BAB III
METODE PENELITIAN

3.1 Metode Penelitian

Penelitian ini menggunakan metode *Research & Development* (R&D) yang bertujuan untuk menghasilkan solusi praktis dengan menggunakan model *Random Forest* dalam *Machine Learning*. Pendekatan penelitian ini difokuskan pada perancangan dan implementasi sistem deteksi DDoS dengan integrasi pemberitahuan cepat melalui *Telegram*. Pada Gambar 3.1 menggambarkan alur penelitian.

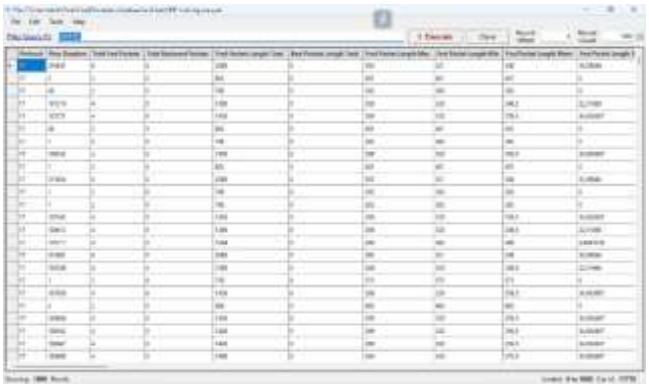
Gambar 3.1 Alur Penelitian

Pada Gambar 3.1 mengilustrasikan alur penelitian yang sistematis dan terstruktur, dimulai dari tahap awal hingga penyelesaian penelitian. Tahapan-tahapan yang ditunjukkan meliputi: Mulai Penelitian, Eksplorasi dan Identifikasi Masalah, Pengumpulan dan Analisis Data, Perancangan Model, Pengembangan Model dan Sistem, Pengujian dan Validasi, Evaluasi dan Penyempurnaan, serta Dokumentasi dan Publikasi. Setiap tahapan saling berhubungan dan membentuk sebuah proses yang berkesinambungan, memastikan bahwa penelitian dilakukan dengan metodologi yang tepat dan hasil yang diperoleh dapat dipertanggungjawabkan secara ilmiah.

Kemudian, terdapat deskripsi pada Gambar 3.1 tahapan alur penelitian yang terdiri dari beberapa tahapan, meliputi:

1. Eksplorasi dan Identifikasi Masalah
    a. Identifikasi Kebutuhan: Menentukan kebutuhan akan sistem yang mampu mendeteksi serangan DDoS secara efektif dan memberikan pemberitahuan cepat kepada pengguna atau administrator jaringan.
    b. Studi Literatur dan Best Practices: Mengkaji penelitian sebelumnya, teknologi yang ada, dan praktik terbaik dalam deteksi serangan DDoS serta metode pemberitahuan cepat. Fokus pada penggunaan *Machine Learning* dan *Telegram* sebagai *platform* notifikasi.
    c. Penentuan Tujuan dan Ruang Lingkup: Menetapkan tujuan penelitian untuk mengembangkan sistem deteksi DDoS berbasis algoritma *Random Forest* dan mendefinisikan ruang lingkup pengembangan, termasuk fitur dan komponen utama sistem.

2. Pengumpulan dan Analisis Data
    a. Pengumpulan Dataset: Data lalu lintas jaringan diambil dari dataset publik CICDDoS2019. Dataset ini dikumpulkan oleh *Canadian Institute for Cybersecurity* yang mencakup berbagai jenis serangan DDoS serta aktivitas jaringan normal, dipilih karena kelengkapannya dalam menyediakan berbagai pola serangan dan non-serangan. Pada Gambar 3.2 data mentah dari dataset CICDDoS2019 diproses untuk menghilangkan data yang tidak relevan atau yang rusak guna memastikan kualitas data yang baik. Selanjutnya, fitur-fitur penting dari data diidentifikasi dan diekstraksi untuk digunakan dalam model deteksi DDoS, termasuk jumlah paket per detik, sumber dan tujuan IP, serta pola lalu lintas lainnya.



