**KLASIFIKASI SERANGAN *DISTRIBUTED DENIAL OF SERVICE* (DDoS) DENGAN ALGORITMA *K-NEAREST NEIGHBOR* (KNN)**

**SKRIPSI**

****

**NI KADEK SUKMA PUTRI RAHAYU**

**1708561038**

**PROGRAM STUDI INFORMATIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS UDAYANA**

**JIMBARAN**

**2021**

# **SURAT PERNYATAAN KEASLIAN KARYA ILMIAH**

# **LEMBAR PENGESAHAN**

# **ABSTRAK**

# **KATA PENGANTAR**

Proposal penelitian dengan judul Klasifikasi Serangan *Distributed Denial Of Service* (DDoS) dengan Algoritma *K-Nearest Neighbor* (KNN) ini disusun dalam rangkaian pelaksanaan Tugas Akhir di Program Studi Informatika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Udayana. Proposal ini disusun dengan harapan dapat menjadi pedoman dan arahan dalam melaksanakan penelitian selanjutnya.

Sehubungan dengan telah selesainya pengerjaan dari proposal ini, maka diucapkan terimakasih dan penghargaan kepada berbagai pihak yang telah membantu untuk pengusulan proposal ini, antara lain:

1. Bapak I Komang Ari Mogi, S.Kom., M.Kom. sebagai Pembimbing I yang telah banyak membantu menyempurnakan proposal ini;
2. Bapak Cokorda Rai Adi Pramartha,ST.MM.PhD sebagai Pembimbing I yang telah banyak membantu menyempurnakan proposal ini;
3. Bapak-bapak dan Ibu-ibu dosen pengajar di Program Studi Informatika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Udayana yang telah meluangkan waktu untuk memberikan saran dan masukan dalam menyempurnakan proposal ini;
4. Kawan-kawan di Program Studi Informatika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Udaayana yang telah memberikan dukungan moral dalam penyelesasian proposal ini.

Disadari pula bahwa sudah tentu proposal ini masih memiliki kelemahan dan kekurangan didalamnya. Memperhatikan hal ini, maka dari itu masukan dan saran-saran penyempurnaan untuk proposal ini sangat diharapkan.

Jimbaran, Juni 2020

Penyusun

# **DAFTAR ISI**

# **DAFTAR TABEL**

# **DAFTAR GAMBAR**

# **DAFTAR LAMPIRAN**

# **BAB I**

# **PENDAHULUAN**

## **Latar Belakang**

Pada era milenial ini teknologi berkembang dengan sangat pesat, bahkan kehidupan manusiapun sangat bergantung dengan teknologi yang ada. Teknologi digunakan dalam berbagai bidang mulai dari bidang kesehatan, perbankan, sosial, dan lain-lain. Namun semakin berkembangnya teknologi sekarang ini, maka semakin banyak juga resiko yang akan dihadapi. Salah satu resikonya adalah serangan yang dikirimkan oleh oknum yang tidak bertanggungjawab. Serangan yang dikirimkan terkadang susah dideteksi sehingga membuat tingkat keamanan dari jaringan komputer sangatlah rentan. Banyak jenis serangan yang dapat dikirimkan, tentu saja serangan ini akan membuat komputer menjadi lambat. Jenis serangan yang dikirimkan oleh onum yang tidak bertanggung jawab tersebut tentunya dapat merugikan pihak lain. Untuk meminimalisir resiko dari serangan, maka dibutuhkan sebuah sistem yang dapat digunakan untuk membaca pola yang menyimpang dari jaringan komputer. Sistem tersebut diharapkan dapat mendeteksi pola yang menyimpang dari sebuah jaringan komputer dengan hasil akurasi yang tinggi.

Salah satu kelas serangan yang digunakan oleh penyerang adalah Distributed Denial of Service (DDoS). Serangan Distributed Denial of Service (DDoS) merupakan serangan yang dikirimkan dari beberapa penyerang terhadap sebuah komputer atau server dalam jumlah yang melebihi kemampuan komputer itu sendiri (Satmoko, Sukarno, & Jadied, 2018). DDoS merupakan suatu ancaman yang serius karena tujuan utama dari serangan DDoS yaitu untuk membuat sumber daya jaringan target tidak dapat digunakan (Prakash & Priyadarshini, 2018). Serang DDoS dapat berasal dari software yang dapat di download secara gratis di Internet kemudian software tersebut memberi script kiddie (sebuah program yang digunakan untuk membantu menyerang sebuah server) untuk melakukan serangan DDoS. Serangan DDoS diluncurkan dengan cara menemukan server yang ingin diserang, kemudian dengan bantuan komputer-komputer zombie (botnet) atau komputer yang telah menginstall script kiddie dari komputer utama (penyerang), secara bersamaan akan membantu menyerang server tujuan (korban) dengan memerintahkan botnet untuk meluncurkan serangan (Hoque, Bhattacharyya, & Kalita, 2015). Tentu saja hal ini dapat mengganggu lalu lintas pada jaringan sehingga server akan sulit untuk diakses. Untuk memudahkan dalam mendeteksi anomali data dalam jaringan maka diperlukan sebuah sistem yang dapat mengklasifikasikan data anomali dalam aktivitas data normal dalam jaringan, sehingga dapat dilakukan pencegahan terhadap serangan DDoS dan mengatasi permasalahan yang dapat ditimbulkan oleh serangan DDoS.

Penelitian tentang deteksi serangan DDoS pada jaringan sebelumnya sudah pernah kembangkan oleh (Ramadhan, Sukarno, & Nugroho, 2019) dengan membandingkan dua algoritma yaitu algoritma K-Nearest Neighbor dan Decision Tree, dalam penelitian tersebut mengunakan dataset CICIDS2017. Hasil dari penelitian tersebut didapatkan bahwa algoritma KNN memiliki akurasi sebesar 98,94% dan hal tersebut menunjukan hasil dari algoritma KNN tidak lebih bagus dibandingkan dengan menggunakan algoritma Decision Tree, karena proses perhitungannya algoritma Decision Tree melibatkan nilai entropi dan information gain sehingga lebih detail daripada KNN yang perhitungannya sederhana.

Dalam penelitian yang dilakukan penulis dalam mengklasifikasikan serangan DDoS, peneliti menggunakan algoritma K-Nearest Neighbor (KNN) dengan penambahan seleksi atribut atau yang sering dikenal juga dengan istilah seleksi fitur menggunakan Information Gain untuk mendapatkan hasil yang lebih akurat. Berdasarkan penelitian yang dilakukan oleh (Kurniabudi, Harris, & Rahim, 2020) yang berjudul Seleksi Fitur dengan Information Gain untuk Meningkatkan Deteksi Serangan DDoS Menggunakan Random Forest dihasilkan bahwa teknik seleksi fitur dengan information gain mampu meningkatkan performa metode klasifikasi pada algoritma Random Forest.

Dalam penelitian yang dilakukan, dataset yang akan digunakan adalah dataset KDD Cup 99 tentang jenis anomali jaringan. Sebelum masuk ke tahap pengujian dataset, terlebih dahulu dilakukan seleksi fitur terhadap dataset sehingga mendapatkan fitur yang paling relevan dalam hasil data uji yang akan digunakan nantinya. Seleksi fitur ini dilakukan menggunakan metode Information Gain. Setelah mendapatkan fitur-fitur yang akan digunakan dalam pengujian terhadap data uji, maka data uji tersebut diklasifikasikan menggunakan metode K-Nearest Neighbor sehingga dapat mengetahui pola serangan DDoS dan dapat mencegah serangan DDoS lebih awal.

## **Rumusan Masalah**

Berdasarkan latar belakang yang disampaikan maka permasalahan dalam pembuatan penelitian ini adalah :

1. Bagaimana cara untuk menseleksi fitur-fitur dari dataset yang digunakan dengan metode Information Gain.
2. Bagaimana cara mengklasifikasikan data yang normal dan data serangan Distributed Denial of Service (DDoS) menggunakan algoritma K-Nearest Neighbor (KNN).
3. Bagaimana akurasi yang dihasilkan oleh program pada pengklasifikasian serangan Distributed Denial of Service (DDoS).

## **Batasan Masalah**

Adapun beberapa batasan masalah yang akan digunakan sebagai acuan dalam penelitian ini adalah sebagai berikut :

1. Penelitian ini menggunakan data riset yang bersumber dari dataset KDD Cup 99.
2. Fitur-fitur dari dataset dipilih sebanyak 9 fitur, dimana fitur tersebut adalah fitur dengan nilai entropy tertinggi.
3. Penelitian ini hanya mengklasifikasikan data yang masuk termasuk kedalam data serangan Distributed Denial of Service (DDoS) atau data normal.
4. Pengukuran akurasi pada program diukur dengan menggunakan metode K-Fold Cross Validation.

## **Tujuan Penelitian**

Berdasarkan rumusan masalah tersebut maka adapun tujuan dalam penelitian ini adalah :

1. Mengetahui fitur-fitur yang paling relevan dan berpengaruh terhadap jenis data yang masuk.
2. Mendeteksi data yang masuk apakah data tersebut adalah data yang bersifat serangan DDoS atau data normal.
3. Mengetahui akurasi klasifikasi serangan DDoS dengan menggunakan algoritma K-Nearest Neighbor dan seleksi fitur *information gain.*

## **Manfaat Penelitian**

Beberapa manfaat yang dapat diambil dalam penelitian ini adalah sebagai berikut :

1. Mendapatkan informasi lebih awal terhadap data yang masuk yang berpotensi serangan DDoS.
2. Mendapatkan hasil yang akurat terhadap pendeteksian serangan DDoS.

## **Metodelogi Penelitian**

Metodelogi penelitian ini adalah langkah yang dilakukan oleh peneliti dalam mengumpulkan informasi atau data serta melakukan investigasi pada data yang diperoleh. Pada bagian ini, peneliti menjelaskan penyusunan tugas akhir secara sistematis. Adapun bahasan yang akan dijelaskan dalam penelitian ini yaitu desain penelitian, data penelitian, dan metode yang digunakan dalam penelitian ini.

### **Analisis Sistem**

Pada tahap ini, dilakukan identifikasi perkiraan kebutuhan yang diperlukan untuk penelitian ini berupa materi yang berhubungan dengan penelitian yang dilakukan. Hal yang diperlukan yaitu menyangkut pada penerapan metode Information Gain dalam melakukan seleksi terhadap fitur-fitur yang digunakan dalam dataset dan juga penerapan metode K-Nearest Neighbor dalam melalukan klasifikasi dalam sebuah dataset.

