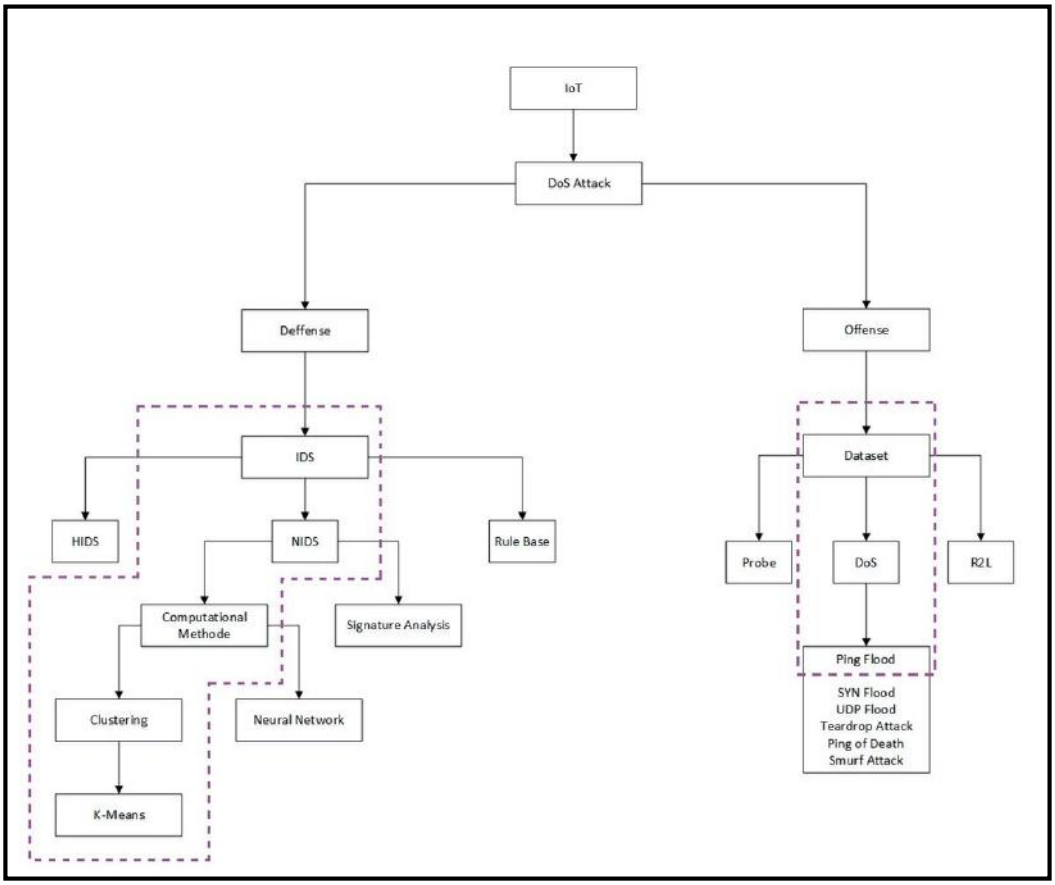
# BAB II. TINJAUAN PUSTAKA

## 2.1 Diagram Konsep Penelitian

Dalam penelitian tugas akhir ini, terdapat beberapa sub materi yang akan dibahas. Sehingga perlu dibuat suatu kerangka konsep secara hirarki untuk menampilkan keseluruhan pembahasan dari penelitian ini. Gambar 2.1 di bawah ini merupakan diagram konsep penelitian yang digunakan. Dalam gambar tersebut terdapat bagian yang diberi garis putus-putus pada sisi *defense* (pertahanan) dan *offense* (penyerang), bagian inilah yang akan dibahas dalam tugas akhir ini.




Gambar 2.1 Diagram Konsep Penelitian

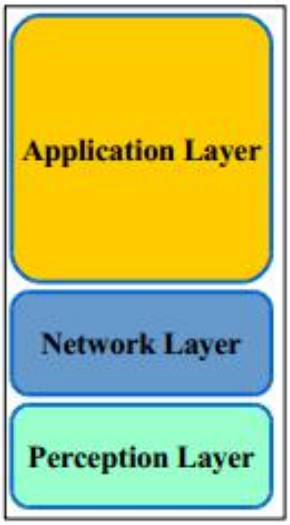
## 2.2 *Internet of Things* (IoT)

*Internet of Things* (IoT) adalah jaringan *hybrid* dari perangkat kecil biasanya

WSN dan internet konvensional. Berbeda dengan WSN biasa, dimana perangkatnya yang kebanyakan dibatasi sumber daya, dan tidak seperti di internet dimana kebanyakan perangkatnya *powerful*, *node* atau *things* pada IoT adalah perangkat *heterogen*. Suatu perangkat IoT bisa berupa bola lampu, *microwave*, bagian mobil, *smartphone*, PC atau laptop, mesin *server* yang kuat atau *cloud* [7] [8]. Fakta bahwa perangkat di IoT sangat *heterogen* dan banyak di antaranya dibatasi sumber daya terhubung secara global membuatnya jauh lebih menantang untuk mengamankan IoT. Ketersediaan IDS dan/atau untuk WSN mungkin tidak cocok untuk melindungi perangkat IoT, karena keduanya *heavyweight* untuk perangkat yang dibatasi atau tidak dikembangkan dalam IoT [7].

Dengan sejumlah penelitian yang telah dilakukan, visi IoT mungkin akan segera menjadi kenyataan. Menurut Gertner, sekitar 25 miliar objek unik yang dapat diidentifikasi diharapkan menjadi bagian dari jaringan komputasi global pada tahun 2020 [2], yang jumlahnya sangat banyak, namun pprevelensi jaringan peranti interkoneksi yang begitu besar akan menimbulkan ancaman keamanan dan privasi baru dan menempatkan semua perangkat pada resiko tinggi terhadap *hacker* karena mereka memegang celah keamanan untuk membuat perangkat bekerja sesuai keinginan mereka.

