Deteksi Anomali Trafik Menggunakan Algoritma BIRCH dan DBSCAN menggunakan Streaming Traffic

Anomaly Traffic Detection with BIRCH dan DBSCAN Algorithm for Streaming Traffic

M. Aldo Shauma , Yudha Purwanto , Astri Novianty

Prodi S1 Sistem Komputer, Fakultas Teknik Elektro, Telkom University
Bandung, Indonesia

shaumaaldo@students.telkomuniversity.ac.id, omyudha@telkomuniversity.ac.id, astrinov@telkomuniversity.ac.id

Abstrak

Pertumbuhan internet yang sangat pesat memiliki dampak positif maupun dampak negatif bagi seluruh pengguna. Begitupula dengan serangan ataupun ancaman yang dapat terjadi terhadap sebuah komputer maupun *server* didalam sebuah jaringan, salah satunya yaitu berupa anomali trafik. Salah satu bentuk anomali trafik adalah *flashcrowd* yang ditandai dengan adanya kenaikan trafik secara signifikan. Ini disebabkan karena banyaknya user yang mengakses dan menimbulkan kepadatan trafik. Banyak penelitian mengenai anomali trafik namun data yang digunakan masih bersifat *damped* atau *offline*, artinya data yang sudah disimpan sebelumnya dan sudah diketahui isi datanya. Oleh karena itu, dibutuhkan suatu sistem deteksi untuk mengenali dan mendeteksi anomali trafik dengan *streaming traffic* (data *online*).

Pada Tugas Akhir ini , akan dilakukan pendeteksian anomali yang terjadi pada suatu jaringan menggunakan algoritma BIRCH (*Balanced Iterative Reducing and Clustering Using Hierarchies*). Sistem deteksi trafik ini mempunyai kemampuan untuk mendeteksi anomali yang terjadi dengan cara membentuk klaster trafik anomali dan klaster trafik normal. Selanjutnya akan dibantu dengan algoritma *clustering* DBSCAN (Density Based Spatial Clustering of Applications with Noise) pada bagian *Clustering feature tree* (CF Tree) BIRCH dalam mengelompokkan datastream berdasarkan kepadatan data *streaming traffic* yang masuk serta pelabelan trafik.

Hasil dari penelitian ini, algoritma BIRCH dan DBSCAN memiliki performansi yang baik dalam mendeteksi anomali trafik. Hal ini dapat ditunjukkan dengan pengujian yang dilakukan terhadap Akurasi dari hasil klaster, dimana nilai rata rata akurasinya adalah 98.45 % serta memakan waktu kurang lebih 600 detik atau sekitar 10 menit dalam sekali proses 30.000 data.

Kata Kunci : BIRCH, DBSCAN, flashcrowd, streaming traffic

Abstract

Internet has grown so fast that made positive and negative effect for it's user. It same as threat that can be happen to computer or server in a network, one of them is about anomaly traffic, for example is flashcrowd that happen when the demand of traffic in a network is increase significally. It's happen because too many user that access the same network and cause traffic density to increase.There are so many research about traffic anomaly but the data that used in the research still in damped mode or we call it offline, it means the data not real and determine by the researcher.Therefore, a system detection is needed to detect and identify traffic anomaly based on streaming traffic.

In this Final Task, anomali detection will be research with BIRCH (*Balanced Iterative Reducing and Clustering Using Hierarchies*) algorithm. This system can detect and identify anomaly that happen in the network with cluster based system to make anomaly cluster and normal cluster traffic.Then, it will continued with DBSCAN (Density Based Spatial Clustering of Applications with Noise) clustering algorithm for labeling traffic packet.

Result from this study, BIRCH and DBSCAN algorithm has good performance in detecting anomalous traffic. It can be demonstrated by tests the accuracy of detecting datastream, where the average value of accuracy is 98.4% and it takes about 600 seconds or 10 minute for one process from 30.000 data.