Gambar 3.2 Dataset CICDDoS2019

b. Analisis Data: Melakukan analisis awal untuk memahami distribusi kelas, karakteristik data, serta mengidentifikasi fitur yang relevan untuk mendeteksi serangan.

Setelah data dikumpulkan, langkah berikutnya adalah pembersihan dan normalisasi data untuk memastikan kualitas dan konsistensi data sebelum digunakan dalam pelatihan model. Proses ini mencakup beberapa langkah berikut:

1) Pembersihan Data (*Data Cleaning*):
   a. Menghapus Duplikasi: Menghapus *entri* data yang duplikat untuk menghindari bias dalam model.
   b. Mengatasi *Missing Values*: Mengisi atau menghapus data yang hilang (*missing values*) menggunakan teknik seperti imputasi rata-rata, modus, atau nilai median.
   c. Menghapus *Outliers*: Mengidentifikasi dan menghapus *outliers* yang dapat mengganggu pelatihan model.
2) Normalisasi Data (Data *Normalization*):
   a. *Scaling*: Mengubah skala fitur-fitur data menggunakan teknik seperti *Min-Max Scaling* atau *Standardization* agar semua fitur berada dalam skala yang sama. Ini penting untuk algoritma pembelajaran mesin yang sensitif terhadap skala fitur.
   b. *Encoding*: Mengubah fitur kategorikal menjadi format numerik menggunakan teknik seperti *one-hot encoding* atau label *encoding*.

3) Pemisahan Fitur dan Label (*Feature and Target Separation*):
   a. Fitur *(Features)*: Kolom-kolom dalam dataset yang berfungsi sebagai input untuk model pembelajaran mesin. Contoh fitur meliputi jumlah paket, ukuran paket, dan waktu antar paket.
   b. *Label* (Target): Kolom yang menunjukkan kategori atau kelas dari data, seperti serangan DDoS atau lalu lintas normal.
4) Pembagian Data (Data *Splitting*):
   a. *Training Data*: Sebagian 80% dataset yang digunakan untuk membangun dan melatih model.
   b. *Testing Data*: Sebagian 20% dari dataset yang digunakan untuk menguji kinerja model.

Dengan mengikuti langkah-langkah pembersihan dan normalisasi data ini, data yang digunakan dalam penelitian ini menjadi siap untuk tahap berikutnya, yaitu pelatihan dan pengujian model deteksi serangan DDoS menggunakan algoritma *Random Forest*. Proses ini memastikan bahwa data yang digunakan adalah berkualitas tinggi dan dapat memberikan hasil yang akurat dan andal.

Berikut merupakan penjelasan dari jenis-jenis data penelitian ini, meliputi:

1) Data Simulasi Serangan DDoS
   a. Dataset CICDDoS2019: digunakan untuk simulasi serangan DDoS terdiri dari 431,371 sampel, di mana 345,096 sampel (80%) dialokasikan untuk pelatihan dan 86,275 sampel (20%) untuk pengujian model. Dataset ini mencakup pola serangan DDoS yang umum serta variasi lalu lintas yang mungkin muncul selama serangan, dengan tujuan memberikan representasi yang komprehensif dari skenario dunia nyata.
   b. *Generated* data: digunakan untuk mensimulasikan berbagai skenario serangan yang mungkin belum tercakup dalam dataset yang ada. *Generated* data ini mencakup variasi dalam 72 parameter serangan, seperti *Protocol* (nilai 6 untuk TCP dan 17 untuk UDP), *Flow Duration* (2.000.000 hingga 5.000.000

mikrodetik), Total Fwd *Packets* (5.000 hingga 20.000 paket), dan Flow *Bytes/s* (8.000.000 hingga 20.000.000 *byte/s*). *Generated* data ini diilustrasikan pada Gambar 3.3 dan digunakan untuk memastikan bahwa model deteksi dapat mengidentifikasi serangan DDoS dengan berbagai karakteristik, sehingga meningkatkan keandalan model dalam situasi dunia nyata.

Gambar 3.3 *Generated* Data

2) Data Penggunaan Aplikasi *Telegram*

Data tentang *penggunaan* aplikasi pesan *Telegram*, diperoleh untuk mengevaluasi kecepatan dan kehandalan pemberitahuan cepat. Ini mencakup waktu *respons* notifikasi, frekuensi penggunaan, dan tanggapan pengguna terhadap pemberitahuan.

