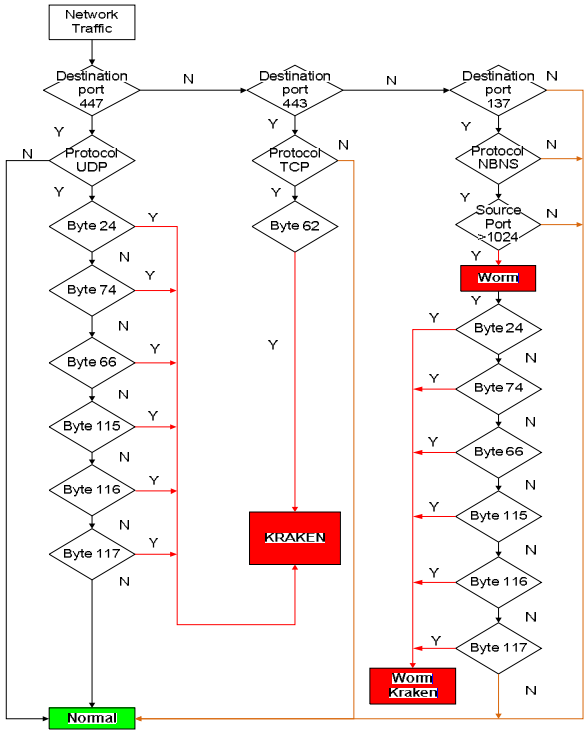
No.	*Feature*	*Item*	*Description*
1.	*Destination port*	*137*	*NBNS - NetBIOS Name Service*
		138	*Browser - NetBIOS Datagram Service*
		139	*TCP - NetBIOS Session Service*
		445	*Microsoft - DS*
		Between 1024-10000	
2.	*Source port*	*53*	*Domain Name Server (DNS)*
		Between 1024-10000	
3.	*Protocol*	*TCP*	*Transmision Control Protocol*
		DNS	*NetBios Name Service*
		NBNS	*NetBios Name Service*
		NB-dgm	*NetBios Datagram Service*
		NB-ssn	*NetBios Session Service*

Berdasarkan table 2, feature yang digunakan untuk mengidentifikasi *conficker* adalah *destination port, source port dan protocol*. Masing-masing feature memiliki perilaku sendiri. Untuk mengidentifikasi *conficker, destination port* yang digunakan adalah port 137, 138, 139, 445 dan port antara 1024 dan 10000. Selain itu, *byte source port* yang digunakan adalah 53 dan port antara 1024 dan 10000. *Protocol* yang digunakan adalah NBNS, NB-dgm dan NB-ssn.

4.2 Membuat Rule Module berdasarkan Hasil Analisa Perilaku

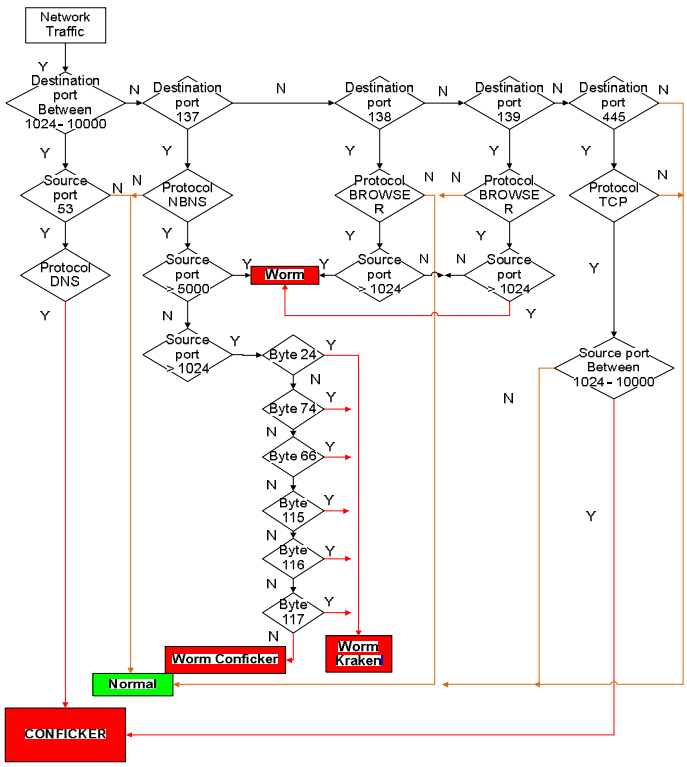
Berdasarkan gambar 2, untuk mengidentifikasi *kraken* menggunakan 3 indikator *destination port*. Jika destination port adalah 447 dan *protocol* UDP, harus mengikuti kondisi dengan memilih *byte size* mana yang dimiliki trafik. Terdapat beberapa pilihan yaitu 24 bytes, 74 bytes, 66 bytes, 115 bytes, 116 bytes dan 117 bytes. Ketika trafik termasuk salah satu dari pilihan tersebut, maka trafik tersebut terinfeksi *kraken*. Selain itu ketika tidak ada *byte size* yang sesuai, maka trafik normal. Selain port 447, *kraken* bisa diindikasikan dengan port 443,ketika *protocol* TCP dan byte size adalah 62 bytes. Selain itu, ketika kondisi tersebut tidak terpenuhi maka trafik tersebut normal.

