UNIVERSITAS INDONESIA

IMPLEMENTASI SENSOR WIDS DAN ANALISA TRAFIK RTT PADA PENDETEKSIAN ROGUE ACCESS POINT

SKRIPSI

M. ADHITYA AKBAR

NPM 0806 316 801

FAKULTAS TEKNIK

PROGRAM STUDI TEKNIK KOMPUTER

DEPOK

JUNI 2012

UNIVERSITAS INDONESIA

IMPLEMENTASI SENSOR WIDS DAN ANALISA TRAFIK RTT PADA PENDETEKSIAN ROGUE ACCESS POINT

SKRIPSI
Diajukan sebagai syarat untuk memperoleh gelar Sarjana.

M. ADHITYA AKBAR
0806 316 801

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK KOMPUTER
DEPOK
JUNI 2012

HALAMAN PERNYATAAN ORISINALITAS

Skripsi ini adalah hasil karya saya sendiri, dan semua sumber baik yang dikutip maupun dirujuk telah saya nyatakan dengan benar.

Nama	**: M. Adhitya Akbar**
NPM	**: 0806316801**
Tanda Tangan	**:**
Tanggal	**: 13 Juni 2012**

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh :

Nama	: M. Adhitya Akbar
NPM	: 0806316801
Program Studi	: Teknik Komputer
Judul Skripsi	: Implementasi Sensor WIDS dan Analisa Trafik RTT pada Pendeteksian Rogue Access Point

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Teknik pada Program Studi Teknik Komputer, Fakultas Teknik, Universitas Indonesia

DEWAN PENGUJI

Pembimbing : Muhammad Salman S.T., M.IT ( )

Penguji : Yan Maraden ST. MSc. ( )

Penguji : I Gde Dharma Nugraha ST. MT ( )

Ditetapkan di : Depok
Tanggal : 11 Juli 2012

UCAPAN TERIMA KASIH

Puji syukur saya panjatkan kepada Tuhan Yang Maha Esa, karena atas berkat dan rahmat-Nya, saya dapat menyelesaikan skripsi ini. Penulisan skripsi ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Sarjana Teknik Jurusan Teknik Komputer pada Fakultas Teknik Universitas Indonesia. Oleh karena itu, saya mengucapkan terima kasih kepada:

1. Muhammad Salman, ST. M.I.T selaku pembimbing akademis dan dosen pembimbing atas segala bimbingan, ilmu, dan arahan baik dalam penulisan skripsi maupun selama masa studi di Teknik Komputer.
2. Bapak (Ir. Yurisman, MSc), Ibu (Dra. Betty Sailun MEd), yang telah memberikan bantuan dukungan moral, do'a serta materil dan adik-adik yang selalu menjadi sumber inspirasi dan semangat.
3. Pak Budi Sudiarto, yang telah berjasa mengajarkan penulis konsep dasar jaringan komputer di CCNA.
4. Saleh Iskandar ST, atas pengetahuan *wireless security*.
5. Aneta yang telah membenarkan pengerjaan tulisan buku skripsi ini.
6. Arie, Fikri, Rika, Hari, Faris, Adityo, Irvanda dan Daus selaku teman-teman satu bimbingan, Norman PND, A. Sunandar, Y.D Prima, H. Napit, R. Armayasa, thanks we did it guys.
7. Ruki, Burhan, Alfa, Archie, Erwin, Agung, Irfan, yang telah membimbing penulis dalam menyelesaikan skripsi ini.

Akhir kata, saya berharap Tuhan Yang Maha Esa berkenan membalas segala kebaikan semua pihak yang telah membantu. Semoga skripsi ini membawa manfaat bagi pengembangan ilmu.

Depok, 13 Juni 2012

Penulis,

M. ADHITYA AKBAR
NPM. 0806316801

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI
TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Universitas Indonesia, saya yang bertanda tangan di bawah ini:

Nama : M. Adhitya Akbar

NPM : 0806316801

Program Studi : Teknik Komputer

Departemen : Teknik Elektro

Fakultas : Teknik

Jenis karya : Tugas akhir

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia **Hak Bebas Royalti Noneksklusif** (*Non-exclusive Royalty-Free Right*) atas karya ilmiah saya yang berjudul:

IMPLEMENTASI SENSOR WIDS DAN ANALISA TRAFIK RTT PADA PENDETEKSIAN ROGUE ACCESS POINT

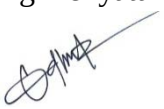
beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan memublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di: Depok

Pada tanggal: 13 Juni 2012

Yang menyatakan,



(M. Adhitya Akbar)

ABSTRAK

Nama : M. Adhitya Akbar

Program Studi : Teknik Komputer

Judul : Implementasi sensor WIDS dan analisa trafik RTT pada Pendeteksian *Rogue Access Point*

Pembimbing : Muhammad Salman, ST. M.I.T

Perkembangan jaringan *wireless internet* sangat mengagumkan selama beberapa dekade ini. Hal ini mendorong pertumbuhan *hotspot* di tempat-tempat umum. Dibalik kemudahannya, terdapat ancaman yang sangat berbahaya, salah satu bentuk ancaman terbesarnya adalah *Rogue Access Point* (RogueAP). *Evil-twin-attack* merupakan sebuah *proof-of-concept* ancaman dari RogueAP. User biasa akan mudah tertipu dan terhubung ke akses poin palsu tersebut. Ketidakpahaman mendalam mengenai *network* oleh *user* semakin mempermudah aktifitas *hacker*. Dibutuhkan suatu sistem yang tepat untuk mengetahui keberadaan RogueAP ini. Metode yang diusulkan juga bermacam-macam seperti pendekatan wired-side, wireless-side dan gabungan keduanya. Pada tulisan ini akan memberikan gambaran dua metode tersebut yaitu analisa trafik RTT dan WIDS sensor. Pada skenario 1 dan 2, kenaikan RTT Ping berkisar rata-rata 7.5% dari RTT *Legitimate Access Point*. *Response time* pendeteksian RAP di WIDS sensor tergantung pada jarak dan kekuatan sinyal antara WIDS dengan RogueAP. Pendeteksian WIDS Sensor mencapai keakuratan hingga 100% mendeteksi RogueAP dalam jangkauannya akan tetapi masih berbasiskan identitas mac address. Pada Area 1 dan 2 rata-rata *response time* berkisar 1-2 detik sedangkan pada Area 3 sebesar 3.7 detik dan Area 4 (1%-24%) sekitar 10.4 detik. Analisa trafik RTT sangat berpotensi menjadi alternatif pendeteksian *Rogue Access Point*.

Kata kunci: *Rogue Access Point, Evil-twin-attack, Round Trip Time, Wireless Intrusion Detection System*

ABSTRACT

Name : M. Adhitya Akbar

Study Program : Computer Engineering

Title : Implementation WIDS Sensor and RTT Traffic Analysis on Rogue Access Point Detection

Supervisor : Muhammad Salman, ST. M.I.T

The development of wireless data networks are very impressive for several decades. This encourages the growth of hotspots in public area. Behind the simplicity, there is a very dangerous threat, one of the greatest threats is the Rogue Access Point. Evil-twin-attack is a proof-of-concept threat of RogueAP. Regular user would be easily fooled and wil be connected to the fake access point. Not understanding the depth of the network by the user enhances the threat from hackers. Therefore we need a proper system for the presence of Rogue Access Point. The proposed method as well as a variety approaches of wired-side, wireless-side and a combination of both (hybrid). In this paper will provide an overview of two methods, namely the analysis of RTT traffic and WIDS sensor. In scenario 1 and 2, average increasing ranged Ping RTT is 7.5% of the RTT Legitimate AP. Response time detection of RAP in WIDS sensor depends on the distance and signal strength between the WIDS with Rogue Access Point. WIDS detection sensor reaches up to 100% accuracy in detecting RogueAP range but still based on the identity of the mac address. Average response time Area 1 and 2 ranges from 1-2 seconds while in Area 3 of 3.7 seconds and Area 4 (1% -24%) at about 10.4 seconds. RTT traffic analysis is a potential alternative to Rogue Access Point detection.

Keywords: Rogue Access Point, Evil-twin-attack, Round Trip Time, Wireless Intrusion Detection System

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Pertumbuhan Teknologi Informasi dan media transmisi telah mengubah wajah telekomunikasi. Indonesia sendiri baru merasakan dampaknya sekitar 10 tahun terakhir ini. Dampak yang diberikan sangat besar bagi pertumbuhan sektor-sektor penting skala besar atau kecil, bahkan untuk urusan-urusan pribadi sekalipun.