3) Data Evaluasi Sistem

Data evaluasi sistem akan mencakup hasil dari uji coba dan pengujian sistem deteksi DDoS yang diimplementasikan. Ini mencakup akurasi deteksi, waktu *respons* sistem, dan keberhasilan *respons* terhadap serangan.

Data yang diperoleh dari hasil deteksi dan pemberitahuan digunakan untuk mengevaluasi keefektifan sistem secara keseluruhan, mencakup analisis kinerja model serta kecepatan dan keakuratan pemberitahuan. Analisis dilakukan untuk menilai performa model *Random Forest* dalam mendeteksi serangan DDoS dan efektivitas pemberitahuan cepat melalui

Telegram, termasuk pengukuran metrik evaluasi serta interpretasi hasil untuk mendapatkan wawasan yang lebih dalam tentang kinerja sistem.

3. Perancangan Model
    a. Pemilihan Algoritma: Memilih algoritma *Random Forest* berdasarkan studi literatur dan pertimbangan teknis, termasuk keandalan dan performa dalam klasifikasi.
    b. Pemilihan Fitur: Menentukan fitur-fitur penting yang akan digunakan dalam model, seperti *traffic volume* (Gbps), *packets per second* (pps), dan *requests per second* (rps).
    c. Desain Arsitektur Sistem: Merancang arsitektur sistem yang mencakup modul deteksi serangan, analisis data, dan sistem pemberitahuan melalui *Telegram*.

4. Pengembangan Model dan Sistem
    a. Praproses Data: Melakukan pembersihan dan normalisasi data, mengelola data yang hilang, dan membagi dataset menjadi set pelatihan dan pengujian.
    b. Pelatihan Model: Melatih model *Random Forest* dengan dataset yang telah diproses, melakukan *tuning hyperparameter*, dan menguji model dengan data pengujian.
    c. Integrasi Sistem Pemberitahuan: Mengembangkan sistem pemberitahuan cepat dengan integrasi *bot Telegram* untuk mengirim pesan otomatis ketika serangan terdeteksi.

Setelah sistem mendeteksi adanya serangan DDoS menggunakan model *Random Forest*, langkah selanjutnya adalah mengirimkan notifikasi ke *Telegram* untuk memberitahukan adanya serangan tersebut. Proses pengiriman notifikasi ke *Telegram* dilakukan dengan menggunakan *Bot* API yang disediakan oleh *Telegram*. Berikut adalah langkah-langkah detailnya:

1) Membuat Bot di *Telegram*:
    a. Buka aplikasi *Telegram* dan cari *BotFather*.

b. Buat *bot* baru dengan mengikuti instruksi dari *BotFather* dan dapatkan token API *bot*.

2) Menggunakan API *Telegram*:
   a. Gunakan token API yang diperoleh dari *BotFather* untuk mengirim pesan melalui bot.
   b. Gunakan *endpoint* API *Telegram* `https://api.telegram.org/bot<token>/sendMessage` untuk mengirim pesan.

3) Mengirim Notifikasi:
   a. Ketika serangan DDoS terdeteksi, sistem akan memanggil fungsi `send_telegram_notification` dengan pesan yang berisi informasi tentang serangan.
   b. Fungsi ini akan membuat permintaan HTTP *GET* ke *endpoint* API *Telegram* dengan parameter *chat_id* dan pesan.

Berikut merupakan contoh kode untuk mengirimkan notifikasi ke *Telegram* Pada Gambar 3.4:

```
import requests

def send_telegram_notification(message):
    bot_token = 'YOUR_TELEGRAM_BOT_TOKEN'
    bot_chatID = 'YOUR_CHAT_ID'
    send_text = f'https://api.telegram.org/bot{bot_token}/sendMessage?chat_id={bot_chatID}&parse_mode=Markdown&text={message}'
    response = requests.get(send_text)
    return response.json()

# Contoh penggunaan
message = "DDoS attack detected! Immediate action required."
send_telegram_notification(message)
```

Gambar 3.4 Kode untuk mengirim notifikasi ke *Telegram*

Proses ini pada Gambar 3.4 memastikan bahwa setiap kali serangan DDoS terdeteksi, notifikasi cepat dikirimkan melalui *Telegram*, sehingga administrator jaringan dapat segera mengambil tindakan yang diperlukan untuk menangani serangan tersebut.