### **Desain Penelitian**

Penelitian ini mengambil judul “Klasifikasi Serangan *Distributed Denial Of Service* (DDoS) dengan Algoritma K-Nearest Neighbor (KNN)”, penelitian bertujuan untuk mengetahui jenis data yang masuk ke suatu perangkat apakah data tersebut adalah data yang dikirim sebagai data anomali atau data serangan atau data tersebut adalah data normal atau data yang memang benar diinginkan oleh user. Desain penelitian yang digunakan dalam penelitian ini adalah studi kasus *(Case Studi Research)* dan eksperimental. Studi kasus merupakan penelitian yang memusatkan perhatian pada suatu kasus tertentu dengan menggunakan individu atau kelompok sebagai bahan studi. Penggunaan penelitian studi kasus ini biasanya difokuskan untuk menggali dan mengumpulkan data yang lebih dalam terhadap objek yang diteliti untuk dapat menjawab permasalahan yang sedang terjadi. Sehingga bisa dikatakan penelitian bersifat deskriptif dan eksploratif (Hasibuan, 2007). Desain penelitian ini dipilih karena penelitian ini difokuskan pada sebuah kasus tertentu, dimana penelitian ini menggunakan dataset KDD Cup 99 yang didapat dari situs web yaitu kanggle.com. Aplikasi pendeteksi anomali jaringan akibat serangan DDoS *(Distributed Denial of Service)* menggunakan algoritma KNN *(K-Nearest Neighbor)* untuk klasifikasi serangan ini dibangun berdasarkan tingkat kedekatan satu data dengan data lainnya. Sehingga, isi dari klasifikasi yang terbentuk meliputi data-data yang memiliki kedekatan dan kemiripan yang serupa.

Data yang digunakan dalam penelitian ini adalah data sekunder. Data sekunder adalah data yang diperoleh melalui media perantara. Dalam proses pengenalan data ini, tahap yang dilakukan terlebih dahulu adalah memilih fitur-fitur yang paling baik digunakan. Dalam tahap pemilihan fitur ini, metode yang akan digunakan yaitu Information Gain. Setelah mendapatkan fitur-fitur tersebut selanjutkan akan dilakukan proses pelatihan terhadap data, kemudian data yang sudah dilatih digunakan sebagai pembanding terhadap data uji yang dimasukan dalam program.

### **Data Penelitian**

Tahap selanjutnya adalah tahap pengumpulan data yang akan digunakan dalam penelitian ini. Dataset yang digunakan dalam penelitian ini adalah dataset yang berasal dari situs web kanggle.com yaitu KDD Cup 99. Data KDD Cup 99 dipilih karena data ini sudah banyak digunakan oleh para peneliti sebelumnya. Data yang digunakan dalam penelitian ini berjumlah 9941 data dimana data tersebut terdiri dari 2 jenis data yaitu data normal dan serangan yaitu *nmap* dan *portsweep.*

### **Metode Penelitian**

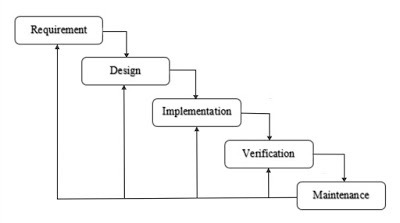
Metode yang digunakan dalam pengklasifikasian serangan DDoS ini adalah metode KNN *(K-Nearest Neighbor*). Sebelum masuk ke tahap pengujian, fitur-fitur yang ada pada data akan diseleksi terlebih dahulu. Tujuan dari seleksi fitur ini adalah untuk mencari fitur yang paling berpengaruh terhadap data sehingga akurasi yang dihasilkan dalam pengklasifikasian ini juga akan maksimal. Metode yang digunakan untuk seleksi fitur ini adalah Information Gain. Information Gain melakukan seleksi fitur dengan teknik scoring dengann menggunakan maksimal entropy pada masing-masing fitur. Jika fitur-fitur sudah diurutkan dari entropy tertinggi ke entropy terendah, selanjutkan akan dipilih 9 fitur teratas untuk pengklasifikasin dengan metode K-Nearest Neighbor.

Pada tahap klasifikasi dengan K-Nearest Neighbor, data dengan fitur-fitur yang telah dileksi akan dilatih terlebih dahulu. Kemudian data yang sudah dilatih tersebut akan digunakan untuk memprediksi data uji yang diinputkan. Dan selanjutnya ditampilkan hasil prediksi yang dilakukan dengan metode K-Nearest Neighbor tersebut.

Setelah melakukan prediksi terhadap data uji, selanjutnya dilakukan pengukuran performa terhadap program. Pengujian performa ini berupa pengujian akurasi dari program. Untuk pengujian performa sendiri digunakan metode K-Fold Cross Validation. Pada pengujian ini data akan dibagi menjadi 5 bagian. Setelah itu akan muncul seberapa besar akurasi yang dihasilkan oleh program.

### **Metodologi Pengembangan Perangkat Lunak**

Metode air terjun atau yang sering disebut metode waterfall sering dinamakan siklus hidup klasik (classic life cycle), dimana hal ini menggambarkan pendekatan yang sistematis dan juga berurutan pada pengembangan perangkat lunak, dimulai dengan spesifikasi kebutuhan pengguna lalu berlanjut melalui tahapan-tahapan perencanaan (planning), permodelan (modeling), konstruksi (construction), serta penyerahan sistem ke para pelanggan/pengguna (deployment), yang diakhiri dengan dukungan pada perangkat lunak lengkap yang dihasilkan (Pressman, 2012). Tahapan metode waterfall dapat dilihat pada gambar di bawah ini.



Gambar 1 Tahapan Metode Waterfall

Sumber : PengetahuanDanTeknologi.com

Dalam pengembangannya metode waterfall memiliki beberapa tahapan yang berurut yaitu:requirement (analisis kebutuhan), design system (desain sistem), Coding (pengkodean) & Testing(pengujian), Penerapan Program, pemeliharaan. Tahapan tahapan dari metode waterfall adalah sebagai berikut :

1. Requirement Analisis

Tahap ini pengembang sistem diperlukan komunikasi yang bertujuan untuk memahami perangkat lunak yang diharapkan oleh pengguna dan batasan perangkat lunak tersebut. Informasi ini biasanya dapat diperoleh melalui wawancara, diskusi atau survei langsung. Informasi dianalisis untuk mendapatkan data yang dibutuhkan oleh pengguna.

1. System Design

Spesifikasi kebutuhan dari tahap sebelumnya akan dipelajari dalam fase ini dan desain sistem disiapkan. Desain Sistem membantu dalam menentukan perangkat keras(hardware) dan sistem persyaratan dan juga membantu dalam mendefinisikan arsitektur sistem secara keseluruhan.

1. Implementation

Pada tahap ini, sistem pertama kali dikembangkan di program kecil yang disebut unit, yang terintegrasi dalam tahap selanjutnya. Setiap unit dikembangkan dan diuji untuk fungsionalitas yang disebut sebagai unit testing.

1. Integration & Testing

Seluruh unit yang dikembangkan dalam tahap implementasi diintegrasikan ke dalam sistem setelah pengujian yang dilakukan masing-masing unit. Setelah integrasi seluruh sistem diuji untuk mengecek setiap kegagalan maupun kesalahan.

1. Operation & Maintenance

Tahap akhir dalam model waterfall. Perangkat lunak yang sudah jadi, dijalankan serta dilakukan pemeliharaan. Pemeliharaan termasuk dalam memperbaiki kesalahan yang tidak ditemukan pada langkah sebelumnya. Perbaikan implementasi unit sistem dan peningkatan jasa sistem sebagai kebutuhan baru.

### **Evaluasi dan Validasi Hasil**

Pada tahap ini program diuji secara keseluruhan sebagai sistem yang lengkap untuk memastikan bahwa persyaratan perangkat lunak telah terpenuhi dan sesuai dengan kebutuhan serta tujuan awal dari penelitian ini. Skenario dari pengujian ini yaitu yang pertama akan dilakukan pemilahan atau penseleksian fitur dengan menggunakan metode *Information Gain.* Penseleksian fitur ini digunakan untuk mengurutkan fitur yang paling berpengaruh dalam dataset yang digunakan. Setelah mengurutkan fitur dari yang terbaik ke yang terendah maka selanjutnya akan dipilih 9 fitur yang akan digunakan untuk pengklasifikasian anomali jaringan dengan menggunakan metode KNN *(K-Nearest Neighbor).* Program akan mengklasifikasikan data yang dibagi menjadi data uji terhadap data latih yang digunakan sebagai patokan dalam pengujian data uji tersebut. Kemudian akurasi dari keberhasilan pendeteksian akan dihitung untuk mengetahui seberapa besar akurasi yang dapat dihasilkan oleh program. Pada penelitian ini akurasi diperoleh dengan membandingkan jumlah data yang berhasil dengan benar diklasifikasikan kedalam 2 buah kelas normal dan kelas serangan DDoS. untuk mendapatkan nilai akurasi dihitung dari jumlah data yang tepat dibagi dengan jumlah data. Perhitungan nilai akurasi sebagai berikut (Vienna, 2009) :

# **BAB II**

# **TINJAUAN PUSTAKAN**

## **Anomali Jaringan**

Anomali jaringan adalah suatu keadaan yang terjadi pada sebuah network traffic yang menyebabkan kondisi menjadi tidak normal. Anomali yang terjadi bisa dilihat melalui kenaikan lonjakan pengguna Internet, melalui serangan pada suatu traffic dan lonjakan yang tidak disengaja. Kenaikan lonjakan dapat dilihat pada saat adanya bencana yang terjadi di dunia, kompetisi atau pertandingan dan kejadian yang tidak biasa terjadi setiap hari. Secara tidak sadar, kondisi kenaikan lonjakan ini memberikan dampak negatif bagi beberapa pihak. Kenaikan lonjakan yang terjadi tersebut menimbulkan penurunan performansi dari suatu jaringan. Untuk itu, perlu dilakukan deteksi terhadap anomali yang terjadi(1). Dalam penelitian (Imam, Sukarno, & Nugroho, 2019), disebutkan bahwa anomali jaringan adalah keadaan dimana dalam sebuah *network traffic* (lalu lintas jaringan) tidak normal. Anomali jaringan ini terkadang bisa dirasakan atau dilihat melalui kelonjakan pengguna Internet. Secata tidak sadar, kondisi lonjakan ini dapat berdampak negattif bagi beberapa pihak. Kenaikan lonjakan ini dapat mempengaruhi *bandwidth* sehingga menimbulkan penurunan peformansi dari suatu jaringan.