 ISBN 979 - 26 - 0255 - 0



Gambar 2: *Rule* untuk menentukan karakteristik *botnet kraken*

Berikut ini dalam gambar 3 akan dijelaskan *rule* untuk menentukan karakteristik *botnet conficker*.



Gambar 3: *Rule* untuk menentukan karakteristik *botnet conficker*

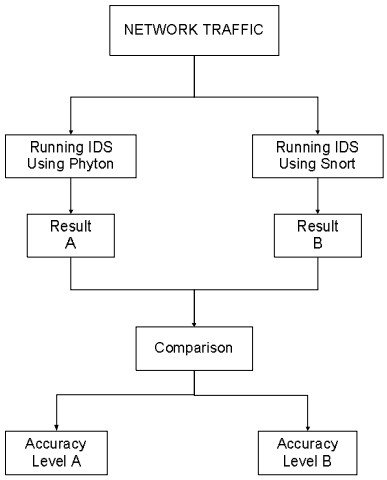
Dibandingkan dengan *kraken*, untuk mengidentifikasi *conficker*, terdapat beberapa port sebagai indikatornya. Indikator tersebut diantaranya *destination port* antara 1024 sampau 10000, port 137, port 138 dan port 139 dan port 445. Ketika *destination port* antara 1024 sampai 10000 and *source port* 53 dan *protocol* adalah DNS maka trafik tersebut terdeteksi sebagai *conficker*. Ketika *destination port* adalah 137, *protoco*l adalah NBNS (NetBIOS Name Service) dan *source port* melebihi dari 5000, maka trafik tersebut terdeteksi sebagai *worm*. Selain itu, jika *source port* antara 1024 dan 5000 dan *byte*

size adalah 24, 74, 66, 115, 116 atau 117 byte, maka trafik tersebut terdeteksi sebagai *worm kraken*, jika hal itu terjadi dengan kondisi yang sama namun *byte size* yang berbeda dengan yang disebutkan di atas, maka trafik tersebut terdeteksi sebagai *worm conficker*. Kondisi normal dari port 137 terjadi jika trafik tidak memenuhi kondisi seperti yang disebutkan dalam gambar 3.

Rule yang dihasilkan dari hasil analisa perilaku *kraken* dan *conficker* di atas, diimplementasikan ke dalam bahasa pemrograman. Dalam penelitian ini, peneliti menggunakan bahasa pemrograman python dan alat monitoring jaringan Snort. Bahasa python digunakan untuk membuat *rule* berdasarkan analisa perilaku *kraken* dan *conficker*. Sedangkan snort digunakan sebagai bahan pembanding untuk memperoleh nilai tingkat akurasi.

4.3 Proses Testing

Dalam tahap ini, peneliti membuat flow untuk proses testing dan disebutkan dalam gambar 4 berikut ini :



Gambar 4: flow untuk proses testing

Proses running dimulai dengan import file csv yang terdiri dari file trafik jaringan dari *dataset*. File ini sengaja disimpan dalam format csv file dengan tujuan untuk memudahkan python membaca trafik jaringan. Script akan menjalankan proses pengkondisian yang terdiri dari beberapa parameter pengkondisian yaitu *source port, destination port, protocol dan byte size* untuk masing-masing trafik flow. Pengkondisian tersebut terdiri atas perilaku masing-masing dengan kondisi tertentu yang berbeda satu sama lain. Dari proses tersebut, menghasilkan hasil yang berhasil untuk mendeteksi trafik normal dan trafik abnormal, khususnya *kraken* dan *conficker*. Hasil ini berupa file log dari python yang otomatis tersimpan dalam format csv file, peneliti menyebutnya sebagai result A. hasil tersebut terdiri dari flow trafik jaringan dengan beberapa penjelasan, yang mana yang merupakan trafik normal, dan yang mana yang merupakan trafik yang terinfeksi *kraken* maupun *conficker*. Dari hasil ini bisa dilihat prosentase berapa banyak yang terserang *kraken* dan *conficker* serta berapa banyak trafik normal dan abnormal.

