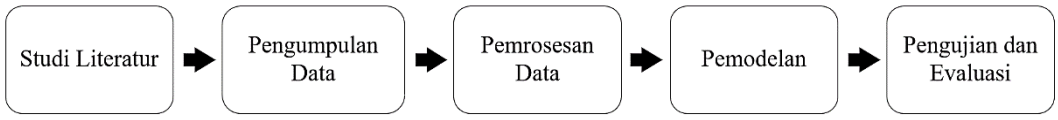
# BAB III

# METODOLOGI PENELITIAN

Alur penelitian yang akan dilakukan ditunjukan pada gambar 3.1 sebagai berikut :




Gambar 3.1 Alur Metodologi Penelitian

## 3.1 Studi Literatur

Studi literatur ini berisi uraian tentang teori, temuan, dan bahan penelitian lainnya yang diperoleh dari jurnal internasional dan jurnal nasional. Studi literatur ini akan dijadikan landasan kegiatan penelitian dalam menyusun kerangka pemikiran yang jelas dari perumusan masalah yang ingin diteliti. Studi literatur yang digunakan adalah jurnal mengenai analisis serangan *malware* dan *machine learning*.

## 3.2 Pengumpulan Data

Data yang digunakan dalam penelitian ini adalah data sekunder, yaitu data yang tidak diperoleh secara langsung melainkan dikumpulkan oleh pihak lain. Data yang digunakan adalah dataset publik deteksi serangan *botnet* pada perangkat *Internet of Things*.

Dataset ini dihasilkan dari data lalu lintas jaringan mentah dalam format *packet capture* (.pcap) menggunakan *mirroring port* pada *switch*. Data latih ini bersih dari perilaku jahat, lalu lintas jaringan normal dikumpulkan segera setelah pemasangan perangkat di jaringan. Data yang digunakan merupakan data lalu lintas

*real*, yang dikumpulkan dari 7 perangkat *IoT* komersial yang terinfeksi secara otentik oleh *Mirai*.

Data tersebut dapat diunduh melalui halaman website *UCI Repository* ini https://archive.ics.uci.edu/ml/datasets/detection_of_IoT_botnet_attacks_N_BaIoT. Dataset yang diambil adalah data lalu lintas normal dan lalu lintas serangan *Malware Mirai* pada jenis kamera keamanan. Perangkat IoT yang akan digunakan sebagai penelitian ditunjukkan pada tabel 3.1 sebagai berikut :

Tabel 3.1 Perangkat IoT (berjenis *security camera*)

| *No* | *Model* | *Type* |
|---|---|---|
| *1* | *Provision PT-737E* | *Security Camera* |
| *2* | *Provision PT-838* | *Security Camera* |
| *3* | *SimpleHome XCS7-1002-WHT* | *Security Camera* |
| *4* | *SimpleHome XCS7-1003-WHT* | *Security Camera* |

Perangkat yang berjenis kamera keamanan terdiri dari 2 model, masing-masing terdiri dari 2 versi keluaran yang berbeda. Pengujian akan dilakukan pada 5 jenis serangan *Malware Mirai*, serangan yang akan digunakan sebagai penelitian ditunjukkan pada tabel 3.2 sebagai berikut :

Tabel 3.2 Jenis Serangan

| *No* | *Attacks* | *Description* |
|---|---|---|
| *1* | *Scan* | *Automatic scanning for vulnerable devices* |
| *2* | *ACK* | *Ack flooding* |
| *3* | *SYN* | *Syn flooding* |
| *4* | *UDP* | *UDP flooding* |
| *5* | *UDPplain* | *UDP flooding with fewer options, optimized for higher packets per second* |

*Mirai* menyerang dengan cara *scan* kerentanan *port* pada perangkat IoT dan 4 serangan lainnya menggunakan tipe serangan *Distributed Denial of Sevices* (DDoS),

cara kerjanya yaitu membanjiri kinerja perangkat dengan cara mengirim paket yang sangat banyak pada perangkat yang menjadi target.

