REPOSITOR, Vol. 2, No. 3, Maret 2020, Pp. 339-350
ISSN : 2714-7975
E-ISSN : 2716-1382

Pengelompokan Notifikasi Alert Intrusion Detection System Snort Pada Bot Telegram Menggunakan Algoritma K-Means

Bagus Alfiansyah*[1], Syaifuddin[2], Diah Risqiwati[3]
[1,2,3]Teknik Informatika/Universitas Muhammadiyah Malang
bagus_437149@webmail.umm.ac.id[*1],saifuddin@umm.ac.id[2],risqiwati@umm.ac.id[3]

Abstrak

Dengan semakin luasnya pengetahuan dan meningkatnya kejahatan internet maka dibutuhkan Intrusion Detection System (IDS) salah satunya adalah Snort yang dapat mendeteksi serangan. Dibutuhkan notifikasi serangan agar administrator tahu jika adanya serangan. Pengelompokan alert menggunakan metode K-Means untuk membagi 2 kelompok alert yaitu low dan high. Bot Telegram akan mengirimkan alert yang memiliki label high saja. Notifikasi akan muncul pada aplikasi Telegram. Dataset 4SICS digunakan untuk proses penegelompokan agar menghasilkan 2 centroid yang akan digunakan pada serangan real. Proses pengujian serangan real dilakukan selama 2 hari. Terdapat total 10352 serangan diantaranya 1096 memiliki label high dan 9256 memiliki label low serta terdapat 771 notifikasi yang dikirimkan. Persentase hasil serangan selama satu jam berdasarkan label serangan. 60,38% serangan memiliki label "high" dan 39,62% memiliki label "low". Persentase hasil serangan selama dua hari berdasarkan label serangan. 89% serangan memiliki label "low" dan 11% memiliki label "high".

Kata Kunci: *K-Means, IDS, Snort, Telegram, Bot*

Abstract

With the increasing knowledge and cybercrime, Intrusion Detection System (IDS) is needed. One of which is Snort that can detect the attack. Notification when there is attack is needed so the administrator knows. Alert clustering uses K-Means to divide 2 cluster of alerts namely "low" and "high". Telegram Bots will send alerts that having a "high" label only. Dataset from 4SICS is used for the grouping process to produce 2 centroid that will be used in real attacks. The real attack testing process is carried out for 2 days. There were a total of 10352 attacks including 1096 having a "high" label and 9256 having a "low" label and there were 771 notifications sent. Percentage of results of one hour attack results based on attack labels was 60.38% of attacks had the label "high" and 39.62% had the label "low". Percentage of results of two days attack results based on attack labels was 89% of attacks had the label "low" and 11% had the label "high".

Keywords: *K-Means, IDS, Snort, Telegram, Bot*

1. Pendahuluan

Dalam perkembangan teknologi internet pada jaman sekarang memberikan banyak kemudahan dan manfaat bagi manusia untuk mendapatkan informasi. Pada segi yang lain, muncul sebuah problematika pada keamanan informasi dengan seiring berkembangnya teknologi dan informasi. Mengamankan informasi adalah salah satu tantangan karena meningkatnya ancaman dan serangan yang dilakukan pada keamanan jaringan.

Dengan semakin luasnya pengetahuan tentang kejahatan internet, didukung dengan kemudahan mendapatkan aplikasi atau alat bantu untuk melancarkan para pelaku kejahatan internet untuk melancarkan aksinya. Untuk mengatasi permasalahan tersebut pada setiap jaringan dianjurkan untuk mempunyai Intrusion Detection System (IDS). IDS adalah suatu perangkata atau aplikasi yang mempunyai keahlian untuk mendeteksi serangan secara otomatis pada jaringan serta juga dapat menganalisis jika adanya gangguan terhadap keamanan jaringan pada sistem [1].

Salah satu aplikasi yang dapat difungsikan sebagai IDS adalah Snort yang merupakan aplikasi opensource. Snort dapat mengawasi dan mendeteksi jika terjadi adanya intrusi atau serangan pada jaringan sesuai dengan rule atau ketentuan yang telah ditetapkan sebelumnya [2]. Di samping itu, perlu adanya pengawasan dari seorang administrator keamanan jaringan agar

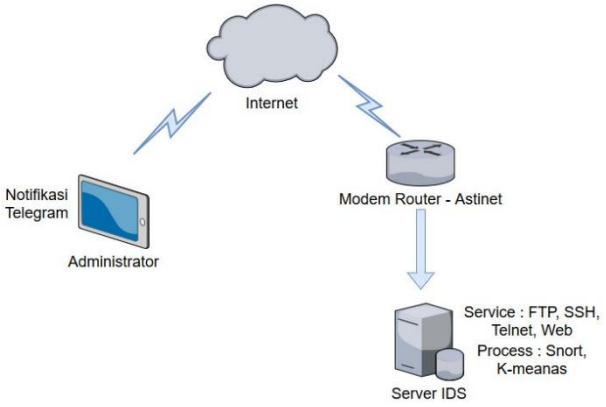
dapat melihat hasil serangan yang tertangkap oleh IDS, tetapi seorang administrator tidak selalu berada di depan layar komputer untuk melakukan pengawasan. Dibutuhkan sebuah notifikasi secara real-time saat serangan terjadi. Pada penelitian ini memanfaatkan aplikasi chatting untuk media notifikasi yang dikirimkan kepada administrator agar dapat mengetahui adanya indikasi serangan.

