ISSN : 2407 - 3911

METODE TRANSFORMASI "CAKAR AYAM" UNTUK MEREDUKSI *SEARCH-SPACE* PADA *INTRUSION DETECTION SYSTEM* BERBASIS *K-NEAREST NEIGHBOR*

Kharisma Muchammad, Thomas Brian
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Kampus ITS Sukolilo, Surabaya 60111
kharisma_muchammad@yahoo.co.id, thomasbrian2112@yahoo.com

Abstrak

Penggunaan *Intrusion Detection System* (IDS) pada jaringan komputer merupakan hal yang diperlukan untuk menjaga keamanan jaringan. Beberapa IDS berbasis *K-nearest neighbor* (KNN) memiliki akurasi yang relatif baik namun jika data *training* terlalu besar, waktu yang diperlukan untuk mendeteksi serangan juga meningkat. Waktu untuk deteksi bisa ditekan dengan mereduksi search space pada data *training*. Namun problem reduksi *search space* dengan mempertahankan kualitas deteksi masih merupakan problem terbuka. Pada artikel ini diajukan suatu metode transformasi "cakar ayam" berbasis jumlah jarak data ke *centroid* dan jarak data ke dua *sub-centroid* untuk mereduksi *search space* pada IDS berbasis *K-nearest neighbor*. *Localized K-nearest neighbor* dilakukan pada data yang telah tertransformasi. Eksperimen menggunakan *agglomerative hierarchial clustering* dengan *Unweighted Pair-Group Method of Centroid* pada dataset *NSL-KDD 20%* menunjukkan penurunan *search space* maksimum sebesar 38% dengan tingkat akurasi sebesar 77.5%. Tingkat akurasi dan *specificity* maksimum yang dicapai pada eksperimen sebesar 88% dan 88.3% dengan tingkat reduksi sebesar 12% dan tingkat *sensitifity* maksimum yang dicapai sebesar 80.2% pada tingkat reduksi 11%. Berdasarkan eksperimen, luas *search space* dapat dikurangi sambil menjaga akurasi deteksi. Rasio *tradeoff* antara akurasi dan search space mungkin dapat diperbaiki dengan mengganti algortima *clustering* dengan *divisive hierarchial clustring*.

Kata kunci :

clustering, deteksi intrusi, keamanan jaringan

Abstract

Intrusion detection System (IDS) for computer network has became an essential needs to ensure network security. Some K-nearest neighbor (KNN) based IDS have a relatively good accracy in detecting attack, but the need to use all training data costs time consumption . Detection time cost can be reduced by reducing search space needed for the algorithm. The problem of search space reduction while maintaining decent accuracy still an open problem. In this Paper we propose a new transformation method "chiken claw" method. which based on sum of two distances. The first distance is the distance of data and its cluster. The later is distance of data to 2 of its cluster's sub-centroid..This method is proposed to reduce the search space on K-nearest neighbor based IDS because the search is based on resulted one dimentional transformed data. Experiment using Unweighted Pair-group Method of centroid on NSL-KDD 20% shows maximum search space reduction 38% with 75% accuracy. Maximum accuracy and sensitivity in the experiment is 88% and 88.3% respectively with space reduction 12%. Maximum sensitivity from experiment is at 80.2% with 11% space reduction. Based on experiments, search space can be reduced while maintaining accuracy. Search space-accuracy trade off might be improved by using different clustering algorithm such as divisive hierarchial clustering

Keywords:

clustering, intrusion detection, network security

I. PENDAHULUAN

Kebutuhan akan keamanan pada sistem komputer telah menjadi suatu kebutuhan yang esensial. Salah satu permasalahan keamanan yang sering terjadi adalah serangan pada jaringan komputer. Salah satu cara untuk mengatasi permasalahan tersebut adalah dengan menggunakan sistem deteksi intrusi (*Intrusion detection System/IDS*).