### 2.2.1 *Architecture* IoT




Gambar 2.2 Arsitektur *3-layer* pada IoT [9]

IoT harus mampu menghubungkan miliaran benda *heterogen* melalui internet, jadi ada kebutuhan kritis akan arsitektur *layer* yang fleksibel [9]. Dari model-model yang diusulkan, model dasar dari IoT adalah model arsitektur 3-*layer* [10] yang terdiri dari *Application Layer, Network Layer*, dan *Perception Layer* [11].

a. *Perception Layer*. Layer ini berisi jenis data yang berbeda pada data sensor seperti RFID, *barcode* atau jaringan sensor lainnya [12]. Tujuan dasar pada layer ini alahah untuk mengenali objek yang unik dan berurusan dengan data yang dikumpulkan dari dunia nyata dengan bantuan sensor masing-masing [9].
b. *Network Layer*. Tujuan dari layer ini adalah untuk mengirimkan kumpulan informasi yang diperoleh dari lapisan persepsi, ke sistem pemrosesan informasi tertentu melalui jaringan komunikasi yang ada seperti *Internet*, *Mobile Network*, atau jaringan terpercaya lainnya [9].
c. *Application Layer*. *Application layer* menyediakan layanan yang diminta oleh pengguna. Contohnya, *application layer* dapat memberikan pengukuran suhu dan kelembaban udara kepada pengguna yang meminta data tersebut. Yang penting dalam *layer* ini pada IoT adalah bahwa *layer* ini memiliki kemampuan untuk menyediakan *smart service* berkualitas tinggi untuk memenuhi kebutuhan pengguna [13][10][14]. Layer ini menyadari berbagai *practical application* pada IoT berdasarkan kebutuhan pengguna dan berbagai jenis industri seperti *Smart Home, Smart Environment, Smart Transport, Smart Hospital*, dan sebagainya [15].

### 2.2.2 Elemen IoT

Memahami blok bangunan IoT membantu untuk mendapatkan wawasan yang lebih baik tentang makna dan fungsi dari IoT. Berikut ini akan membahas elemen utama yang diperlukan untuk mengirim fungsi dari IoT, seperti yang diilustrasikan pada Gambar 2.3, dan Tabel 1 menunjukkan kategori dari elemen-elemen ini beserta contohnya pada setiap kategori.


Gambar 2.3 Elemen IoT [9]

a. ***Identification.*** *Identification* sangat penting bagi IoT untuk memberi nama dan mencocokkan layanan dengan permintaan. Banyak metode *identification* yang tersedia untuk IoT seperti *Electronic Product Codes* (EPC) dan *Ubiquitous Codes* (uCode) [16]. Selanjutnya, menangani objek IoT sangat penting untuk membedakan antara ID objek dan alamatnya. Sebagai tambahan, mengatasi metode objek IoT termasuk IPv6 dan IPv4. membedakan antara identifikasi dan alamat objek sangat penting karena metode identifikasi tidak unik secara global, sehingga membantu mendeteksi objek secara unik. Selain itu, objek di dalam jaringan mongkin menggunakan IP publik dan bukan domain pribadi. Metode identifikasi digunakan untuk memberikan identitas yang jelas untuk setiap objek dalam jaringan [9].

b. ***Sensing.*** Penginderaan *(sensing)* IoT berarti mengumpulkan data dari benda-benda terkait di dalam jaringan dan mengirimkannya kembali ke *warehouse, database*, atau *cloud.* Data yang terkumpul dianalisis untuk mengambil tindakan spesifik berdasarkan layanan yang dibutuhkan. Sensor IoT bias berupa *smart sensor,* actuator atau perangkat penginderaan yang dapat dipakai. *Single Board Computer* (SBC) yang terintegrasi dengan sensor dan *built-in* TCP/IP dan fungsi keamanan biasanya digunakan untuk mewujudkan produk IoT seperti Arduino Yun, Raspberri PI, BeagleBone Black, dan sebagainya. Perangkat seperti itu biasanya terhubung ke portal manajemen pusat untuk menyediakan data yang dibutuhkan oleh pelanggan [9].

c. ***Communication.*** Teknologi komunikasi IoT menghubungkan objek *heterogen* bersama untuk memberikan *smart service* tertentu. Biasanya, *node* IoT harus beroperasi dengan menggunakan daya rendah di hadapan *link* komunikasi yang *lossy* dan berisik. Contoh dari protokol komunikasi yang digunakan untuk IoT adalah *WiFi*, Bluetooth, IEEE 802.15.4, Zwave, dan LTE-*Advanced.* Beberapa teknologi komunikasi yang spesifik juga digunakan oleh RFID, *Near Field*

*Communication* (NFC) dan *ultra-wide bandwidth* (UWB). RFID adalah teknologi pertama yang mengetahui konsep M2M (RFID *tag* dan *reader*). *Tag* RFID merupakan chip atau label sederhana yang melekat untuk memberikan identitas objek. RFID *reader* mentransmisikan sinyak kueri ke *tag* dan menerima sinyal yang dipantulkan dari *tag,* yang kemudian beralih ke *database*. *Database* terhubung ke pusat pemrosesan untuk mengidentifikasi objek berdasarkan sinyal yang dipantulkan dalam jarak 10 cm sampai dengan 200 m [9]. Protokol NFC bekerja pada pita frekuensi tinggi pada 13,56 MHz dan mendukung kecepatan data hingga 424 kbps. Rentang yang berlaku adalah sampai 10 cm dimana komunikasi antara pembaca aktif dapat terjadi [17]. Teknologi komunikasi UWB dirancang untuk mendukung komunikasi dalam area jangkauan rendah dengan menggunakan energi rendah dan bandwidth tinggi yang mana aplikasi untuk menghubungkan sensor telah meningkat baru-baru ini [9]. Teknologi komunikasi lainnya adala *WiFi*, yang menggunakan gelombang radio untuk bertukar data antar benda dalam jarak 100 m [9]. *WiFi* memungkinkan *smart device* untuk berkomunikasi dan bertukar informasi tanpa menggunakan router dalam beberapa konfigurasi *ad hoc. Bluetooth* menghadirkan teknologi komunikasi yang digunakan untuk bertukar data antar perangkat jarak dekat menggunakan radio gelombang pendek untuk meminimalkan konsumsi daya [18]. Akhir-akhir ini, *Bluetooth special interest group* (SIG) mencitakan *Bluetooth* 4.1 yang menyediakan *Bluetooth Low Energy* serta konektivitas dengan kecepatan tinggi dan IP untuk mendukung IoT. *Long-Term Evolution* (LTE) sebenarnya adalah *standard wireless communication* untuk pengririman data dengan kecepatan tinggi antara *mobile phone* berbasisteknologi jaringan GSM/UMTS. Hal ini dapat mencakup perangkat yang melaju cepat dan menyediakan layanan *multicasting* dan *broadcasting.* LTE-A (LTE *Advanced*) adalah versi yang lebih baik dari LTE termasuk perpanjangan *bandwidth* yang mendukung hingga 100 MHz, *downlink* dan multiplexing spasial *uplink,* cakupan yang diperluas, *throughput* yang lebih tinggi dan *latency* yang lebih rendah [9].

