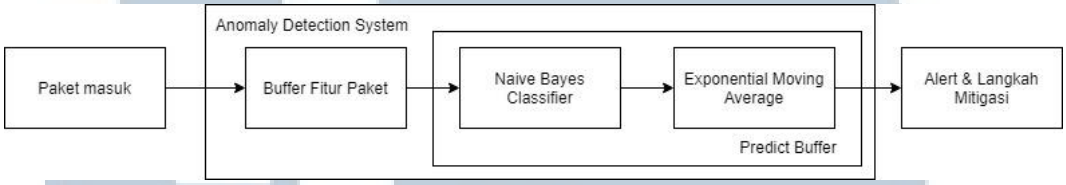
BAB III

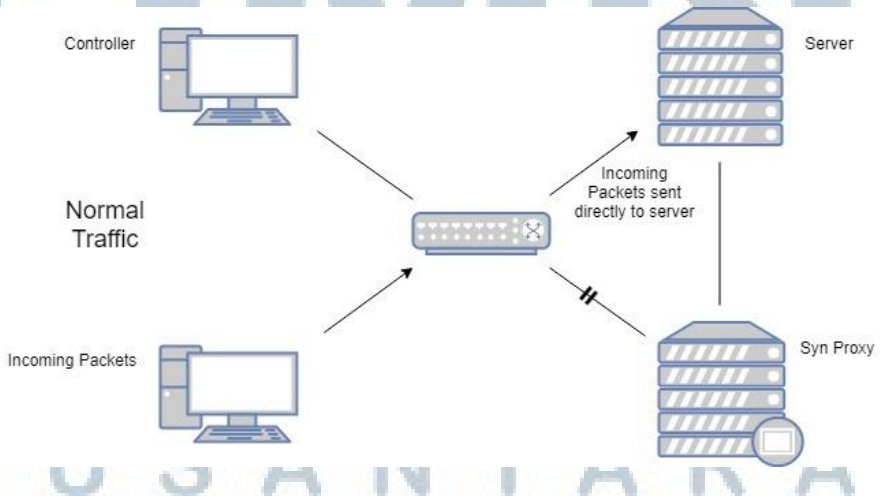
METODE PENELITIAN

3.1. Diagram Blok Arsitektur Sistem



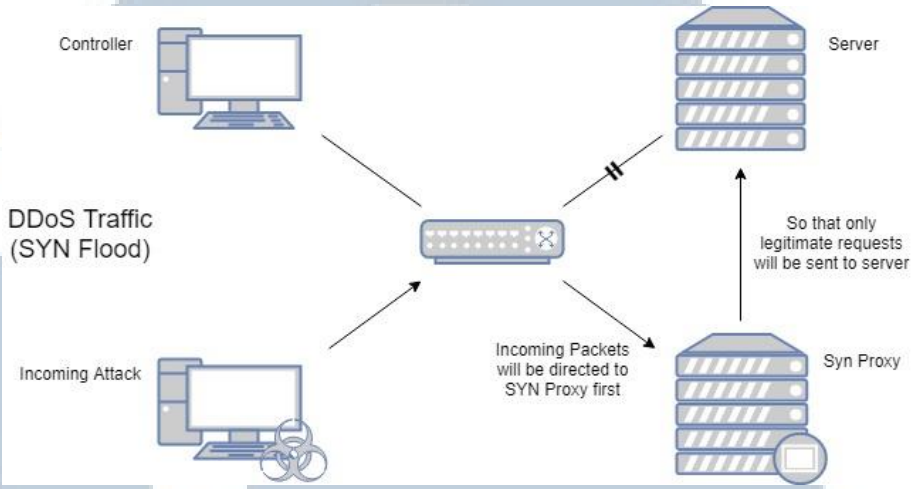
Gambar 3.1 Diagram Blok *Anomaly Detection System*

Gambar 3.1 menunjukkan diagram blok *anomaly detection system*. Paket yang masuk akan diambil fitur-fiturnya (*source* dan *destination* IP, *port* tujuan, ukuran paket, TCP *flag*) dan disimpan dalam *buffer* sambil terus menerus diakumulasi. Ketika *buffer* mencapai 100 paket maka hasil akumulasi fitur-fitur yang ada akan digunakan oleh *Naïve Bayes Classifier* untuk menentukan nilai probabilistik masing-masing kelas (normal dan anomali) dan kemudian dikalkulasi menggunakan *exponential moving average* untuk menentukan kondisi jaringan serta *rule* yang berlaku pada perangkat jaringan.