IDS dapat dibagi menjadi 2 kelompok. Kelompok pertama adalah IDS berbasis *signature* dan kelompok kedua berbasis anomali. Kelompok pertama menggunakan basis data serangan yang dikenali untuk membedakan antara aktifitas normal dan aktifitas serangan. Keunggulan pendekatan ini adalah sumber daya yang diperlukan untuk memeriksa aktifitas yang dipakai relatif kecil (Garcia-Teodoro dkk, 2009). Kelemahan pendekatan ini adalah tidak mampu mengenali serangan yang tidak terdapat dalam basis datanya. Pendekatan kedua menggunakan suatu model normal untuk mendeteksi serangan. Aktifitas yang memiliki deviasi terlalu jauh akan dideteksi sebagai aktifitas serangan. Salah satu kelebihan pendekatan ini adalah mampu mendeteksi serangan yang baru karena tidak bergantung pada basis data (Garcia-Teodoro dkk, 2009). Kelemahan pendekatan berbasis anomali adalah sumber daya yang diperlukan untuk membentuk model maupun deteksi relatif lebih tinggi dibandingkandengan pendekatan metode pertama dan tingkat *false positive* yang lebih tinggi dibandingkan dengan IDS berbasis *signature* (Garcia-Teodoro dkk, 2009).

IDS berbasis anomali sendiri dapat dibagi menjadi beberapa kelompok berdasarkan metode untuk membentuk model normal maupun klasifikasi aktifitas. Salah satu kelompok tersebut adalah IDS dengan pendekatan *machine learning* dengan *K-nearest neighbor*. Beberapa contoh IDS yang menggunakan *K-nearest neighbor* untuk proses klasifikasi antara lain CANN (Lin dkk, 2014) dan TANN (Tsai dan Lin, 2009). Masalah yang dialami oleh kedua IDS tersebut antara lain *search space* yang dipakai adalah seluruh data *training*. Jika ukuran data *training* terlalu besar, waktu yang diperlukan untuk mendeteksi serangan akan meningkat secara drastis. Penelitian ini mengajukan skema transformasi baru untuk mengakomodasi pencarian secara lokal dengan mereduksi *search space*. Secara intuitif penurunan *search space* dengan metode *clustering*, secara umum dapat mempercepat proses pencarian terdekat. Hal ini terjadi karena data yang dicari dalam KNN biasanya berada dalam satu *cluster* meski pada beberapa kasus tertentu tetangga terdekat bisa saja berada pada cluster yang berbeda. Hal ini dapat terjadi jika ukuran cluster terlalu kecil dan jarak antar *cluster* relatif kecil.

Penggunaan centroid dikarenakan *centroid* dapat dianggap sebagai "wakil" dari data dalam suatu cluster dan jarak data dari centroid dapat dipakai unutk membedakan data dalam suatu dataset. *Subcentroid* dipakai karena jika jarak dari centroid dapat dipakai untuk membedakan data dalam suatu dataset, maka jarak ke subcentroid dapat dipakai untuk membedakan data dalam suatu *cluster*.

II. KAJIAN LITERATUR

Penggunaan *K-nearest neighbor* untuk mendeteksi intrusi jaringan telah dilakukan oleh beberapa penelitian, beberapa diantaranya (Lin dkk, 2014), (Tsai dan Lin, 2009), (Liao dan Vemuri, 2002), dan (Ma dan Kaban , 2013). Beberapa penelitian fokus pada transformasi fitur (Lin dkk, 2014), (Tsai dan Lin, 2009) dan beberapa pada penggunaan jarak baru selain jarak *euclidean* (Ma dan Kaban , 2013).