Keyword : BIRCH, DBSCAN, flashcrowd, streaming traffic

1. Pendahuluan

Pertumbuhan pesat yang dialami oleh dunia jaringan internet memberikan dampak positif dalam pengolahan informasi serta pengaksesan data. Pada saat ini, pengguna internet sudah terdiri dari berbagai kalangan usia dan kebutuhan. Para pengguna internet pada umumnya belum paham mengenai keamanan pada jaringan internet yang sehari – hari merekan gunakan. Banyak kemungkinan yang akan terjadi pada data personal mereka apabila terkoneksi ke internet. Ini diakibatkan oleh pihak – pihak yang tidak bertanggung jawab dalam memanfaatkan teknologi secara negatif. Dari berbagai aspek inilah menyebabkan serangan pada jaringan komputer harus dideteksi agar dapat dicegah, sehingga kenyamanan akses data tetap terjaga.

Denial of Service (DoS) dan *Disributed Denial of Service* (DDoS) merupakan bentuk serangan *flooding* yang berusaha membuat suatu *host* atau *service* menjadi tak dapat diakses oleh user yang berhak. Sasaran serangan oleh DoS/DDoS adalah *link/bandwidth* untuk membuat sumber daya bandwidth penuh dan sumber daya komputasi pada server agar sistem pengolah kehabisan sumber daya yang berujung oleh jaringan *down* atau *crash* [1]. deteksi menggunakan algoritma BIRCH (*Balanced Iterative Reducing and Clustering Using Hierarchies*) memperkenalkan dua konsep yaitu *clustering feature* dan *clustering feature tree* (*CF tree*) [2]

Deteksi serangan pada saat ini lebih banyak menggunakan data non real time, artinya data masih bersifat offline dengan cara di *damped.* Sehingga ketika diaplikasikan ke data yang bersifat realtime akan menimbulkan perbedaan yang signifikan. Dalam pengembangannya sistem deteksi anomali banyak pendekatan untuk mengetahui pola trafik normal sebagai acuan deteksi anomali trafik. Penelitian ini menggunakan algoritma BIRCH dan DBSCAN dalam mengelompokkan trafik apakah tergolong pada trafik normal ataukah trafik anomali.

2. Dasar Teori

2.1 Deteksi Anomali Trafik

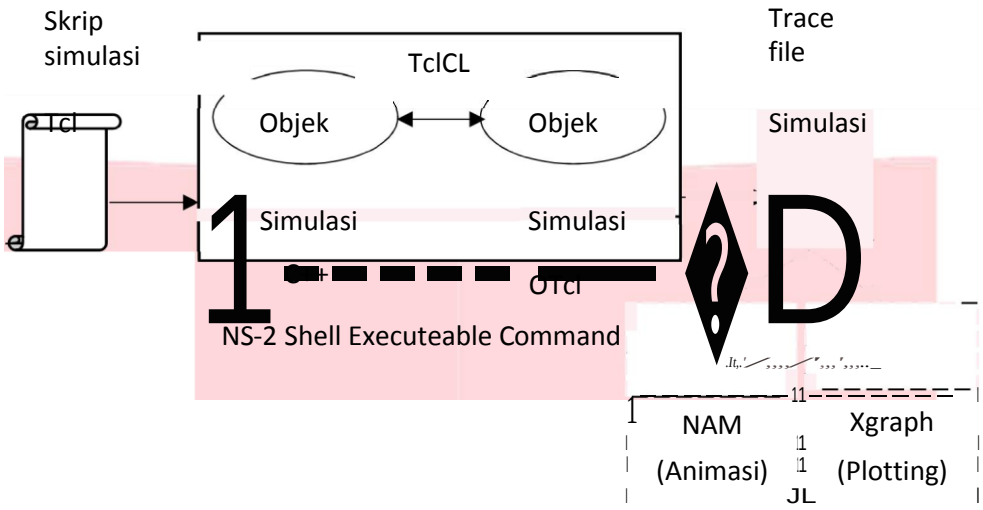
Anomali trafik merupakan suatu keadaan yang menyebabkan abnormalitas pada lalu lintas jaringan. Penyebab dari anomali ini bisa saja faktor dari banyaknya pengguna internet atau serangan pada suatu jaringan. Kondisi ini dapat menyebabkan penurunan performansi jaringan sehingga rentannya sebuah jaringan untuk diserang. Dampak dari anomali trafik ini dapat melumpuhkan jaringan, dapat juga disisipi file yang membahayakan target dari penyerang. Maka dari itu, perlu dilakukan deteksi terhadap lalu lintas jaringan dalam mencegah ataupun mengatasi anomali didalam lalu lintas jaringan.