Aplikasi berbasis chat messaging sedang banyak dan umum digunakan oleh masyarakat. Salah satu aplikasi tersebut yang sedang banyak digunakan adalah Telegram. Aplikasi tersebut merupakan aplikasi chat messaging yang menyediakan Bot yang dapat digunakan untuk pengiriman notifikasi [3]. Penelitian sebelumnya yang sejenis, terdapat platform lain yang telah dimanfaatkan untuk mengirim notikasi *alert* Snort seperti *SMS Gateway* [4]. Tetapi permasalahan muncul saat banyaknya serangan atau alert saat Snort menemukan adanya intrusi akan mengirimkan semua notifikasi, masalah tersebut dapat mengakibatkan administrator kesulitan untuk melihat dan menganalisa serangan melalui notifikasi tersebut. Untuk menangani masalah tersebut diperlukan pengumpulan data log serangan dan kemudian dilakukan Analisa. Pada penelitan sebelumnya [5] metode K-Means digunakan untuk melakukan klasifikasi serangan yang tertangkap oleh Snort berdasarkan jumlah IP Address dan Port.

Pada penelitian ini memanfaatkan metode clustering K-Means. Algoritma tersebut dapat mengelompokkan kedalam beberapa cluster atau kelompok berdasarkan kemiripan dari data tersebut [6]]. Algoritma yang umum digunakan dalam pengaplikasiannya, mampu mengelompokkan objek besar dengan sangat cepat sehingga mempercepat proses pengelompokkan merupakan kelebihan dari K-Means itu sendiri. Pada penilitian ini K-Means digunakan untuk mengelompokkan alert sebelum dikirim agar tidak semua notifikasi yang dikirim oleh Bot Telegram sehingga administrator dapat dengan mudah menganalisa. Hanya alert yang mempunyai label “high” yang akan dikirim menjadi notifikasi melalu Bot Telegram.

2. Metode Penelitian

2.1 Gambaran Umum Sistem



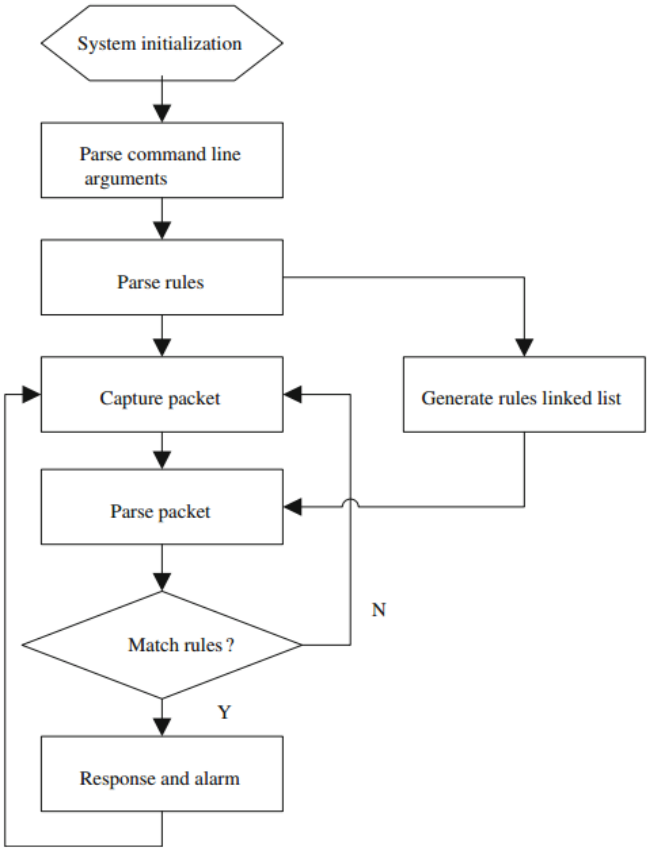
Gambar 1. Gambaran Umum Sistem

Pada Gambar 1, sistem Intrusion Detection System menggunakan Snort yang berfungsi sebagai sistem yang dapat mendeteksi adanya serangan atau intrusi pada Server. Snort tersebut akan menangkap semua trafik jaringan dan dilakukan analisa. Server yang digunakan berbasis virtual machine yang berarti server tersebut berbentuk virtual yang kemampuanya serupa dengan komputer asli atau bukan komputer virtual. Pada Server memiliki layanan Web Server, FTP dan SSH. Pada server tersebut juga terdapat proses pengelompokan alert menggunakan K-Means. Serangan yang telah ditangkap oleh Snort berupa alert yang akan di lakukan pengelompokan dan pemberian label dan dikirm ke admin melalui Telegram yang dapat diakses pada Perangkat Mobile.