Dengan di bebaskannya penggunaan frekuensi 2.4 GHz pada tahun 2000 oleh pemerintah pada keputusan DIRJEN POSTEL 241/2000 telah memulai sebuah era Internet *wireless* di Indonesia. Pertumbuhan Internet *wireless* meningkat drastis pada tahun-tahun berikutnya.

Penggunaan Wi-Fi pada tempat-tempat umum ini dikenal dengan sebutan *Hotspot*. *Hotspot* sendiri ada yang gratis dan berbayar. Kebanyakan *Hotspot* publik tidak menggunakan keamanan lebih seperti penggunaan WEP atau WPA/WPA2 sehingga semua *user* dapat mengakses dengan bebas termasuk *hacker*. Hal ini menyebabkan *hotspot* publik menjadi sasaran empuk para *hacker* untuk melakukan aktifitas *hacking*.

Pengetahuan yang dimiliki *user* sangat bervariatif dimana pada umumnya tidak memiliki pengetahuan jaringan yang dalam sedangkan dalam pekerjaannya menuntut untuk selalu terhubung ke Internet. Jenis *user* seperti ini akan sangat mudah menjadi korban para *hacker* karena tidak menyadari lalu lintas jaringan komputer yang dimilikinya.

Berbagai macam serangan dapat terjadi pada jaringan *Wireless* seperti ARP *Poisoning* pada *access point* maupun *wireless router*, *Man-in-the-middle* (MITM), *Denial of Service* seperti SYN *Flood*, dan *Rogue/Trojan access point*.

Berdasarkan dari semua serangan yang mungkin terjadi, yang menjadi perhatian dan paling berbahaya adalah *Rogue/Trojan access point*. Hal ini dikarenakan trafik dari *user* akan dapat ditangkap dan diteruskan oleh *hacker* dan kondisi itu tidak terdapat keamanan.

Untuk membuat *Rogue/Trojan access point* sangatlah mudah sehingga tingkat serangan dengan menggunakan metode ini sangatlah tinggi. Bahkan pada Windows 7 dapat menciptakan sebuah *virtual interface* yang dapat menjadi sebuah *access point*.

Oleh karena itu, maka sangat dibutuhkan sebuah metode untuk mengatasi permasalahan tersebut. Mendeteksi sebuah *Rogue Access Point* sangatlah sulit dikarenakan kemungkinan *MAC address*, SSID, *channel* dan beberapa parameter *access point* yang dapat di palsukan/disamakan.

1.2 Rumusan Masalah

Dengan melihat mudahnya serangan RogueAP di lakukan dan tingkat kesuksesan dalam menjebak *user* sangatlah tinggi, diperlukan sebuah sistem pendeteksian yang mendeteksi konektivitas yang di lakukannya melalui *access point* yang sebenarnya atau yang palsu.

Pada tulisan ini, permasalahan hanya berfokus pada pendeteksian *Rogue Access Point* yang berjalan di IEEE 802.11g. Pendeteksian menggunakan *Wireless Intrusion Detection System* (WIDS) berupa *sensor drone* berbasiskan *Open Source* dan analisa trafik RTT DNS ICMP.

1.3 Tujuan Penulisan

Tujuan dari penulisan ini adalah untuk menguji dan menentukan metode-metode yang tepat dalam pendeteksian *Rogue Access Point*. Sistem sensor WIDS dan analisis trafik RTT ini diharapkan dapat menjadi solusi dalam menemukan benang merah antara metode pendekatan *wired-side* dan *wireless-side* dalam pendeteksian *Rogue Access Point*.

1.4 Batasan Masalah

Permasalahan dalam penulisan skripsi ini dibatasi pada perancangan dan implementasi sensor WIDS serta analisa trafik RTT dalam pendeteksian *Rogue Access Point*. Penelitian ini membahas konsep dan metode-metode Pendeteksian *Rogue Access Point*. Jenis *access point* yang digunakan berjalan di frekuensi 2.4 GHz dengan standar IEEE 802.11g dan 802.11n. *Access Point* dan *Rogue Access Point* menggunakan 802.11g dimana WIDS sensor menggunakan 802.11n. Jenis lingkungan yang digunakan bersifat *local area network* menggunakan IPv4. Tidak terdapat pembatasan port pada lingkungan percobaan.

1.5 Metode Penelitian

Metode yang digunakan untuk Skripsi ini adalah :

1. Studi literatur, dengan membaca dan menggunakan buku sebagai referensi untuk skripsi ini
2. Perancangan jaringan, setelah studi literature dan , maka dilakukan perancangan jaringan secara teori untuk mengimplementasikan WIDS dan menganalisis trafik RTT dengan menggunakan 3 topologi skenario *rogue access point* (*Rogue*AP) yang berbeda
3. Pengujian dan analisis, menguji WIDS mendeteksi *Rogue*AP dan menganalisis parameter RTT dengan menggunakan *ping* dan *dns lookup*.

1.6 Sistematika Penulisan

Agar penulisan skripsi ini dapat terarah dengan baik, maka penulisan skripsi ini dibagi atas beberapa bab, yaitu:

a. Bab 1 Pendahuluan

   Pada bab ini, akan dijelasan mengenai Latar Belakang, Tujuan, Pembatasan Masalah, Metodologi Penulisan, dan Sistematika penulisan.

b. Bab 2 Landasan Teori

   Pada bab ini, akan dijelaskan mengenai karakteristik TCP/IP, DNS, jaringan WLAN, Sistem *access point*, *Rogue Access Point*, Keamanan Jaringan *Wireless*, Kismet WIDS, Linux *Server* dan Round-Trip-Time.

c. Bab 3 Perancangan dan Implementasi Pendeteksian *Rogue Access Point*

   Pada bab ini, akan dijelaskan mengenai perancangan dan metode pendeteksian *Rogue Access Point* serta topologi rancang awal. Pada bab

ini dijelaskan tentang cara implementasi WIDS dan rancangan pengujian RTT serta cara pembuatan *Rogue Access Point*.

d. Bab 4: Pengujian dan Analisis

   Dari hasil implementasi tersebut, akan dilakukan pengujian dengan hasil yang akan dibahas pada bab 4. Pengujian dan analisis hasil dilakukan berdasarkan hasil pengujian teknis dan pengujian oleh penguji dan dilakukan terhadap *Rogue Access Point* menggunakan Kismet dan analisa RTT.

e. Bab 5: Kesimpulan

   Bab ini berisi mengenai kesimpulan dari uraian bab-bab sebelumnya serta saran untuk tahapan kelanjutan dalam pendeteksian *Rogue Access Point*.

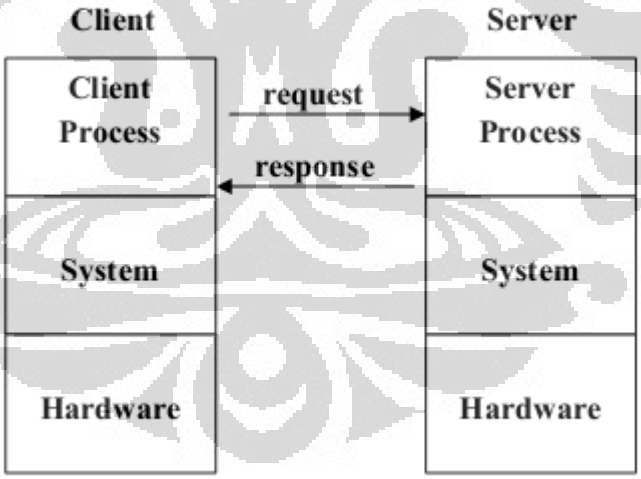
BAB 2
LANDASAN TEORI

2.1 Pengertian TCP/IP

TCP/IP (singkatan dari *Transmission Control Protocol*/*Internet Protocol*) adalah standar komunikasi data yang digunakan oleh komunitas internet dalam proses tukar-menukar data dari satu komputer ke komputer lain di dalam jaringan Internet . Protokol ini tidaklah dapat berdiri sendiri, karena memang protokol ini berupa kumpulan protokol (*protocol suite*). Protokol ini juga merupakan protokol yang paling banyak digunakan saat ini. Data tersebut diimplementasikan dalam bentuk perangkat lunak (*software*) di sistem operasi. Istilah yang diberikan kepada perangkat lunak ini adalah TCP/IP *stack*

2.2 Pengertian *Client-Server*

Client-Server adalah arsitektur jaringan yang memisahkan *Client* (biasanya aplikasi yang menggunakan GUI) dengan *server*. Masing-masing *Client* dapat meminta data atau informasi dari *server*



Gambar 2.1 Ilustrasi komunikasi pada Jaringan *Client-Server*

2.3 *Wireless Local Area Network* (WLAN)

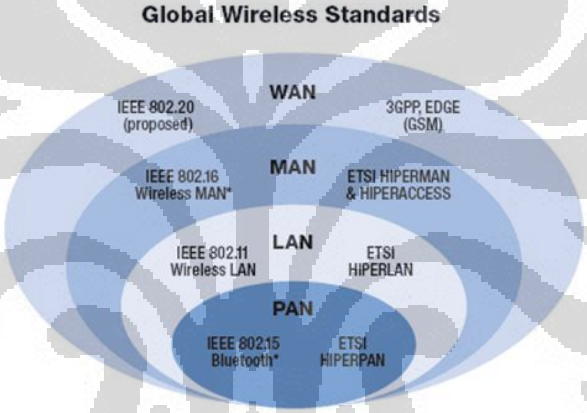
2.3.1 Pengertian Jaringan *Wireless*

Jaringan *Wireless* adalah sebuah sistem jaringan yang dapat mengirimkan dan menerima data dengan menggunakan media ruang bebas.