Dalam pendeteksian serangan atau anomaly yang terjadi di dalam jaringan, dikenal dua istilah pendekatan yaitu *Intrusion Detection System* (IDS) dan *Intrusion Prevention System* (IPS) [1]. IDS dan IPS memiliki sistem yang bekerja dalam mengawasi keadaan jaringan serta memberikan peringatan kepada administrator apabila mendeteksi serangan datang ataupun anomaly. Pada IDS/IPS menggunakan metode *traffic anomaly based* dan *intrusion signature.* Metode *intrusion signature* mendeteksi serangan berdasarkan *database* yang dimiliki, sehingga kelemahannya jika sebuah serangan tidak ada di *database* serangan, maka akan dianggap sebagai trafik normal. Pada *traffic anomaly based,* proses pendeteksian tidak membutuhkan *database* serangan, sehingga proses pendeteksian tidak bergantung pada *database* serangan

2.2 Network Simulator – 2 (NS2)

Dalam buku [3], *Network Simulator* (*Version 2*), dikenal sebagain NS2, merupakan alat simulasi yang berguna dalam mempelajari sifat dinamis jaringan komunikasi. NS2 terdiri dari dua bahasa utama : C++ dan *Object-oriented Tool Comand* (*OTcl*). Sementara C++ mendefinisikan mekanisme internal (yaitu, *backend*) simulasi. Bahasa Otcl membentuk simulasi dengan merakit dan mengkonfigurasi objek serta melakukan menjadwalan diskrit (yaitu, *Frontend*). C++ dan Otcl dihubungkan bersama menggunakan TclCL. Gambar 2.2 menunjukkan arsitektur dasar NS2 yang menyediakan user dengan perintah eksekusi “ns” yang mengambil satu argumen input dalam bentuk file *.tcl.*

Gambar 2.1 Arsitektur dasar NS [3]



2.3 *Preprocessing*

Preprocessing dilakukan untuk mentransformasi *raw data* menjadi data yang mudah diinterpretasikan sebagai inputan algoritma sistem deteksi anomali untuk dianalisis. *Preprocessing* akan mempengaruhi dalam segi kualitas output akhir. Jika data inputan tidak berkualitas, maka hasil deteksi juga tidak berkualitas dan diragukan keakuratannya. Dalam proses *clustering* selanjutnya, dibutuhkan data *preprocessing* dari output file NS2 berupa file *.tracefile* (*.tr*). Pada penelitian ini dilakukan proses *preprocessing data transformation*, yaitu mengubah suatu data agar diperoleh data yang lebih berkualitas. Dengan adanya *preprocessing* akan meningkatkan hasil analisis yang dilakukan. Tujuan *preprocessing* output file NS2 ini yaitu sebagai inputan *clustering* pada algoritma BIRCH.