2.2 Proses Deteksi Serangan

Intrusion Detection System menggunakan Snort bekerja dalam mendeteksi intrusi dan memberikan alert secara real time. Setiap paket yang masuk akan ditangkap dan dianalisa oleh Snort berdasarkan rule yang telah ditetapkan. Jika paket tersebut terdeteksi sebagai intrusi akan dilakukan pencatatan file log dan memunculkan sebuah alert. Namun jika paket tersebut tidak

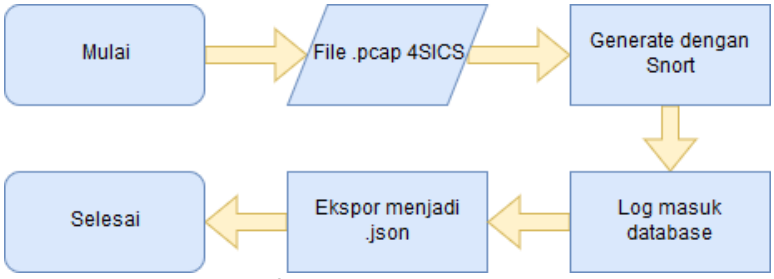
terdeteksi sebagai intrusi maka akan diabaikan. Log alert yang sudah terbentuk akan diterjemahkan oleh Barnyard2 untuk di store ke database. Alur proses deteksi serangan dapat dilihat pada Gambar 2.



Gambar 2. Proses Deteksi Snort

2.3 Dataset 4SICS SCADA

Konferensi keamanan cyber industri 4SICS adalah pertemuan tahunan yang mengumpulkan pemangku kepentingan keamanan cyber ICS / SCADA yang paling penting di berbagai industri penting [7]. Data trafik jaringan diambil pada tanggal 20-22 Oktober 2015, Nalen, Stockholm. Netresec telah bekerja dengan kru 4SICS untuk menangkap lalu lintas jaringan di lab ICS. Dataset yang disediakan oleh 4SICS ini dapat digunakan secara umum dan gratis.

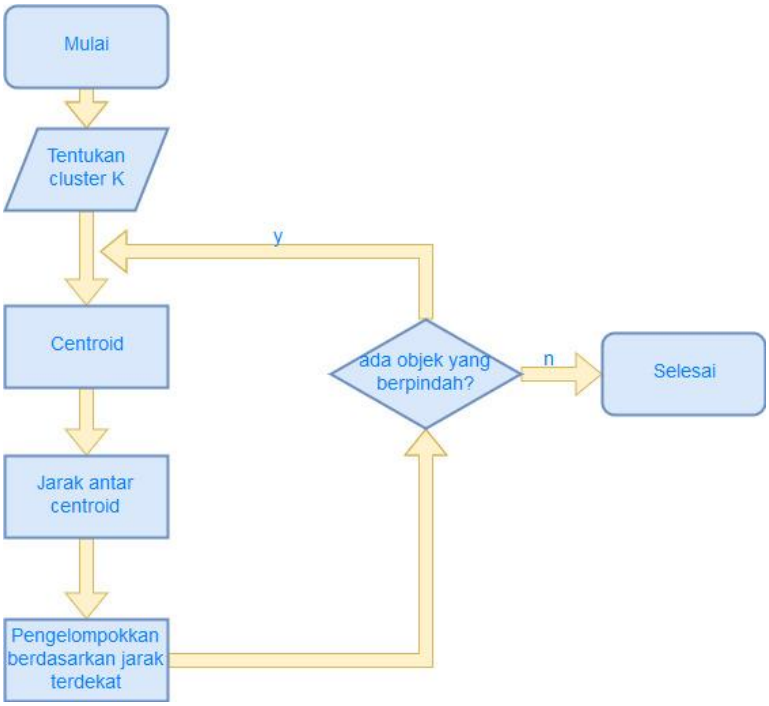


Gambar 3. Generate Dataset

Pada Gambar 3 terdapat proses generate dataset. Dataset berupa file .pcap dimana file ini akan di-generate dengan Snort menggunakan rules yang telah disediakan oleh 4SICS. Dari proses generate akan menghasilkan alert berdasarkan rules yang telah ditetapkan, alert tersebut akan masuk dalam database yang kemudian di ekspor menjadi file .json melalui phpmyadmin. Untuk proses clustering K-Means pada dataset menggunakan hasil file .json tersebut.

2.4 Perhitungan K-Means

Algoritma K-Means digunakan untuk mengelompokkan Alert serangan yang tertangkap oleh Snort. Pada penelitian ini untuk pengelompokan awal menggunakan dataset dari 4SICS SCADA yang berupa file .json.