### 3.3 Pemrosesan Data

Pemrosesan data yaitu melakukan pengolahan dataset yang tadinya tidak seimbang dan dimensi dataset yang tinggi. Dataset tersebut dilakukan proses *sampling* untuk menyeimbangkan datasetnya, jumlah datanya di*sampling* menjadi 1000 data pada masing-masing dataset dan dilakukan proses *features selection* untuk menyeleksi fitur yang sesuai untuk proses klasifikasi serangan *Mirai*, fitur yang dipilih yaitu 5 fitur terbaik berdasarkan kriteria *Gini index*.

### 3.4 Pemodelan

Pemodelan merupakan tahapan dalam pembuatan model untuk proses *features selection* yaitu model untuk meseleksi fitur terbaik menggunakan kriteria *Gini index* dan proses pengujian untuk menentukan mengolah data latih dan data uji dengan *output* evaluasi dan deteksi anomali.

### 3.5 Pengujian dan Evaluasi

Penelitian ini melakukan evaluasi menggunakan metode *confusion matrix*. *Confusion matrix* digunakan untuk menganalisis seberapa baik *classifier* mengenali *tuple* dari kelas yang berbeda (Han & Kamber, 2011). Nilai dari *True-Positive* dan *True-Negative* memberikan informasi ketika *classifier* melakukan klasifikasi data bernilai benar, sedangkan *False-Positive* dan *False-Negative* memberikan

informasi ketika *classifier* salah dalam melakukan klasifikasi data (Fibrianda & Bhawiyuga, 2018). C*onfusion matrix* ditunjukkan pada Tabel 3.3 sebagai berikut:

Tabel 3.3 *Confusion Matrix*

| Aktual / Prediksi | *Attack* | Normal |
|---|---|---|
| *Attack* | TP | FN |
| Normal | FP | TN |

*True-Positive* (TP) yaitu jumlah prediksi benar dari *flow attack*. *False-Positive* (FP) yaitu jumlah kesalahan prediksi *flow normal* pada *flow attack*. *False- Negative* (FN) yaitu jumlah kesalahan prediksi *flow attack* pada *flow* normal. *True-Negative* (TN) yaitu jumlah prediksi benar pada *flow normal*.

Hasil *confusion matrix* digunakan untuk mengukur akurasi, *precision*, *recall* dan *F-measure* untuk menganalisa kinerja dari algoritma dalam melakukan klasifikasi untuk mendeteksi *botnet mirai* dengan persamaan (3.1) (3.2) (3.3) (3.4):

$$\text{Akurasi} = \frac{TP+TN}{TP+TN+FP+FN} \quad (3.1)$$

$$\textit{Recall} = \frac{TP}{TP+FN} \quad (3.2)$$

$$\textit{Precision} = \frac{TP}{TP+FP} \quad (3.3)$$

$$\textit{F-Measure} = 2 \text{ x } \frac{Precision\ x\ Recall}{Precision + Recall} \quad (3.4)$$

Akurasi yaitu kedekatan antara nilai prediksi dan nilai aktual. *Precision* yaitu tingkat ketepatan antara informasi yang diminta oleh pengguna dengan jawaban yang diberikan oleh sistem. *Recall* ialah tingkat keberhasilan sistem dalam menemukan kembali sebuah informasi. *F-Measure* merupakan salah satu perhitungan evaluasi yang mengkombinasikan *recall* dan *precission*.

Proses selanjutnya melakukan *anomaly detection*, yaitu teknik untuk mendeteksi *outlier*. Istilah *outlier* mengacu pada himpunan bagian dari serangkaian titik data yang gagal untuk menyesuaikan dengan pola konvensional. *Outlier* tersebut biasanya dihapus ataupun dihilangkan untuk membuat pola konvensional yang sudah dibuat menjadi lebih baik. *Anomaly detection* yang digunakan menggunakan *unsupervised learning* menggunakan teknik *detect outlier (Distances)* yaitu deteksi anomali dengan menggunakan algoritma *k-Nearest Neighbor* (k-NN) untuk proses klasifikasinya.