2.4 Algoritma Clustering BIRCH (*Balanced Itterative Reducing and Clustering Using Hierarchies*)

Algoritma BIRCH (Balanced Itterative Reducing and Clustering Using Hierarchies) menggunakan konsep Cluster Feature (CF) yang meringkas informasi tentang subcluster – subcluster yang ada sehingga dapat mengurangi skala dari clustering. Algoritma BIRCH memiliki keunggulan dalam menangani data yang besar dan akurat karena banyak outlier yang dieliminasi. Area kepadatan dari suatu data menentukan apakah dianggap sebagai outlier atau dianggap subcluster. Jika sebuah area padat dengan data, maka akan dianggap sebagai subcluster, sedangkan area yang sangat jarang akan diperlakukan sebagai outlier. Dalam clustering hierarkis, data dipartisi ke dalam beberapa level / tingkatan secara berturut turut. Saat terjadi peningkatan level, data di klasterkan bersamaan dengan cara hirarkis. Pendekatan BIRCH dalam pengklasteran data secara optimal dengan menggunakan konsep cluster feature untuk menyimpan ringkasan informasi mengenai subcluster – subcluster dan membangun sebuah pohon hierarkis (dendogram) dari subcluster tersebut.

Fase 1 : Membangun CF tree
inisialisasi

Fase 2 (optional) : Membangun CF tree
yang lebih kecil

Fase 3 : Clustering global

Fase 4 (optional and offline) : Perbaikan
cluster

Gambar 2.2 Fase – fase pada BIRCH ***[4]***

Pada fase pertama, BIRCH menampilkan scan linear kesemua data point untuk membangun CF Tree inisialisasi. Point di-insert dengan mencari leaf terdekat pada tree. Jika inputan pada leaf bisa menampung point baru tanpa melanggar kondisi threshold, maka nilai CF untuk masukan ini diupdate,

sebaliknya akan dibuat node leaf yang baru. Dalam kasus ini, jika node leaf berisi lebih dari L masukan, setelah proses insert, kemungkinan node daun dipecah.

Pada fase kedua yang bersifat opsional, ukuran CF Tree bisa dikurangi sampai jumlah node daun yang ditentukan telah dicapai. Selain itu, disini juga mengeluarkan outlier yang lebih banyak dan mengelompokkan subcluster yang lebih padat ke dalam bentuk yang lebih besar. Lalu dilanjutkan fase ketiga menggunakan algoritma clustering hierarkis untuk meng-klasterkan nilai CF dari node leaf. Sekumpulan data point bisa diadaptasi dalam bentuk subcluster – subcluster yang tiap tiap subcluster diwakilkan dengan vektor CF. Secara umum bisa dengan menerapkan algoritma yang ada secara langsung untuk mengelompokkan subcluster subcluster karena informasi pada vektor Cfnya biasanya cukup untuk menghitung jarak. Terakhir pada fase 4 bersifat opsional yaitu perbaikan cluster.

2.5 Algoritma DBSCAN *(Density Based Spatial Clustering of Application with Noise)*

Terdapat banyaknya metode dalam *density based clustering*, salah satunya adalah DBSCAN. DBSCAN merupakan algoritma menumbuhkan area – area dengan kepadatan yang cukup tinggi ke dalam cluster – cluster dan menemukan cluster – cluster dalam bentuk yang sembarang dalam suatu database *spatial* yang memuat noise [5]Ide dasar dari DBSCAN adalah memanfaatkan jumlah titik minimal yang harus dimiliki untuk menentukan suatu poin untuk digolongkan menjadi *core point, border point,* atau *noise point* yang disebut MinPts. Selain itu, juga menggunakan *threshold* yang harus dipenuhi untuk menentukan titik tersebut menjadi *core point, border point, atau noise point* yang disebut juga Eps.

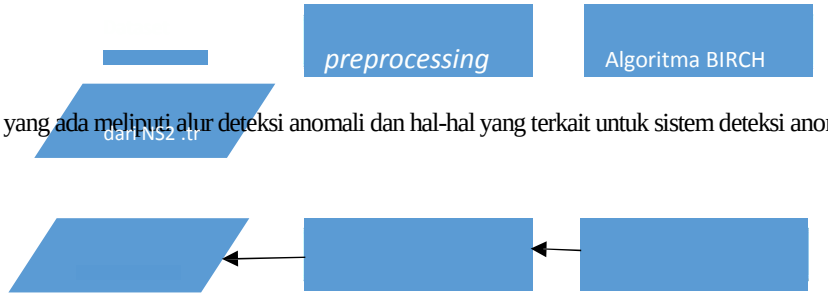
Kelebihan dari DBSCAN adalah [5]:

1. Dapat menghasilkan suatu cluster yang lebih akurat dari bentuk data yang tidak beraturan (*arbitrary shape*) jika dibandingkan dengan *partitional clustering*
2. Dapat menangani noise
3. Baik untuk data dalam jumlah besar

3. Pembahasan

3.1 Deskripsi Sistem

Pada perancangan sistem menjelaskan mengenai alur dari proses yang dikerjakan pada tugas akhir ini.