Gambar 3.2 Alur Paket Masuk pada *Traffic* Jaringan Normal

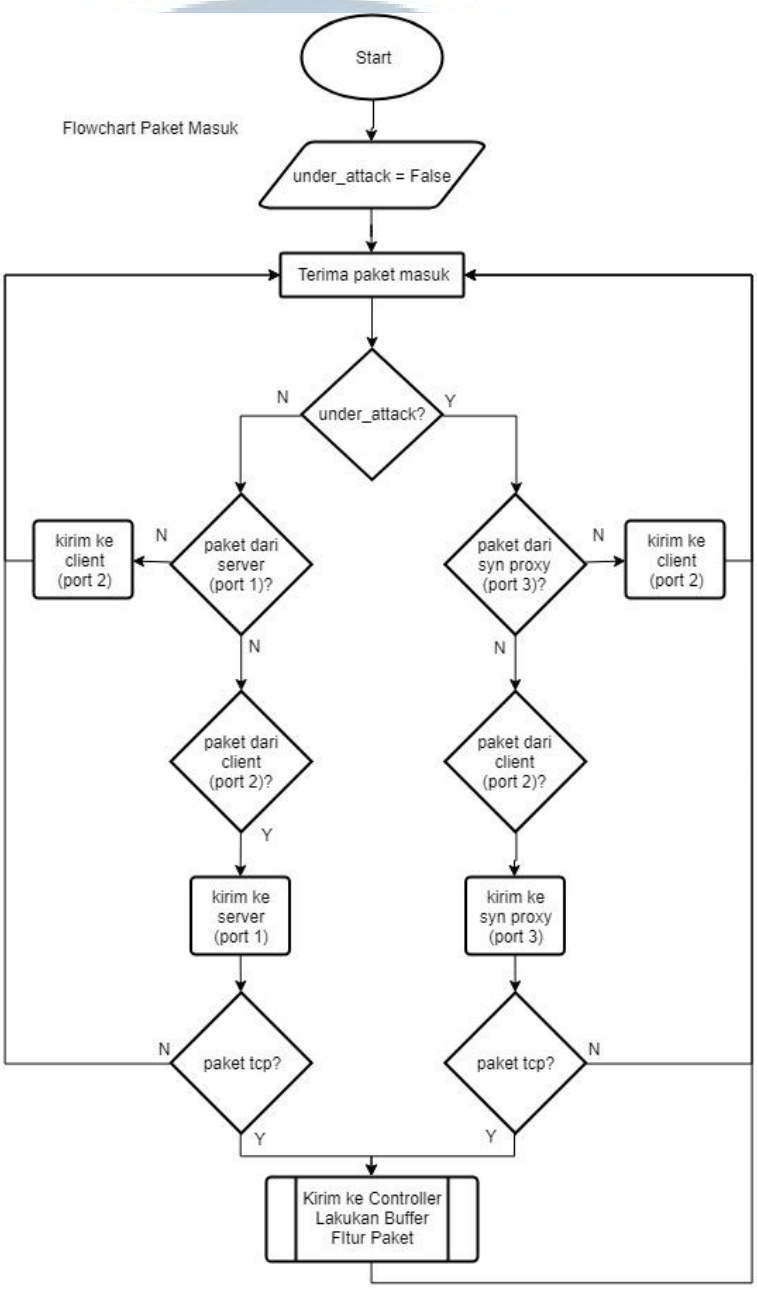
Gambar 3.2 memperlihatkan alur paket yang masuk langsung dikirim ke *server* pada kondisi *traffic* jaringan yang normal.