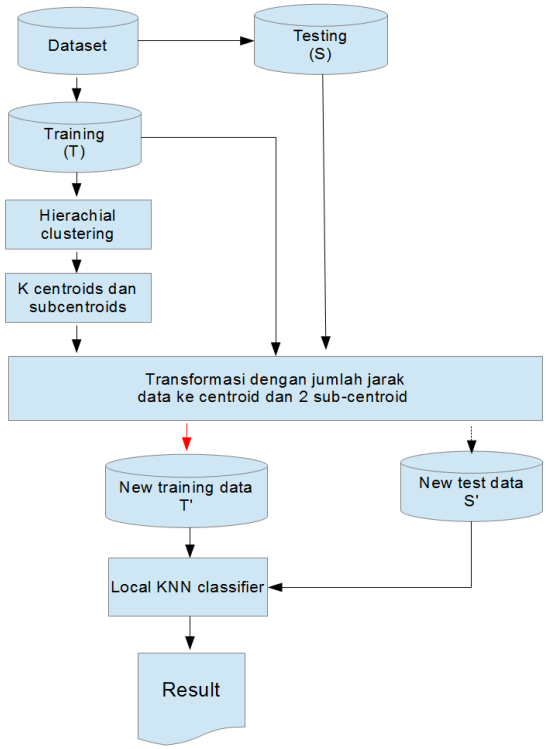
Pada TANN (Tsai dan Lin, 2009), transformasi fitur dilakukan dengan mencari luas total segitiga yang terbentuk antara data dan 2 dari 5 *centroid*. 5 *centroid* tadi dicari dengan algoritma *K-means*. Sedangkan pada CANN (Tsai dkk, 2014), transformasi fitur dilakukan dengan menjumlahkan jarak dari 5 *centroid* ke data dan satu tetangga terdekat dari data tersebut yang masih satu *cluster*. Hasil transformasi TANN maupun CANN kemudian dimasukkan ke *KNN* untuk mendeteksi apakah data termasuk aktifitas serangan atau aktifitas normal.

Permasalahan yang terjadi pada kesemua metode tersebut adalah *search space* yang dipakai untuk *KNN* tumbuh seiring dengan ukuran data *training* yang digunakan. Beberapa peneliti seperti Garcia dkk (2008) mengatasi hal ini dengan menggunakan komputasi parallel dengan teknologi CUDA dari NVIDIA namun tidak semua komputer dilengkapi dengan kartu grafis yang dapat melakukan hal tersebut sehingga teknik reduksi *search space* masih bisa dikatakan relevan.

Pada makalah ini, diajukan suatu skema transformasi fitur yang mengakomodasi reduksi *search space* dengan menjaga akurasi deteksi.

III. METODE YANG DIAJUKAN

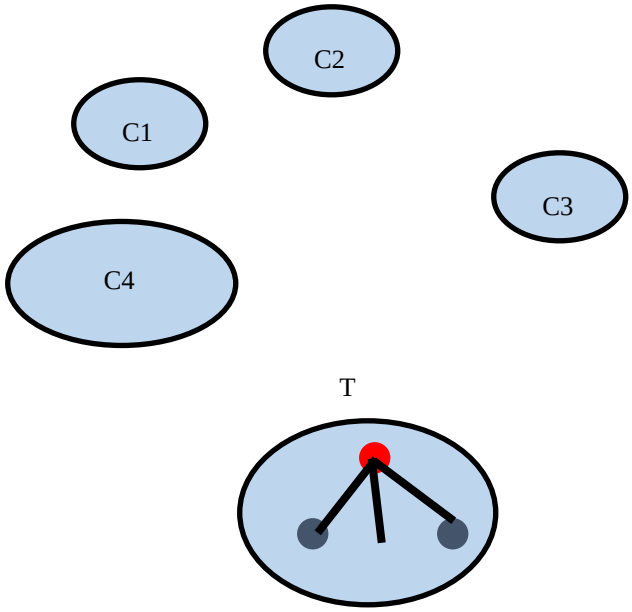
Alur metode yang diajukan dapat dilihat pada Gambar 1. Data *training* di-*cluster* dengan suatu algoritma *clustering* hingga didapatkan *K cluster* dan *K centroid*. Langkah kedua yang dilakukan adalah mencari 2 *sub centroid* dari tiap-tiap *cluster*. Langkah ini tidak perlu dilakukan pada *cluster* yang hanya memiliki 1 anggota. Langkah ketiga yang dilakukan adalah mentransformasi data pada tiap *cluster*.