Tipe-tipe *Wireless Network* terbagi atas 3 yaitu :

1. *Wireless* PAN (*Personal Area Network*)
2. *Wireless* LAN
3. *Wireless* Mesh *Network*
4. *Wireless* MAN
5. *Wireless* WAN

(Personal Area Network, 2011)



Gambar 2. 2 Standar Global *Wireless*[1]

2.3.2 Jaringan *Wireless* LAN

Jaringan *Wireless LAN* menghubungkan dua atau lebih *divais* dengan menggunakan metode distribusi (pada umumnya menggunakan *spread-spectrum* atau OFDM) dan biasanya memberikan sebuah koneksi ke jaringan Internet kabel melalui sebuah *access point*. Memberikan kemudahan bagi *user* untuk melakukan mobilitas pada daerah cakupan lokal dan tetap terkoneksi ke jaringan. Kebanyakan WLAN yang *modern* merujuk ke sebuah standar yaitu IEEE 802.11, dan di pasarkan dalam nama Wi-Fi.

Sebuah *Wireless* LAN terdiri atas sebuah *node* dan *access point*. Sebuah *node* tersebut adalah komputer atau periferal seperti printer yang memiliki sebuah adapter jaringan, misalnya sebuah antena. *access point* berfungsi sebagai *transmitter* dan *receiver* antara *node* atau antara *node* pada jaringan yang berbeda.

Wireless Fidelity atau Wi-Fi adalah jenis mekanisme jaringan *Wireless* (IEEE 802.11) sebagai produk dari Wi-Fi Alliance yang sering digunakan pada industri. Jaringan ini digunakan pada tempat-tempat umum seperti tempat perbelanjaan, restoran, perpustakaan, kampus, sekolah dan tempat-tempat umum lainnya.

2.3.3 Cara Kerja *Wireless* LAN

WLAN menggunakan transmisi radio, infra merah, dan gelombang mikro untuk mengirmkan data dari satu titik ke titik lainnya tanpa menggunakan perantara kabel.

Wireless LAN menggunakan *electromagnetic airwaves* (radio atau infra merah) untuk menukarkan informasi dari satu titik ke titik lainnya tanpa harus tergantung pada sambungan secara fisik.Gelombang radio biasa digunakan sebagai pembawa karena dapat dengan mudah mengirimkan daya ke penerima. Data ditransmikan dengan cara ditumpangkan pada gelombang pembawa sehingga bisa diextract pada ujung penerima.

Data ini umumnya digunakan sebagai pemodulasi dari pembawa oleh sinyal informasi yang sedang ditransmisikan. Ketika datanya sudah dimodulasikan pada gelombang radio pembawa, sinyal radio akan menduduki lebih dari satu frekuensi, hal ini terjadi karena frekuensi atau *bit rate* dari informasi yang memodulasi ditambahkan pada sinyal *carrier*.

Multiple radio carrier bisa ada dalam suatu ruang dalam waktu yang bersamaan tanpa terjadi interferensi satu sama lain jika gelombang radio yang ditransmisikan berbeda frekuensinya. Untuk mengekstrak data, radio penerimanya diatur dalam satu frekuensi dan menolak frekuensi-frekuensi lain.

Pada konfigurasi *Wireless* LAN tertentu, *transmitter/receiver* (*transceiver*) device, biasa disebut *access point*, terhubung pada jaringan kabel dari lokasi yang fixed menggunakan kabel standar. Sebuah *access point* bisa mendukung sejumlah group kecil dari *user* dan bisa dipakai dalam jarak antara seratus sampai beberapa ratus kaki. *access point* (atau antena yang terhubung pada *access point*) biasanya diletakkan pada tempat yang tinggi tapi dapat juga diletakkan dimana saja untuk mendapatkan cakupan yang dikehendaki. *End user* mengakses *Wireless* LAN menggunakan *Wireless*-LAN adapters, biasa terdapat pada PC *card* pada *notebook* atau palmtop *computer*, atau sebagai *card* dalam komputer *desktop*, atau terintegrasi dalam *hand-held computer*.

Lima Proses dasar yang terjadi pada WLAN :

1. *Scanning*

Proses mencari jaringan. Untuk dapat menggunakan sebuah jaringan, langkah pertama yang harus dilakukan adalah mencarinya. Seperti halnya jaringan kabel, perlu menemukan sebuah kabel yang terhubung ke *patch panel* untuk mendapatkan koneksi ke Inter/Intranet. Dalam dunia *Wireless* diperlukan identifikasi jaringan terlebih dahulu di suatu *area*. Setelah melakukan *scanning*, komputer akan memilih untuk bergabung ke salah satu dari BSS.

Pada proses *scanning* terdapat 7 parameter. Kebanyakan *Wireless card*

Menggunakan konfigurasi dasar pada parameter tersebut. Berikut adalah penjelasan singkat 7 parameter tersebut :

a.) Tipe-tipe BSS

Terdapat 2 tipe BSS yaitu ***Adhoc*** dan ***Infrastructure Network***

b.) BSSID

Terbagi atas 2 yaitu ***Individual* BSSID** dan ***Broadcast* BSSID**. *access point* akan memilih BSSID yang mana yang akan ditransfer. Jika *access point* memilih *Individual* BSSID, maka AP tersebut

akan menjadi "*invisible"* sedangkan sebaliknya untuk *Broadcast* BSSID maka AP akan dapat terlihat oleh *divais* lain.

c.) SSID

SSID atau nama jaringan merupakan sebuah *string* dimana *Client-Client* dapat berasosiasi ke jaringan.

d.) Scan Type

Pada parameter ini terbagi atas 2 yaitu ***Active*** dan ***Passive***. Dengan *Active* dan Passive menggunakan *frame* yang berbeda. Tipe *Active* akan menghemat penggunaan energi.

e.) Channel List

Scan dapat menggunakan *Probe Request* atau *Listen* pada *channel* yang spesifik.

f.) Probe Delay

Waktu *delay* dalam mikrodetik sebelum aktif *scanning* dimulai.

g.) *Minimum* dan *Maximum Channel Time*

Waktu minimal dan maksimal sebelum *station* melakukan *scanning channel* tertentu.

2. *Joining*

Proses ini tidak memberikan kepastian mendapatkan akses jaringan. Setelah *joining* diperlukan *authenticate* dan *associate*, dan barulah mendapatkan akses ke jaringan.

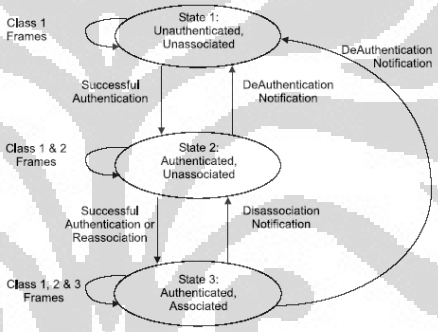
Terdapat 2 cara bagaimana sebuah *station* (komputer) dapat bergabung ke BSS tertentu.

- *Station* memilih BSS secara manual
- Otomatis, berdasarkan *power* level dan kekuatan sinyal, *station* akan memilih *access point* mana yang akan dipilih. Paramater pada *station* haruslah sama.