Gambar 4. Algoritma K-Means

Pada Gambar 4 terdapat tahapan proses clustering menggunakan K-Means. Proses clustering dilakukan menggunakan Algoritma K-Means. Berikut merupakan proses clustering terhadap data serangan yang telah terdeteksi oleh Snort dan kemudian disimpan pada Database

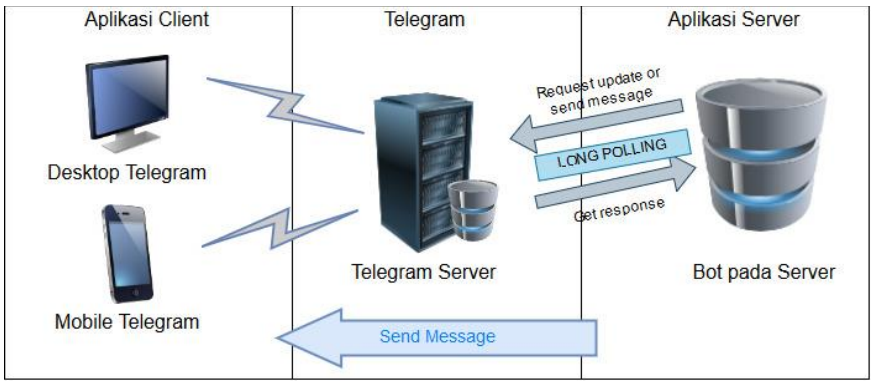
1. Menentukan nilai K atau jumlah cluster dan membangkitkan centroid. Variable yang akan digunakan untuk clustering adalah signature class dan priority. Dimana signature class merupakan kategori serangan. Sedangkan Priority adalah tingkatan serangan pada tiap serangan. K yang dimasukkan adalah dua, sehingga ada dua cluster/kelompok. Membangkitkan centroid awal dari objek yang sudah tersedia pada sebanyak cluster yang di tetapkan.
2. Mengelompokkan data ke dalam cluster setelah centroid telah ditentukan maka akan dihitung jarak antara titik data ke titik centroid dengan menggunakan Euclidence Distance pada Persamaan 1.

$$D(x_2, x_1) = \sqrt{\sum_{j=1}^{p} (x_{2j} - x_{1j})^2} \quad (1)$$

3. Pengelompokan anggota berdasarkan jarak terdekat telah dilakukan, maka selanjutnya adalah menghitung kembali titik pusat tiap cluster dengan anggota sekarang. Pusat cluster atau centroid yang baru dihitung dari rata-rata semua objek / data dalam cluster.
4. Kembali ke tahap 2, dilakukan iterasi atau perulangan proses sampai nilai titik pusat atau centroid yang dihasilkan tetap dan keanggotaan cluster tidak berpindah ke kelompok lain.

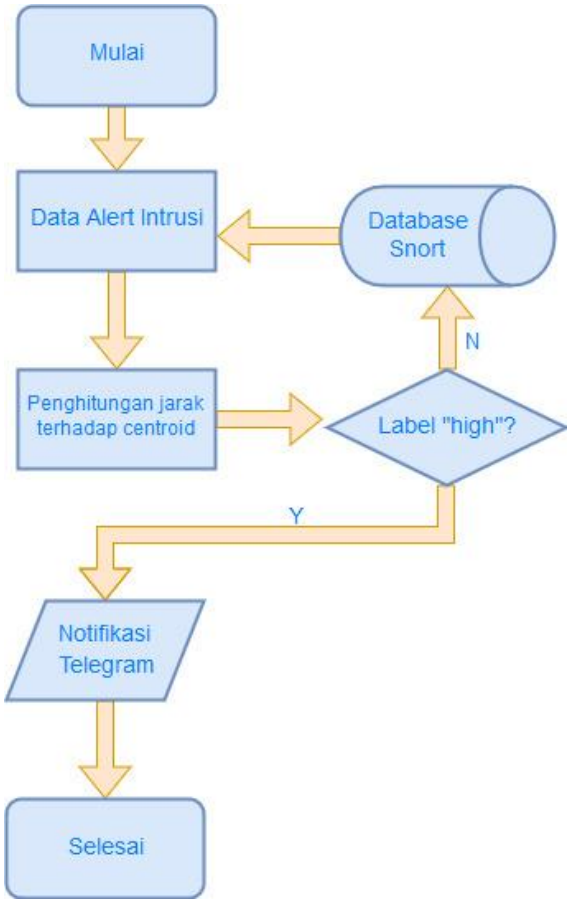
Setelah proses clustering pada dataset selesai maka akan didapatkan dua centroid yang terakhir. Dua centroid tersebut memiliki label “low” dan “high”. Centroid yang telah didapatkan tersebut digunakan pada pengujian impelemntasi serangan yang real. Setiap serangan baru maka akan dihitung jaraknya terhadap masing-masing centroid menggunakan persamaan (1). Data tersebut akan memilih jarak terpendek pada centroid dan menjadi anggota cluster tersebut.

2.5 Pengiriman Notifikasi



Gambar 5. Proses Bot Telegram

Proses yang terdapat pada Gambar 5 menjelaskan bagaimana cara kerja Bot Telegram yang berguna untuk mengirim pesan. Pengiriman pesan pada Bot harus menggunakan Token API. Pada penelitian ini untuk mendapatkan update dan pengirimian notifikasi dari bot maka menggunakan metode long-polling. Server akan menunggu data yang diminta tersedia dengan perulangan atau loops dan koneksi akan terus dibuka dan setelah data tersedia dan siap untuk dikirim maka client akan mendapatkan update secepatnya.