Gambar 3.3 Alur Paket Masuk pada *Traffic* Jaringan Anomali

Sedangkan pada Gambar 3.3 ditunjukkan alur paket masuk yang dialihkan ke *SYN Proxy server* karena *rule* yang diterapkan pada perangkat jaringan ketika berada dalam kondisi *traffic* anomali dikarenakan serangan *SYN Flood*. Dengan menggunakan *SYN Proxy*, dampak serangan *SYN Flood* pada *server* akan dapat diminimalisir karena *server* hanya akan menerima *legitimate request*.

3.1.1. Diagram Alur Paket Masuk

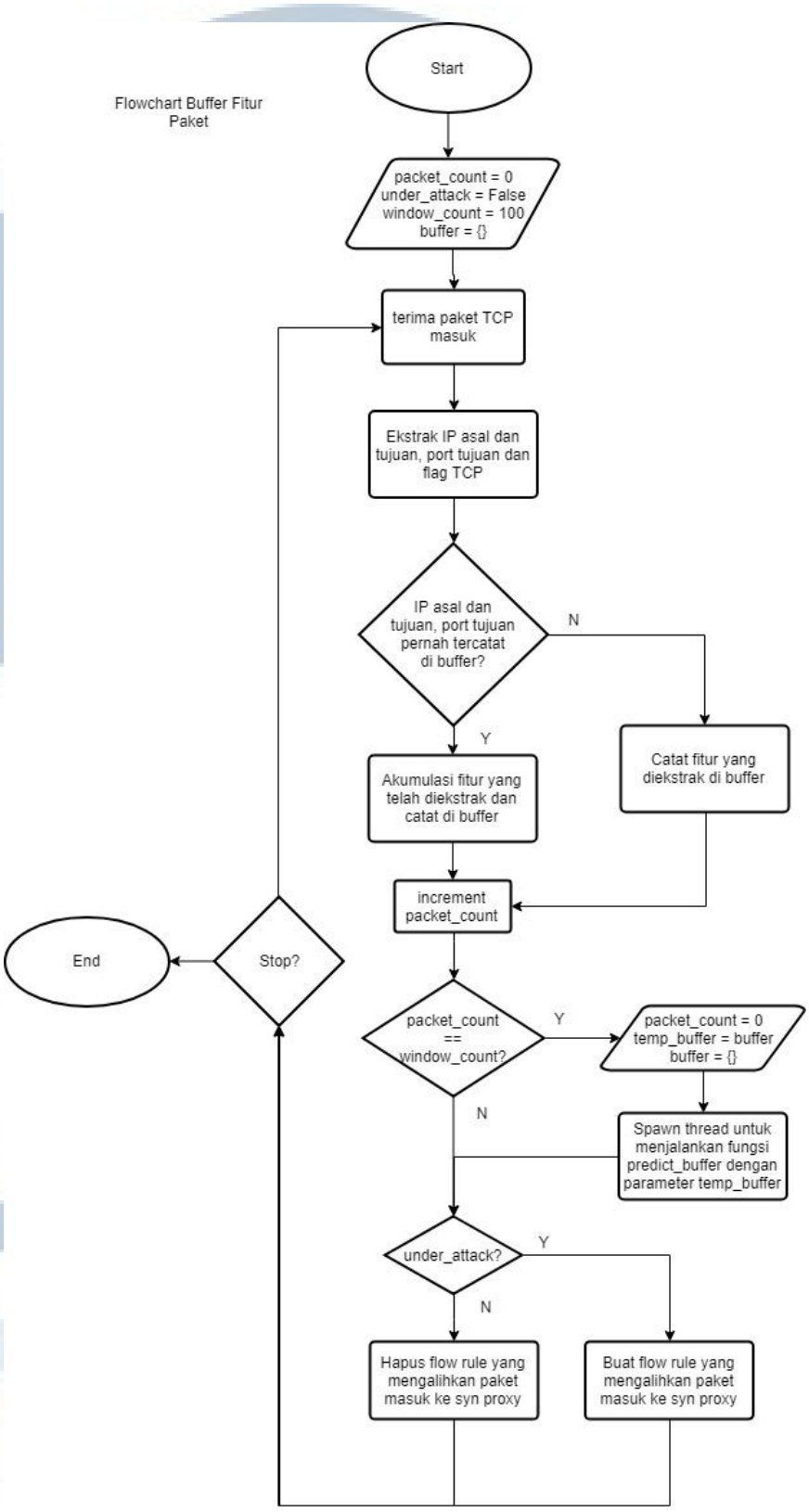


Gambar 3.4 Flowchart Paket Masuk

Gambar 3.4 menunjukkan diagram alur ketika ada paket yang masuk ke *switch*. Dalam kondisi normal, semua paket yang masuk dari *client* akan dikirim ke *server* dan jika paket tersebut menggunakan protokol TCP maka akan dikirim ke *controller* untuk di ekstrak dan diakumulasi fiturnya, sedangkan semua paket

dari *server* akan dikirim ke *client*. Ketika dalam serangan, paket yang masuk dari *client* akan dikirimkan ke *SYN Proxy* dan jika menggunakan protokol TCP juga akan dikirim ke *controller* untuk di ekstrak dan diakumulasi fiturnya, dan semua paket yang masuk dari *SYN Proxy* akan dikirimkan langsung ke *client*.