5. Pengujian dan Validasi
    a. Pengujian Model: Mengevaluasi kinerja model dengan menggunakan metrik seperti akurasi, presisi, *recall*, dan *F1-score* untuk menilai kemampuan deteksi.
    b. Simulasi Serangan DDoS: Melakukan simulasi serangan DDoS untuk menguji model dalam kondisi realistis dan mengevaluasi respons sistem.
    c. Pengujian Sistem Pemberitahuan: Mengukur kecepatan dan akurasi notifikasi yang dikirimkan melalui *Telegram*, serta mengevaluasi efektivitasnya.

6. Evaluasi dan Penyempurnaan
    a. Analisis Hasil: Menganalisis hasil pengujian dan simulasi untuk menilai efektivitas model deteksi dan sistem pemberitahuan.
    b. Identifikasi Kelemahan: Mengidentifikasi kelemahan seperti *false positives* dan *false negatives* serta variasi dalam waktu pengiriman notifikasi.
    c. Perbaikan dan Optimasi: Melakukan iterasi untuk memperbaiki dan mengoptimalkan model dan sistem berdasarkan hasil evaluasi.

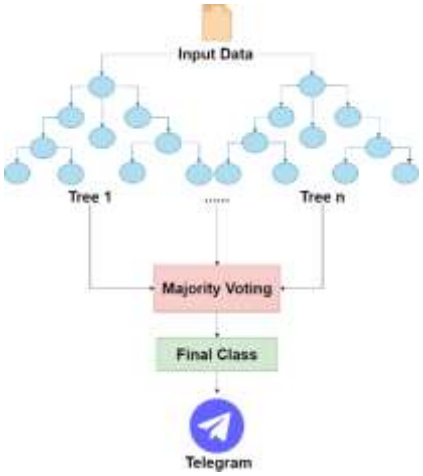
7. Dokumentasi dan Publikasi
    a. Dokumentasi Teknis: Menyusun dokumentasi lengkap dari desain, implementasi, dan penggunaan sistem, termasuk panduan pengguna dan dokumentasi teknis.
    b. Publikasi dan Diseminasi: Menyusun hasil penelitian dalam bentuk skripsi.

Alur penelitian ini mencakup seluruh tahapan penting dalam siklus R&D, mulai dari identifikasi masalah hingga evaluasi dampak dan rencana pengembangan lanjutan, serta memberikan dasar yang kuat

untuk pengembangan dan evaluasi sistem deteksi DDoS yang efisien dan efektif.

3.1.1 Model Arsitektur Random Forest

Pada penelitian ini, untuk memastikan bahwa sistem deteksi DDoS ini dapat memberikan *respons* yang cepat dan akurat, arsitektur model yang digunakan harus mampu menangani volume data yang besar dan melakukan klasifikasi dengan efisiensi tinggi. Algoritma *Random Forest* dipilih karena kemampuannya dalam menggabungkan prediksi dari berbagai pohon keputusan, yang menghasilkan tingkat akurasi yang tinggi. Gambar 3.5 menggambarkan arsitektur model *Random Forest* yang digunakan dalam sistem ini. *Input* data lalu lintas jaringan diproses melalui sejumlah pohon keputusan (*Tree* 1 hingga *Tree* n). Setiap pohon menghasilkan prediksi, yang kemudian digabungkan menggunakan mekanisme "*Majority Voting*" untuk menentukan kelas akhir dari *input* tersebut. Jika serangan terdeteksi, notifikasi akan segera dikirimkan melalui *Telegram*.