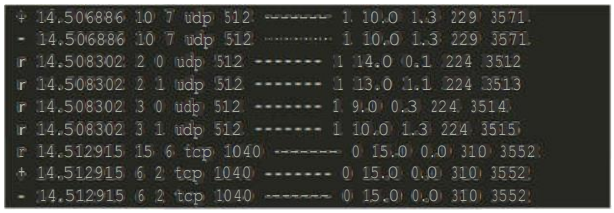


Penjelasan yang ada meliputi alur deteksi anomali dan hal-hal yang terkait untuk sistem deteksi anomali.

Gambar 3.1 Alur Deteksi Anomali Trafik

3.2 Dataset NS2 .tracefile

Dalam penelitian tugas akhir ini membutuhkan *dataset* sebagai data yang diolah pada metode deteksi. *Dataset* ini adalah data *real time traffic* yang diduga terdapat anomali didalamnya.



Gambar 3.2 isi .tracefile

Berdasarkan *raw data* dengan format diatas, akan dilakukan tahap *preprocessing.* Tahap *preprocessing* ini dilakukan untuk mendapatkan fitur – fitur yang relevan dan akan digunakan sebagai inputan dari algoritma deteksi.

3.3 Pre-processing

Pada tahap preprocessing, *raw data* hasil dari file .tr yang merupakan *output* dari skenario di NS2 akan menjadi inputan. Preprocessing pada penelitian ini mengacu pada fitur dalam dataset KDDCUP 1999. Pada dataset KDDCUP, terdapat 41 fitur yang dimiliki dalam satu paket, sedangkan yang digunakan pada penelitian ini adalah 4 fitur. Berdasarkan jurnal [6], trafik normal pada KDDCUP 99 memiliki 11 fitur paling relevan dalam menentukan sebuah trafik tergolong pada trafik normal. 4 fitur ini mengacu pada fitur dengan tingkat kecendrungan paling tinggi pada sebuah trafik normal.

Tabel 3.1 Fitur *Dataset* KDD99

Feature name	*Description*	*Type*
1. Duration	Duration of the Connection	Continuous
6. Destination Bytes	Bytes sent from source to destination	Continuous
31. srv_diff_host_rate	% of connections to different hosts	Continuous
32. dst_host_count	Count of connections having the same destination host	Continuous

Pengambilan 4 fitur ini dengan proses *preprocessing* dengan inputan dari hasil output NS2. Berikut merupakan algoritma *preprocessing* yang dilakukan :

Algoritma 1 : *Preprocessing* fitur *Duration*

```
1: Input out.tr
2: if src_node(x) = src_node(y) and event = "+" then
3:     time1 ← time;
4: endif
5: if src_node(x) = src_node(y) and event = "r" or "d" then
6:     time2 ← time;
7: endif
8: if time2 - time1 <> "0" then
9:     print time2 – time1
10: endif
11: end
12: until semua data di out.tr
```

Algoritma 2 : *Preprocessing* fitur *Destination Bytes*

```
1: Input out.tr
2: if src_addr(x) = src_addr(y) and dest_addr(x) = dest_addr(y) then
3:     destbytes ← packet_size;
4:   else
5:     destbytes ← packet_size
6: endif
7: end
8: until semua data di out.tr
```

Algoritma 3 : *Preprocessing* fitur *srv_diff_host_rate*

```
1: Input out.tr
2: if src_addr(x) = src_addr(y) then
3:     count ++;
4:   if dest_addr(x) = dest_addr(y) then
5:     diffhost ++
6:     endif
7:   else
8:   (diffhost/count)*100
10: endif
11: end
12: until semua data di out.tr
```