Adapun kelas anomali yang dapat ditemukan dalam jaringan :

1. DoS

*Denial of Service* (DoS) adalah kategori serangan, yang victim’s resources sehingga membuatnya tidak dapat menangani legitimate requests, contohnya syn flooding. Relevant features: “source bytes” dan “percentage of packets with errors”. Jenis anomali pada DoS: Back, Land, Neptune, Pod, Smurf, Teardrop, Apache2, Udpstrom, Processtable, Worm.

1. Probe

Surveillance dan serangan probing lainnya bertujuan untuk mendapatkan informasi tentang korban jarak jauh. Contohnya, port scanning. Relevant features: “duration of connection” dan “source bytes”. Jenis anomali pada PROBE: Satan, Ipsweep, Nmap, Portsweep, Mscan, Saint.

1. R2L

Unauthorized access, dari remote machine, attacker masuk ke remote machine dan mendapatkan local access dari mesin korban. Contohnya, password guessing. Relevant features: Network level features – “duration of connection” dan “service requested” dan host level features - “number of failed login attemp- ts”. Jenis anomali pada R2L: Guess passwd, Ftp write, Imap, Phf, Multihop, Warezmaster, Warezclient, Spy, Xlock, Xsnoop, Snmpguess, Snmpgetattack, Httptunnel, Sendemail, named

1. U2R

Unauthorized access untuk local super user (root) privileges adalah jenis serangan, dimana attacker meng- gunakan akun normal untuk masuk ke sistem korban dan mencoba untuk mendapatkan hak istimewa root / administrator dengan mengeksploitasi beberapa vulnerability pada korban, mis. serangan buffer overflow. Relevant features: “number of file creations” dan “number of shell prompts invoked”. Jenis anomali pada U2R: Buffer Overflow, Loadmodule, Rootkit, Perl, Sqlattack, Xterm, Ps.

## **Serangan DoS**

Denial of Service (DoS) adalah salah satu serangan paling umum di jaringan komputer, serangan ini mencegah pengguna dari akses layanan normal karena penyerang mengkonsumsi sumber daya jaringan komputer, memori, prosesor, dan lain-lain yang lebih baik (Cahyaningtyas, Sukarno, & Nugroho, 2019). Serangan DoS dilakukan dengan membanjiri target serangan dengan lalu lintas, atau mengirim informasi untuk memicu crash (Gunawan, Sukarno, & Putrada, 2018). Kelompok serangan DoS memiliki dua kelas yang berbeda : serangan logika dan serangan sumber daya(Gunawan et al., 2018).

Serangan logika menyerang dengan cara mengeksploitasi kelemahan dari software yang ada, hal ini dilakukan agar perangkat yang jauh mengalami crash atau secara substansial akan menurunkan kinerja dari perangkat tersebut. Sedangkan serangan sumber daya dilakukan dengan membanjiri sumber daya dari komputer korban (misalkan pada CPU dan memori). Serangan sumber daya juga dapat menyerang jaringan korban dengan mengirimkan aliran paket palsu yang dengan jumlah yang sangat banyak atau tidak berhenti dalam waktu beberapa lama.

Serangan DoS dapat diatasi dengan cara memblokir IP perangkat yang membanjiri perangkat korban. Tetapi karena mudah untuk memblokir serang DoS, kini muncullah serangan baru dengan metode yang sama dengan DoS tetapi menggunakan banyak perangkat. Serangan ini disebut dengan istilah Distributed Denial of Service (DDoS)

## **Serangan Distributed Denial of Services**

Serangan *Distributed Denial of Service* (DDoS) merupakan serangan yang mudah dilakukan namun sulit untuk ditanggulangi (Pratama, 2019). Sebelum melakukan serangan DDoS, penyerang akan menyiapkan komputer untuk membantu dalam penyerangan tersebut. Komputer-komputer yang membantu dalam penyerangan tersebut disebut dengan komputer zombie atau botnet, dimana komputer tersebut dikendalikan oleh sebuah server atau komputer utama untuk membantu menyerang korban dan mengakibatkan server menjadi *down* dan mengakibatkan *system error* (Adrian & Isnianto, 2016).

### **Konsep Serangan DDoS**

Konsep dari serangan DDoS adalah membanjiri lalu lintas jaringan dengan data dengan jumlah yang sangat banyak. Konsep serangan DDoS sama halnya seperti serangan DoS akan tetapi serangan DDoS dibantu dengan komputer zombie atau botnet. Adapun konsep serangan DoS adalah sebagai beriktu :

1. *Request Flooding,* konsep ini merupakan teknik yang digunakan dengan membanjiri jaringan menggunakan banyak *request.* Hal ini mengakibatkan pengguna lain yang terdaftar tidak dapat dilayani oleh server tersebut.
2. *Traffic Flooding,* konsep ini merupakan teknik yang digunakan dengan membanjiri lalu lintas jaringan dengan data dengan jumlah yang banyak. Akibatnya, pengguna lain tidah bisa dilayani oleh server.
3. Mengubah sistem konfigurasi atau bahkan merusak komponen dan server, tetapi tidak banyak yang menggunakan cara ini karena sulit untuk dilakukan.

### **Metode Serangan DDoS**

Serangan DDoS adalah teknik penyerangan yang memiliki banyak cara untuk melakukan serangan, seperti menggunakan virus dan botnet.

1. Virus

Penyerang sengaja menciptakan virus untuk menjalankan bot melalui script yang berjalan pada sistem operasi. Penyerang akan menyebarkan virus melalui file yang dibagikan ke berbagai situs yang terhubung dengan internet. Bahkan beberapa virus juga dapat mengambil hak akses dari perangkat yang sudah men*download script* dan dijalankan pada sistem operasi. Jika komputer sudah terinveksi virus DDoS maka virus tersebut akan melakukan serangan DDoS secara aktif ke server atau ke alamat IP tertentu yang sudah ditentukan.

1. Botnet

Dalam melakukan serangan DDoS, serangan DDoS dibantu dengan kumpulan bot yang dijalankan secara bersama-sama. Bot disisipkan pada malware yang kemudian di tanam ke komputer yang terhubung ke jaringan internet. Jumlah komputer yang dijadikan bot bisa mencapai puluhan sampai dengan jutaan, hal ini tergantung dari banyaknya komputer yang telah terinfeksi malware. Semua komputer yang terinfeksi malware tersebut disebut dengan botnet, sedangkan satu komputer yang terinfeksi malware tersebut disebut dengan komputer zombie.

## **Data Mining**

Data mining adalah langkah analisis terhadap proses penemuan pengetahuan di dalam basis data atau *Knowledge Discovery in Database* yang disingkat KDD. *Data mining* didefinisikan sebagai penemuan dari pola-pola baru yang berasal dari kumpulan data yang sangat besar, meliputi metode-metode dari *artificial intelligence, machine learning, statistic,* dan *database system.* Data mining ditujukan untuk mengekstrak (mengambil intisari) pengetahuan dari sekumpulan data sehingga didapatkan struktur yang dapat dimengerti menusia serta meliputi bisis data dan menagemen data, prapemrosesan data, pertimbangan model dan inferensi, ukuran keterkatikan, pertimbangan kompleksitas, pascapemrosesan terhadap struktur yang ditemukan, visualisasi, dan *online updating* (Suyanto, 2017)*.* Data mining atau penambangan data adalah teknik yang relative cepat dan mudah untuk menemukan pengetahuan, pola dan/atau relasi antar data, secara otomatis.

Secara umum, kegunaan dari data mining dapat dibagi menjadi dua yaitu deskriptif dan prediktif. Deskriptif berarti data mining digunakan untuk mencari pola-pola yang dapt fipahami manusia mrnjrlaskan karakteristik data. Sedangkan prediktif berarti data mining digunakan untuk membentuk sebuah model pengetahuan yang akan digunakan untuk melakukan prediksi. Berdasarkan fungsionalitasnya, tugasnya data mining dikelompokkan menjadi beberapa bagian (Mardi, 2017).

1. Description (Deskripsi)

Pada bagian ini, biasanya akan dicoba untuk pencarian cara untuk menggambarkan pola serta kecenderungan yang terdapat dalam data. Sebagai contoh, petugas pengumpulan suara mungkin tidak dapat menemukan keterangan atau fakta bahwa siapa yang tidak cukup profesional akan sedikit didukung dalam pemilihan presiden.

1. Estimation (Estimasi)

Model ini dibangun dengan menggunakan record lengkap yang menyediakan nilai dari variabel target sebagai nilai prediksi. Selanjutnya, pada peninjauan berikutnya estimasi nilai dari variabel target dibuat berdasarkan nilai variabel prediksi. Contohnya melakukan estimasi IPK dengan melihat nilai indeks prestasi dari mahasiswa.

1. Prediction (Prediksi)

Prediksi serupa dengan klasifikasi dan estimasi, hanya saja nilai dari hasil akan ada dimasa mendatang. Contohnya memprediksi tingkat pengangguran pada lima tahun kedepan.

1. Classification (Klasifikasi)

Pada bagian klasifikasi biasanya terdiri dari target variabel kategori. Contohnya penggolongan kriteria penjelajahan internet dipisahkan dalam dua kategori yaitu berupa serangan, atau tidak.

1. Clustering (Pengklusteran)

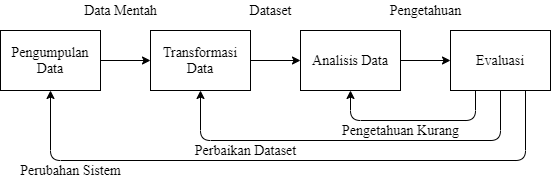
Pengklusteran berbeda dengan klasifikasi karena dalam clustering tidak terdapat variabel target. Pengklsteran merupakan pengelompokan record, pengamatan, atau memperhatikan dan membentuk kelas objek-objek yang memil iki kemiripan. Sebagai contoh, melakukan pengklusteran terhadap ekspresi dari gen, untuk mendapatkan kemiripan perilaku dari gen dalam jumlah besar.