Gambar 3.5 Arsitektur Model *Random Forest*

Arsitektur pada Gambar 3.5 menunjukkan berbagai elemen yang bekerja bersama untuk mendeteksi serangan DDoS dan mengirimkan pemberitahuan cepat melalui *Telegram*. Pada Tabel 3.1 menjelaskan lebih lanjut mengenai deskripsi arsitektur model *Random Forest* tersebut sebagai beirkut:

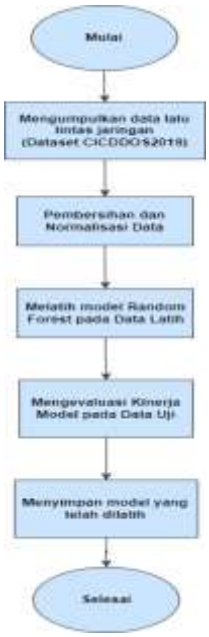
Tabel 3.1 Deskripsi Arsitektur *Random Forest*

Langkah	Deskripsi
Input Data	Data yang digunakan sebagai input untuk model, seperti data lalu lintas jaringan (misalnya dari CICDDoS2019).
Tree 1 hingga *Tree n*	Data *input* diproses melalui beberapa pohon keputusan dalam model *Random Forest*. Setiap pohon menghasilkan prediksinya sendiri berdasarkan subset data yang berbeda.
Majority Voting	Prediksi dari semua pohon keputusan dikombinasikan menggunakan *majority voting* untuk menentukan hasil akhir.
Final Class	Prediksi akhir atau kelas yang diputuskan oleh model, misalnya apakah lalu lintas adalah serangan DDoS atau tidak.
Telegram	Jika serangan terdeteksi, pemberitahuan dikirim melalui *Telegram* untuk memberikan peringatan kepada pengguna atau administrator.

3.1.2 *Flowchart* Perancangan Model

Flowchart Perancangan Model Deteksi DDoS menggunakan algoritma *Random Forest* dimulai dengan pengumpulan data lalu lintas jaringan dari dataset CICDDoS2019. Langkah berikutnya adalah pembersihan dan normalisasi data untuk memastikan kualitasnya. Model *Random Forest* kemudian dilatih menggunakan data latih. Setelah itu, kinerja model dievaluasi menggunakan data uji. Model yang telah dilatih kemudian disimpan untuk digunakan dalam deteksi serangan DDoS di masa mendatang. Proses ini dimulai dari tahap pengumpulan data hingga penyimpanan model yang telah dilatih, dan

berakhir pada tahap evaluasi dan penyimpanan model. Proses ini ditunjukkan pada Gambar 3.6.



Gambar 3.6 *Flowchart* Perancangan Model

Berikut merupakan penjelasan dari langkah-langkah *Flowchart* pada Gambar 3.6, yaitu:

1. Mengumpulkan data lalu lintas jaringan (Dataset CICDDOS2019).
2. Pembersihan dan normalisasi data.
3. Melatih model *Random Forest* pada data latih.
4. Mengevaluasi kinerja model pada data uji.
5. Menyimpan model yang telah dilatih.

3.2 Model *Machine Learning*

Pada penelitian ini, algoritma *Random Forest* digunakan sebagai model pembelajaran mesin utama untuk mendeteksi serangan DDoS. *Random Forest* dipilih karena kemampuannya yang kuat dalam menangani dataset yang kompleks dan beragam serta memberikan hasil prediksi yang akurat. Beberapa parameter penting yang digunakan dalam model *Random Forest* antara lain:

1. *n_estimators*: Jumlah pohon dalam hutan. Semakin banyak pohon, semakin stabil prediksi model, namun juga semakin tinggi

kebutuhan komputasi. Dalam penelitian ini, nilai default sebanyak 100 pohon digunakan.

2. *max_features*: Jumlah fitur maksimum yang dipertimbangkan untuk membelah setiap node. Defaultnya adalah akar kuadrat dari jumlah total fitur, yang membantu menjaga keseimbangan antara bias dan *varians*.
3. *max_depth*: Kedalaman maksimum dari pohon keputusan. Parameter ini membantu mengontrol *overfitting* dengan membatasi pertumbuhan pohon.
4. *min_samples_split*: Jumlah minimum sampel yang dibutuhkan untuk membagi node. Hal ini mencegah pohon menjadi terlalu rumit dengan memastikan bahwa node hanya akan terbagi jika memiliki cukup data.
5. *min_samples_leaf*: Jumlah minimum sampel yang dibutuhkan untuk menjadi daun akhir dalam pohon. Hal ini mencegah pohon dari mempelajari detail yang terlalu spesifik pada data pelatihan.