Gambar 1. Alur metode yang diajukan

Data ditransformasikan dengan persamaan (1) dimana *T* adalah data yang akan ditransformasikan, *C* adalah *centroid* yang paling dekat dengan *T*, *SC1* adalah *sub-centroid* pertama dari *C* dan *SC2* adalah *sub-centroid* kedua dari C. ||T-C|| adalah jarak antara data cetroid. ||T-SC1|| dan ||T-SC2|| adalah jarak antara data dengan *sub-centroid* 1 dan *sub-centoid* 2. Skema transformasi yang diajukan dapat dilihat pada Gambar 2. Seperti TANN dan CANN jarak ke *centroid* dipakai sebagai fitur untuk membentuk fitur baru namun karena tujuan dari metode ini adalah pencarian lokal, hanya jarak ke satu *centroid* yang dipakai. Dua sub-*centroid* dipakai karena jumlah *sub-centroid* yang paling kecil dari cluster adalah 2, jika *cluster* tersebut tidak berisi satu data saja.

$$T' = ||T\text{-}C|| + ||T\text{-}SC1|| + ||T\text{-}SC2|| \quad \text{... [1]}$$



Gambar 2. Skema transformasi yang diajukan

Pada fase *testing*, tiap data *testing S* diletakkan pada *cluster* yang *centroid*-nya paling dekat, kemudian ditransformasi dengan persamaan [1] dan menghasilkan *S'*. Data *testing* yang telah ditransformasikan tadi kemudian dicari *N* tetangga terdekatnya yang masih 1 *cluster* dari data *training* yang telah tertransformasi.

IV. DESAIN EKSPERIMEN

IV.1 *Dataset*

Dataset yang dipakai adalah dataset NSL-KDD 20% yang didapatkan dari (Cheng Hong). NSL-KDD merupakan hasil pengolahan dari KDD99 yang dipakai untuk kontes data mining KDD nuggets. KDD99 sendiri berasal dari dataset DARPA. Data NSL-KDD ini dihasilkan Tavallaee dkk (2009) karena dataset aslinya (KDD99) dianggap terlalu mudah untuk diolah sehingga mereka memilih

sebagian data yang dianggap representatif dari dataset tersebut dan yang sulit untuk dideteksi.

Dataset ini berisi atribut koneksi dari klien ke server. Atribut yang dicatat terdiri dari 41 fitur antara lain tipe koneksi (TCP,UDP), lama koneksi (detik), jenis protokol (HTTP,FTP,SSH, dll) dan lain-lain.

Kolom yang dibuang adalah kolom protocol_type, service dan flag ketiga kolom ini dibuang karena data kategorikal tidak bisa diolah. Data duplikat kemudian dibuang dengan perintah Unix sort dan uniq. Data duplikat dibuang agar model yang dihasilkan tidak bias ke sekelompok data yang sering muncul. Statistik atribut pada dataset dapat dilihat pada tabel 1. Pada tabel tersebut dapat dilihat atribut apa saja yang dipakai dalam eksperimen ini, nilai rata-rata, nilai maksimum, nilai minimum, dan variannya. Untuk keperluan 10 *fold cross validation* data kemudian dipecah menjadi 10 partisi dengan komposisi tiap partisi dapat dilihat pada Tabel 2. Pada kolom pertama dapat dilihat label serangan dan kolom berikutnya distribusi jumlah data serangan yang terdapat pada partisi tersebut. Dapat dilihat pada Tabel 2 komposisi tiap partisi hampir sama.

IV.2 *Hierachial Clustering*

Algoritma *hierarchial clustering* yang digunakan dalam eksperimen adalah metode yang diusulkan oleh Bougettaya dkk (2014) dengan menggunakan *UPGMC (Unweighted Pair-Group Method of Centroid)* untuk mencari pasangan *cluster* yang paling dekat. Algoritma ini memerlukan 2 parameter dari pengguna. Parameter pertama adalah jumlah *cluster* awal (K_{awal}) dan jumlah *cluster* final (K_{akhir}) dimana $K_{awal}>K_{akhir}$. Eksperimen ini akan menggunakan beberapa nilai K_{awal} dan K_{akhir} untuk melihat pengaruh kedua parameter tersebut terhadap akurasi deteksi.

IV.3 *K-Nearest Neighbor*

Nilai *K* pada *KNN* yang dipakai adalah 1. Hal ini untuk memudahkan pembuatan *confusion matrix*, karena terdapat beberapa kelas dalam dataset.