1. Association (Asosiasi)

Tugas asosiasi dalam data mining adalah menemukan atribut yang muncul dalam satu waktu. Sebagai contoh asosiasi adalah Meneliti jumlah pelanggan dari perusahaan telekomunikasi seluler yang diharapkan untuk memberikan respons positif terhadap penawaran upgrade layanan yang diberikan.

## **Proses Data Mining**

Proses data mining tersusun atas tiga tahapan, yaitu pengumpulan data *(data collection),* transformasi data *(data transformation),* dan analisis data *(data analysis)* (17). Proses tersebut diawali dengan *preprocessing* yang terdiri atas pengumpulan data untuk menghasilkan data mentah (raw data) yang akan digunakan dan dibutuhkan oleh data mining. Kemudian setelah data metah didapatkan tahap selanjutnya adalah transformasi data. Transformasi data ini bertujuan untuk mengubah data mining, misalkan melalui filtrasi atau agregasi. Kemudian hasil dari transformasi data tersebut akan digunakan oleh analisis data untuk membangkitkan pengetahuan dengan menggunakan teknik seperti analisis statistic, *machine learning,* dan visualisasi informasi.



Gambar 2. 1 Kerangka Proses Data Mining

Pada Gambar 2.1 ditunjukkan diagram yang menggambarkan aliran informasi damlam proses data mining yang diadaptasi dari (17). Proses data mining pada gambar tersebut ditunjukkan sebagai proses yang iterative. Hasil evaluasi pengetahuan yang dihasilkan data mining dapat menimbulkan kebutuhan pengetahuan yang lebih lengkap, perbaikan kumpulan data (dataset) atau perubahan pada sistem.

## **Seleksi Atribut**

Seleksi atribut atau yang sering juga dikenal dengan istilah seleksi fitur adalah salah satu bagian dari *preprocessing* data. Seleksi atribut ini adalah teknik yang digunakan untuk menghilangkan atribut yang kurang berpengaruh terhadap data sehingga atribut tersebut boleh saja tidak digunakan untuk tahap selanjutnya. Jika pada suatu kasus memiliki ratusan atau bahkan ribuan atribut namun tidak semua atribut tersebut penting untuk diperhatikan, artinya atribut atau dimensi data bisa direduksi.

## **Klasifikasi**

Dalam buku Dr. Suyanto, S.T, M.Sc, klasifikasi yaitu bagaimana mempelajari sekumpulan data sehingga dihasilkan aturan yang bisa mengklasifikasikan atau mengenali data-data baru yang belum pernah dipelajari. Klasifikasi dapat didefinisikan sebagai proses untuk menyatakan suatu objek data sebagai salah satu kategori (kelas) yang telah didefinisikan sebelumnya (Zaki et al. 2013). Klasifikasi banyak digyunakan dalam bebagai aplikasi, diantaranya adalah deteksi kecurangan *(fraud detection),*  pengelolaann pelanggan, diagnosis medis, prediksi penjualan, dan sebagainya.

Model kladifikasi lebih sering dibangun menggunakan teknik pembelajaran dalam bidang *machine learning.* Proses pembelanjaran secara otomatis terhadap suatu himpunan data mampu menghasilkan model klasifikasi (fungsi target) yang memetakan objek data x (input) ke dalah satu kelas y yang telah didefinisikan sebelumnya. Jadi, proses pembelajaran memerlukan masukan *(input)* berupa himpunan data latih *(training set)* yang berlabel (memiliki atribut kelas) dan mengeluarkan output yang berupa sebuah model klasifikasi.

Ada banyak teknik klasifikasi yang diusulkan oleh para ahli, yang dapat dikelompokkan kedalam dua kategori, yaitu teknik klasifikasi global atau memperhitungkan semua data latih dan teknik klasifikasi lokal atau teknik ini hanya memperhitungkan sebagian dari data latih. Masing-masing teknik tersebut mepunyai kelebihan dan kekurangan, serta kekuatan dan kelemahan.

## ***Information Gain***

Information gain merupakan metode seleksi fitur yang paling sederhana, yaitu dengan cara melakukan perangkaian atribut. Information gain mendeteksi fitur-fitur yang paling banyak memiliki informasi berdasarkan kelas tertentu, penentuan atribur ini dilakukan dengan cara menghitung nilai entropy terlebih dahulu (Aini, Sari, & Arwan, 2018). Entropy adalah suatu parameter untuk mengukur heterogenitas (keberagaman) dalam suatu himpunan data, semakin heterogen suatu himpunan data, semakin besar pula nilai entropy-nya (Suyanto, 2017). Secara matematis, *entropy* dirumuskan pada persamaan (1)

(2)

Dimana c adalah jumlah nilai yang terdapat pada atribut target (juumlah kelas). Sedangkan menyatakan posisi atau ratio antara jumlah sampel di kelas dengan jumlah semua sampel pada himpunan tersebut.

Secara sistematis, *information gain* dirumuskan pada persamaan (2) (Suyanto, 2017).

(2)

Dimana :

: atribut

: menyatakan suatu nilai yang mungkin untuk atribut A

: himpunan nilai-nilai yang mungkin untuk atribut A

: jumlah sample untuk nilai v

: jumlah seluruh sampel data

: entropy untuk sampel-sampel yang memiliki nilai v

## ***K-Nearest Neighbor***

Algoritma dibagi menjadi dua jenis, yaitu algoritma supervised dan algoritma unsupervised. Algoritma supervised learning memiliki tujuan untuk menemuka pola baru dalam sebuah data dengan mrnghubungkan pola data yang sudah ada dengan data baru, sedangkan pada algoritma unsupervised learning, data belum memiliki pola apapun dan tujuan dari algoritma unsupervised learning adalah uuntuk menemukan pola dalam sebuah data (Liantoni, 2015). Algoritma K-Nearest Neighbor termasuk algoritma supervised karena algoritma ini bertujuan untuk mengklasifikasika objek baru berdasarkan atribut dan data training.

Dalam buku yang dikarang oleh Dr. Suyanto, S.T., M.Sc., algoritma K-Nearest Neighbor bekerja dengan mencari sejumlah kelompok k objek data atau po;a (dari semuaa pola latih yang ada) yang paling dekat dengan pola masukan, kemudian memilih kelas dengan jumlah pola terbanyak di antara k pola tersebut. Untuk menentukan atau mendefisinikasn jarak antar objek maka digunakan rumus *Euclidean* pada persamaan (3) (Mustakim & Oktaviani, 2016)

(3)

Dimana :

: jarak antara titik pada data training dan data testing yang akan diklasifikasikan

: nilai dari data training

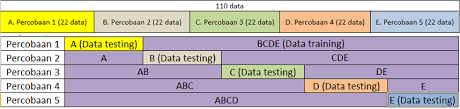
: nilai dari data testing

: nilai atribut

: dimensi data atribut

## ***K-fold Cross Validation***

*K-Fold Cross Validation* merupakan salah satu metode yang digunakan untuk mengetahui rata-rata keberhasilan dari suatu sistem dengan cara melakukan perulangan dengan mengacak atribut masukan sehingga sistem tersebut teruji untuk beberapa atribut input yang acak (Banjarsari, Budiman, & Farmadi, 2016). Metode *k-Fold Cross Validation* mempartisi himpunan data *D* secara acak menjadi *k fold* (subhimpunan) yang saling bebas, sehingga masing-masing *fold* berisi 1/*k* bagian data. Untuk himpunan data yang akan dibangun, masing-masing dari himpunan data tersebut akan memiliki (k – 1) *fold* data latih atau data *training* dan 1 *fold* data untuk data uji atau data *testing* (Suyanto, 2017)*.* Contoh pembagian dataset kedalam proses 5-*Fold Cross Validation* dapat dilihat pada Gambar .



Gambar 6. 1 5-Fold Cross Validation

Sumber : Pencarian k-Optimal pada Algoritma KNN untuk Prediksi Kelulusan Tepat Waku Mahasiswa Berdasarkan IP Sampai Dengan Semester 4 (Studi Kasus : Mahasiswa Program Studi Ilmu Komputer FMIPA Unlam). 2015.

Cara kerja *k-Fold Cross Validation* adalah sebagai berikut :

1. Total instance dibagi menjadi n bagian
2. *Fold* ke-1 adalah ketika bagian ke-1 menjadi data uji (testing data) dan sisanya menjadi data latih (training data). Selanjutnya, hitung akurasi berdasarkan porsi data tersebut. Perhitungan akurasi tersebut dengan menggunakan persamaan (4) :

(4)

1. Fold ke-2 adalah ketika bagian ke-2 menjadi data uji (testing data) dan sisanya menjadi data latih (training data). Selanjutnya, hitung akurasi berdasarkan porsi data tersebut.
2. Demikian seterusnya hingga mencapai fold ke-K. Hitung rata-rata akurasi dari K buah akurasi. Dan rata-rata akurasi ini menjadi akurasi final.

# **BAB III**

# **ANALISIS DAN PERANCANGAN**

## **Variable Penelitian**

Penelitian yang dilakukan oleh penulis dengan judul “Klasifikasi Serangan *Distributed Denial Of Service* (DDoS) dengan Algoritma *K-Nearest Neighbor* (KNN)”, dimana variable yanag digunakan sebagai tolak ukur keberhasilan dari implementasi sistem klasifikasi data ini adalah pengelompokan atau klasifikasi data berdasarkan tingkat kemiripan tertinggi dari data uji terhadap data latih. Yang perlu diperhatikan dalam sistem ini adalah apakah klasifikasi yang dihasilkan oleh sistem mencangkup data dari paket data dengan tingkat kemiripan tertinggi melalui perhitungan jarak *Eucledian*. Data yang digunakan pada penelitian ini yaitu dataset yang berasal dari KDD Cup 99.

Data intrusi yang digunakan untuk penelitian diambil dari dataset KDD Cup 99 yang mana dataset ini juga sudah banyak digunakan sebagai patokan oleh banyak peneliti. Tabel 1 menunjukan jumlah paket data yang digunakan dalam penelitian untuk mengevaluasi rules dan pengujian data guna mendeteksi intrusi. Jumlah fitur dari dataset KDD Cup 99 yang digunakan dalam penelitian ini adalah sebanyak 41 fitur (dapat dilihat pada Tabel 2, Tabel 3, dan Tabel 4), koneksi diberi label *attack* atau normal. Data dikategorikan menjadi dua kelas ketegori utama yaitu DoS *(Deniala of Service)* dan normal (Tabel 1). Dalam dataset ini ada 3 group fitur, yaitu fitur dasar *(basic)*, *content based* dan *time based features*.