3. *Authenticating*

Otentikasi adalah hal yang paling penting dalam jaringan *wireless*. Pada jaringan kabel, akses fisik dapat dibatasi, namun pada jaringan *wireless* hal tersebut tidak dapat dilakukan sehingga diperlukan *authentication.*

Pada jaringan yang *Open*, setiap komputer akan di otentifikasi tanpa adanyan pengecekan. Komputer A akan mengirimkan *Authentication Managemen frame* yang mengindikasikan pengenalan, ditujukan ke A. Pada jaringan *closed* (dengan menggunakan Enkripsi), komputer mesti mengetehui SSID dari *access point* (AP) dengan maksud dapat terkoneksi ke AP. *Shared Key Authentication* menggunakan standar *challenge* dan *response* dengan sebuah *Shared Secret Key*.



Gambar 2.3 State dan Service Authentication[11]

4. ***Association***

   Setelah *Authentication,* maka tahapan berikutnya adalah *Association* dimana akan memberikan akses ke jaringan. Hal ini sama dengan saat kita memasukkan RJ45 ke *port* yang ada koneksi Internet pada jaringan kabel. 802.11 membatasi *divais* untuk berasosiasi lebih dari satu *access point*.

   Data dapat bertukar antara komputer dan AP hanya jika telah terjadi *Association* antara komputer tersebut dengan AP. Setiap AP mengirimkan *frame beacon* beberapa kali setiap detik yang berisikan SSID, *Supported Rates*, dan informasi lainnya. Komputer akan memilih untuk bergabung dengan sebuah AP pada sinyal yang besar. *association* memiliki 2 langkah proses. Komputer yang mana masih dalam kondisi *unauthenticated* dan *Unassociated* mendengarkan adanya *frame beacon*. Komputer akan memilih sebuah BSS untuk bergabung. Komputer dan AP akan melakukan otentifikasi dengan bertukaran *frame authentication Management*. Lalu setelah otentifikasi, maka komputer akan mengirimkan sebuah *frame association request* ke AP, dan AP membalasanya dengan mengirimkan *Association Response Frame* yang berisikan sebuah *Association* ID ke komputer. Lalu komputer tersebut telah berhasil Otentifikasi dan Asosiasi dengan AP.

   Sebuah komputer dapat otentifikasi ke beberapa AP pada waktu yang sama tapi hanya dapat berasosiasi pada satu AP pada satu waktu.

5. **The Re*assoc*iat*ion* Procedure**

   Terjadi ketika sebuah *station* berpindah ke jaringan dengan banyak *access point*. *Station* akan memonitor kualitas sinyal *access point* yang pernah berasosiasi dan *access point* lainnya pada ESS yang sama. Ketika *station* mendeteksi bahwa *access point* lain memiliki kualitas yang lebih baik, *station* akan melakukan Prosedur *Reassociation*.

2.3.4 Komponen Jaringan *Wireless* LAN

Dalam membangun sebuah WLAN diperlukan beberapa komponen antara lain sebagai berikut :

a. *Wireless access point* (WAP)

Sebuah *Wireless access point* (WAP) adalah *divais* yang mengizinkan *divais Wireless* untuk terhubung ke jaringan kabel menggunakan Wi-Fi, *bluetooth* atau standar yang berhubungan. WAP biasanya terkoneksi ke sebuah *router* (melalui jaringan kabel) dan dapat melanjutkan data antara *divais Wireless* (seperti komputer atau printer) dan *divais* dengan media kabel pada sebuah jaringan.

Gambar 2.4 Industrial *Wireless Access Point* [3]

Gambar 2.5 Cisco Linksys WAP [17]

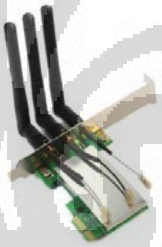
b. *Wireless Router* (WRT)

Wireless Router merupakan *divais* yang melakukan fungsi sebagai sebauh *router* dan sekaligus sebagai *wireless access point* dan *Network Switch*.

Gambar 2.6 Linksys WRT Black [2]

c. *Wireless Adapter*/WLAN *Card*

Wireless adapter merupakan sebuah divais yang menambah konektivitas *Wireless* komputer atau PDA. *Wireless adapter* dipakai melalui *port* USB, PC *card*, *memory card* atau dimasukkan ke dalam PCI bus di dalam komputer. Ada tiga tipe w*ireless adapter*, yaitu Wi-Fi, seluler, dan *bluetooth*.



Gambar 2. 7 *Wireless Adapter* [16]

d. *Mobile*/*Desktop* PC

PC merupakan *divais end-user* yang berfungsi sebagai media akses oleh pengguna. Sekarang *mobile* PC sudah *built-in wireless adapter* sehingga tidak perlu lagi menggunakan *wireless adapter* USB tambahan. Untuk *desktop* PC, biasanya masih membutuhkan *interface* tambahan berupa USB atau PCI *card*.

e. Antena

Antena merupakan sebuah perangkat yang terdiri dari susunan kabel, batang besi, dan lain-lain yang berfungsi untuk memancarkan dan menerima sinyal radio. Fungsi antena pada WLAN adalah untuk memperkuat daya pancar. Antena biasanya digunakan pada WAP, WR, dan *wireless adapter* dengan tujuan mendapatkan akses lebih stabil dan jauh.

1. Antena Omni-*directional*

Antena *omni-directional* merupakan antena yang dapat menciptakan pola radiasi *horizontal* 360^{o}. Antena ini digunakan ketika ingin mencakup semua arah (*horizontal*) dari antena dengan memvariasikan derajat cakupan vertikal.

Gambar 2. 8 TP-Link Omni-Directional *Antenna* [16]

2. Antena *directional*

Antena *directional* merupakan antena yang berfokus kepada energi frekuensi radio di arah tertentu saja. Karena *gain* antena *directional* meningkat, cakupan jaraknya pun meningkat, tetapi cakupan sudut efektifnya berkurang.

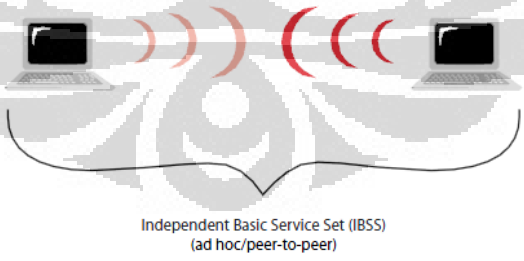
Gambar 2. 9 Antena TPLink ANT 2409B 2,4 GHz outdoor directional ~9dBi [16]

2.3.5 Mode Jaringan *Wireless* LAN

Spesifikasi 802.11 mendefinisikan dua tipe mode jaringan WLAN, yaitu mode *ad-hoc* (*peer-to-peer*) dan mode infrastruktur.

a. Mode Ad-Hoc

Jaringan *ad-hoc* juga kadang dinamakan *Independent Basic Set Identifier* (IBSS). Pada jaringan *ad-hoc*, jaringan *Wireless* sangat sederhana. Jaringan ini menghubungkan beberapa komputer yang menggunakan *Wireless* NIC *card* ke dalam sebuah jaringan tanpa menggunakan peralatan tambahan seperti *access point* (AP) melalui gelombang radio.



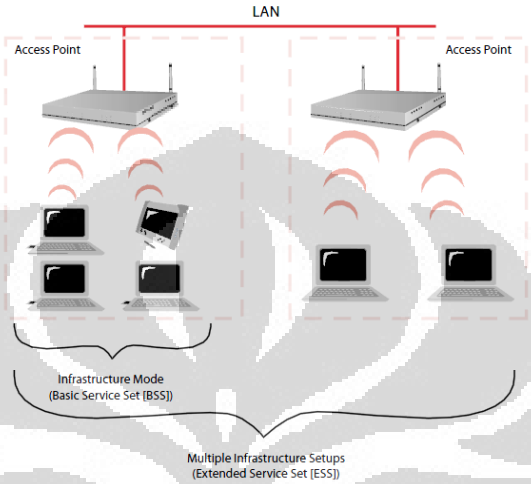
Gambar 2. 10 Mode ad-hoc [16]

b. Mode Infrastruktur

Pada mode infrastruktur, semua perangkat *client* berkomunikasi dengan AP yang menghubungkan *wireless* milik *client* dengan jaringan kabel yang telah ada. AP mengubah paket 802.11 ke paket *ethernet* LAN 802.3. Semua *Wireless*

client dan *divais* dalam jangkauan dapat menerima paket, tetapi hanya *Client* yang cocok dengan alamat tujuan yang akan menerima dan memproses paket.

Infrastruktur *wireless* sederhana dengan satu AP disebut *Basic Service Set* (BSS). Jaringan dimana lebih dari satu AP yang membentuk satu sub jaringan disebut *Extended Service Set* (ESS).