Gambar 6. Proses Pengiriman Notifikasi

Pada paket yang ditangkap Snort dan terdeteksi sebagai intrusi selanjutnya akan disimpan dalam database. Kemudian data intrusi tersebut akan dihitung jarak centroid terhadap kelompok yang telah dilakukan pada dataset. Saat proses perhitungan jarak selesai maka data yang berlabel “high” akan langsung dikirim ke administrator melalui Notifikasi Telegram seperti pada

Gambar 6. Fungsi pada Telegram Bot yang digunakan untuk mengirim pesan adalah sendMessage.

3. Hasil Penelitian dan Pembahasan

Pada bab ini dijelaskan tentang implementasi dari proses K-Means, deteksi serangan dan pengiriman notifikasi.

3.1. Implementasi K-Means

Untuk proses pengelompokan K-Means menggunakan Dataset dari 4SICS dan hasil dari pengelompokan itu akan dijadikan patokan terhadap serangan yang real.

a. Generate Dataset

File dataset yang didapatkan berupa file .pcap, dimana file ini di-generate menggunakan Snort dan hasil log masuk ke database. Pada database tersebut file di ekspor menjadi file .json melalui phpMyAdmin. File .json tersebut dijadikan bahan proses K-Means tetapi hanya variable sig_class_id dan sig_priority yang diambil. Berikut merupakan contoh data dari kedua variable tersebut yang terdapat dalam dataset dari total 24802 data alert:

Tabel 1. Data Alert dari Dataset

Signature Class	Signature Priority
1	2
23	3
30	2
23	3
1	2
1	2
1	2
1	2
33	2
18	2

b. Perhitungan K-Means

Setelah semua data telah didapat maka data tersebut dikelompokkan menggunakan algoritma K-Means. Kelompok yang dibentuk adalah 2 kelompok yang akan menjadi kelompok berlabel “high” dan “low”. Penentuan centroid awal atau titik pusat awal pada penelitian ini dilakukan pada data pertama dan kedua dapat dilihat pada Tabel 2.

Tabel 2. Titik Pusat Awal

Titik Pusat Awal	Signature Class	Signature Priority
Kelompok 1	23	3
Kelompok 2	1	2

Jumlah kelompok dan centroid awal sudah diketahui maka dilakukan perhitungan jarak menggunakan rumus Persamaan 1 pada iterasi pertama.

Jarak data pertama terhadap kelompok 1:

$$d_{11} = \sqrt{(23-23)^2 + (3-3)^2} = 0{,}00$$

Jarak data pertama terhadap kelompok 2:

$$d_{12} = \sqrt{(23-1)^2 + (3-2)^2} = 22{,}0227$$

Jarak data kedua terhadap kelompok 1:

$$d_{21} = \sqrt{(1-23)^2 + (2-3)^2} = 22{,}0227$$

Jarak data kedua terhadap kelompok 2:

$$d_{22} = \sqrt{(1-1)^2 + (2-2)^2} = 0{,}00$$

Jarak data ketiga terhadap kelompok 1:

$$d_{31} = \sqrt{(30-23)^2 + (2-3)^2} = 7{,}07106$$

Jarak data ketiga terhadap kelompok 2:

$$d_{32} = \sqrt{(30-1)^2 + (2-2)^2} = 29{,}00$$

Tabel 3. Hasil Perhitungan K-Means

Signature Class	Signature Priority	Jarak Kelompok 1	Jarak Kelompok 2	Jarak Terpendek
23	3	0	22,022	Kelompok 1
1	2	22,022	0	Kelompok 2
30	2	7,071	29	Kelompok 1
23	3	0	22,022	Kelompok 1
1	2	22,022	0	Kelompok 2
1	2	22,022	0	Kelompok 2
1	2	22,022	0	Kelompok 2
1	2	22,022	0	Kelompok 2
33	2	10,049	32	Kelompok 1
18	2	5,099	17	Kelompok 1

Pada iterasi kedua digunakan titik pusat yang baru berdasarkan jumlah rata-rata data keanggotaan pada tiap kelompok:
Titik pusat kelompok 1 yang baru:

$$\text{Signature Class: } \frac{23+30+23+33+18}{5} = 25{,}4$$

$$\text{Signature Priority: } \frac{3+2+3+2+2}{5} = 2{,}4$$

Titik pusat kelompok 2 yang baru:

$$\text{Signature Class: } \frac{1+1+1+1+1}{5} = 1$$

$$\text{Signature Priority: } \frac{2+2+2+2+2}{5} = 2$$

Setelah terbentuk titik pusat yang baru maka tiap data akan dihitung kembali dengan titik pusat yang baru. Iterasi berhenti pada iterasi ketiga karena pada iterasi ketiga tidak ada data yang berpindah lagi. Pada penilitian ini hanya diambil centroid atau titik pusat terakhir untuk digunakan sebagai patokan terhadap data Alert yang baru. Centroid atau titik pusat terakhir dapat dilihat pada Tabel 4.