Tabel 3. 1 Jumlah Paket Data

|  |  |  |
| --- | --- | --- |
| **No** | **Nama** | **Jumlah** |
| 1 | Normal | 67343 |
| 2 | Back | 956 |
| 3 | Land | 18 |
| 4 | Neptune | 41214 |
| 5 | POD *(Ping of Death)* | 201 |
| 6 | Smurf | 2646 |

Pada penelitian ini, jumlah dataset yang digunakan adalah sebanyak 112378 data dimana data ini adalah data yang terdiri dari data normal dan data serangan. Serangan yang dideteksi oleh sistem hanya serangan DoS *(Denial of Service).* Nama dari serangan DoS *(Denial of Service)* yang diteksi yaitu *Back, Land, Neptune, Ping of Death (POD), dan Smurf.*

Tabel 3. 2 Fitur Dasar (Basic) Tiap-tiap Koneksi TCP

|  |  |  |
| --- | --- | --- |
| **Nama Fitur** | **Keterangan** | **Tipe** |
| duration | lamanya (detik) koneksi | continuous |
| protocol\_type | Tipe protokol (ICMP, TCP, UDP) | discrete |
| service network | service pada tujuan (http, telnet, dll) | discrete |
| src\_bytes | Jumlah rata-rata byte, termasuk informasi header yang diterima oleh destination host | continuous |
| dst\_bytes | Jumlah rata-rata byte termasuk informasi header yang diterima oleh source host | continuous |
| flag status, | normal atau error dari koneksi | discrete |
| Land | 1 jika koneksi bersal dari/ke host yang sama/port; 0 jika tidak | Discrete |
| wrong\_fragment | jumlah ``wrong'' fragments | continuous |
| urgent | jumlah dari paket urgent | continuous |

Tabel 3. 3 Fitur Konten pada Koneksi (Conection Based)

|  |  |  |
| --- | --- | --- |
| **Nama fitur** | **Keterangan** | **Tipe** |
| hot | jumlah dari dari indikator ``hot'' | Continuous |
| num\_failed\_logins | jumlah dari percobaan login yang gagal | continuous |
| logged\_in | 1 jika berhasil login; 0 jika tidak | Discrete |
| num\_compromised | jumlah dari kondisi ``compromised'' | Continuous |
| root\_shell | 1 jika root shell dijalankan; 0 jika tidak | Discrete |
| su\_attempted | 1 jika ada percobaan perintah ``su root'' ; 0 jika tidak | Discrete |
| num\_root | jumlah akses ``root'' | Continuous |
| num\_file\_creations | jumlah dari file yang dimunculkan oleh operasi | Continuous |
| num\_shells | jumlah dari shell prompts | Continuous |
| num\_access\_files | jumlah dari operasi pada file kontrol akses | Continuous |
| num\_outbound\_cmds | jumlah dari perintah outbound dalam sesi ftp | Continuous |
| is\_hot\_login | 1 jika login termasuk daftar``hot'' ; 0 jika tidak | Discrete |
| is\_guest\_login | 1 jika login adalah login ``guest''; 0 jika tidak | Discrete |

Tabel 3. 4 Fitur Trafik Waktu (Time Based Features)

|  |  |  |
| --- | --- | --- |
| **Nama fitur** | **Keterangan** | **Tipe** |
| count | Fitur yang mengindikasikan jumlah koneksi pada host yang sama | Continuous |
| serror\_rate | % dari koneksi ``SYN'' errors | continuous |
| rerror\_rate | % dari koneksi yang mempunyai ``REJ'' errors | Continuous |
| same\_srv\_rate | % dari konkesi ke service yang sama | Continuous |
| diff\_srv\_rate | % dari koneksi ke services yang berbed | Continuous |
| srv\_count | Jumlah koneksi pada service sama untuk koneksi yang sama dalam 2 detik terakhir | Continuous |
| srv\_serror\_rate | % dari koneksi yang mempunyai ``SYN'' errors | Continuous |
| srv\_rerror\_rate | % dari koneksi yang mempunyai ``REJ'' errors | Continuous |
| srv\_diff\_host\_rate | % dari koneksi ke hosts yang berbeda | Continuous |
| num\_outbound\_cmds | jumlah dari perintah outbound dalam sesi ftp | Continuous |
| is\_hot\_login | 1 jika login termasuk daftar``hot'' ; 0 jika tidak | Discrete |
| is\_guest\_login | 1 jika login adalah login ``guest''; 0 jika tidak | Discrete |

Tabel 3. 5 Fitur Dataset

|  |  |
| --- | --- |
| **No** | **Nama** |
| 1 | Duration |
| 2 | protocol\_type |
| 3 | Service |
| 4 | Flag |
| 5 | src\_bytes |
| 6 | dst\_bytes |
| 7 | Land |
| 8 | wrong\_fragment |
| 9 | Urgent |
| 10 | Hot |
| 11 | num\_failed\_logins |
| 12 | logged\_in |
| 13 | num\_compromised |
| 14 | root\_shell |
| 15 | su\_attempted |
| 16 | num\_root |
| 17 | num\_file\_creations |
| 18 | num\_shells |
| 19 | num\_access\_files |
| 20 | num\_outbound\_cmds |
| 21 | is\_host\_login |
| 22 | is\_guest\_login |
| 23 | Count |
| 24 | srv\_count |
| 25 | serror\_rate |
| 26 | srv\_serror\_rate |
| 27 | rerror\_rate |
| 28 | srv\_rerror\_rate |
| 29 | same\_srv\_rate |
| 30 | diff\_srv\_rate |
| 31 | srv\_diff\_host\_rate |
| 32 | dst\_host\_count |
| 33 | dst\_host\_srv\_count |
| 34 | dst\_host\_same\_srv\_rate |
| 35 | dst\_host\_diff\_srv\_rate |
| 36 | dst\_host\_same\_src\_port\_rate |
| 37 | dst\_host\_srv\_diff\_host\_rate |
| 38 | dst\_host\_serror\_rate |
| 39 | dst\_host\_srv\_serror\_rate |
| 40 | dst\_host\_rerror\_rate |
| 41 | dst\_host\_srv\_rerror\_rate |

## **Analisis Kebutuhan**

Pada tahap ini, dilakukan analisis kebutuhan terhadap penelitian yang dilakukan oleh peneliti. Analisis kebutuhan ini meliputi kebutuhan yang diperlukan oleh sistem yang di bangun, kebutuhan sistem tersebut meliputi data yang digunakan pengoperasian sistem dan referensi sebagai sumber pembelajaran yang digunakan sebagai patokan dalam pembuatan sistem. Teknik pengumpulan data yang digunakan dalam penelitian ini menggunakan metode studi literatur. Studi literatur digunakan untuk mengetahui kebutuhan dari penelitian pada sistem klasifikasi yang dibangun.

1. Jenis data yang digunakan pada penelitian ini adalah data primer. Data primer adalah data yang dikumpulkan oleh peneliti sendiri. Dalam penelitian ini data yang digunakan adalah data riset bersumber dari KDD Cup 99.
2. Teknik Studi Literatur, teknik studi literatur dilakukan dengan cara mempelajari referensi baik itu dari jurnal maupun buku yang berkaitan dengan anomali jaringan, metode *Information Gain,* dan metode *K-Nearest Neighbor* (KNN) yang dapat menunjang penelitian ini. Studi literatur dilakukan dengan mengambil literatur – literatur pendukung dari jurnal – jurnal ilmiah, baik itu dari jurnal dalam negeri maupun jurnal luar negeri dan juga dapat dilakukan dengan mempelajari buku – buku terkait. Dalam studi literatur ini, penulis mencari sumber yang berkaitan dengan permasalahan yang perlu menjadi perbaikan dalam penelitian selanjutnya.

## **Analisis Sistem**

Pada tahap ini, dilakukan analisis kebutuhan sistem berupa perangkat yang digunakan untuk menganalisis proses dari sistem tersebut. Perangkat yang digunakan untuk membantu menganalisis kebutuhan sistem adalah *flowchart* (diagram alir). *Flowchart* data digunakan untuk menyusun ibput, proses, dan juga output dari sistem klasifikasi serangan DDoS yang dikembangkan. Hasil dari tahapan ini adalah spesifikasi kebutuhan sistem dan analisa dari model proses sistem yaitu penentuan input – input dari sistem, penentuan proses – proses dari sistem, dan juga penentuan output dari pemrosesan input sistem.

### **Penentuan Input – input Sistem**

Input – input yang digunakan dalam sistem yaitu data yang dimasukan ke sistem kemudian data tersebut akan diproses oleh sistem dan selanjutnya menghasilkan output yang sudah ditentukan sebelumnya. Data yang dimasukkan kedalam sistem adalah data berupa kumpulan data riset yang akan dilakukan tahap klasifikasi pada data tersebut. Dataset yang digunakan dalam penelitian ini berperan sebagai kumpulan data – data yang belum diolah dan akn dijadikan sebagai objek penelitian. Dataset yang digunakan adalah dataset yang berasal dari KDD Cup 99 dengan format file .csv.

Fitur dari input data pada sistem ini akan dipilih sesuai dengan nilai *Entropy* terbesarnya sebelum masuk ke tahap pembagian kumpulan data. Metode yang digunakan dalam pemilihan atau seleksi fitur ini adalah *Information Gain.* Pembagian kumpulan data ini dibagi menjadi dua, yaitu data pelatihan dan data pengujian. Dataset yang dibagi menjadi data pelatihan ini digunakan oleh metode *K-Nearest Neighbor* (KNN) untuk membuat kelas – kelas. Sedangkan data pengujian digunakan untuk mengevaluasi keakuratan model dari sistem ini. Kemudian sistem membagi kumpulan data secara acak kedalam kereta dan dataset dengan menggunakan metode *K-Fold Cross Validation.*

Dalam dataset KDD Cup training dari tiga protokol yang memiliki 32 tipe serangan ini dibagi menjadi dua kelas, yaitu normal dan serangan. Setiap penyimpangan dari prilaku normal dikatakan sebagai serangan.