Selain menjalankan trafik jaringan menggunakan python, peneliti juga menggunakan *Intrusion Detection System* (IDS) Snort. Ketika menggunakan snort, file log otomatis tersimpan dalam bentuk format file tcpdump atau format file IDS. Setelah mengeksekusi beberapa command pada snort, maka dari 100.000 flow trafik, menunjukkan bahwa status alert dan log sejumlah 1340 (1,333 % dari seluruh flow trafik). Hasil yang dihasilkan command dalam snort tersebut berupa log history yang tersimpan dalam format file IDS. Hasil ini disebut sebagai result B. Untuk dijadikan bahan pembandingan dalam proses komparasi, file result B harus dikonversi ke dalam bentuk format excel.

Dari kedua hasil result A yang dijalankan menggunakan python dan result B menggunakan snort, akan dibandingkan untuk memperoleh tingkat akurasi dari keduanya. Tingkat akurasi akan menunjukkan prosentase berapa banyak trafik jaringan yang berhasil dideteksi sebagai *botnet* attack. Tingkat akurasi menggunakan pengukuran *true positive* dan *false positive*. Dalam penelitian ini, peneliti hanya fokus pada dua kualitas dari IDS, yaitu *true positive rate* dan *false positive rate* [11]. Penjelasan lebih lanjut untuk pengukuran tingkat akurasi akan dijelaskan pada bab selanjutnya.

4.4 Hasil

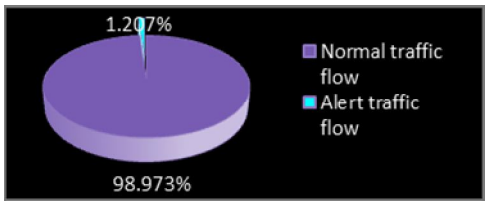
Adapun hasil yang diperoleh dari proses testing dapat dilihat dalam table 3 di bawah ini.

 ISBN 979 - 26 - 0255 - 0

Tabel 3: Result A

Item	*Total*	*%*
Conficker	*1002*	*1.002*
Kraken	*296*	*0.296*
Worm	*24*	*0.024*
Worm Conficker	*1748*	*1.748*
Worm Kraken	*0*	*0*
Normal	*96930*	*96.93*
TOTAL	*100000*	*100*

Table 3 tersebut menunjukkan bahwa dari 100.000 flow trafik jaringan, attack terbanyak adalah *worm conficker* sebanyak 1.748 flow, yaitu sekitar 1,748 %. Dan attack terbanyak kedua adalah *conficker* sebanyak 1002, yaitu sekitar 1,002 %. Lalu diikuti *kraken* sebanyak 296 flow, yaitu sekitar 0,296 %. Dan attack paling jarang dijumpai yaitu *worm*, hanya sekitar 24 atau 0.024 %. Di dalam result A tidak ditemukan *worm kraken*. Selain dari trafik yang terinfeksi attack tersebut, merupakan trafik normal. Total flow trafik jaringan yang ditemukan sebagai total trafik abnormal sekitar 3070 flow. Sama halnya dengan result A yang menggunakan dua dari lima *dataset* yang digunakan dalam proses testing, result B juga menggunakan file yang sama. Hasil dari proses running dengan *dataset* yang sama menggunakan Snort dapat dilihat dalam gambar 5 di bawah ini.



Gambar 5: Result B

Sesuai dengan gambar 5 tersebut, alert trafik flow yang terinfeksi *botnet* lebih kecil dari pada trafik normalnya. Angka tersebut sekitar 1.207 % atau sekitar 1.207 flow trafik dari total 100.000 flow trafk jaringan.