Tabel 4. Titik Pusat Akhir

Titik Pusat Terakhir	Signature Class	Signature Priority
Kelompok 1	25,47	2,604
Kelompok 2	1,22	1,96

3.2. Pengiriman Notifikasi

Alert yang dikirimkan menjadi notifikasi melalui Bot Telegram dihitung jaraknya terlebih dahulu terhadap centroid yang telah terbentuk terdapat pada Tabel 4. Centroid diurutkan terhadap variable Signature Priroty. Rumus jarak yang digunakan untuk mengihutung tiap jarak pada masing-masing centroid menggunakan Euclidean distance, dengan begitu setiap alert dihitung jarak terhadap tiap centroid dan yang memiliki jarak terdekat dengan Kelompok 1 diberikan label “high” dan dikirimkan menjadi sebuah notifikasi dan jika terdekat dengan Kelompok 2 diberikan label “low” dan tidak diteruskan menjadi sebuah notifikasi.

```
function sendMessage($chatID, $messaggio, $token) {
    $url = " https://api.telegram.org/"
            . $token . "/sendMessage?chat_id="
            . $chatID."&parse_mode=html";
    $url = $url . "&text=" . urlencode($messaggio);
    $ch = curl_init();
    $optArray = array(
            CURLOPT_URL => $url,
            CURLOPT_RETURNTRANSFER => true
    );
    curl_setopt_array($ch, $optArray);
    $result = curl_exec($ch);
    curl_close($ch);
}
```

Gambar 7. Fungsi method sendMessage

Pengiriman notifikasi Gambar 7 menggunakan *method* yang telah disediakan oleh Telegram Bot yaitu *sendMessage*, parameter yang digunakan yaitu *Token, chatId* dan Teks. *Token* yang telah didapatkan melalui pendaftaran adalah *bot505420106:AAECH-bNaHwfOW1YisDjp7pc3ic dq2w5VVE* , *chatID* yang telah didapatkan melalui *getUpdates* adalah *403430642* dan teks berisi dari konten *alert*.

Agar pengiriman notifikasi dapat dilakukan secara otomatis maka digunakan crontab untuk penjadwalan secara otomatis. Penjadwalan dilakukan agar beberapa proses dapat berjalan tanpa harus di-run secara manual. Pengaturan crontab sendiri dapat dilakukan dengan mengetikkan perintah crontab -e.

```
* * * * * php /var/www/html/kmeans_telegram/core.php
* * * * * wget --spider localhost/base/index.php
0 * * * * php /var/www/html/kmeans_telegram/satujam.php
0 0 * * * php /var/www/html/kmeans_telegram/harian.php
```

Gambar 8. Konfigurasi Cronjob

Pada Gambar 8 terdapat beberapa penjadwalan proses. Baris 1 untuk melakukan refresh terhadap website BASE agar selalu mengupdate alert terbaru yang telah di store ke database. Pada baris 2-4 merupakan proses pengiriman notifikasi alert melalui Bot Telegram dari tiap menit, tiap jam dan tiap 24 jam.

3.3 Hasil Deteksi Serangan

Pada pengujian sistem, serangan merupakan serangan real yang telah ditangkap oleh Snort. Pengujian ini merekam serangan pada rentang waktu selama 2 hari

a. Rule Snort

Serangan terdeteksi berdasarkan rule yang digunakan. Pada pengujian ini rule menggunakan snort-community-ruleset, yaitu sekumpulan rule yang disediakan oleh Official Snort yang dibuat oleh para contributor Snort open-source. Rule ini di-generate menggunakan pulledpork dengan mengaktifkan pada file enablesid.conf seperti pada Gambar 9.

```
# Example of modifying state for specific categories entirely (see README.CATEG
# VRT-web-iis,ET-shellcode,ET-emergingthreats-smtp,Custom-shellcode,Custom-emer
#VRT-indicator-scan
Snort-Community-community
```

Gambar 9. Penggunaan Rule Community

Kemudian pulledpork dijalankan menggunakan perintah /usr/local/bin/pulledpork.pl -c /etc/snort/pulledpork.conf -l dan akan muncul jumlah rule yang digunakan sebanyak 13147 seperti pada Gambar 10.

```
Rule Stats...
        New:-------14
        Deleted:---14
        Enabled Rules:----13147
        Dropped Rules:----0
```

Gambar 10. Jumlah Rule

b. BASE

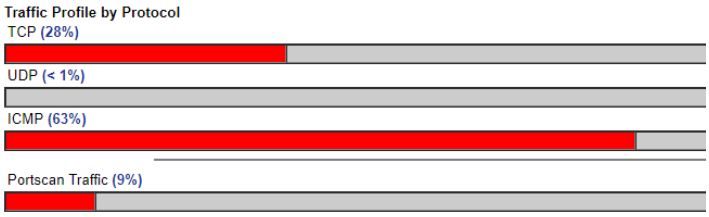
Halaman BASE menunjukkan hasil dari deteksi serangan selama 2 hari yang memiliki total alert adalah 10352 seperti pada Gambar 11.