Tabel 3. 6 Nama Serangan setiap Tipe Protokol

|  |  |
| --- | --- |
| **Protocol tipe** | **Nama Serangan** |
| UDP | normal, teardrop, satan, nmap, rootkit |
| TCP | normal, neptune, guess\_passwd, land, portsweep, buffer\_overflow, phf, warezmaster, ipsweep, multihop, warezclient, perl, back, ftp\_write, loadmodule, satan, spy, imap, rootkit |
| ICMP | normal, portsweep, ipsweep, smurf, satan, pod, nmap |

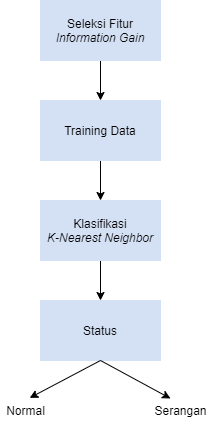
### **Penentuan Proses-proses dalam Operasi Sistem**

Secara garis besar, proses – proses yang dilakukan pada sistem klasifikasi serangan *Distributed Denial of Service* (DDoS) ini adalah sebagai berikut :

1. Proses input dataset ke dalam sistem.
2. Proses ekstrasi fitur dari dataset menggunakan metode *Information Gain* untuk menghilangkan fitur – fitur yang tidak dibutuhkan.
3. Proses normalisasi dataset dan pengujian terhadap data uji pada rentang 0 – 1.
4. Proses menghitung kesamaan data uji dengan dataset untuk menentukan kelas data uji dengan menggunakan perhitungan jara *Eucledian.*
5. Proses klasifikasi dengan metode *K-Nearest Neighbor* (KNN).
6. Proses penghitungan akurasi sistem dengan metode *K-Fold Cross Validation.*

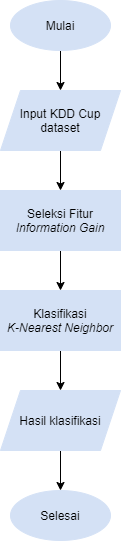
## **Perancangan Sistem**

Pada Gambar 3.1 dijelaskan desain sistem untuk pendeteksian anomali jaringan akibat serangan *Distributed Denial of Service* (DDoS) menggunakan metode *K-Nearest Neighbor (KNN).* Dalam sistem ini terdapat dua proses utama. Pertama, sistem melakukan preprosesing terhadap dataset. Preprosesing yang digunakan pada tahap ini adalah seleksi fitur, tujuan dari seleksi fitur ini adalah untuk mendapatkan fitur yang paling relevan terhadap data. Metode yang digunakan dalam tahap seleksi fitur ini adalah metode *Information Gain*. Setelah mendapatkan fitur – fitur yang digunakan dalam sistem, selanjutnya dilakukan pengelompokkan dataset dimana data ini pada tahap ini data dikelompokkan ke dalam 2 tipe yaitu normal dan serangan. Tipe serangan disini adalah serangan DDoS yaitu *back, land, neptune, ping of death* (POD)*,* dan *smurf.* Klasifikasi data tersebut dilakukan dengan menggunakan metode *K-Nearest Neighbor* (KNN). Sistem ini menggunakan dataset KDD Cup 99, dimana dataset ini juga sudah banyak digunakan dalam penelitian sistem deteksi anomali jaringan.



Gambar 3. 1 Perancangan Sistem

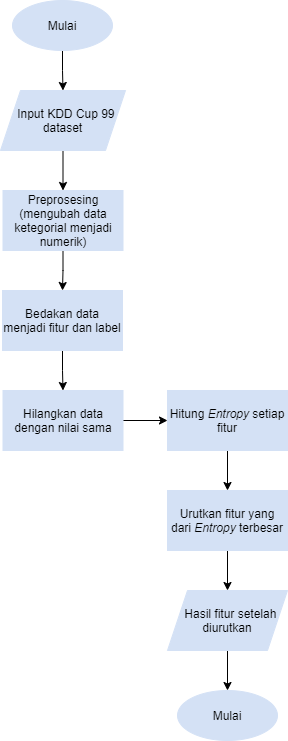
## **Flowchart Sistem**

****

Gambar 3. 2 Flowchart Sistem

Sistem menerima input dataset yaitu dataset KDD Cup 99. Kemudian fitur – fitur dari data tersebut diseleksi dengan menggunakan metode *Informatin Gain*. Selanjutnya dilakukan normalisasi terhadap atribut dataset yang memiliki tipe data kategorial. Setelah dilakukan normalisasi data, selanjutnya data tersebut dihitung dengan menggunakan metode *K-Nearest Neighbor* (KNN). Kemudian hasil dari perhitungan KNN tersebut akan dideteksi apakah paket dari data uji atau normal atau serangan.

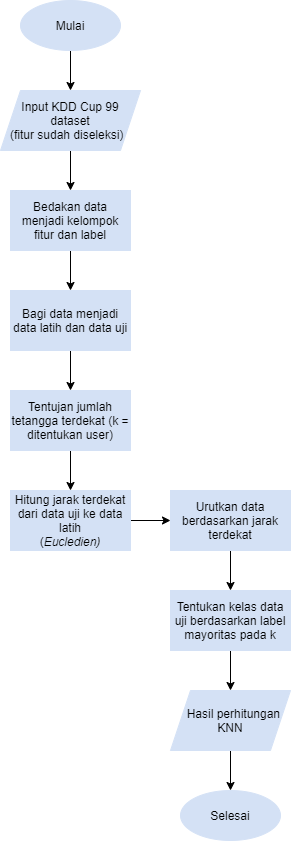
## **Flowchart Information Gain**

****

Gambar 3. 3 Flowchart Information Gain

Pada proses seleksi fitur dengan menggunakan metode *Information Gain,* tahap pertama yang dilakukan adalah memasukkan dataset KDD Cup 99. Kemudian data tersebut memasuki tahap *preprocessing* untuk mengubah data yang memiliki nilai kategorial menjadi nilai numeric. Selanjutnya, data dibedakan sesuai kebutuhan yaitu menjadi fitur dan label. Pada data fitur ini, data yang memiliki nilai yang sama akan dihilangkan karena nilai ini akan berpengeruh sangat kecil terhadap fitur yang lainnya. Selanjutnya sistem menghitung nilai *Entropy* dari masing – masing fitur dan kemudian mengurutkan fitur yang memliki nilai *Entropy* paling besar sampai paling kecil. Dan tahap terakhir pada seleksi fitur ini adalah sistem menampulkan diagram nilai *Entropy* dari semua fitur kemudian dapat dipilih fitur mana saja yang digunakan untuk proses selanjutnya.

## **Flowchart K-Nearest Neighbor**



Gambar 3. 4 Flowchart K-Nearest Neighbor

Tahap pertama yang dilakukan pada proses ini adalah melakukan input data. Setelah dilakukan input data, selanjutnya dilakukan tahap normalisasi terhadap input data agar semua data seragam yaitu mengubah data kategorial menjadi numerik. Jika semua data sudah bertipe numerik maka tahap selanjutnya adalah membagi data kedalam dua kategori yaitu fitur dan label. Kategori fitur ini digunakan untuk menginialisasi data yang terdapat pada fitur – fitur pada data. Sedangkan label digunakan untuk membedakan kelas pada sistem yaitu kelas normal yang diinisialisasi dengan 0 dan kelas serangan yang diinisialisasi dengan 1, perlu diketahui serangan yang digunakan pada sistem ini adalah *back, land, neptune, ping of death* (POD), dan *smurf.* Selanjutnya data tersebut dibagi menjadi dua kelompok, yaitu data latih dan data uji. Kemudian tentukan jumlah tetangga terdekat, disini jumlah tetangga terdekat diinisialisasi dengan k dan jumlah k ditentukan oleh user. Selanjutnya menghitung jarak data uji terhadap jarak data latih terdekat. Kemudian urutkan data berdasarkan jarak terdekat dan tentukan kelas data uji berdasarkan mayoritas pada jumlah k. tahap terakhir pada metode ini, ditampilkan hasiil dari klasifikasi data uji terhadap data latih.

## **Skenario Pengujian Sistem**

Menurut (20), pengujian perangkat lunak sangat diperlukan dalam sebuah sitem, dengan melakukan pengujian perangkat lunak ini dapat diketahui kesalahan atau error yang terjadi dalam sistem itu sendiri. Pada tahap pengujian sistem pada penelitian ini, peneliti menggunakan scenario pengujian sebagai berikut.

Skenario dari pengujian ini yaitu yang pertama akan dilakukan pemilahan atau penseleksian fitur dengan menggunakan metode *Information Gain.* Penseleksian fitur ini digunakan untuk mengurutkan fitur yang paling berpengaruh dalam dataset yang digunakan. setelah mengurutkan fitur dari yang terbaik ke yang terendah maka selanjutnya akan dipilih 9 fitur yang akan digunakan untuk pengklasifikasian anomali jaringan dengan menggunakan metode KNN *(K-Nearest Neighbor).* Program akan mengklasifikasikan data yang dibagi menjadi data uji terhadap data latih yang digunakan sebagai patokan dalam pengujian data uji tersebut. Kemudian akurasi dari keberhasilan pendeteksian akan dihitung untuk mengetahui seberapa besar akurasi yang dapat dihasilkan oleh program. Pada penelitian ini akurasi diperoleh dengan membandingkan jumlah data yang berhasil dengan benar diklasifikasikan kedalam 2 buah kelas normal dan kelas serangan DDoS. untuk mendapatkan nilai akurasi dihitung dari jumlah data yang tepat dibagi dengan jumlah data. Perhitungan nilai akurasi sebagai berikut (Vienna, 2009) :

# **BAB IV**

# **HASIL DAN PEMBAHASAN**

## **Gambaran Umum Sistem**

Pada bab ini membahas gambaran umum dari sistem klasifikasi serangan *Distributed Denial of Service* (DDoS). Dalam penelitian ini, penulis menggunakan data yang berasal dari KDD Cup 99 dengan jumlah data sebanyak 112378 data dimana data tersebut terdiri dari data normal dan data serangan *Denial of Service* (DoS) yaitu *back, land, neptune, ping of death* (POD), dan *smurf.* Penelitian ini juga menggunakan 41 atribut yang ada pada dataset KDD Cup 99.