IV.4 Kriteria Penilaian

Kriteria penilaian yang dipakai dalam eksperimen ini antara lain *accuracy*, *sensitivity*, *specificity* dan *search space reduction*. Rumus untuk mendapatkan ketiganya dapat dilihat pada persamaan [2], [3], [4] dan [5] dimana *TP* adalah *true positive*, *TN* adalah *true negative*, *FP* adalah *false positive* dan *FN* adalah *false negative*. S_{awal} adalah jumlah data *training* awal dan S_{max} adalah ukuran *cluster* paling besar pada seluruh skenario untuk parameter uji tersebut.

$$Accuracy = TP+TN/(TP+TN+FP+FN) \quad \text{... [2]}$$
$$Sensitivity = TP/(TP+FN) \quad \text{... [3]}$$
$$Specificity = TN/(TN+FP) \quad \text{... [4]}$$
$$Reduction = (S_{awal}-S_{max})/ S_{awal} \quad \text{... [5]}$$

IV.5 Hasil Eksperimen dan Diskusi

Hasil eksperimen dapat dilihat pada tabel 3 dengan nilai paling tinggi untuk tiap kriteria ditebalkan. Dari tabel tersebut dapat dilihat pengaruh nilai parameter K_{awal} dan K_{akhir} pada akurasi ,*sensitivity* dan *specificity*. Makin besar K_{awal} dan makin kecil K_{akhir} meningkat pula akurasi,*sensitivity*, dan*spesificity*. Nilai akurasi terbesar didapatkan dari nilai K_{awal} 60 dan nilai K_{akhir} 40 sebesar 0.838 begitu pula nilai *specificity* sebesar 0.883 namun nilai *sensitivity* paling tinggi diraih pada nilai K_{awal} 60 dan K_{akhir} 30 sebesar 0.802. Perbedaan minor akurasi pada 60 40 dan 60 30 mungkin terjadi karena fluktuasi pada algoritma *clustering*.

Nilai akurasi tersebut masih dibawah metode TANN (Tsai dan Lin, 2009) dan CANN (Lin dkk, 2014) sebesar 0.990 dan 0.997. Meski terjadi penurunan *search space*, penurunan akurasi pada metode ini terjadi lebih besar dari pada penurunan *search space*. Pada beberapa kasus mungkin hal ini tidak dapat ditolerir. Salah satu dugaan yang muncul untuk menjelaskan performa metode ini adalah data yang tidak dinormalisasi. Dugaan ini muncul melihat pada tabel 1 nilai varian, maksimum dan minimum tiap atribut tidak terdistribusi secara merata. Meskipun beberapa metode seperti CANN (Lin dkk, 2014) dan TANN (Tsai dan Lin, 2009) tidak memerlukan normalisasi data, mungkin metode ini memerlukan proses normalisasi untuk mencapai hasil yang lebih baik.

Kemungkinan lain yang mungkin menyebabkan performa yang sub-optimal adalah distribusi data pada dataset yang menumpuk pada satu *cluster* saja. Hal ini menyebabkan *cluster* tersebut berisi campuran data serangan dan normal. Hal ini sangat berpengaruh pada akurasi deteksi karena pada eksperimen ini pencarian hanya dilakukan secara lokal. Penggunaan algoritma *clustering* yang lain seperti *divisive hierarchial clustering* dengan

minimisasi *entropy* mungkin dapat membantu meningkatkan akurasi deteksi. Dengan *cluster* yang homogen, secara intuitif, akurasi deteksi dapat ditingkatkan dengan mempertahankan tingkat reduksi karena kemungkinan besar ukuran *cluster* yang dihasilkan relatif kecil.

Nilai *search space reduction* paling tinggi didapatkan pada parameter K_{awal} 60 dan K_{akhir} 60 dengan nilai 0.386. Hasil ini sesuai ekspektasi dimana *reduction rate* akan meningkat seiring nilai K_{akhir}. Berdasarkan eksperimen, nilai *space reduction* masih jauh dari batas teoritis pada eksperimen dengan K_{akhir} 60 sebesar 0.983 dimana seluruh *dataset* dibagi secara merata menjadi 60 *cluster*. Tingkat *reduction rate* ini terjadi akibat penumpukan data pada satu *cluster* saja. Penggunaan *divisive hierachial clustering* mungkin dapat memperbaiki nilai ini.