***d. Computation.*** Unit-unit pengolahan (seperti mikrokontrolerdan mikroprosesor) dan aplikasi perangkat lunak mewakili “otak” dan kemampuan

komputasi dari IoT. Berbagai platform perangkat keras dikembangkan untuk menjalankan aplikasi IoT seperti Arduino, UDOO, FriendlyARM, Intel Galileo, Raspberry Pi, Gadgeteer, BeagleBone, Cubieboard, Z1, WiSense, Mulle, dan T-Mote Sky. Selanjutnya, banyak platform perangkat lunak digunakan untuk menyediakan fungsionalitas IoT. Diantara platform-platform ini, *Real Time Operating System* (RTOS) sangat penting karena dijalankan untuk seluruh waktu aktivasi perangkat. Ada banyak *Real Time Operating System* (RTOS) yang merupakan kandidat untuk pengembangan aplikasi IoT berbasis RTOS. Seperti RTOS Contiki yang telah banyak digunakan dalam skenario IoT. Contiki memiliki simulasi yang disebut Cooja yang memungkinkan peneliti dan pengembang untuk mensimulasi dan meniru IoT dan *Wireless Sensor Network* (WSN) [19].

***e.*** ***Services.*** Secara keseluruhan, layanan IoT dapat dikategorikan atas 4 kelas [20]: *Information Aggregation, Identity-related Services, Collaborative Aware Services* dan *Ubiquitous Services. Identity-related Services* adalah layanan paling dasar dan paling penting yang digunakan pada tipe layanan lainnya. Setiap aplikasi yang perlu membawa hal-hal dari dunia nyata ke dunia maya harus mengidentifikasi objek tersebut. Layanan *Information Aggregation* mengumpulkan dan meringkas pengukuran sensorik mentah yang perlu diproses dan dilaporkan ke aplikasi IoT. *Collaborative Aware Services* bertindak di atas layanan *Information Aggregation* dan menggunakan data yang telah dikumpulkan untuk membuat keputusan dan bertindak sesuai. *Ubiquitous Services* bertujuan untuk menyediakan *Collaborative Aware Services* kapanpun dan dimanapun dibutuhkan, serta untuk siapa saja yang membutuhkan. Tujuan akhir seluruh aplikasi IoT adalah untuk mencapai tingkat *Ubiquitous Services.* Namun tujuan akhir ini tidaklah mudah untuk dicapai karena ada banyak kesulitan dan tantangan. Sebagian besar aplikasi yang ada menyediakan layanan *identity-related*, *Information Aggregation*, dan *collaborative aware. Smart healthcare* dan *smart grid* termasuk kategori *information aggregation. Smart home, smart buildings, intelligent transportation system,* dan *industrial automation* lebih dekan dengan kategori *collaborative aware* [9].

***f.*** ***Semantic.*** Semantik dalam IoT mengacu pada kemampuan untuk mengekstrak pengetahuan dengan cerdas oleh mesin yang berbeda untuk menyediakan layanan yang dibutuhkan. Pengekstrakan *knowledge* termasuk penemuan dan penggunaan sumber daya dan informasi permodelan. Selain itu, termasuk mengenali dan menganalisis data untuk memahami keputusan yang tepat untuk memberikan layanan yang tepat [21]. Dengan demikian, semantic mewakili otak IOC dengan mengirimkan permintaan ke sumber daya yang tepat [9].

**Tabel 1**

Elemen IoT [9]

| **IoT Elements** | | **Samples** |
|---|---|---|
| **Identification** | Naming | EPC, uCode |
| | Addressing | IPv4, IPv6 |
| **Sensing** | | Smart Sensors, Wearable sensing devices, Embedded sensors, Actuators, RFID tag |
| **Communication** | | RFID, NFC, UWB, Bluetooth, BLE, IEEE 802.15.4, Z-Wave, WiFi, WiFiDirect, LTE-A |
| **Computation** | Hardware | SmartThings, Arduino, Phidgets, Intel Galileo, Raspberry Pi, Gadgeteer, BeagleBone, Cubieboard, Smart Phones |
| | Software | OS (Contiki, TinyOS, LiteOS, Riot OS, Android); Cloud (Nimbits, Hadoop, etc.) |
| **Services** | | Identity-related (shipping), Information Aggregation (smart grid), CollaborativeAware (smart home), Ubiquitous (smart city) |
| **Semantic** | | RDF, OWL, EXI |

## 2.3 *Denial of Services* (DoS)

*Denial of Services* (DoS) adalah suatu upaya yang dilakukan oleh penyerang untuk mengkonsumsi sumber daya atau *bandwidth* pengguna. Serangan semacam ini, jika dilakukan dari berbagai *node* secara bersamaan dapat disebut sebagai

serangan *Distributed Denial of Services* (DDoS) [22].

### 2.3.1 Teknik Serangan DoS

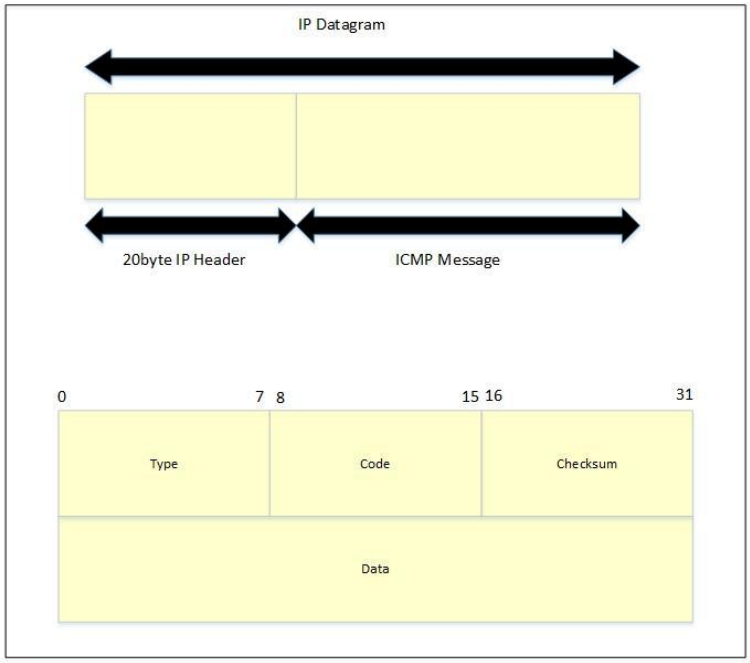
Pada serangan DoS, *attacker* berusaha untuk mencegah pengguna untuk menggunakan suatu layanan. Ada beberapa bentuk serangan DoS, tapi ada dua bentuk serangan DoS yang umum, yaitu:

1. Pertama, serangan itu berusaha untuk membanjiri *service* atau jaringan tidak menghabiskan *bandwidth* sehingga *request* dari pengguna tidak sampai pada *service* atau jaringan tersebut. Hal ini disebut juga sebagai *flooding.*
2. Kedua, serangan itu berusaha untuk merusak *hardware* atau *software* agar *service* atau mesin itu *unavailable* untuk pengguna. Beberapa target dari serangan ini adalah *server, DNS look up,* dan *routing devices* [23].