Pada tahapan ini dibutuhkan masukkan data berupa dile dataset yang memiliki format \*.csv. Sebelum masuk ke proses klasifikasi sebelumnya fitur – fitur dari dataset diseleksi terlebih dahulu dan ditentukan jumlah fitur yang digunakan untuk tahap selanjutnya. Penulis menggunakan label 0 dan 1 untuk membedakan kelas pada sistem, dimana 0 artinya data tersebut adalah data yang masuk kedalam kelas normal dan 1 artinya data tersebut adalah data yang masuk pada kelas serangan.

## **Lingkungan Perancangan dan Implementasi Sistem**

Perancangan sistem dilakukan pada sistem operasi *Windows 10.* Sistem dirancang di sebuah laptop yang memiliki prosessor Intel® Core™ i7-7700HQ CPU @ 2.80GHz (8 CPUs), ~2.8GHz dan RAM sebesar 8 GB.

Sistem implementasi dengan menggunakan bahasa pemrograman Pyhton. Dalam perancangan dan implementasi sistem ini, digunakan beberapa perangkat lunak untuk dapat memenuhi kebutuhan implementasi sistem. berikut adalah perangkat lunak yang digunakan.

1. Anaconda Navigator
2. Jupyter Notebook
3. Microsoft Excel
4. Google Chrome

## **Implementasi Program**

Implementasi program pada sub bab ini membahas mengenai implementasi perancangan yang sebelumnya telah dibuat kedalam bahasa pemrograman. Adapun pembahasan dalam kode program ini adalah sebagai berikut.

### **Preprocessing Dataset**

Dalam proses preprosesing ini tahapan yang dilakukan adalah mengubah tipe data dari data kategorial yang ada pada dataset menjadi tipe data numerik. Pada dataset ini terdapat tiga atribut yang harus diubah kedalam bentuk numerik yaitu atribut protocol\_type, service, dan flag.

1. Tahap pertama yang dilakukan adalah menginputkan dataset KDD Cup 99 yang sudah disimpan kedalam file dengan format \*.csv.

Tabel 4. 1 Memasukkan File Dataset

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | data = pd.read\_csv('data.csv') |

1. Langkah selanjutnya adalah mengambil nilai dari masing – masing kolom dengan tipe data ketegorial. Kolom yang diambil adalah protocol\_type, service, dan flag

Tabel 4. 2 Nilai Kolom pada Atribut protocol\_type

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | protocol\_type\_columns=['protocol\_type'] |
| 2 | protocol\_type\_values = data[protocol\_type\_columns] |
| 3 | protocol\_type\_values.head() |

Tabel 4. 3 Nilai Kolom pada Atribut service

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | service\_columns=['service'] |
| 2 | service\_values = data[service\_columns] |
| 3 | service\_values.head() |

Tabel 4. 4 Nilai Kolom pada Atribut flag

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | flag\_columns=['flag'] |
| 2 | flag\_values = data[flag\_columns] |
| 3 | flag\_values.head() |

1. Kemudian buat kode unik di setiap nilai pada atribut.

Tabel 4. 5 Nilai Unik Atribut protocol\_type

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | unique\_protocol = sorted(data.protocol\_type.unique()) |
| 2 | string1 = 'protocol\_type\_' |
| 3 | unique\_protocol2 = [string1 + x for x in unique\_protocol] |
| 4  5 | dumcols\_protocol\_type = unique\_protocol2 |

Tabel 4. 6 Nilai Unik Atribut service

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | unique\_service = sorted(data.service.unique()) |
| 2 | string2 = 'service\_' |
| 3 | unique\_service2 = [string2 + x for x in unique\_service] |
| 4  5 | dumcols\_service = unique\_service2 |

Tabel 4. 7 Nilai Unik Atribut flag

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | unique\_flag = sorted(data.flag.unique()) |
| 2 | string3 = 'flag\_' |
| 3 | unique\_flag2 = [string3 + x for x in unique\_flag] |
| 4  5 | dumcols\_flag = unique\_flag2 |

1. Konversi nilai setiap atribut menjadi nilai numerik.

Tabel 4. 8 Konversi Nilai dari Atibut protocol\_type

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | protocol\_type\_values\_enc = protocol\_type\_values.apply(LabelEncoder().fit\_transform) |

Tabel 4. 9 Konversi Nilai dari Atibut service

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | service\_values\_enc = service\_values.apply(LabelEncoder().fit\_transform) |

Tabel 4. 10 Konversi Nilai dari Atibut flag

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | flag\_values\_enc = flag\_values.apply(LabelEncoder().fit\_transform) |

1. Gabungkan semua nilai yang sudah dikonversi kedalam dataset.

Tabel 4. 11 Penggabungan Atribut protocol\_type

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | data['protocol\_type'] = protocol\_type\_values\_enc |

Tabel 4. 12 Penggabungan Atribut service

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | data['service'] = service\_values\_enc |

Tabel 4. 13 Penggabungan Atribut flag

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | data['flag'] = flag\_values\_enc |

1. Konversi nilai dari label pada dataset. Dimana kelas dari data normal dikonversi menjadi 0 dan kelas dari data serangan dikonversi menjadi 1.

Tabel 4. 14 Konversi Nilai Atribut

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | label\_data = data['label'] |
| 2 | new\_label\_data = label\_data.replace({'normal' : 0, 'back' : 1, 'land' : 1, 'pod' : 1, 'neptune' : 1, 'smurf' : 1}) |
| 3 | data['label'] = new\_label\_data |

### **Proses Pemilihan Fitur dengan Information Gain**

Pada tahap seleksi fitur ini metode yang digunakan dalam melakukan seleksi fitur adalah *Information Gain.* Seleksi fitur ini bertujuan untuk memilih fitur yang paling relevan terhadap proses klasifikasi yang selanjutnya dilakukan.

Adapun tahapan dari pengklasifikasian dataset KKD Cup 99 adalah sebagai berikut.

1. Tahap pertama yang dilakukan adalah memisahkan kolom fitur dengan kolom label. Pemisahan ini bertujan untuk menandai label yang akan digunakan pada tahap pengklasifikasian

Tabel 4. 15 Pemisahan Fitur dan Label Dataset

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | fitur = data.drop('label', axis = 1) |
| 2 | target = data['label'] |
| 3 | fitur.shape, target.shape |

1. Tahap berikutnya adalah membagi data menjadi dua yaitu data latih dan data uji.

Tabel 4. 16 Pembagian Data Latih dan Data Uji

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | X\_train, X\_test, y\_train, y\_test = train\_test\_split(fitur, target, test\_size = 0.2, random\_state = 0) |

1. Selanjutnya lakukan ekstrasi fitur dengan threshold = 0.01 (ditentukan oleh user).

Tabel 4. 17 Ekstraksi Fitur

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | constant\_filter = VarianceThreshold(threshold=0.01) |
| 2 | constant\_filter.fit(X\_train) |
| 3 | X\_train\_filter = constant\_filter.transform(X\_train) |
| 4 | X\_test\_filter = constant\_filter.transform(X\_test) |

1. Kemudian hilangkan fitur yang memiliki nilai yang sama karena fitur tersebut akan memiliki pengaruh yang sama.

Tabel 4. 18 Menghilangkan Fitur dengan Nilai Sama

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | features\_to\_keep = [not index for index in duplicated\_features] |
| 2 | X\_train\_unique = X\_train\_T[features\_to\_keep].T |
| 3 | X\_test\_unique = X\_test\_T[features\_to\_keep].T |
| 4 | X\_train\_unique.shape, X\_test\_unique.shape |

1. Hitung nilai *Entropy* pada setiap fitur. Disini perhitungan nilai *Entropy* menggunakan library *Random Forest Classifier*.

Tabel 4. 19 Menghitung Nilai Entropy Tiap Fitur

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | mi = mutual\_info\_classif(X\_train\_unique, y\_train) |

1. Tahap selanjutnya adalah urutkan nila *Entropy* dari fitur mulai dari nilai paling besar hingga nilai paling kecil.

Tabel 4. 20 Mengurutkan Nilai Entropy Tiap Fitur

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | mi.sort\_values(ascending=False, inplace = True) |

1. Tampilkan diagram untuk melihat besar nilai *Entropy* pada setiap fitur.

Tabel 4. 21 Menampilkan Diagram Nilai Entropy

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | plt.title('Mutual information with respect to features') |
| 2 | mi.plot.bar(figsize = (16,5)) |
| 3 | plt.show() |

### **Proses Prediksi Data Uji dengan K-Nearest Neighbor**

Pada tahap ini, dilakukan proses klasifikasi pada dataset KDD Cup 99 dengan menggunakan metode *K-Nearest Neighbor* (KNN). Fitur yang digunakan pada tahap ini berjumlah 20 fitur dimana fitur tersebut dipilih berdasarkan urutan nilai *Entropy* pada fitur – fitur yang ada pada dataset KDD Cup 99.

Berikut adalah tahapan klasifikasi dataset KDD Cup 99 dengan menggunakan metode *K-Nearest Neighbor* (KNN).

1. Tahap pertama yaitu masukkan dataset yang sudah melalui tahap seleksi fitur dengan format dataset \*.csv.

Tabel 4. 22 Input Dataset

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | data = pd.read\_csv('feature\_selection.csv') |

1. Ulangi langkah pada tahap preeprosesing untuk mengubah nilai fitur ke dalam nilai numerik.
2. Selanjutnya pisahkan kolom label dan kolom fitur untuk memudahkan proses klasifikasi dan untuk memberi tanda pada setiap dataset.

Tabel 4. 23 Memisahkan Kolom Fitur dengan Kolom Label

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | label = data.iloc[:,-1:].values.ravel() |
| 2 | feature = data.iloc[:,:-1] |

1. Kemudian bagi dataset kedalam dua jenis data yaitu data latih dan data uji. Data latih disini digunakan sebagai patokan dalam pengklasifikasian data uji. Sedangkan data uji adalah data yang diklasifikasikan berdasarkan kemiripan dengan data latih. Untuk data latih sistem menggunakan 70% dari dataset dan untuk data uji sistem menggunakan 30% dari dataset.