Confusion matrix tiap eksperimen dapat dilihat pada tabel 4 hingga 7. Pada kolom pertama adalah data aktual dan kolom berikutnya adalah hasil klasifikasi data tersebut. Pada seluruh eksperimen jenis serangan *u2r* adalah yang paling sulit diklasifikasi. Hanya 1 data *u2r* yang secara benar diklasifikasi yaitu pada eksperimen 60 60 pada tabel 4. kebanyakan jenis serangan *u2r* diklasifikasi sebagai aktifitas normal. Dalam hal ini performa CANN tidak jauh berbeda dengan performa metode ini dengan tingkat deteksi sebesar 3.85%. Sedangkan dibandingkan TANN metode ini masih jauh dibawah dengan tingkat deteksi sebesar 60%.

V. KESIMPULAN

Berdasarkan eksperimen diatas, metode yang diajukan menunjukkan nilai akurasi dan *spesificity* yang paling tinggi didapatkan pada nilai parameter K_{awal} 60 dan K_{akhir} 40 sebesar 0.838 dan 0.883 dan tingkat reduksi sebesar 0.126. Nilai *sensitivity* yang paling tinggi didapatkan dengan nilai parameter K_{awal} 60 dan K_{akhir} 30 sebesar 0.802 dan tingkat reduksi sebesar 0.113. Tingkat reduksi paling tinggi didapatkan dengan nilai parameter K_{awal} 60 dan K_{akhir} 60 sebesar 0.386.

Penelitian lanjutan dengan *dataset* yang berbeda diperlukan untuk validasi metode lebih lanjut. Berdasarkan hasil eksperimen penurunan *search space* dengan metode ini menimbulkan penurunan akurasi yang sama besarnya. Beberapa langkah yang mungkin dapat memperbaiki performa metode ini antara lain penggunaan *divisive hierachial clustering*.

REFERENSI

Bougettaya, A. Yu, Q., Liu,X., Zhou, X., Song, A.. (2014). Efficient Agglomerative Hierachial Clustering. *Expert System with Applications*, vol 42, pp. 2785-2798.

Cheng Hong Low: Tersedia : https://github.com/defcom17/NSL_KDD

Garcia-Teodoro. P, Diaz-Verdejo. J, Marcia-Fernandez. G, Vazquez, E. (2009). Anomaly-based network intrusion detection: Technique, system and challenges. *Journal of Computers and Security*, vol 28, no. 1, pp. 18–28.

Garcia. V, Debreuve. E, dan Barlaud. M. (2008). Fast K Nearest Neighbor Search using GPU. *Computer Vision and Pattern Recognition Workshop*. Pp. 1-6.

Lin, W.C., Ke, S.W., Tsai, C.F. (2014). CANN: An intrusion detection system based on combining cluster center and nearest neighbors". *Knowledge-based systems*. Vol 78, pp. 13-21.

Liao, Y., Vemuri, V.R. (2002). Using k-nearest neighbor classifier for intrusion detection. *Computers & Security*, vol 21, no. 5, pp. 439-448.

Ma, Z., Kaban, A. (2013). K-Nearest-Neighbours with a Novel Similarity Measure for Intrusion Detection. *UK Workshop for Computational Intelligence (UKCI)*, 2013, pp. 266-271.

Tavallae. M, Bagheri. E, Lu. W, Ghorbani A.A. (2009). A Detailed Analysis of the KDD CUP 99 Dataset.*IEEE Symposium on Computational Intelligence for Security and Defense Applications.* Pp. 1-6.

Tsai, C.F., Lin, C.Y. (2009). A triangle area based nearest neigbors approach to intrusion detection. *Pattern recognition*, vol 43, no 1. Pp. 222- 229.