4.5 Pengukuran Tingkat Akurasi

Untuk mengukur tingkat akurasi dapat menggunakan formula berikut ini :

$$\text{True Positive Rate} = \frac{\text{Number of pattern detected as attack}}{\text{Number of input attack pattern}} \quad (1)$$

$$\text{False Positive Rate} = \frac{\text{Number of pattern mistaken as attack}}{\text{Number of input normal pattern}} \quad (2)$$

Dari formula tersebut, hasil pengukuran bisa dilihat pada table 4 di bawah ini.

Tabel 4: Hasil Pengukuran Tingkat Akurasi

No.	*Accuracy Measurement*	*Tools*	
		Python	***Snort***
1	*True Positive rate*	*(18/18) *100 %*	*(3/18) * 100 %*
	Result	***100 %***	***16,6 %***
2	*False Positive rate*	*(6/8) * 100 %*	*(8/8) * 100 %*
	Result	***75 %***	***100 %***

Berdasarkan [11] *true positive rate* dikatakan lebih baik jika mendekati angka 1 (=100 %) dan dalam kualitas teknik bahwa karakteristik dikatakan lebih besar jika memiliki karakteristik yang lebih baik. Sesuai table 4 tersebut, hasil *true positive rate* dari python mendekati 1 (=100 %). Dari total 18 IP address sebagai attack, *rule based* yang merupakan hasil implementasi dalam kode python mampu mendeteksi semua IP address tersebut. Dengan kata lain, *rule based* berhasil mendeteksi *botnet* attack. Dibandingkan dengan *rule based* pada snort yang mendeteksi *botnet* hanya 3 IP address dari 18 IP address. *Rule based* pada snort menghasilkan angka hanya 0,166 (=16,6 %), ini berarti angka tersebut masih jauh dari angka 100 % dan bisa disimpulkan bawa *rule based* dari snort kurang baik. Berdasarkan table 4, hasil perbandingan antara *rule based* dari python lebih baik dari *rule based* dari snort.

Selain *true positive rate*, terdapat pula *false positive rate*. Berdasarkan [11], *false positive rate* dikatakan lebih baik jika mendekati angka 0 (=0 %) dan dalam kualitasi teknik, karakteristik yang lebih kecil itu merupakan karakteristik yang lebih

baik. Berdasarkan table 4, dari 8 total IP address yang yang ditetapkan sebagai IP normal, *rule based* dari python telah mampu mendeteksi 6 normal IP address sebagai attack, hasilnya sekitar 0,75 (75 %). Angka tersebut masih di atas 50 % dan masih jauh dari angka 0 %. Hal ini disebabkan karena feature yang digunakan dalam penelitian ini terbatas pada 6 feature yaitu *source IP, destination IP, protocol, byte size, source port* dan *destination port.* Di sisi lain, *dataset* yang digunakan juga terbatas pada data hasil eksperimen pada laboratorium security dari Universiti Teknikal Malasua Melaka selama 96 jam (4 hari) untuk abnormal trafik dan 24 jam (1 hari) untuk normal trafik. Oleh karena itu, bisa disimpulkan bahwa *rule based* dari python memiliki angka yang bagus pada perhitungan *false positive* dikarenakan angka tersebut tidak sampai mencapai angka 100 %. Selain itu, dibandingkan dengan *rule based* pada snort yang mampu mendeteksi 8 normal IP address dari total 8 IP address yang ditetapkan sebagai attack. *Rule based* dari snort telah dihitung angka *false positive rate*-nya, dan hasilnya adalah 1 (=100 %). Artinya angka tersebut sangat jauh dari angka 0 dan bisa disimpulkan bahwa *rule based* dari snort tidak baik. Berdasarkan table 4, perbandingan antara *rule based* dari python dan *rule based* dari snort menunjukkan bahwa *rule based* dari python lebih baik dari *rule based* dari snort.

5. KESIMPULAN

Penelitian ini berdasarkan pada masalah *botnet* yang terdiri dari berbagai macam tipe dengan perilaku masing-masing yang menyebabkan user kesulitan dalam klasifikasi tipe *botnet* yang digunakan untuk deteksi *botnet*. Keberadaan beberapa IDS yang digunakan untuk mendeteksi *botnet* attack, tidak ada yang mengklasifikasikan kedua jenis *botnet kraken* dan *conficker*. Berangkat dari masalah inilah, peneliti melakukan penelitian dengan tujuan utama untuk menyelesaikan problem tersebut. Oleh karena itu, peneliti mengusulkan *rule based* untuk membantu membedakan antara *kraken* dan *conficker* berdasarkan pada perilaku masing-masing dan mengimplementasikannya ke dalam *rule based* menggunakan kode python untuk mendeteksi *botnet*. Proses testing menghasilkan hasil yang menunjukkan bahwa *rule based* yang diusulkan dan diimplemetasikan dalam python berhasil mendeteksi *botnet* khususnya *kraken* dan *conficker* dalam jaringan.