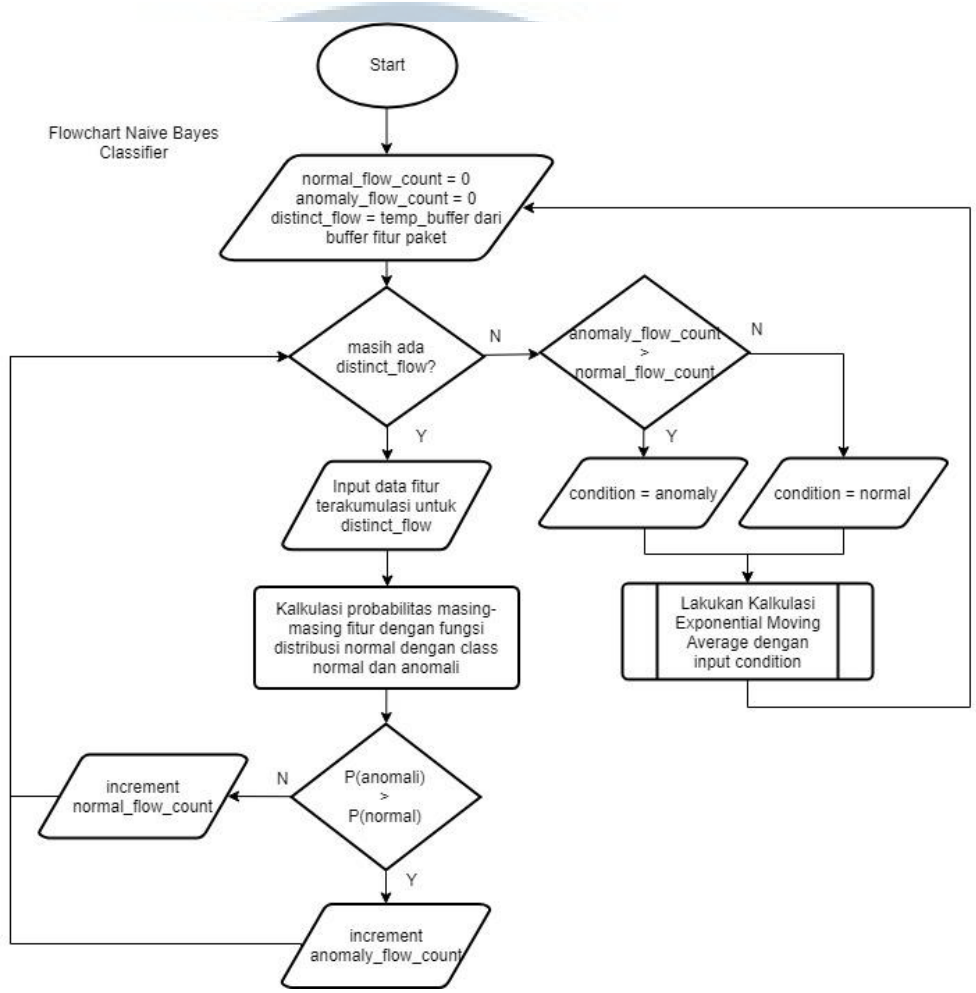
3.1.2. Diagram Alur Buffer Fitur Paket



Gambar 3.5 Flowchart *Buffer* Fitur Paket

Gambar 3.5 menunjukkan diagram alur pada bagian *buffer* fitur paket dalam *Anomaly Detection System*. Pertama-tama, akan diinisialisasi packet_count, alert, window_count, dan *buffer*. Kemudian dari setiap paket TCP yang masuk akan diekstrak fiturnya dan disimpan dalam *buffer* hingga counter mencapai 100. Setiap 100 paket, *buffer* akan dipindahkan ke dalam sebuah variabel penampung yang kemudian dikirim ke sebuah thread yang menjalankan fungsi prediksi serangan sambil terus melakukan *buffer* fitur paket TCP yang datang. Hasil dari prediksi yang telah dimasukkan kedalam modul *exponential moving average* akan menentukan kondisi saat itu, jika dalam serangan maka akan dibuat *flow rule* untuk mengalihkan paket yang masuk ke *SYN Proxy*, sedangkan jika tidak berada dalam serangan maka akan menghapus *flow rule* yang mengalihkan paket ke *SYN Proxy*.