Sensors/Total: 1 / 1
Unique Alerts: 51
Categories: 10
Total Number of Alerts: 10352

- Src IP addrs: 411
- Dest. IP addrs: 5
- Unique IP links 481
- Source Ports: 1228
  - TCP (1210) UDP (19)
- Dest Ports: 26
  - TCP (20) UDP (7)

Gambar 11. Jumlah Serangan yang Terdeteksi

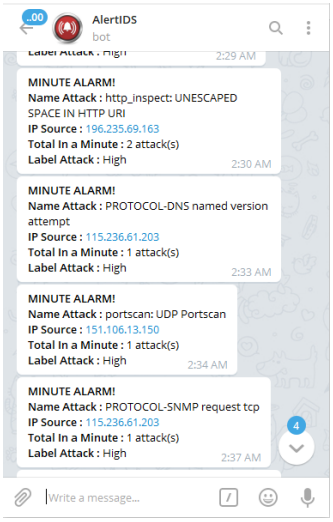
BASE juga menangkap serangan berdasarkan protocol yang digunakan seperti pada Gambar 12.



Gambar 12. Serangan Berdasarkan Protocol

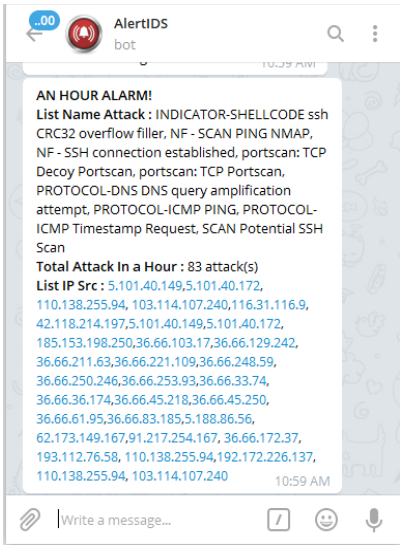
c. Notifikasi Telegram

Setelah terdeteksi maka pengelompokan dan pemberian label dilakukan agar hanya alert yang memiliki label “high” dikirimkan menjadi notifikasi melalui Bot Telegram.



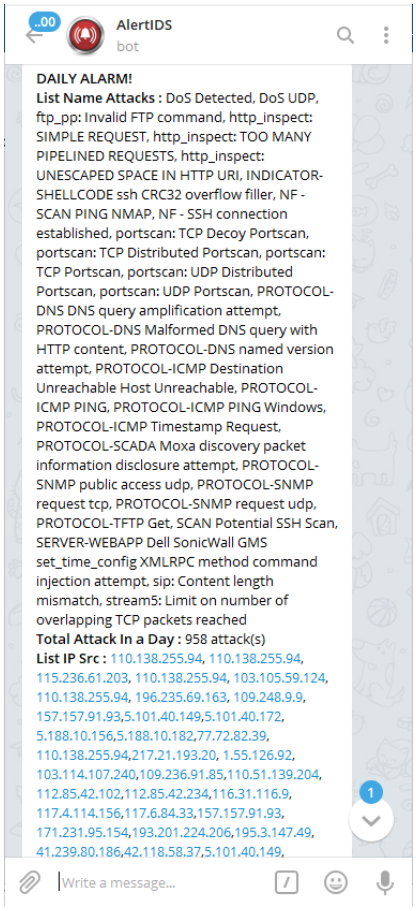
Gambar 13. Notifikasi Tiap Menit

Pada Gambar 13 merupakan Notifikasi yang dikirim melalui Bot Telegram dalam rentang waktu satu menit. Pengiriman dalam rentang satu menit dikirim sesuai nama tiap serangan.



Gambar 14. Notifikasi Tiap Jam

Sedangkan pada Gambar 14 merupakan alert yang dikirimkan melalui Bot Telegram selama rentang waktu satu jam terakhir. Bot Telegram juga akan melakukan pengiriman notifikasi dalam rentang waktu 24 Jam terakhir seperti pada Gambar 15.



Gambar 15. Notifikasi Tiap 24 Jam

3.4 Hasil Pengujian Sistem

Hasil dari pengelompokan terhadap alert yang tertangkap oleh Snort. Setiap *Alert* akan dikelompokkan menurut jarak terpendek terhadap *centroid* yang telah terbentuk seperti pada Tabel 5.

Tabel 5. Hasil Pengelompokkan

No	Nama Serangan	Signature Class	Signature Priority	Jarak Centroid 1	Jarak Centroid 2	Label
1	portscan: TCP Decoy Portscan	4	2	2,78	29,41	High
2	INDICATOR-SHELLCODE ssh CRC32 overflow filler	15	1	13,81	10,6	Low
3	portscan: TCP Decoy Portscan	4	2	2,78	29,41	High
4	NF - SSH connection established	30	2	28,78	4,56	Low
5	DoS Detected	3	2	1,78	22,49	High
6	SCAN Potential SSH Scan	23	3	21,8	2,51	Low

7	DoS UDP	3	2	1,78	22,49	High
8	SCAN Potential SSH Scan	23	3	21,8	2,51	Low
9	DoS Detected	3	2	1,78	22,49	High
10	DoS Detected	3	2	1,78	22,49	High
11	SSH Brute Force Attack	30	2	28,78	4,56	Low
12	SSH Brute Force Attack	30	2	28,78	4,56	Low
13	portscan: UDP Distributed Portscan	4	2	2,78	29,41	High
14	http_inspect: TOO MANY PIPELINED REQUESTS	2	3	1,3	23,48	High

Pada tabel diatas merupakan sampel dari pengelompokan dan pemberian label serangan real. Total terdapat 10352 serangan dan dari data tersebut sejumlah 1096 serangan yang memiliki label “high” dan 9256 serangan yang memiliki label “low”.