Gambar 2.11 Basic Service Set (BSS) dan Extended Service Set (ESS) [16]

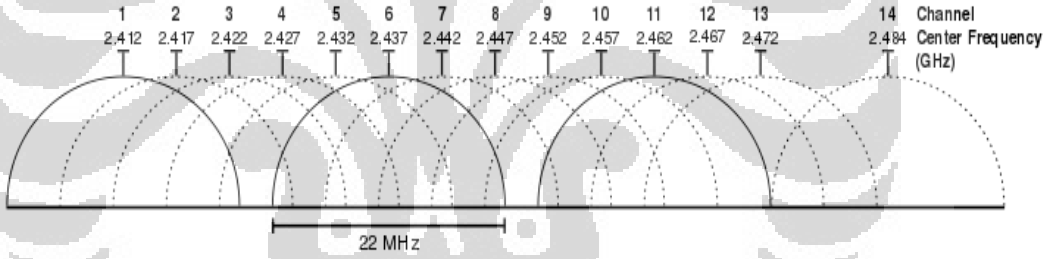
Di dalam jaringan *Wireless* terdapat *password* (SSID). SSID adalah 32 karakter pengenal unik yang terdapat pada *header* atau paket yang dikirim pada jaringan yang bertindak sebagai *password* ketika *divais mobile* mencoba untuk membuat koneksi dengan BSS. SSID membedakan WLAN yang satu dengan yang lain, jadi semua AP dan *divais* yang mencoba untuk membuat koneksi ke WLAN tertentu harus menggunakan SSID yang sama. SSID juga dianggap sebagai nama jaringan karena SSID merupakan nama yang dapat mengidentifikasi suatu jaringan *wireless*.

Pada jaringan ESS, jaringan-jaringan BSS tidak harus menggunakan SSID yang sama namun tanpa SSID yang sama, pengguna tidak akan dapat menggunakan fungsi *roaming*. *Roaming* adalah fitur yang memungkinkan *Client* berpindah daru sebuah jaringan BSS ke jaringan BSS yang lain secara otomatis tanpa terputus koneksinya. Untuk menggunakan *roaming*, harus terdapat *overlapping area* atau *area* dimana sinyal dari kedua BSS bisa diakses.

2.3.6 *Channel Wireless*

Sebelum membahas lebih jauh tentang WLAN, ada komponen utama yang menyebabkan WLAN dapat bekerja, yaitu *channel*. Frekuensi yang digunakan untuk radio adalah FM dan AM. Di dalam frekuensi FM, terdapat ratusan *channel* tempat dimana masing-masing stasiun radio menggunakannya. Jaringan *Wireless* menggunakan konsep yang sama dengan stasiun radio, dimana saat ini terdapat dua alokasi frekuensi yang digunakan, yaitu 2,4 GHz dan 5 GHz. Frekuensi 2,4 GHz digunakan oleh 802.11b/g/n. Frekuensi ini dibagi menjadi *channel-channel* seperti pembagian frekuensi untuk stasiun radio.

International Telecommunication Union (ITU) membagi frekuensi ini menjadi 14 *channel* namun setiap negara mempunyai kebijakan tertentu terhadap pembagian *channel* ini. Amerika hanya mengijinkan penggunaan *channel* 1 – 11, Eropa hanya menggunakan *channel* 1 – 13, Jepang diperbolehkan menggunakan semua *channel*, yaitu 1 – 14.



Gambar 2.12 Channel Wi-Fi pada frekuensi 2,4 GHz [1]

Pada Gambar 2.8 terlihat frekuensi tengah *channel* 1 adalah 2,412. *Range* dari *channel* ini adalah 2,401 – 2.423, sehingga setiap *channel* mempunyai lebar 22 MHz. Tabel di bawah ini merupakan alokasi pembagian *channel* pada frekuensi 2,4 GHz.

Tabel 1 Alokasi pembagian *channel* pada frekuensi 2,4 GHz

Channel	Frekuensi (GHz)	*Range*	Channel *Range*
1	2,412	2,401 – 1,423	1 – 3
2	2,417	2,406 – 2,428	1 – 4
3	2,422	2,411 – 1,433	1 – 5
4	2,427	2,416 – 2,438	2 – 6
5	2,432	2,421 – 1,443	3 – 7
6	2,437	2,426 – 2,448	4 – 8
7	2,442	2,431 – 1,453	5 – 9
8	2,447	2,436 – 2,458	6 – 10
9	2,452	2,441 – 1,463	7 – 11
10	2,457	2,446 – 2,468	8 – 11
11	2,462	2,451 – 1,473	9 – 11
12	2,467	2,456 – 2,478	tidak ada di US
13	2,472	2,461 – 1,483	tidak ada di US
14	2,484	2,473 – 2,495	tidak ada di US

2.3.7 Standar Wi-Fi

Sebelum 802.11, teknologi *wireless* LAN memiliki kecepatan yang sangat rendah yang dapat ditawarkan. Disaat itu kebanyakan menggunakan 902-928 MHz ISM band. Berikut adalah beberapa standar dari IEEE

- 802.1 > LAN/MAN Management and Media Access Control Bridges
- 802.2 > Logical Link Control (LLC)
- 802.3 > CSMA/CD (Standar untuk Ehernet Coaxial atau UTP)
- 802.4 > Token Bus
- 802.5 > Token Ring (bisa menggunakan kabel STP)
- 802.6 > Distributed Queue Dual Bus (DQDB) MAN
- 802.7 > Broadband LAN
- 802.8 > Fiber Optic LAN & MAN (Standar FDDI)
- 802.9 > Integrated Services LAN Interface (standar ISDN)
- 802.10 > LAN/MAN Security (untuk VPN)
- 802.11 > Wireless LAN (Wi-Fi)
- 802.12 > Demand Priority Access Method
- 802.15 > Wireless PAN (Personal Area Network) > IrDA dan Bluetooth
- 802.16 > Broadband Wireless Access (standar untuk WiMAX)

Wi-Fi bermula dari IEEE. Komite IEEE 802 LAN/MAN mendefinisikan standar untuk Local Area Network (LAN) dan Metro Area Network (MAN) termasuk ethernet, token ring dan Wi-Fi.

Semenjak IEEE 802.11 standard diadopsi pada tahun 1997, kecepatan data meningkat dari 1 atau 2 Mbps menjadi 11 Mbps dan sekarang hingga 10 Gbps.

Pembagian standar 802.11 menjadi beberapa *working-group* didasarkan terhadap teknologi yang dihasilkan dimulai dari sufiks 802.11 yaitu 802.11a, 802.11b hingga kini 802.11n.

a. 802.11b

Sempat menjadi dominasi pemakaian tipe b. Standard 802.11b mengunakan frekuensi 2.4GHz. Standard ini sempat diterima oleh pemakai didunia dan masih bertahan sampai saat ini. Tetapi sistem b bekerja pada band yang cukup kacau, seperti gangguan pada Cordless dan frekuensi Microwave dapat saling menganggu bagi daya jangkaunya. Standard 802.11b hanya memiliki kemampuan tranmisi standard dengan 11Mbps atau rata rata 5MBbit/s yang dirasakan lambat, mendouble (turbo mode) kemampuan *wireless* selain lebih mahal tetapi tetap tidak mampu menandingi kemampuan tipe a dan g. Perangkat ini dapat menjangkauan jarak maksimum 300 kaki (91 m).

b. 802.11a

Standard 802.11a, adalah model awal yang dibuat untuk umum. Mengunakan kecepatan 54Mbps dan dapat mentranfer data double dari tipe g dengan kemampuan bandwidth 72Mbps atau 108Mbps. Sayangnya sistem ini tidak terlalu standard, karena masing masing vendor atau pabrikan memberikan standard tersendiri. 802.11a mengunakan frekuensi tinggi pada 5Ghz sebenarnya sangat baik untuk kemampuan tranfer data besar. Tetapi 802.11a memiliki kendala pada harga , komponen lebih mahal ketika perangkat ini dibuat untuk publik dan jaraknya dengan frekuensi 5GHz konon lebih sulit menembus ruang untuk kantor. Pemilihan 5Ghz cukup beralasan, karena membuat pancaran signal frekuensi 802.11a jauh dari gangguan seperti oven microwave atau cordless phone pada 2GHz, tetapi frekuensi tinggi juga memberikan

dampak pada daya jangkau relatif lebih pendek. Standar ini bisa menjangkau jarak maksimum 150 kaki (45,7 m).

c. 802.11g

Standard yang cukup kompatibel dengan tipe 802.11b dan memiliki kombinasi kemampuan tipe a dan b. Mengunakan frekuensi 2.4GHz mampu mentransmisi 54Mbps bahkan dapat mencapai 108Mbps bila terdapat inisial G atau turbo. Standar 802.11g memiliki jangkauan maksimum 300 kaki (91 m). Untuk *hardware* pendukung, 802.11g paling banyak dibuat oleh vendor. Secara teoritis mampu mentranfer data kurang lebih 20Mbit/s atau 4 kali lebih baik dari tipe b dan sedikit lebih lambat dari tipe a.Karena mengunakan carrier seperti tipe b dengan 2.4Ghz, untuk menghadapi gangguan frekuensi maka ditempatkan sistem OFDM

d. 802.11n

802.11n merupakan perubahan terbaru yang mengembangkan 802.11 sebelumnya dengan menambahkan MIMO (multiple input multiple output) dan beberapa fitur lainnya. IEEE telah menyetujui dan telah di luncurkan dan d publikasikan pada oktober 2009. Dengan munculnya 802.11n telah disetejui pula oleh para produsen perangkat yang mendukung teknologi ini. 802.11n memiliki bandwidth mencapai 540 Mbps dan beroperasi baik dalam frekuensi 2,4 GHz ataupun 5 GHz. Jarak maksimum jangkauan bisa mencapai 984 kaki (250 m).