3.1.3. Diagram Alur Naïve Bayes Classifier

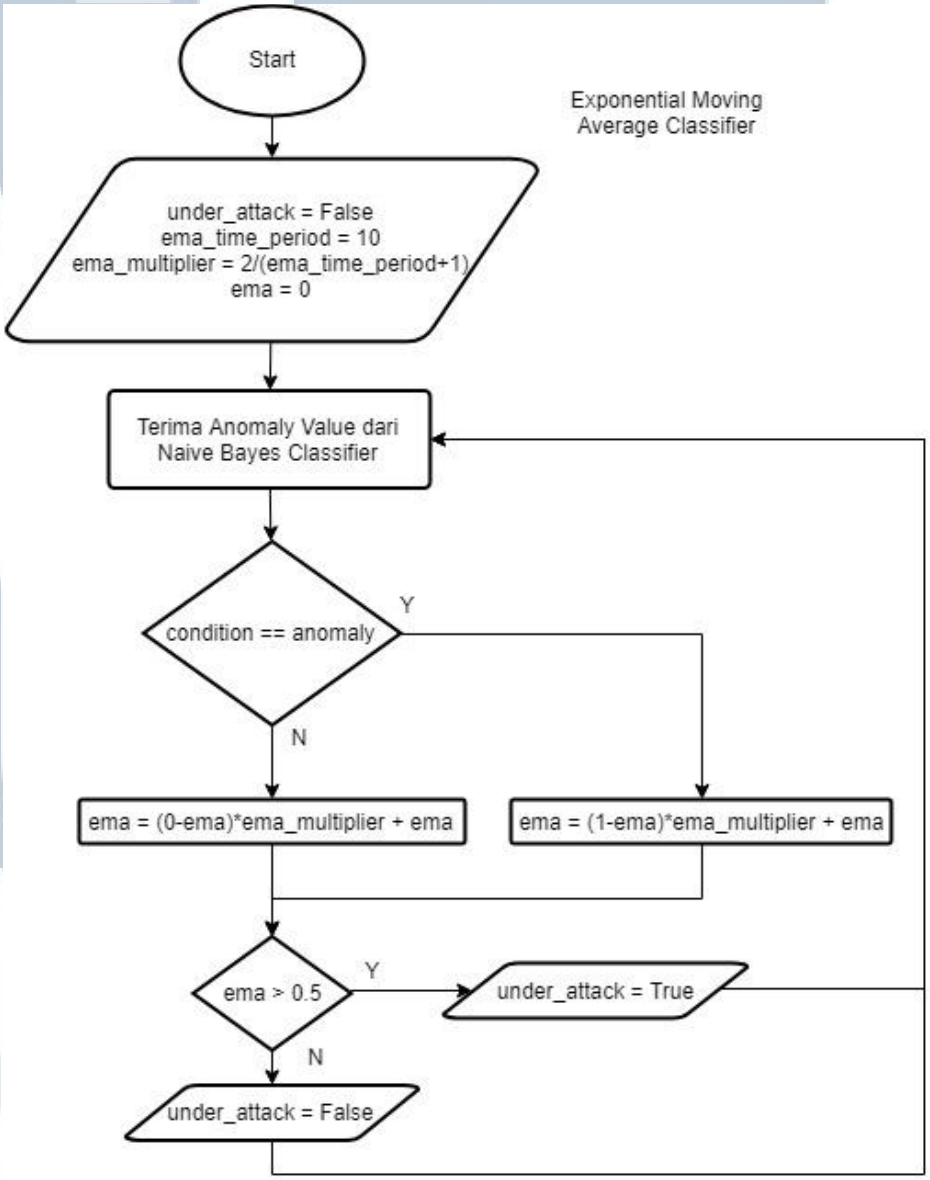


Gambar 3.6 Diagram Alur *Naïve Bayes Classifier*

Gambar 3.6 menunjukkan diagram alur n*aïve bayes classifier*. Pertama inisialisasi variabel yang digunakan dalam *naïve bayes classifier* dan *exponential moving average* yang akan dijelaskan pada Gambar 3.7. Setelah itu *classifier* akan secara berkala menerima input dari *buffer* fitur paket dan dilakukan input fitur yang telah terakumulasi dari masing-masing *distinct flow* kedalam *classifier* untuk dilakukan kalkulasi probabilitas. Jika probabilitas anomali *flow* tersebut lebih besar dari probabilitas normal maka anomaly_flow_count akan ditambah 1, demikian juga sebaliknya, maka normal_flow_count akan ditambah 1. Jika semua *distinct flow* telah

dikalkulasi, maka akan dibandingkan jumlah *flow* anomali dengan jumlah *flow* normal, jika anomali lebih banyak maka dianggap kondisi anomali dan jika normal lebih banyak maka kondisi normal. Variabel condition kemudian akan dikirim ke dalam modul *exponential moving average* untuk dilakukan kalkulasi *ema* sehingga mengurangi kemungkinan *false alarm*.

3.1.4. Diagram Alur Exponential Moving Average