Kemudian dilakukan analisa jumlah notifikasi yang terkirim melalui Bot Telegram dalam jangkauan waktu 10 Menit, 1 Jam, 6 Jam, 12 Jam, 24 Jam

Tabel 6. Analisa Pengujian Waktu

No	Jangkauan Waktu	Jumlah Serangan yang Terdeteksi	Jumlah Notifikasi
1	10 Menit	14	6
2	1 Jam	56	22
3	6 Jam	1202	230
4	12 Jam	1615	443
5	24 Jam	8530	606
6	48 Jam	10352	771

Pada Tabel 6 dapat dilihat pada 10 Menit pertama serangan yang terdeteksi adalah 14 serangan dan notifikasi yang dikirim adalah 6 notifikasi. Pada rentang waktu 1 Jam serangan yang terdeteksi adalah 56 dan notifikasi yang dikirim sejumlah 22. Rentang waktu 6 Jam serangan yang terdeteksi berjumlah 1202 dan notifikasinya adalah 230. Pada rentang waktu 12 Jam sistem mendeteksi ada 1615 serangan dan mengirimkan notifikasi sejumlah 443 dan yang terakhir pada rentang waktu 24 Jam sistem mendeteksi adanya 8530 serangan dan mengirimkan notifikasi yang berjumlah 606.

Pada Gambar 16 terdapat persentase hasil serangan selama satu jam berdasarkan label serangan. 60,38% serangan memiliki label “high” dan 39,62% memiliki label “low”.

Gambar 16. Persentase Label Serangan Selama 1 Jam

Sedangkan pada Gambar 17 terdapat persentase hasil serangan selama total 2 hari berdasarkan label serangan. 11% serangan memiliki label “high” dan 39,62% memiliki label “low”.

Gambar 17. Persentase Label Serangan Selama 2 Hari

4. Kesimpulan

Setelah dilakukan impelmentasi dan pengujian pada sistem maka dapat ditarik beberapa kesimpulan seperti:

1. Intrusion Detection System Snort dapat mendeteksi serangan selama 2 hari berdasarkan rule yang telah diterapkan.
2. Hasil Pengujian menunjukkan bahwa sistem dapat mengelompokkan jenis serangan menjadi dua kelompok atau cluster yaitu “high” dan “low” berdasarkan hasil proses K-Means dari dataset dengan menggunakan variable signature class dan priority terhadap masing-masing serangan.
3. Bot Telegram dapat mengirimkan data alert menjadi notifikasi pada Aplikasi Telegram.
4. Hasil Pengujian menunjukkan bahwa Telegram Bot berhasil mengirimkan data alert yang berlabel “high” saja.

Merujuk dari hasil pengujian pada penilitian ini memiliki banyak kekurangan dan kelemahan. Saran dari penulis untuk pengembangan penelitian ini sebagai berikut:

1. Menambahkan beberapa aturan atau rule yang digunakan oleh Snort agar dapat mendeteksi adanya serangan yang lebih beragam.
2. Sebaiknya ditambahkan metode pencegahan serangan agar sistem tidak hanya mendeteksi serangan saja.
3. Menambahkan fitur pada Telegram sehingga Bot dapat melakukan komunikasi dua arah pada sistem sehingga tidak hanya mengirimkan notifikasi saja tetapi dapat menerima perintah dari user.

Daftar Notasi

D : Jarak Euclidean
x_1 : Titik objek 1
x_2 : Titik objek 2
P : Banyaknya Data

Referensi:

[1] A. Anitha, “Network Security Using Linux Intrusion,” *Int. J. Res. Comput. Sci.*, vol. 2, no. 1, pp. 33–38, 2011.
[2] R. Rafiudin, *Mengganyang Hacker dengan snort*. Penerbit Andi, 2010.
[3] M. I. Kurniawan, U. Sunarya, and R. Tulloh, “Internet of Things : Sistem Keamanan Rumah berbasis Raspberry Pi dan Telegram Messenger,” *ELKOMIKA*, vol. 6, no. 1, pp. 1–15, 2018.
[4] T. Wahyudi and R. Efendi, “Perancangan keamanan jaringan komputer menggunakan snort dengan notifikasi sms,” *J. Teknol. Inf. dan Komun.*, pp. 1–8, 2012.
[5] F. C. Wulur, “Klasifikasi Alert pada Intrusion Detection System Menggunakan Algoritma K-Means,” *Progr. Stud. Tek. Inform. FTI-UKSW*, 2015.
[6] Y. Agusta, “K-Means – Penerapan, Permasalahan dan Metode Terkait,” *J. Sist. dan Inform.*, vol. 3, no. Pebruari, pp. 47–60, 2007.
[7] “Capture files from 4SICS Geek Lounge.” [Online]. Available: https://www.netresec.com/index.ashx?page=PCAP4SICS. [Accessed: 07-Apr-2018].