2.4 Keamanan Jaringan *Wireless*

Keamanan jaringan *wireless* adalah sebuah metode pencegahan dari akses ilegal atau serangan ke komputer dengan media *wireless*. Beberapa jenis keamanan pada jaringan *wireless* yang telah di terapkan antara lain ***Wired Equivalent Privacy*** **(WEP)** dan ***Wi-Fi Protected Access*** **(WPA)**. WEP adalah keamanan terendah yang dapat diimplementasikan di jaringan *Wireless*. Jaringan dengan menggunakan pertahanan WEP dapat dengan mudah di retas dalam waktu kurang lebih 3 menit. WPA merupakan alternatif sebagai jawaban dari permasalahan keamanan pada WEP.

Kebanyakan laptop sudah memiliki *Wireless Card* terinstall. Kemampuan untuk dapat masuk kejaringan dalam keadaan mobile atau bergerak sangatlah

memberikan keuntungan. *Hacker* dapat dengan mudah menjalankan serangannya dengan mobilitas yang tinggi. Sebagai hasilnya, sangat penting dalam bidang *Enterprise* menerapkan keamanan jaringan dengan sangat baik dalam usaha menghadang serangan dari pihak yang tidak bertanggung jawab. ***Wireless Intrusion Systems*** **(WIPS)** atau ***Wireless Intrusion Detection System*** **(WIDS)** biasanya digunakan untuk menerapkan *Wireless Security Policies*.

Beberapa kemungkinan serangan pada jaringan *Wireless* :

1. ***Wireless Network Sniffing***

   Wireless Sniffing hanya dapat terjadi pada satu jaringan sehingga jika sudah beda *subnet* maka hampir mustahil untuk melakukan *sniffing*.

   Jaringan *wireless* menggunakan teknologi radio untuk mengirimkan paket—paketnya. Paket tersebut akan di enkapsulasi melalui *network stack* dari aplikasi yang digunakan. *driver* dan *hardware wireless* lalu mengubah paket tersebut (kode biner) menjadi gelombang radio. Gelombang ini kemudian dipancarkan dari perangkat *Wireless* ke arah yang telah didefinisikan sebelumnya oleh antena.

   Pada kondisi normal, semua *divais wireless* akan mendeteksi gelombang radio yang bertebaran di udara, dan *decode* paket data tersebut untuk mendeterminasikan jika informasi tersebut benar-benar untuknya. jika paket tersebut ditujukan untuk *client* tersebut, maka data akan di teruskan ke *hardware* dan *driver*, nantinya akan di *decode* hingga layer aplikasi.

   Permasalahan terjadi ketika energi *wireless* dipancarkan dari *wireless card* biasanya dipancarkan ke segala arah. Hasil akhirnya adalah setiap orang dengan *divais wireless* yang dalam status *listen*ing pada *area* tersebut dapat mendeteksi energi radio dan menggunakan *software*/*hardware* untuk *decode* gelombang radio tersebut menjadi paket. Dikarenakan hal tersebut, maka *wireless vendor* menggunakan enkripsi untuk menjaga data ketika akan/telah ditransmisikan.

   Wireless Sniffing dapat menggunakan berbagai macam *software sniffer* baik di lingkungan Windows maupun Linux. Untuk *wireless sniffer basic*, bisa menggunakan Wireshark (Windows/Linux) dan Kismet (Linux).

2. Spoofing *MAC Address access point*

Untuk mendapatkan *MAC address* dari AP maupun *client*, *hacker* cukup menggunakan Airmon-ng pada sistem operasi Linux. Airmon-ng akan mengubah *wireless adapter* ke dalam modus monitor dan mulai menangkap paket-paket yang bisa dilihatnya.

Tahapan dalam melakukan *spoofing MAC address* adalah:

- Mencari AP yang akan dijadikan target
- Memilih *Client* yang akan dijadikan target
- Mengetahui *MAC Address* dari *Client* yang dipilih
- Mengganti *MAC Address* dengan menggunakan macchanger (Linux) dan SMAC (Windows).
  - Linux : macchanger wlan0 [mac *Address* target]
  - Windows : GUI pada SMAC

3. *Denial of Service* terhadap *access point* atau *Client*

Denial of Service (DoS) terjadi ketika sebuah sistem tidak dapat menangani servis ke authorized *Client* karena kekurangan *resource* oleh *unauthorized Client*. Pada *Wireless Network*, serangan DoS sangat susah untuk di cegah, di hentikan ketika serangan terjadi dan korban (*access point*) serta *Client* tidak akan merasakan serangan terjadi. Durasi serangan DoS berkisar dari milidetik hingga jam. Contoh serangan DoS pada Jaringan *Wireless* sebagai berikut :

- **Jamming Air Wave**

  Beberapa peralatan rumah tangga atau sehari-hari seperti Microwave *ovens*, *baby monitors*, dan *cordless phone* beroperasi pada frekuensi 2.4 GHz. *Hacker* dapat mengeluarkan noise dalam jumlah yang besar menggunakan peralatan tersebut dan memacetkan trafik data pada udara sehingga sinyal AP menjadi *drop* hingga sangat rendah dan mematikannya. Satu-satunya solusi untuk hal ini adalah RF *proofing* pada lingkungan yang ada.

- **Flooding with *Association*s**

  AP memasukkan data tentang *association request* yang dikirimkan oleh *client* ke sebuah tabel yang dinamakan

association table yang dikelola AP pada memorinya. IEEE 802.11 menspesifikasikan sebuah nilai maksimum 2007 concurent *association* ke sebuah AP. Besarnya tabel tergantung model AP. Ketika tabel tersebut *overflow*, AP akan menolak *client* yang akan melakukan asosiasi berikutnya.

Setelah berhasil masuk ke jaringan *wireless* melalui proses *cracking* enkripsi, *hacker* dapat melakukan otentifikasi beberapa *non-existing Client* yang terlihat *legitimate* tapi dengan MAC *Address* yang random. *Hacker* akan mengirimkan dengan *massive* sehingga tabel tersebut menjadi *overflow*.

Mengaktifkan *MAC filtering* pada AP akan mencegah serangan ini.

- ***Forged Dissociation***

  Hacker dapat mengirimka sebuah *frame* palsu *disassociation* dimana *source MAC Address* merupakan *MAC Address* AP. *Client* tetap dalam kondisi telah terotentifikasi akan tetapi hanya butuh melakukan reasosiasi dan mengirimkan *reassociation request* ke AP. AP dapat saja mengirimkan sebuah *frame reassociation response* sehingga *client* dapat meneruskan pengiriman data. Akan tetapi, *hacker* dapat meneruskan pengiriman fram *disassociation frames* pada periode yang telah ditentukan dan menyebabkan *client* akan terus menerus melakukan pengiriman *reassociation request* ke AP.

- **Forged Deauthentication**

  Hacker akan memonitor semua *frame* yang bertebaran dan memastikan *MAC address* yang akan jadi korban. Ketika sebuah data atau *association response* telah berhasil di observasi, *hacker* akan memalsukan *frame deauthentication* dimana *source MAC Address* telah dipalsukan menjadi salah satu AP. Setelah hal tersebut terjadi maka *Client* dalam keadaan *unassociation* dan *unauthenticated* dan harus

melakukan hubungan kembali. *Hacker* akan mengirimkan terus menerus *frame deauthentication* pada periode tertentu.