Gambar 3.7 Diagram Alur *Exponential Moving Average*

Gambar 3.7 menunjukkan diagram alur *exponential moving average*. Pertama akan dilakukan inisialisasi nilai awal *ema* dan kemudian nilai tersebut akan digunakan untuk perhitungan *ema* selanjutnya dengan menitikberatkan pada kondisi terbaru sehingga nilai yang didapat akan kecil kemungkinan *false positive* yang diakibatkan *fluktuasi* sesaat. Jika hasil penghitungan *ema* diatas 0.5 maka jaringan dianggap sedang berada dalam serangan dan value under_attack menjadi True dan semua *incoming TCP packets* akan dialihkan melalui *SYN Proxy*, sedangkan sebaliknya, maka jaringan dianggap normal dan value under_attack menjadi False dan *incoming TCP packets* dapat langsung mengakses *server* tanpa melalui *SYN Proxy*.

3.2. Instrumen Penelitian

Penelitian dilakukan dengan menggunakan instrumen perangkat keras dan perangkat lunak berikut:

1. Sebuah laptop dengan prosesor Intel core i7, RAM 8GB dengan OS ubuntu linux yang dijalankan diatas VMWare pada Windows 10 sebagai *OpenFlow Controller*
2. 3 buah pc dengan prosesor Intel core i5, RAM 8GB yang menjalankan Windows10 (salah satunya menjalankan ubuntu linux diatas VMWare sebagai *SYN Proxy*)
3. ZODIAC FX OpenFlow Switch
4. Wireshark