Pada umumnya serangan DoS dapat dikelompokkan kedalam dua kategori umum, kategori pertama yaitu *flooding attack* yang mengirimkan banyak paket kepada target untuk memenuhi CPU, memori atau jaringan target. Salah satu contoh dari *flooding attack* adalah *ICMP flooding*. Kategori kedua yaitu *logic attack* yang memanfaatkan kelemahan sistem untuk menyebabkan kerusakan sistem [22].

### 2.3.2 *Internet Control Message Protocol* (ICMP) *Flood*

ICMP biasanyan digunakan dalam teknologi jaringan, untuk mendiagnosa, ataupun melaporkan kesalahan, namun kini protokol ini digunakan *attacker* untuk menyerang korban dengan mengirim *payloads.* Dalam Gambar 2.4 pada dua kolom pertama, menentukan apakah suatu paket ICMP merupakan sebuah pesan permintaan atau pesan *error.* Pesan *error* ICMP tidak dikirim sebagai tanggapan terhadap ICMP *error.* Saat ICMP *error* dikirim, maka akan mengirim *IP header* dan *datagram* yang menyebabkan *error*. Jadi unit penerima dapat mengasosiasikan kesalahan dengan proses. Maka saat tipe 0 *(echo reply)* terkirim, balasannya bukan lagi tipe 8 *(echo request). Field* terakhir pada format ICMP ialah *checksum. Field* ini digunakan untuk memeriksa kesalahan. Sebelum pesan ICMP ditransmisikan, *checksum* dihitung dan dimasukkan kedalam *field.* Jadi pada bagian penerima, *checksum* dihitung lagi dan diverifikasi pada *checksum field.* Jika ada hal yang tidak sesuai ditemukan, maka itu menegaskan terdapat kesalahan atau telah terjadi perubahan [24].




Gambar 2.4 Format Paket ICMP [24]

Beberapa contoh dari ICMP *flood* yaitu *smurf attack, ping flood, ping of death.* Ada beberapa mekanisme pertahanan yang cukup efektif untuk melawan *smurf* dan *ping attack*, namun metode ini dapat menyebabkan kerusakan pada jaringan LAN kecil.

1. *smurf attack*

   Serangan ini membanjiri *bandwidth* korban. Dalam hal ini, *attacker* mengirim permintaan ICMP dalam jumlah besar ke alamat *broadcast* dan menggunakan alamat *spoofing* untuk berpura-pura sebagai pengguna [25]. Sehingga membanjiri alamat pengguna tersebut dan dengan cepat menghabiskan *bandwidth*, menghalangi paket pengguna untuk sampai ke tujuan. Biasanya digunakan untuk menyerang mesin atau server tanpa keamanan ISP.

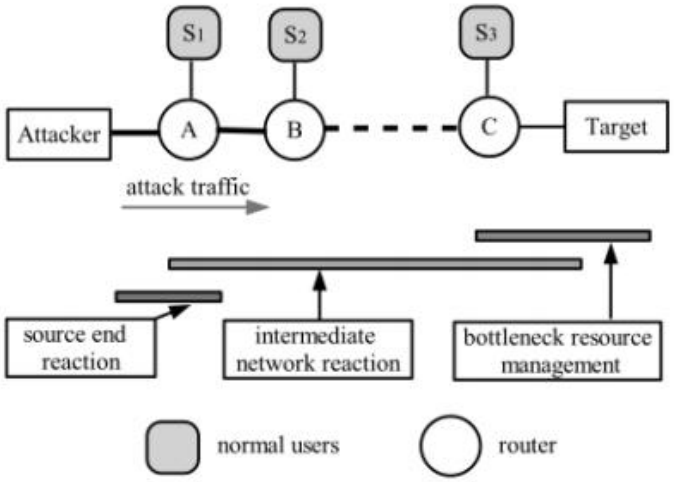
2. *ping flood*

   Hampir sama dengan *smurf attack,* dimana *bandwidth* jaringan korban akan dibanjiri dengan banyak paket ping. Pada *hosts* seperti Unix dapat menggunakan *command* "ping". Mudah dilakukan dengan akses *bandwidth* yang lebih besar dari pada korban.

3. *ping of death*

   Jika *smurf* dan *ping flood* membanjiri *bandwidth* jaringan korban, yang menyebabkan pelayanan yang melambat, *ping of death* dapat merusak sistem

korban. Dalam metode ini, penyerang mengirim paket ping yang *corrupt* ke korban yang dapat merusak sistem. Tidak seperti serangan yang lebih spesifik seperti serangan *remote to local*, serangan DoS bertujuan untuk merusak korban sebanyak mungkin. Gambar 2.5 merupakan model sederhana dari skema serangan DoS, dimana garis tipis menunjukkan *link bandwidth* rendah dan garis tebal menunjukkan *link bandwidth* tinggi. Serangan DoS mulai berlaku ketika sumber daya *bottleneck* tercapai.