Algoritma 4 : *Preprocessing* fitur *dst_host_count*

```
1: Input out.tr
2: if src_addr (x) = src_addr(y) then
3:     countclient ++;
4:       if dest_addr(x) = dest_addr(y) then
5:           destcountnorm++
6:       endif
7: endif
8: if src_addr (x) = src_addr(z) then
9:     countattack ++;
10:       if dest_addr(x) = dest_addr(z) then
11:           destcountatt++
12:       endif
13: endif
14: end
15: until semua data di out.tr
```

3.4 Algoritma BIRCH

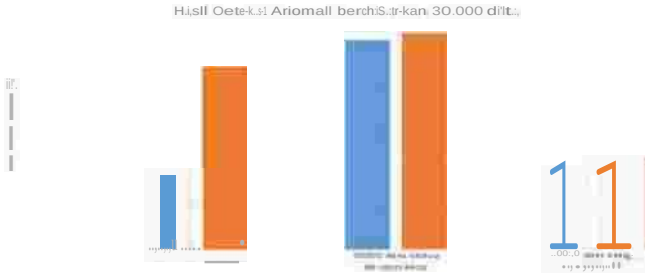
Pada tahap ini, hasil data *preprocessing* berupa fitur yang relevan per satuan paket akan diproses dalam algoritma BIRCH. Fitur yang diambil sebelumnya merupakan fitur – fitur relevan terhadap trafik normal. Sehingga nantinya hasil cluster akan membandingkan jika tidak sesuai dengan trafik normal, maka trafik akan tergolong ke cluster anomali. Pada algoritma ini, data yang di proses hanyalah data *numeric* dan sangat sensitif terhadap urutan data.

3.5 Algoritma DBSCAN

Pada tahap algoritma DBSCAN, hasil algoritma BIRCH berupa *clustering* point yang akan menjadi inputan. Nilai *Centroid* yang didapatkan pada algoritma sebelumnya akan menjadi acuan utama dari algoritma ini dalam meningkatkan kualitas cluster. Selanjutnya, akan diberikan sebua pelabelan berdasarkan masing masing node tergolong ke cluster normal ataupun anomali. Dari pelabelan ini akan dibandingkan dengan label prediksi untuk menguji *detection rate* serta akurasi dari hasil cluster.

3.6 Pengujian *Accuracy* dan *Detection Rate*

Validasi suatu cluster sangat dibutuhkan untuk menguji seberapa baik hasil clusterisasi yang dihasilkan. Dalam penelitian ini menggunakan *confusion matrix* untuk menguji tingkat *Accuracy* dan *Detection Rate* yang telah dilakukan. Perbandingan data hasil akan dilakukan dengan data prediksi sesuai dengan pelabelan apakah trafik termasuk normal ataukah anomali. Data total yaitu 90.000 data akan dibagi ke 3 bagian dan akan diuji *Detection Rate* dan *Accuracy* nya. Hasil sesuai pada gambar grafik dibawah ini.