Setelah menentukan parameter-parameter ini, langkah berikutnya adalah mempersiapkan data yang akan digunakan. Proses ini mencakup pengumpulan, pembersihan, dan normalisasi data. Data yang telah diproses selanjutnya digunakan untuk melatih model *Random Forest*, di mana setiap pohon dalam hutan dibentuk dengan menggunakan subset acak dari dataset.

3.3 Metode Evaluasi

Metode evaluasi digunakan untuk menilai kinerja model klasifikasi dalam mendeteksi serangan *Distributed Denial of Service* (DDoS) menggunakan algoritma *Random Forest*. Dalam penelitian ini, beberapa metrik evaluasi penting digunakan untuk memberikan pemahaman yang komprehensif tentang efektivitas model. Berikut adalah menurut Pai, V., & Adesh, N. D. (2021) dan Vikram, A. (2020) definisi dan rumus untuk metrik yang digunakan:

1. Akurasi (*Accuracy*)

Definisi: Akurasi adalah ukuran yang menunjukkan seberapa sering model membuat prediksi yang benar dibandingkan dengan semua prediksi yang dibuat. Ini memberikan pandangan umum tentang kinerja model.

Rumus: $Akurasi = \frac{(TP + TN)}{(TP + TN + FP + FN)}$ (1)

Penjelasan:

True Positives (TP): Kasus di mana model memprediksi positif dan itu benar.

True Negatives (TN): Kasus di mana model memprediksi negatif dan itu benar.

False Positives (FP): Kasus di mana model memprediksi positif tetapi sebenarnya negatif.

False Negatives (FN): Kasus di mana model memprediksi negatif tetapi sebenarnya positif.

2. Presisi (*Precision*)

   Definisi: Presisi mengukur proporsi prediksi positif yang benar dibandingkan dengan semua prediksi positif yang dibuat oleh model. Ini sangat penting dalam konteks di mana kesalahan positif (FP) harus diminimalisir.

   Rumus: $Presisi = \frac{TP}{(TP + FP)}$ (2)

3. *Recall* (*Sensitivity* atau *True Positive Rate*)

   Definisi: *Recall* mengukur kemampuan model untuk mendeteksi semua instance positif yang ada. Ini sangat penting dalam mendeteksi semua serangan yang terjadi.

   Rumus: $Recall = \frac{TP}{(TP + FN)}$ (3)

4. *F1-Score*

   Definisi: *F1-Score* adalah rata-rata harmonik dari presisi dan *recall*. Ini memberikan gambaran tentang keseimbangan antara presisi dan *recall*, dan penting dalam situasi di mana keseimbangan antara kedua metrik ini kritis.

   Rumus:$F1 - Score = 2\ x\ \frac{(Presisi\ x\ Recall)}{(Presisi + Recall)}$ (4)

Dalam konteks penelitian dengan judul "Perancangan Model Deteksi *Distributed Denial of Service* (DDoS) Menggunakan Algoritma *Random Forest* dengan Pemberitahuan Cepat Melalui Telegram", penggunaan metrik ini bertujuan untuk mengevaluasi seberapa baik model dapat membedakan antara lalu lintas normal dan serangan DDoS. Metrik-metrik ini penting untuk memberikan gambaran yang jelas mengenai efektivitas model, baik dalam hal ketepatan prediksi (presisi) maupun kemampuan mendeteksi semua kasus serangan (*recall*).

Untuk menguji kinerja model secara lebih umum dan menghindari *overfitting*, teknik *Cross Validation* diterapkan. *Cross validation* memungkinkan model dievaluasi pada berbagai subset data yang berbeda, memberikan wawasan lebih mendalam tentang performa model pada data yang belum pernah dilihat sebelumnya. Metode ini memastikan bahwa model tidak hanya bekerja dengan baik pada data pelatihan, tetapi juga memiliki kemampuan generalisasi yang kuat pada data uji.