Tabel 1. Komposisi Dataset

Nama Atribut	Rata-rata	Maksimum	Minimum	Varian
Duration	$3.10 \times 10^{+2}$	$4.29 \times 10^{+4}$	0.00	$7.33 \times 10^{+6}$
src_bytes	$2.47 \times 10^{+4}$	$3.82 \times 10^{+8}$	0.00	$5.91 \times 10^{+12}$
dst_bytes	$3.55 \times 10^{+3}$	$5.15 \times 10^{+6}$	0.00	$8.02 \times 10^{+9}$
Land	8.07×10^{-5}	$1.00 \times 10^{+0}$	0.00	8.07×10^{-5}
wrong_fragment	2.41×10^{-2}	$3.00 \times 10^{+0}$	0.00	6.88×10^{-2}
urgent	4.03×10^{-5}	$1.00 \times 10^{+0}$	0.00	4.03×10^{-5}
hot	2.01×10^{-1}	$7.70 \times 10^{+1}$	0.00	$4.72 \times 10^{+0}$
num_failed_logins	1.21×10^{-3}	$4.00 \times 10^{+0}$	0.00	2.10×10^{-3}
logged_in	4.01×10^{-1}	$1.00 \times 10^{+0}$	0.00	2.40×10^{-1}
num_compromised	2.32×10^{-1}	$8.84 \times 10^{+2}$	0.00	$1.10 \times 10^{+2}$
root_shell	1.57×10^{-3}	$1.00 \times 10^{+0}$	0.00	1.57×10^{-3}
su_attempted	1.37×10^{-3}	$2.00 \times 10^{+0}$	0.00	2.42×10^{-3}
num_root	2.54×10^{-1}	$9.75 \times 10^{+2}$	0.00	$1.34 \times 10^{+2}$
num_file_creations	1.50×10^{-2}	$4.00 \times 10^{+1}$	0.00	2.85×10^{-1}
num_shells	3.63×10^{-4}	$1.00 \times 10^{+0}$	0.00	3.63×10^{-4}
num_access_files	4.40×10^{-3}	$8.00 \times 10^{+0}$	0.00	9.86×10^{-3}
num_outbound_cmds	0.00	$0.00 \times 10^{+0}$	0.00	0.00
is_host_login	0.00	$0.00 \times 10^{+0}$	0.00	0.00
is_guest_login	9.28×10^{-3}	$1.00 \times 10^{+0}$	0.00	9.19×10^{-3}
count	$8.28 \times 10^{+1}$	$5.11 \times 10^{+2}$	$1.00 \times 10^{+0}$	$1.30 \times 10^{+4}$
srv_count	$2.80 \times 10^{+1}$	$5.11 \times 10^{+2}$	$1.00 \times 10^{+0}$	$5.33 \times 10^{+3}$
serror_rate	2.80×10^{-1}	$1.00 \times 10^{+0}$	0.00	1.97×10^{-1}
srv_serror_rate	2.77×10^{-1}	$1.00 \times 10^{+0}$	0.00	1.97×10^{-1}
rerror_rate	1.16×10^{-1}	$1.00 \times 10^{+0}$	0.00	9.95×10^{-2}
srv_rerror_rate	1.18×10^{-1}	$1.00 \times 10^{+0}$	0.00	1.02×10^{-1}
same_srv_rate	6.69×10^{-1}	$1.00 \times 10^{+0}$	0.00	1.91×10^{-1}
diff_srv_rate	6.19×10^{-2}	$1.00 \times 10^{+0}$	0.00	3.19×10^{-2}
srv_diff_host_rate	9.75×10^{-2}	$1.00 \times 10^{+0}$	0.00	6.67×10^{-2}
dst_host_count	$1.81 \times 10^{+2}$	$2.55 \times 10^{+2}$	0.00	$9.87 \times 10^{+3}$
dst_host_srv_count	$1.17 \times 10^{+2}$	$2.55 \times 10^{+2}$	0.00	$1.23 \times 10^{+4}$
dst_host_same_srv_rate	5.20×710^{-1}	$1.00 \times 10^{+0}$	0.00	2.01×10^{-1}
dst_host_diff_srv_rate	8.16×10^{-2}	$1.00 \times 10^{+0}$	0.00	3.46×10^{-2}
dst_host_same_src_port_rate	1.48×10^{-1}	$1.00 \times 10^{+0}$	0.00	9.53×10^{-2}
dst_host_srv_diff_host_rate	3.24×10^{-2}	$1.00 \times 10^{+0}$	0.00	1.24×10^{-2}
dst_host_serror_rate	2.79×10^{-1}	$1.00 \times 10^{+0}$	0.00	1.95×10^{-1}
dst_host_srv_serror_rate	2.73×10^{-1}	$1.00 \times 10^{+0}$	0.00	1.96×10^{-1}
dst_host_rerror_rate	1.15×10^{-1}	$1.00 \times 10^{+0}$	0.00	9.14×10^{-2}
dst_host_srv_rerror_rate	1.16×10^{-1}	$1.00 \times 10^{+0}$	0.00	9.86×10^{-2}