Penelitian ini tidak hanya berhasil mendeteksi *botnet* attack, namun juga memiliki keterbatasan diantaranya total *dataset* yang digunakan dalam penelitian ini hanya 5 bagian dari file trafik abnormal dan 3 bagian dar file trafik normal. Oleh karena itu, analisa perilaku *kraken* dan *conficker* hanya menggunakan 3 bagian file dari trafik abnormal dan proses testing juga hanya menggunakan 2 bagian file trafik abnormal. Dalam hal ini, bisa disebabkan dari proses analisa yang terbatas pada *dataset* dan kemungkinan terdapat karakter dari perilaku keduanya *kraken* dan *conficker* yang tidak teridentifikasi oleh peneliti. Di sisi lain, peneliti hanya menggunakan 6 feature untuk menganalisa perilaku *kraken* dan *conficker*. Oleh karena itu, proses analisa kurang spesifik. Hal ini menyebabkan *rule* yang dibuat untuk mendeteksi trafik jaringan memiliki parameter yang terbatas.

Untuk penelitian selanjutnya, peneliti berharap ada penelitian yang mendeteksi *botnet* lebih spesifik dibanding dengan penelitian ini. Hal tersebut bisa dilakukan dengan analisa perilaku *kraken* dan *conficker* tidak hanya terbatas pada 6 feature saja, namun lebih dari itu dan menggunakan *dataset* yang lebih bervariasi lagi jenis *botnet*nya. Peneliti berharap tidak hanya untuk mendeteksi *kraken* dan *conficker* saja, namun juga mampu mendeteksi tipe *botnet* lain yang menyerang jaringan.

DAFTAR PUSTAKA

[1] Li, C., Jiang, W., & Zou, X. (2009). *Botnet*: Survey and Case Study. Fourth International Conference on Innovative Computing, Information and Control (pp. 1184-1187). IEEE.
[2] Saha, B., & Gairola, A. (2005). *Botnet*: An Overview. CERT-In White Paper.
[3] Bächer, P., Holz, T., Kötter, M., & Wicherski, G. (2005). Know your enemy: Tracking *Botnet*s. The Honeynet Project. 1-21.
[4] Mizoguchi, S., Kugisaki, Y., Kasahara, Y., Hori, Y., & Sakurai, K. (2010). Implementation and Evaluation of Bot Detection Scheme based on Data Transmission Intervals. (pp. 73-78). IEEE.
[5] Feily, M., Shahrestani , A., & Ramadass , S. (2009). A Survey of *Botnet* and *Botnet* Detection . Third International Conference on Emerging Security Information, Systems and Technologies (pp. 268-273). IEEE.
[6] Sobh, T. S. (2006). Wired and wireless intrusion detection system: Classifications,good characteristics and state-of-the-art. science@direct (pp. 1-25). Computer Standards & Interfaces.
[7] Arjunwadkar, M., & Kulkarni, R. (2010). The *Rule* Based Intrusion Detection and Prevention Model for Biometric System. Journal of Emerging Trends in Computing and Information Sciences, 117-120.
[8] Mizutani, M., Takeda, K., & Murai, J. (2009). Behaviour *Rule* based Intrusion Detection. CoNEXT Student Workshop (pp. 57-58). Rome, Italy: ACM.
[9] Duf eld, N., Haffner, P., Krishnamurthy, B., & Ringberg, H. (2009). *Rule*-Based Anomaly Detection on IP Flows. INFOCOM 2009, IEEE . IEEE.
[10] Tarng, W., Den, L.-Z., Ou, K.-L., & Chen, M. (2011). The Analysis and Identification of P2P *Botnet*'s Traffic Flows. International Journal of Communication Networks and Information Security (IJCNIS) Vol., (pp. 138-148).
[11] Konno, T., & Tateoka, M. (2005). Accuracy Improvement of Anomaly-based Intrusion Detection System using Taguchi Method.