Untuk melakukan Forged *deassociation* dan *Deauthentication* dapat dengan menggunakan mdk3 pada linux dan *wireless card* yang mendukung *injection*.

4. *Man-in-the-Middle* (MITM)

Man-in-the-middle (MITM) adalah serangan yang mengacu pada situasi di mana seorang *hacker* pada *host* _X memasukkan *host* _X tersebut ke komunikasi antara *host* _B dan *host* _C, dan tidak B atau C satupun menyadari kehadiran X. Semua pesan yang dikirim oleh B lakukan mencapai C tetapi melalui X, dan sebaliknya. *Hacker* hanya dapat mengamati komunikasi atau memodifikasinya sebelum mengirimnya keluar. Sebuah serangan MITM dapat mematahkan koneksi yang dinyatakan aman. Pada tingkat TCP, SSH dan VPN, adalah contoh yang rentan terhadap serangan ini.

a. ARP Poisoning pada Jaringan *Wireless*

ARP *cache poisoning* merupakan masalah lama dalam jaringan kabel. Jaringan kabel telah berhasil menerapkan teknik mitigasi terhadap serangan tersebut. Tapi, teknik ARP *Poisoning* aktif kembali sejalan munculnya AP yang terhubung ke switch / hub bersama dengan klien kabel lainnya.

ARP digunakan untuk menentukan alamat *MAC* dari perangkat yang alamat IP diketahui. Terjemahan dilakukan dengan *table lookup*. ARP *cache* bertambah sejalan dengan bertambahnya *host* pada jaringan. Jika *cache* ARP tidak memiliki entri untuk alamat IP, paket IP yang *outgoing* akan di simpan dalam antrian, dan Permintaan paket ARP yang efektif ilustrasinya sebagai berikut

"Jika alamat IP Anda sesuai alamat IP target, maka silakan beritahu saya tahu apa alamat Ethernet anda.

Host dengan IP target diharapkan untuk merespon dengan ARP *reply*, yang berisi alamat *MAC* dari Host. Setelah tabel diperbarui karena menerima tanggapan ini, semua paket IP antri sekarang dapat dikirim.

Entri dalam tabel berakhir setelah waktu yang ditetapkan dalam rangka untuk memperhitungkan perubahan alamat *hardware* mungkin untuk alamat IP yang sama. Perubahan ini mungkin terjadi, misalnya, karena NIC yang diganti.

Sayangnya, ARP tidak menyediakan untuk verifikasi bahwa tanggapan dari *host* yang valid atau bahwa itu adalah menerima respon palsu seolah-olah telah mengirim Permintaan ARP. ARP *poisoning* adalah teknik serangan mengeksploitasi ketidakadaan verifikasi. Ini merusak *cache* ARP bahwa OS mempertahankan dengan alamat *MAC* yang salah untuk beberapa alamat IP. Seorang *hacker* menyelesaikan ini dengan mengirimkan paket ARP *Reply* yang sengaja dibangun dengan alamat *MAC* yang "salah". ARP adalah stateless protocol. Jadi, *Client* menerima *reply* ARP tidak dapat menentukan apakah respon merupakan akibat permintaan itu dikirim atau tidak.

ARP *poisoning* adalah salah satu teknik yang memungkinkan *man-in-the-middle*. Komputer X akan menyisipkan dirinya di antara dua *host* B dan C dengan (i) B di ”racuni”sehingga alamat IP C dikaitkan dengan alamat *MAC* X, (ii) keracunan C sehingga alamat B dikaitkan dengan alamat *MAC* X, dan (iii) menyampaikan paket X menerima.

Serangan ARP *Poisoning* berlaku untuk semua *host* dalam *subnet*. Kebanyakan AP bertindak sebagai lapisan transparan jembatan *MAC*, dan sehingga semua stasiun yang terkait dengan itu adalah rentan. Jika titik akses terhubung langsung ke hub atau switch tanpa *router firewall* / intervensi, maka semua *host* yang terhubung ke hub atau switch yang rentan juga. Perhatikan bahwa perangkat terbaru yang ditujukan untuk pasar konsumen rumah menggabungkan jaringan *switch* dengan mungkin empat atau lima pelabuhan, sebuah AP, *router* dan modem DSL / kabel yang menghubungkan ke Internet pada umumnya. Secara internal, AP terhubung ke saklar. Akibatnya, seorang *hacker* di stasiun *wireless* dapat menjadi MITM antara dua *host* kabel, satu kabel satu *wireless*, atau keduanya *host Wireless*.

Sebuah *software* seperti Ettercap (http://ettercap.*source*forge.net) pada Linux dan Cain Abel (http://www.oxid.it/cain.html) pada Windows mampu melakukan ARP *poisoning*.

b. Session Hijacking

Session Hijacking terjadi dalam konteks "pengguna", baik manusia atau komputer. Pengguna memiliki koneksi berlangsung dengan *server*. *Hijacking* dikatakan terjadi ketika seorang *hacker* menyebabkan pengguna untuk kehilangan koneksi, dan *hacker* menyamakan identitas dan hak istimewa untuk suatu periode.

Hacker menonaktifkan sementara sistem pengguna, katakan dengan serangan DoS atau mengeksploitasi *buffer overflow*. *Hacker* kemudian mengambil identitas pengguna. *Hacker* sekarang memiliki semua akses pengguna. Ketika selesai, *hacker* menghentikan serangan DoS, dan memungkinkan pengguna melanjutkan kembali aktifitas sebelumnya. Pengguna tidak akan mendeteksi gangguan jika gangguan berlangsung tidak lebih dari beberapa detik. Seperti *hijacking* dapat dicapai dengan menggunakan *disassociation* serangan DoS dipalsukan. Jaringan *wireless* perusahaan sering diatur sehingga pengguna akan diarahkan ke *server* otentikasi bila stasiun nya mencoba koneksi dengan AP. Setelah otentikasi, *hacker* menggunakan sesi *hijacking* dijelaskan di atas menggunakan alamat *MAC* palsu.

5. *War Driving*

dilengkapi dengan perangkat *wireless* dan perangkat terkait, dan mengemudi di dalam kendaraan atau parkiran di tempat-tempat umum dengan tujuan untuk menemukan jaringan *wireless* ini dikenal sebagai war-driving. war-*driver* (http://www.wardrive.net/) mendefinisikan war-driving sebagai "tindakan dalam menemukan dan *logging* titik akses *wireless*".

2.4.1 Kelemahan pada *access point*

access point memiliki kelemahan dikarenakan kesalahan dalam mendesain dan pengguna *interface* yang memiliki *password* yang lemah, dst. Hal ini telah didemonstrasikan dengan pemetaan secara umum dari usaha war-driving (www.worldwidewardrive.org) pada beberapa kota di seluruh dunia yang memiliki konfigurasi *access point* yang sangat sederhana dan memiliki WEP bahkan tidak sekalipun, konfigurasi administrasi yang *default* dsb.

Default WEP key yang digunakan terkadang sangatlah berbahaya. AP yang berbeda memiliki teknik yang berbeda pula untuk mengubah inputan *keyboard* dari pengguna menjadi sebuah *bit vector*. Biasanya menggunakan 5 hingga 13 ASCII karakter yang telah di petakan langsung dengan memotong ASCII 8-*bit* menjadi sebuah 40-*bit* atau 104-*bit* WEP key. Kunci yang kuat dapat dibentuk dari 26 heksadesimal. Sangat mungkin untuk membuatnya lebih kuat dari 104 *bit* WEP. Namun, penggunaan WEP key sudah tidak dapat dikatakan sebuah keaamanan lagi, dikarenakan telah terpecahnya algoritma enkripsi WEP. Setidaknya menggunakan WEP key lebih baik daripada tidak menggunakannya sama sekali.

Biasanya AP memberikan akses hanya pada *MAC address* yang telah terdaftarkan. Hal ini sangat mudah untuk di tembus oleh *hacker* yang dapat melakukan *spoofing frame* dengan *MAC address* yang telah terdaftar di AP yang didapatkan dari hasil *monitoring* aktifitas *sniffing* pada jaringan tersebut. Hal itu mungkin, karena *frame-frame* yang berterbangan di udara memiliki informasi *MAC Address* dan tentunya dikarenakan dengan bebas *broadcast* maka dapat dengan mudah di tangkap oleh *hacker*.

access point yang telah di pasang tanpa oterisasi dan verifikasi yang benar serta kebijakan keamanan yang tidak dipatuhi disebut ***Rogue Access Point*** atau biasanya dikenal dengan nama *access point* palsu.