Gambar 3.3 Hasil Deteksi Anomali

Berdasarkan data, tingkat akurasi paling tinggi pada 30.000 data kedua dengan jumlah nilai TP (*True Positive)* mencapai 27.427 dan ternyata memberikan hasil *accuracy* dan *detection rate* hampir mendekati sempurna. Sementara pada 30.000 data pertama dengan selisih antara *detection rate* dan *accuracy* nya hampir mencapai 1.5% dengan kondisi nilai TP pada 30.000 data pertama berjumlah 15.365 dan nilai FN (*False Negative)* berjumlah 14.635. Berdasarkan data grafik diatas serta tabel hasil deteksi sebelumnya, semakin kecil nilai FP (*False Positive*) maka akan meningkatkan nilai *detection rate* dan *accuracy*, namun jika perbandingan antara nila FP dan FN hampir sama, maka akan terlihat juga perbandingan antara nilai *detection rate* dan *accuracy* seperti pada 30.000 data pertama dengan selisih nilanya hampir mendekati 1.5%. Hasil ini dipengaruhi dengan nilai *Threshold* yang pada penelitian ini tidak berubah, sehingga pada proses clustering jika terjadi *out of memory* maka tidak adanya penambahan CF tree yang baru. CF tree hanya bertambah selama masih dibawah batas *Threshold* yang ditentukan. Semakin banyak CF tree maka semakin banyaknya keragaman data dan mempengaruhi tingkat deteksi dan keakuratan cluster. Dari hasil penelitian, dikarenakan data 30.000 kedua lebih dominan kepada data serangan, maka akurasi dan deteksi lebih tinggi dibandingkan dengan dua data lainnya.

4. KESIMPULAN DAN SARAN

4.1 Kesimpulan

Dari hasil yang didapatkan pada penelitian ini, dapat ditarik beberapa kesimpulan sebagai berikut :

1 Akurasi deteksi yang diperoleh dengan menggunakan metode clustering 2 buah algoritma yaitu BIRCH dan DBSCAN adalah 98,45%. Jadi dapat disimpulkan metode dua algoritma ini cukup baik untuk mendeteksi anomali

2 Penggunaan fitur yang tepat sangat mempengaruhi hasil dari deteksi ini, semakin banyak fitur yang digunakan dalam membentuk cluster, akan meningkatkan kompleksitas namun menambah tingkat akurasi dalam proses clustering trafik. Pada penelitian ini menggunakan fitur *Duration, Destination_bytes, srv_diff_host_rate, Diff_host_service.* Fitur *srv_diff_host_rate* Menjadi patokan utama dengan isi nilai rentang 0 – 100 persen

3 Nilai *Threshold* akan berpengaruh terhadap banyaknya CF *tree* yang dihasilkan. Jika nilai *Threshold* selalu di update, maka jumlah data yang menjadi inputan akan lebih banyak karena tidak akan terjadi *out of memory* ketika proses clustering, namun tidak cocok dalam deteksi anomali berbasis data stream karena jika terjadi update maka akan lebih lama proses clusteringnya. Dengan Treshold statik pada penelitian ini, proses clustering memakan waktu hingga 10 menit untuk jumlah data 30.000

4 Dalam pemilihan niai *Threshold,* Semakin kecil nilainya (mendeati 0) maka cluster yag dihasilkan akan lebih baik, namun tetap menyesuaikan sistem dan perlunya training data untuk menyesuaikan nilai *Threshold* yang tepat, sehingga update *Threshold* tidak akan mengganggu lamanya proses *clustering*.

5 Banyaknya CF tree akan berpengaruh kepada nilai Akurasi dan Deteksi. Nilai akurasi akan tinggi apabila keragaman datanya sangat minim (data cendrung ke anomali saja / normal saja). Pada 30.000 data kedua, CF Tree untuk anomali dan normalnya memiliki perbandingan sekitar 80:20 sehingga akurasi dari hasil cluster sangat tinggi, yaitu 99%

4.2 Saran

Saran untuk penelitian selanjutnya adalah :

1. Penggunaan fitur sebagai inputan tidak hanya berpatokan pada trafik normal saja, namun menggunakan fitur lain sehingga dapat dihasilkan cluster tidak hanya dengan label normal dan anomali, melainkan langsung dengan pelabelan serangan
2. Pada penelitian ini hanya mengambil 30.000 data dikarenakan memory error jika melebihi dari 30.000 data, untuk penelitian selanjutnya agar lebih diperhatikan mengenai spesifikasi dari komputer yang digunakan karena mempengaruhi proses banyaknya data yang dapat dideteksi.
3. Pada penelitian ini, hanya terdapat dua tipe paket yaitu paket tcp dan udp, diharapkan adanya pengembangan dalam *generate* data trafik dengan menyerupai trafik pada umumnya, tidak hanya mengacu pada kedua tipe paket ini
4. Pada NS2 dalam penelitian ini tidak mengetahui *service* yang digunakan pada trafik, dengan tidak adanya service dapat mengurangi fitur yang akan di preprocessing. Oleh sebab itu, penelitian selanjutnya dapat menambahkan modul baru pada NS2 untuk dapat menyerupai hasil trafik yang menyerupai sebenarnya.
5. Pada penelitian ini, *Threshold* algoritma BIRCH tidak diupdate karena akan menambah proses waktu *clustering*. Oleh sebab itu, penelitian selanjutnya dapat melakukan peng *update* an terhadap nilai *Threshold* dengan waktu proses yang lebih singkat sehingga dapat lebih meningkatkan akurasi deteksi dengan cepat dan cocok untuk sistem trafik *datastream*.