Gambar 2.5 Model Skema Serangan DoS [23]

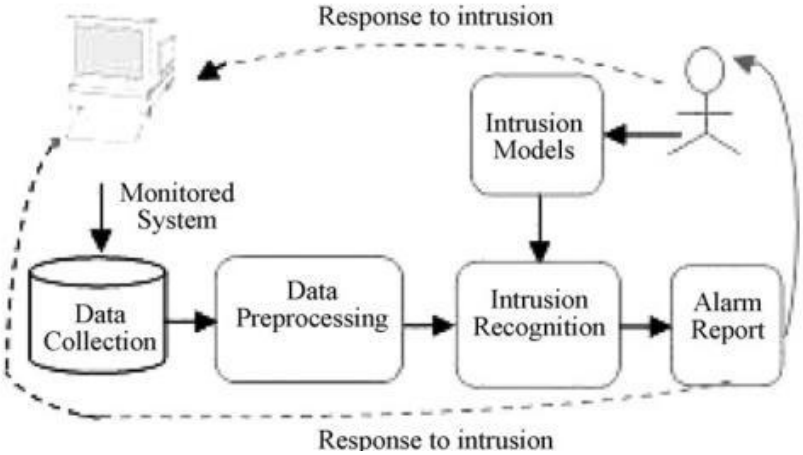
## *2.4 Intrusion Detection System* (IDS)

IDS adalah mekanisme untuk mendeteksi serangan terhadap sistem atau jaringan dengan menganalisis aktivitas di jaringan atau dalam sistem itu sendiri. Setelah serangan terdeteksi, IDS dapat mencatat informasi tentang hal itu dan/atau memberi alarm. Secara garis besar, Mekanisme pendeteksian pada IDS berbasis *signature* Atau berbasis *anomaly*. pendeteksian berdasarkan *signature* sesuai dengan perilaku saat ini dari jaringan terhadap pola serangan yang telah ditentukan sebelumnya. *Signature* telah dikonfigurasikan sebelumnya dan disimpan di perangkat dan masing-masing *signature* cocok dengan serangan tertentu. Secara umum teknik berbasis *signature* lebih sederhana untuk digunakan [8].

### 2.4.1 Arsitektur *Intrusion Detection System* (IDS)

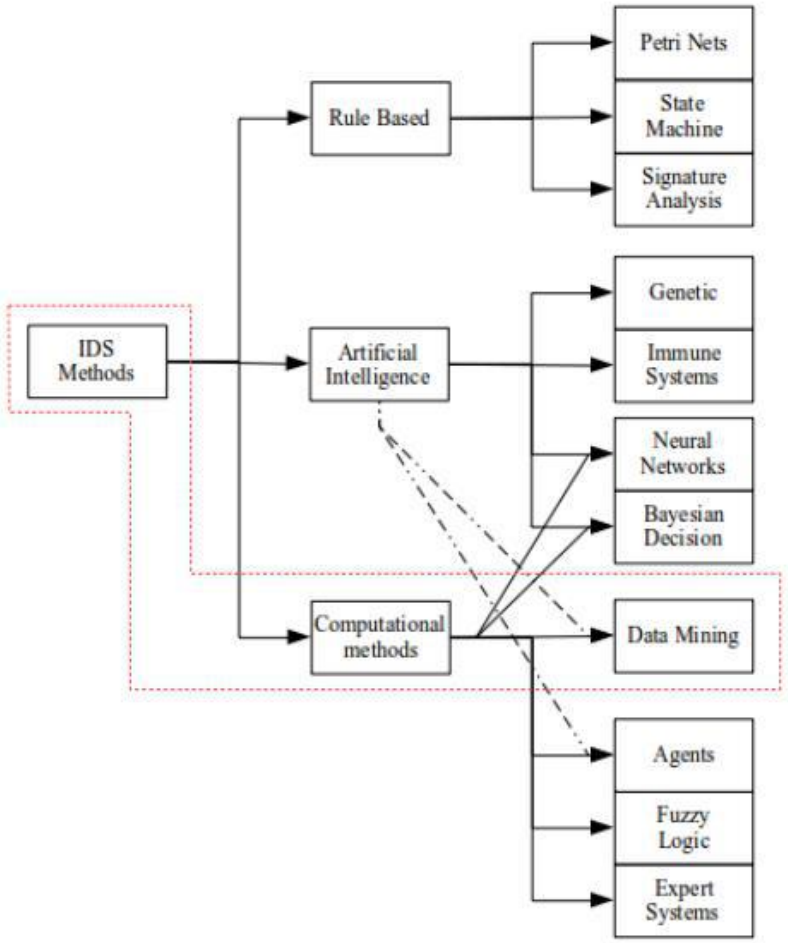
*Intrusion Detection System* (IDS) memonitor aktivitas lingkungan tertentu

dan menentukan suatu aktivitas sebagai aktivitas yang berbahaya atau normal bedasarkan pada integritas sistem, kerahasiaan, dan ketersediaan sumber informasi. Ketika membangun suatu IDS perlu mempertimbangkan pengumpulan data, *data pre-processing, intrusion recognition, reporting*, dan *response* [26]. Berikut ini merupakan gambar organisasi umum dari IDS.




Gambar 2.6 Organisasi Umum IDS [26]

Dalam IDS, metode yang sering digunakan pada penelitian dikelompokkan menjadi tiga, yaitu *Rule Based Method, Artificial Inteliigence,* dan *Computational Method.* Pada metode *Rule Based* terdiri atas *Signature Analysis, State Machine,* dan *Place Transition Net.* Sedangkan pada *Artificial Intelligence* terdapat *Bayesian Decision, Neural Networks, Immune System, Genetic.* Serta pada *Computational Methods* tediri dari *Expert System, Fuzzy Logic, Agents,* dan *Data Mining* [26].




Gambar 2.7 Metode IDS [26]

### 2.4.2 IDS Diklasifikasikan Berdasarkan Penempatan *Deployment*

IDS berdasarkan penempatan *deployment* diklasifikasikan kedalam dua kelompok, yaitu *Network based Intusion Detection system* (NIDS) dan *Host based Intrusion Detection System* (HIDS) [27]. NIDS lebih berfokus pada keamanan infrastruktur jaringan, yang mana bertujuan untuk mengawasi *traffic* jaringan menuju *host.* Fungsi NIDS ialah untuk menganalisis lalu lintas jaringan untuk mendeteksi serangan. Sedangkan HIDS berfungsi untuk mengawasi aktivitas sistem *log*, *host*, informasi program aplikasi, dan sistem operasi [26].