Tabel 2. Komposisi Tiap Dataset

Label	1	2	3	4	5	6	7	8	9	10
normal	1345	1344	1345	1344	1345	1344	1345	1344	1345	1344
rootkit	0	1	0	1	0	1	0	1	0	0
portsweep	56	56	56	56	56	56	56	56	56	57
warezmaster	0	0	0	1	1	1	1	1	1	1
satan	69	69	69	69	69	69	69	69	68	68
multihop	0	0	0	0	0	0	0	0	1	1
back	20	20	20	20	19	19	19	19	20	20
ipsweep	70	71	71	71	71	71	71	71	71	71
buffer_overflow	0	1	1	1	1	0	1	1	0	0
warezclient	19	18	18	18	18	18	18	18	18	18
imap	1	0	0	0	1	0	0	1	1	1
ftp_write	1	0	0	0	0	0	0	0	0	0
spy	0	0	0	0	0	1	0	0	0	0
phf	0	0	0	0	0	0	1	1	0	0
nmap	30	30	30	30	30	30	29	29	29	29
pod	4	4	4	3	3	4	4	4	4	4
smurf	52	53	53	53	53	53	53	53	53	53
teardrop	19	19	19	19	19	19	18	18	19	19
guess_passwd	1	1	1	1	1	1	1	1	1	1
loadmodule	1	0	0	0	0	0	0	0	0	0
land	0	0	0	0	0	0	1	0	0	0
neptune	791	792	792	792	792	792	792	792	792	792

Tabel 3. Hasil Eksperimen

K_{awal}	K_{akhir}	Akurasi	*Sensitivity*	*Specificity*	*Reduction*
60	60	0.775	0.712	0.828	**0.386**
60	40	**0.838**	0.785	**0.883**	0.126
40	40	0.738	0.671	0.795	0.271
60	30	0.834	**0.802**	0.862	0.113

Tabel 4. Confusion Matrix pada 60 60

Aktual	Prediksi				
	Dos	**probe**	**u2r**	**r2l**	**normal**
dos	5580	973	0	3	2315
probe	473	939	0	10	832
u2r	0	0	1	0	10
r2l	1	16	0	90	102
normal	1205	1111	24	144	10961

Tabel 5. Confusion Matrix pada 60 40

Aktual	Prediksi				
	Dos	**probe**	**u2r**	**r2l**	**normal**
dos	5259	1801	0	31	1780
probe	1029	725	0	14	486
u2r	1	0	0	0	10
r2l	6	12	0	39	152
normal	1011	410	9	143	11872

Tabel 6. Confusion Matrix pada 40 40

Aktual	Prediksi				
	Dos	**probe**	**u2r**	**r2l**	**normal**
dos	5074	937	0	48	2812
probe	586	870	0	22	776
u2r	0	1	0	0	10
r2l	26	11	2	41	129
normal	1447	1081	22	199	10696

Tabel 7. Confusion Matrix pada 60 30

Aktual	Prediksi				
	Dos	**probe**	**u2r**	**r2l**	**normal**
dos	5412	1860	2	38	1559
probe	996	721	0	6	531
u2r	2	1	0	0	8
r2l	11	5	0	46	147
normal	1072	555	11	211	11596