DAFTAR PUSTAKA

[1] Yudha Purwanto, Kuspriyanto, Hendrawan, dan Budi Rahardjo, "Traffic Anomaly Detection in DDoS Flooding Attack," *THE 8TH INTERNATIONAL CONFERENCE ON TELECOMMUNICATION SYSTEM,SERVICES, AND APPLICATION 2014,* 2014.

[2] Tian Zhang, Raghu Ramakrishnan, Miron Livny, "BIRCH : An Efficient Data Clustering Method for Very Large Dataset," 1997.

[3] Teerawat Issariyakul, Ekram Hossain, Introduction to Network Simulator NS2, New York: Springer, 2012.

[4] Gina Puspita Sari, Adiwijaya, SSi., MSi, Moch Arif Bijaksana Ir., M.Tech, "Analisis dan Implementasi Metode Clustering BIRCH Untuk Segmentasi Pelanggan," *Telkom University Final Project,* 2007.

[5] Kemas Rahmat, Bimo Aryo Putro, Shaufiah, "Implementasi Density based Clustering Application with Noise (DBSCAN) dalam perkiraan Terjadinya Banjir di Bandung," 2011.

[6] H. Gunes Kayacik, A. Nur Zincir-Heywood, Malcom I. Heywood, "Selecting Features for Intrusion Detection : A Feature Relevance Analysis on KDD 99 Intrusion Detection Datasets," 2005.

[7] I Made Suwija Putra, "Segmentasi Citra Remote Sensing Laut Dengan Metode Clustering DBSCAN," *Majalah Ilmiah Teknologi Elektro,* 2013.

[8] Sami Abbas Nagar, Sulaiman Mohd Nor, Mohamed Saad Boba, "Generation of Real World Traffic Using NS2 Traffic Agents," 2014.

[9] Kleanthis Malialis, "Distributed Reinforcement Learning for Network Intrusion Response," 2014.

[10] Guojun Gan, Chaoqun Ma, Jianhong Wu, Data Clustering, Theory Algorithms, and Application, SIAM, 2007.

[11] T. Zhang, Data Clustering For Very Large Dataset Plus Applications, 1997.

[12] Angger Kartayasa Pribadi Putra, Yudha Purwanto, Astri Novianty, "Analisis Sistem Deteksi Anomali Trafik Menggunakan Algoritma CURE (Clustering Using Representatives) dengan Koefisien Silhouette dalam Validasi Clustering," *Telkom University Final Project,* 2015.

[13] Hanif Nurohman, Yudha Purwanto, Hafiudin, "Analisis Self-similiar untuk Sistem Deteksi Anomali dengan Estimasi Hurst Exponent menggunakan Metode R/S," *Telkom University Final Project,* 2015.

[14] Trinita S.P, Yudha Purwanto, Tito Waluyo Purboyo, "Analisis Metode Covariance Matrix Menggunakan Teknik Sliding Window untuk Sistem Deteksi Anomali Trafik," *Telkom University Final Project,* 2015.

[15] Martin Ester, Hans-Peter Kriegel, Jorg Sander, Xiaowei Xu, "A Density-Based Algorithm for Discovering Cluster in Large Spatial Database with Noise," 1996.