### 2.4.3 IDS dengan *Computational Methodes*

Tujuan dari penelitian menggunakan *Computational Inteligent* ialah untuk adaptasi, pembelajaran, atau mengevaluasi algoritma dalam pembuatan suatu program. Algoritma ini memungkinkan untuk mendeteksi keruasakan sistem

dengan cepat dan beroperasi secara *real time* [28]. Penerapan *Computational Methods* pada IDS adalah topik yang terbilang baru pada *computer science*, memanfaatkan metode komputasi seperti *pattern recognition, machine learning,* dan pendekatan statistik [29]. Yang termasuk kedalam *Computational Methods* salah satunya adalah *Data Mining. Data Mining* merujuk pada teknik yang menggunakan proses ekstraksi data yang sebelumnya tidak diketahui [26]. *Data Mining* adalah bidang ilmu yang sering membahas mengenai pola suatu data. Data yang berukuran besar dapat dianalisis oleh *Data Mining* dengan membuat *rules*, model atau pola tertentu untuk mengenali data baru yang tidak tersimpan pada *database* [29]. Selain itu, *Data Mining* juga berperan untuk menemukan kegiatan *anomaly* yang tidak diketahui, mengidentifikasi *bad sensor signature* dan *false alarm,* serta menghilangkan paket normal dalam *alarm* dan berfokus kepada analisa data serangan. *Data Mining* yang dapat digunakan pada IDS adalah *Clustering, Clasification, Sequence,* dan *Assocition* [28].

## 2.5 Algoritma *Clustering K-means*

*Clustering* adalah suatu klasifikasi tanpa pengawasan yang kelasnya belum ditentukan, dalam pengelompokan, sampel dibagi kedalam kategori anggota yang sama dan disebut *cluster* [29]. *Clustering* juga merupakan masalah mendasar dalam analisis data yang muncul dalam berbagai bidang, seperti pengenalan pola, *machine learning,* bioinformatika dan pengolahan citra [30]. Algoritma ini bertujuan untuk mempartisi suatu kumpulan kedalam kelompok homogen, yaitu kemiripan *intra-cluster* yang tinggi dan kemiripan *inter-cluster* yang rendah, menurut beberapa tujuan pengelompokan .

Salah satu algoritma *clustering* yang banyak dipelajari adalah *K-means*, yang mengurangi jumlah *intra-cluster* yang berbeda. Kesederhanaan dan ketangkasannya telah menjadikan algoritma ini sebagai cara yang terkenal untuk melakukan pengelompokan di berbagai bidang ilmu pengetahuan. *K-means* adalah metode dengan model *centroid,* yaitu suatu model yang membuat *cluster* menggunakan *centroid. Centroid* merupakan titik tengah dari sebuah *cluster* yang berisi nilai. *Centroid* juga digunakan untuk menghitung jarak objek suatu data. Objek data dapat disebut termasuk kedalam *cluster* jika memiliki jarak terdekat

dengan *centroid cluster* [29].

Aturan umum untuk mencari partisi K dengan optimal secara lokal ialah dengan memindahkan titik dari *cluster* 1 ke *cluster* lainnya [31]. Algoritma *K-means* pada umumnya dilakun seperti berikut [28]:

1. Tentukan nilai *k* sebagai jumlah *cluster* yang ingin dibentuk.
2. Bangkitkan *k centroid* sebagai titik pusat *cluster.*
3. Hitung jarak setiap objek ke masing-masing *centroid* menggunakan rumus *Euclidean Distance :*

$$d(x_i,u_i)=\sqrt{(x_i-u_i)^2} \quad \ldots. \quad (2.1)$$

4. Mengelompokkan data berdasarkan jarak minimum dengan *centroid.*
5. Memperbarui nilai *centroid,* nilai *centroid* didapat dari rata-rata *cluster.*
6. Melakukan perulangan dari langkah 2 sampai 5, sehingga tiap *cluster* tidak ada yang berubah.

## 2.6 *Snort*

*Snort* merupakan suatu aplikasi *single threaded* yang mana menggunakan protokol TCP/IP pada system keamanan maupun PC. *Snort* dapat bekerja dalam empat mode, yaitu *Intrusion Prevention System* (IPS), *Network Intrusion Detection System* (NIDS), *packet logger,* dan *sniffer.* Mode *sniffer* berfungsi untuk membaca lalu lintas jaringan. *Packet logger* mencatat seluruh lalu lintas jaringan dengan *log-in* ke file tersebut menggunakan *file log. Mode* IPS berfungsi untuk menangkap paket yang diperoleh dari IP *table* bukan Libpcap (*library* yang digunakan oleh snort). Mode ini juga dapat melakukan tindakan pencegahan terhadap lalu lintas yang berbahaya dalam jaringan. Sedangkan mode NIDS berfungsi untuk menganalisis lalu lintas jaringan. Penggunaan metode NIDS diperlukan *setup* terhadap *rules* untuk membedakan paket normal dan paket serangan sehingga dapat mendeteksi serangan.

*Rules snort* mempunyai tiga prioritas *attack* berdasarkan *classtype,* yaitu *high* (1), *medium* (2), *low* (3), *dan very low* (4). Dimana semakin tinggi tingkat serangan maka semakin rendah nilai prioritasnya. Tabel berikut menunjukkan prioritas *rules* pada *snort* yang dikelompokkan berdasarkan *classtype* [29].

**Tabel 2**

Prioritas *Rules* pada *Snort*

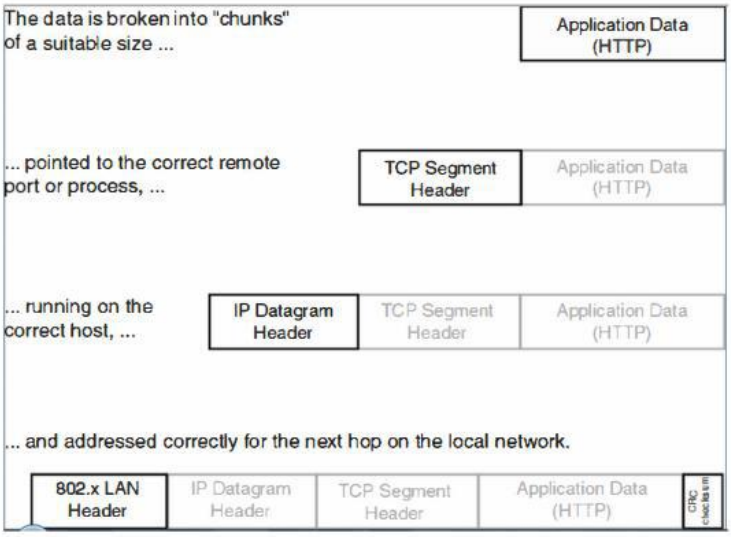
| *Priority* | *Classtype* |
|---|---|
| *High* | *attempted-admin, attempted-user, inappropriate-content, policy violation, shellcode-detect, successful admin, successful-user, Trojan activity, unsuccessful-user, webapplication attack* |
| *Medium* | *attempted-dos, attempted-recon, bad-unknown, defaultlogin-attempt, denial-of-service, misc-attack, non-standardprotocol, rpc-portmap-decode, successful-dos, successfulrecon-largescale, successful-recon-limited, suspicious-login, system-call-detect, unusual-client-port-connection, webapplication-activity* |
| *Low* | *icmp-event, misc-activity, network-scan, not-suspicious, protocol-command-decode, string-detect, unknown* |
| *Very low* | *tcp-connection* |