Tabel 4. 24 Membagi Data Uji dan Data Latih

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | X\_train, X\_test, y\_train, y\_test = train\_test\_split(feature, label, test\_size=0.3) |

1. Langkah selanjutnya yaitu hitung nilai *Eucledian* dari data latih dengan jumlah tetangga sama dengan sembilan (n\_neighbors = 9). Jumlah n\_neighbor ditentukan oleh user. Untuk menggitung nilai *Eucledian* sistem menggunakan library *KNeighborsClassifier.*

Tabel 4. 25 Menghitung Nilai Eucledian Data Latih

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | knn = KNeighborsClassifier(n\_neighbors=9) |
| 2 | knn.fit(X\_train, y\_train) |

1. Selanjutnya prediksi data uji terhadap data latih.

Tabel 4. 26 Prediksi Data Uji

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | pred = knn.predict(X\_test) |
| 2 | pred |

1. Tampilkan data uji yang benar dengan data yang diprediksi oleh sistem.

Tabel 4. 27 Membandingkan Data Uji Sebenarnya dengan Prediksi Sistem

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | cols = { |
| 2 | 'data benar' : y\_test, |
| 3 | 'prediksi' : pred, |
| 4 | } |
| 5 | print("hasil") |
| 6 | pd.DataFrame(cols) |

### **Proses Perhitungan Akurasi dengan K-Fold Cross Validation**

Pada tahap ini, akurasi dari sistem akan diuji untuk mengetahui seberapa besar kebenaran prediksi terhadap suatu data. Perhitungan akurasi ini menggunakan metode *K-Fold Cross Validation.*

Berikut adalah tahapan dari perhitungan akurasi sistem.

1. Dalam sistem akurasi dihitung menggunakan library *KFold.* Untuk jumlah n split yang digunakan yaitu n\_split = 5. Tahap pertama yang dilakukan adalah membagi menjadi 5 bagian dan menghitung akurasi dari setiap data uji dan data testing.

Tabel 4. 28 Perhitung K-Fold Cross Validation

|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | kfold = KFold(n\_splits = 5) |
| 2 | scores = ['accuracy'] |
| 3 | results = cross\_validate(knn, X\_test, y\_test, cv=kfold, scoring=scores, return\_train\_score=True) |
| 4 | results\_data = pd.DataFrame(results) |

1. Selanjutnya adalah menghitung persentase dari akurasi yang dihasilkan oleh sistem. Rumus yang digunakan yaitu.

Tabel 4. 29 Perhitungan Akurasi SIstem

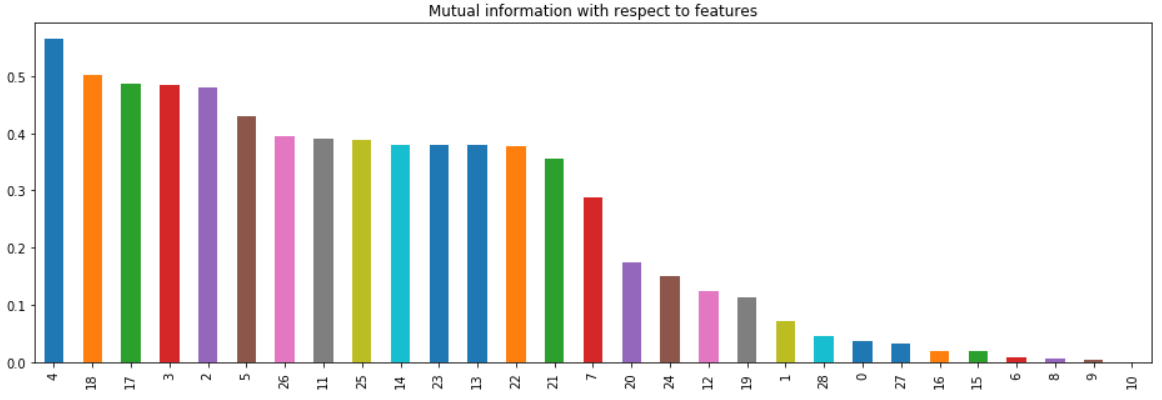
|  |  |
| --- | --- |
| **Baris** | **Kode** |
| 1 | accuracy = results\_data.test\_accuracy.mean() \* 100 |
| 2 | print("accuracy : %0.2f" % accuracy,"%") |

## **Pengujian Sistem**

Pada tahap ini program diuji secara keseluruhan sebagai sistem yang lengkap untuk memastikan bahwa persyaratan perangkat lunak telah terpenuhi dan sesuai dengan kebutuhan serta tujuan awal dari penelitian ini. Skenario dari pengujian ini yaitu yang pertama akan dilakukan pemilahan atau penseleksian fitur dengan menggunakan metode Information Gain. Penseleksian fitur ini digunakan untuk mengurutkan fitur yang paling berpengaruh dalam dataset yang digunakan. Setelah mengurutkan fitur dari yang terbaik ke yang terendah maka selanjutnya akan dipilih 9 fitur yang akan digunakan untuk pengklasifikasian anomali jaringan dengan menggunakan metode KNN (K-Nearest Neighbor). Program akan mengklasifikasikan data yang dibagi menjadi data uji terhadap data latih yang digunakan sebagai patokan dalam pengujian data uji tersebut. Kemudian akurasi dari keberhasilan pendeteksian akan dihitung untuk mengetahui seberapa besar akurasi yang dapat dihasilkan oleh program. Pada penelitian ini akurasi diperoleh dengan membandingkan jumlah data yang berhasil dengan benar diklasifikasikan kedalam 2 buah kelas normal dan kelas serangan DDoS.

### **Seleksi Fitur dengan *Information Gain***

Tahap pertama yang dilakukan pada pengujian sistem ini adalah mencari fitur yang paling relevan terhadap dataset yang digunakan yaitu dataset KDD Cup 99. Jumlah fitur yang terdapat dalam dataset KK Cup 99 ini adalah sebanyak 41 fitur.



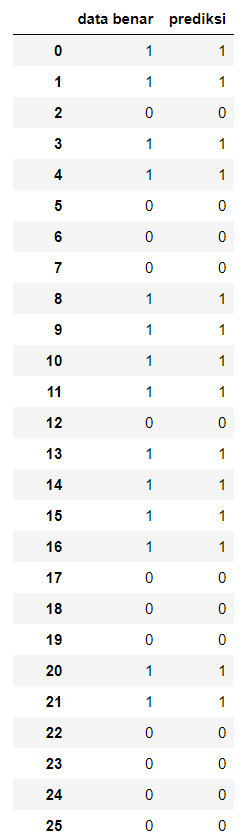
Gambar 4. 1 Grafikk Nilai Entropy Tiap Fitur

Pada Gambar 4.1 dapat dilihat urutan fitur yang memiliki nilai *Entropy* paling besar sampai dengan paling kecil. Pada sistem ini, fitur yang digunakan adalah sebanyak 20 fitur dimana fitur yang digunaka tersebut yaitu :

1. src\_bytes
2. num\_access\_files
3. num\_shells
4. flag
5. service
6. dst\_bytes
7. srv\_rerror\_rate
8. logged\_in
9. srv\_serror\_rate
10. su\_attempted
11. srv\_count
12. root\_shell
13. count
14. is\_guest\_login
15. wrong\_fragment
16. is\_host\_login
17. serror\_rate
18. num\_compromised
19. num\_outbound\_cmds
20. protocol\_type

### **Klasifikasi Serangan DDoS dengan K-Nearest Neighbor**

Klasifikasi serangan *Distributed Denial of Service* (DDoS) menggunakan metode *K-Nearest Neighbor* dengan jumlah tetangga (n\_neighbors) sebanyak 9. Jumlah dataset yang digunakan adalah 112378 dataset yang terdiri dari 67343 data normal dan 45035 data serangan. Dimana data serangan yang diprediksi oleh sistem adalah data serang *Denial of Service* (DoS) yaitu *back, land, neptune, ping of death* (POD), dan *smurf.*

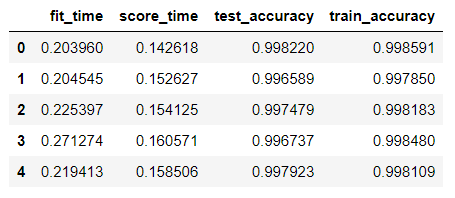


Gambar 4. 2 Hasil Perbandingan Data Uji Sebenarnya dengan Prediksi Data

Pada Gambar 4.2 dapat dilihat perbandingan antara data uji sebenarnya dengan prediksi data yang dilakukan oleh sistem.

### **Pengukuran Akurasi dengan K-Fold Cross Validation**

Pengukuran akurasi pada sistem ini menggunakan metode *K-Fold Cross Validation* untuk mengetahui seberapa baik prediksi yang dilakukan oleh sistem.



Gambar 4. 3 Akurasi Dataset

Gambar menunjukkan akurasi yang dihasilkan dari dataset dengan membagi dataset tersebut menjadi 5 bagian.



Gambar 4. 4 Akurasi Sistem

Gambar menunjukkan akurasi yang dihasilkan oleh sistem yang dibangun.

# **BAB V**

# **PENUTUP**

## **Kesimpulan**

Adapun kesimpulan yang diperoleh dari implementasi algoritma *K-Nearest Neighbor* (KNN) untuk klasifikasi serangan *Distributed Denial of Service* (DDoS) dengan menggunakan dataset yang berasal dari KDD Cup 99.

1. Setiap fitur memiliki nilai Entropy yang berbeda – beda sehingga bisa diketahui fitur – fitur apasaja yang memiliki nilai *Entropy* yang lebih tinggi. Seleksi fitur pada sistem menggunakan metode *Information Gain.*
2. Pada implementasi sistem ini menggunakan 20 fitur untuk dataset yang digunakan. Akurasi yang dihasilkan oleh sistem dengan menggunakan metode *K-Nearest Neighbor* (KNN) ini adalah sebesar 99,74% dalam melakukan klasifikasi.

## **Saran**

Adapun saran yang dapat diberikan untuk pengembangan sistem lebih lanjut adalah sebagai berikut :

1. Dalam sistem klasifikasi ini fitur yang digunakan hanya diseleksi menggunakan metode *Information Gain* sehingga perlu percobaan seleksi fitur dengan menggunakan metode lain juga.
2. Dapat menggunakan metode atau algoritma lain selain *K-Nearest Neighbor* (KNN) atau kombinasikan beberapa algoritma untuk menghasilkan klasifikasi data yang lebih baik.

# **DAFTAR PUSTAKA**

# **LAMPIRAN**