## 2.7 *WiFi*

Teknologi komunikasi IoT menghubungkan objek *heterogeneous* untuk menghasilkan *specific smart services.* Biasanya, *node* IoT harus beroperasi dengan menggunakan daya yang rendah pada *link* komunikasi *lossy* dan *noisy*. Contoh protokol komunikasi yang digunakan untuk IoT adalah *WiFi*, Bluetooth, IEEE 802.15.4, Z-wave, dan LTE-Advanced [9]. *WiFi* didasarkan pada spesifikasi IEEE 802.11, kecepatannya bias mencapai 11 Mbps dengan jarak hingga 100 meter, namun memerlukan daya yang besar, yaitu 400mA pada operasi *standby* [32]. *WiFi* memungkinkan *smart devices* untuk berkomunikasi dan bertukar informasi tanpa menggunakan router di beberapa konfigurasi ad hoc [9].

Struktur paket data *WiFi* terdiri dari protocol layer aplikasi yang digunakan untuk menyediakan akses layanan TCP/IP. Protokol ini termasuk *Dinamic Host Configuration Protocol* (DHCP), *Domain Name System* (DNS), dan *Hypertext Transfer Protocol* (HTTP). Layer komunikasi inter-host berfungsi yang untuk membuat sesi koneksi *connection-oriented* atau *connectionless broadcast*. Protokol

pada layer ini *Transmission Control Protocol* (TCP) dan *User Datagram Protocol* (UDP) [32].

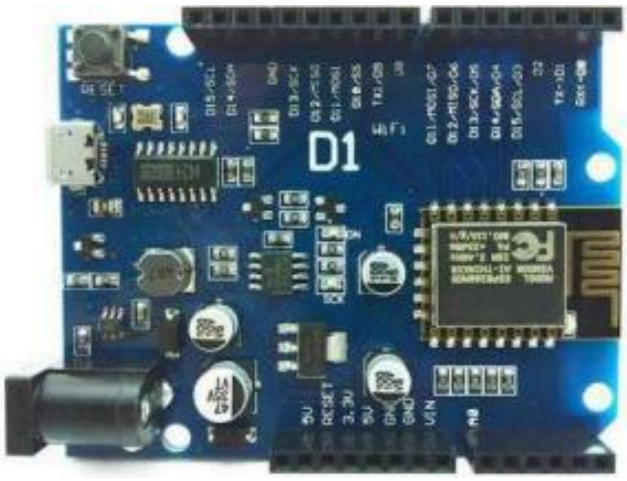



Gambar 2.8 Struktur data Paket *WiFi* [32]

## 2.8 Perangkat pada *Node*

Pada tugas akhir ini, perangkat yang menyusun *nodes* adalah *Wemos D1* dan sensor. Setiap *node* memiliki sensor yang berbeda, sensor-sensor yang digunakan diantaranya adalah DHT22, *Soil Moisture,* dan MQ2.

### 2.8.1 *Wemos D1*




Gambar 2.9 *Wemos D1 Board* [33]

*Wemos D1* adalah *mini wifi board* berbasis ESP-8266EX, dengan 11 pin *input/output* digital, dimana semua pin memiliki *interrupt/pwm/12C/one-wire* yang

mendukung (kecuali D0). Berikut ini merupakan tabel fungsi dan spesifikasi WemosD1 [33]:

**Tabel 3**

Fungsi *Wemos D1*

| Pin | Fungsi | Pin ESP8266 |
|---|---|---|
| **TX** | TXD | TXD |
| **RX** | RXD | RXD |
| **AO** | Analog I/P, MAX 303V I/P | AO |
| **DO** | IO | GPIO 16 |
| **D1** | IO, SCL | GPIO 5 |
| **D2** | IO, SDA | GPIO 4 |
| **D3** | IO 10K, pull up | GPIO 0 |
| **D4** | IO 10K, pull up | GPIO 2 |
| **D5** | IO, SCK | GPIO 14 |
| **D6** | IO, MISO | GPIO 12 |
| **D7** | IO, MOSI | GPIO 13 |
| **D8** | IS, LOK pull down, SS | GPIO 15 |
| **G** | GROUND | GND |

**Tabel 4**

Spesifikasi *Wemos D1*

| Spesifikasi Teknikal | |
|---|---|
| **Microcontroller** | ESP-8266 EX |
| **Operating voltage** | 3.3V |
| **Digital I/O pins** | 1 (max input 3.2V) |
| **Analog input pins** | 80Mh/160MH |
| **Clock speed** | 4m byte |
| **Flash** | 3402mm |
| **Length** | 2506mm |

### 2.8.2 MQ2

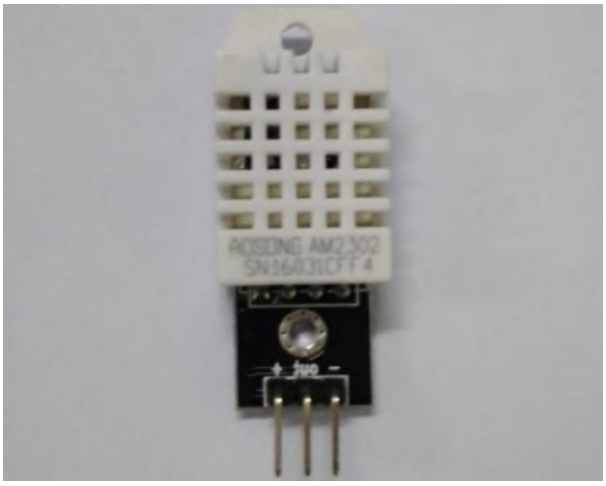
Asap adalah kumpulan udara, partikel cair, dan gas yang dipancarkan saat kumpulan tersebut mengalami pembakaran bersamaan dengan jumlah udara yang masuk atau dicampur ke dalam massa [34]. Untuk pengukuran jumlah asap, dapat menggunakan sensor MQ2 [34][35]. Pada dasarnya modul sensor gas atau asap memiliki pemanas kecil di dalamnya dan memiliki sensor sensitif kimia untuk mendeteksi berbagai gas [35]. MQ2 terdiri dari SnO2 sebagai bahan sensitif, karena sangat efisien mengukur jumlah asap yang ada di lokasi tertentu [34]. sebagai contoh, modul sensor MQ2 dapat membedakan gas-gas *Liquefied Petroleum Gas* (LPG), *Carbon Monoxide* (CO), *Hydrogen* (H2), *Methane* (CH4), *Butane* (C4H10), asap, *Propane* (C3H8) dan *Alcohol* (-OH). Sensor ini terkenal karena ketelitiannya, waktu respon yang cepat, kesensitifannya, tahan uji, awet, dan biaya yang efektif. *Output* dari sensor ini adalah resistensi analog, yang dilekatkan dengan resistor dan dihubungkan dengan *Analog to Digital Converter* (ADC). Dengan membaca tingkat resistensi, kebocoran gas dapat dideteksi. Sensor MQ2 dapat mendeteksi gas 200 ppm sampai dengan 10.000 ppm [35].

Gambar 2.10 Sensor Gas MQ2 [35]

### 2.8.3 DHT22

Sensor ini terhubung langsung dengan udara luar melalui saluran kecil di sisi bawah dekat kabel. Karena saluran kecil tersebut mengisolasi sensor dari volume udara dalam kotak, latensi terhadap perubahan kelembaban relating berkurang [33]. Menurut [36] sensor DHT22 terbukti dapat diandalkan dan stabil. Output dari DHT22 adalah sinyal digital yang dikalibrasi yang dapat dihubungkan langsung ke pin port Arduino Uno. Sensor ini menggunakan teknik pengumpulan

sinyal digital eksklusif dan teknologi penginderaan kelembaban yang mengkalibrasi secara otomatis. Dengan ukurannya yang kecil, konsumsi daya rendah, dan kemampuan untuk berfungsi dalam semua jenis *harsh application occasion*, menjadikan DHT22 cocok untuk digunakan sebagai sensor pemantau kekeringan.




Gambar 2.11 DHT22

### 2.8.4 FC-28

Gambar 2.12 Sensor *Soil Moisture* (FC28) [33]

Terlepas dari pentingnya informasi kelembaban tanah, alat pengukur kelembaban tanah tidak ada sama sekali. Sedangkan pengukuran kelembaban

permukaan tanah sangat dibutuhkan. Sensor *Soil Moisture* FC-28 hadir dengan sepasang probe teknologi yang bisa disisipkan di dalam tanah. Arus-arus melalui probe dan tingkat resistensi akan diukur. Resistensi akan meningkat jika tanah kering. *Output* dari sensor ini adalah *output* analog yang dapat dihubungkan pada salah satu port *Analogue to Digital* (ADC) tersedia pada board mikrokontroler. Modul sensor *Soil Moisture* FC-28 telah dikalibrasi untuk memvefikasi pengoperasian perangkat yang akurat [36].

### 2.8.5 K-0135

Sensor K-0135 atau *water level* ini prinsipnya adalah mengukur ukuran jumlah air melalui saluran dengan rangkaian kawat paralel yang terbuka. Sensor ini kecil, kuat, dan cerdik dirancang dengan fitur berikut [37]:

1. Pertama, jumlah air untuk mensimulasikan Konversi.
2. Kedua, plastisitas, berdasarkan nilai analog output sensor.
3. Ketiga, konsumsi daya rendah, sensitivitas tinggi.
4. Keempat, dapat langsung terhubung ke mikroprosesor atau sirkuit logika lainnya, dan papan pengontrol untuk berbagai, misalnya: *Arduino controller*, mikrokontroler STC, mikrokontroler AVR dan seterusnya.


Gambar 2.13. Sensor *Water Level* K-0135

## 2.9 *Confusion Matrix*

Confusion matrix merupakan suatu perhitungan yang digunakan untuk mengevaluasi hasil deteksi serangan. Jenis-jenis *alert* pada *confusion matrix* adalah *True Positive* (TP), *False Positive* (FP), *False Negative* (FP), dan *True Negative* (TN) [29].

**Tabel 5**

*Alert* pada *Confusion Matrix*

| No | Tipe *alert* | Keterangan |
|---|---|---|
| 1 | *True Positive* | Suatu kategori data yang tertanda atau terindikasi sebagai serangan. |
| 2 | *False Positive* | Suatu kondisi dimana sistem kategori data mendeteksi serangan yang muncul dari trafik normal yang bukan serangan namun terindikasi sebagai serangan. |
| 3 | *False Negative* | Suatu kondisi saat sistem deteksi tidak mendeteksi adanya serangan yang terjadi. |
| 4 | *True Negative* | Suatu kondisi dimana sistem *K-means clustering* dan Naïve Baayes *classification* mendeteksi suatu serangan dan serangan terjadi terhadap sistem jaringan komputer. |

**Tabel 6**

*Confusion Matrix*

| | | Prediksi | |
|---|---|---|---|
| | | Normal | Serangan |
| Aktual | Normal | TN | FP |
| | Serangan | FN | TP |

Dari tabel di atas, terdapat persamaan yang digunakan untuk mengukur akurasi algoritma *K-means*, yaitu persamaan *confusion matrix* berikut ini [29].

a. *True Positive Rate* (TPR)

*True positive rate* didefinisikan sebagai aktual positif dikategorikan dengan benar oleh sistem kategori data.

$$TPR = \frac{TP}{TP+FN} \quad (2.2)$$

b. *True Negative Rate* (TNR)

*True negative rate* didefinisikan sebagai aktual negatif dikategorikan dengan benar.

$$TNR = \frac{TN}{TN+FP} \quad (2.3)$$

c. *False Positive Rate* (FPR)

*False positive rate* adalah aktual negatif dikategorikan sebagai *class* positif.

$$FPR = \frac{FP}{FP+TN} \quad (2.4)$$

d. *False Negative Rate* (FNR)

*False negative rate* adalah aktual positif dikategori sebagai kelas negatif.

$$FNR = \frac{FN}{TP+FN} \quad (2.5)$$

e. *Precision*

*Precision* adalah jumlah informasi relevan dari jumlah dokumen yan ditemukan oleh sistem.

$$percision = \frac{TP}{TP+FP} \quad (2.6)$$

f. *Accuracy*

*Accuracy* didefinisikan sebagai tingkat kedekatan antara nilai pengkategorian dengan nilai aktual.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (2.7)$$