Hacker dapat menguatkan sinyal *Rogue Access Point* daripada *access point* yang sebenarnya. Hasil dari penguatan sinyal tersebut akan menyebabkan *Client* akan memilih *Rogue Access Point* tersebut dan melakukan otentifikasi serta asosiasi. Istilah untuk ini adalah Trojan *access point*. Dengan *Client* terkoneksi pada *Rogue Access Point* tersebut maka dengan mudah *hacker* melakukan *Man-*

in-the-middle dan menangkap semua paket-paket yang dikirimkan oleh *Client*. *Hacker* dapat saja mengambil *password*, akses *network*, menyerang sistem sehingga mendapatkan tingkat *Root*/*Administrator*. Serangan seperti ini dikenal dengan *Evil Twin Attack.*

Sangat mudah untuk membuat sebuah *Rogue Access Point* karena sebuah *access point* adalah sebuah sistem komputer yang telah di optimasikan untuk tujuan tertentu. Sebuah PC dengan *Wireless Card* dapat berubah menjadi sebuah AP. Sebagai contoh *software* yang dapat mewujudkan hal tersebut adalah HostAP (http://*host* ap.epitest.fi/), Connectify (http://www.connectify.com/), FakeAP, bahkan dengan AirBase pada linux juga dapat menciptakan AP.

Kesalahan-kesalahan pada peralatan dan desain *hardware* juga dapat terjadi bahkan pada produsen terkenal. Sebagai contoh, satu AP akan *crash* ketika sebuah *frame* dikirim untuk itu yang memiliki sumber alamat *MAC* palsu dari dirinya sendiri. AP lainnya memiliki fitur tertanam seperti TFTP (*Trivial File Transfer Protocol*) *server*. Dengan meminta sebuah file bernama config.img melalui TFTP, *hacker* menerima citra biner dari konfigurasi AP. Hal tersebut mencakup *password Administrator* yang diminta oleh pengguna *interface* HTTP, kunci enkripsi WEP, alamat *MAC*, dan SSID. AP lain mengembalikan kunci WEP, *MAC* menyaring daftar, *password Administrator* saat mengirim paket UDP ke *port* 27155 yang mengandung *string* "gstsearch".

Tidak jelas bagaimana kekurangan tersebut ditemukan. Kebanyakan produsen merancang peralatan mereka sehingga *firmware*-nya dapat melintas dengan yang baru dan lebih baik. Image *firmware* di-*download* dari situs web produsen. CPU yang digunakan dalam AP dapat dengan mudah dikenali, dan *firmware* dapat dibongkar sistematis mengungkapkan kekurangan hingga tingkat bahasa *assembly*.

2.4.2 Tipe-tipe *Rogue Access Point*

Sebuah *Rogue Access Point* memiliki pola yang beragam. Sebagai contoh yang paling popular termasuk pengunaan SOHO *wireless* router atau mengkonfigurasi sendiri sebuah *wireless divais* atau *client*, akan tetapi terdapat beberapa konfigurasi dari *Rogue Access Point* yaitu :

- ***Wireless Router* terkoneksi melalui *interface* yang "trusted"**

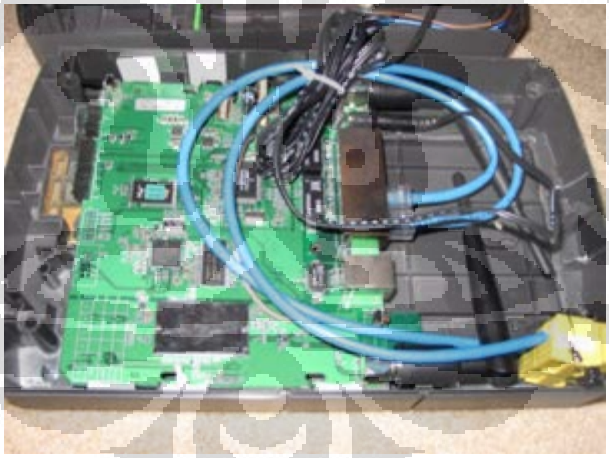
  Pada konfigurasi ini *Rogue Access Point* terkoneksi ke sisi router yang terkoneksi ke jaringan internal. Sebuah DHCP *Server* telah diaktifkan di *Rogue*AP dan dapat menyebabkan bentrokan dengan DHCP *server* yang ada pada jaringan internal.

- ***Wireless Router* terkoneksi melalui *interface* yang "untrusted"**

  Pada konfigurasi ini *Rogue Access Point* terpasang di sisi eksternal jaringan. Jika hal itu terjadi maka kemungkinan untuk mendeteksi sangat kecil.

- **Memasang sebuah kartu *wireless* ke sebuah *divais* yang telah terpasang di jaringan LAN**

  Untuk hal ini dibutuhkan akses fisik ke sistem yang telah terotentikasi dengan sah. Hacker dapat dengan mudah memasang *wireless card* ke sistem yang sah dan mengkonfigurasinya menjadi akses poin dimana fungsi ini di dukung oleh kebanyakan *wireless* chipset, *driver* dan sistem operasi seperti Windows, Linux dan *MAC* OS X.



Gambar 2.13 *Rogue Access Point* Inside UPS [18]

Pada gambar diatas menunjukkan sebuah *Rogue Access Point* yang terinstall di sebuah *desktop* APC UPS. Divais tersebut adalah sebuah *Wireless Router* Linksys WRT54G dengan kemampuan 802.11b/g.

2.5 *Wireless Intrusion Detection System*

2.5.1 Pengertian *Intrusion Detection System*

Penggunaan jaringan *wireless* berkembang dengan sangat cepat. Dengan adanya jaringan *wireless, user* dapat lebih mudah dan nyaman mengakses internet. Penggunaan *client* tidak terbatas dengan adanya kabel seperti layaknya jaringan kabel. Untuk menjaga jaringan *wireless* dari hal-hal yang tidak diinginkan, administrator harus memonitor kejadian-kejadian apa saja yang terjadi di jaringannya dan mendeteksi adanya serangan atau tidak. Untuk melakukannya, administrator harus meng-*install* sebuah *Wireless Intrusion Detection System* (WIDS).

2.5.2 Cara Kerja *Intrusion Detection System*

Secara umum, *Instrusion Detection System* (IDS) didesain dan dibuat untuk memonitor dan melaporkan aktivitas yang berlangsung dalam jaringan kepada administrator. WIDS membutuhkan sensor untuk mengumpulkan data-data dalam jaringan, *server* untuk mengolah data-data yang telah dikumpukan, dan *client* untuk menampilkan hasil dari pengolahan data yang telah dikumpulkan. Interface WIDS ditempatkan pada *client* WIDS.

WIDS dapat mendeteksi serangan pada *frame* 802.11 pada layer dua dari jaringan *wireless.* Ada tiga tipe *frame MAC* 802.11, yaitu *frame* data, kontrol, dan manajemen. Mayoritas serangan *wireless* menjadikan *frame* manajemen sebagai targetnya karena *frame* ini bertugas untuk melakukan autentikasi, asosiasi, disasosiasi, *beacon,* dan *Probe request/Response*. Serangan *wireless* seperti *Man-in-the-Middle* (MIM), *rogue* AP, *war driver*, dan *denial-of-service* berjalan pada *frame* 802.11 dan tidak bisa dideteksi pada layer tiga. IDS biasa (pada jaringan kabel) tidak dapat menerima *frame* ini, karena *frame* manajemen tidak dapat diteruskan ke layer di atasnya.

WIDS membutuhkan *interface* khusus. *Interface wireless* ini harus dioperasikan pada mode monitor, yang dikenal juga sebagai mode RFMON. Mode ini membolehkan *divais* untuk menerima semua lalu lintas yang masuk. Interface yang bertugas memonitor harus terus berganti *channel*, dikenal dengan *channel hopping*, yang tersedia pada jaringan tersebut. Beberapa serangan

wireless bekerja dengan menggunakan *rogue* AP pada *channel* yang berbeda. Sebagai contoh, serangan MIM menggunakan *rogue* AP yang paling sedikit berbeda lima *channel* dari target AP. Tanpa *channel hopping*, WIDS dapat luput terhadap serangan yang dilakukan pada *channel* lain. Walaupun begitu, deteksi ini hanya dapat berjalan pada jaringan *wireless* dengan satu AP saja, karena jaringan yang lebih besar akan mempunyai beberapa AP yang dikonfigurasi pada *channel* yang berbeda untuk menghindari interferensi frekuensi radio dengan AP yang lain.

Untuk mendeteksi serangan pada *range* tertentu, WIDS mencocokkan serangan dengan metodologi deteksi *signature-based*, *knowledge-based*, dan analisis protokol *stateful*. Deteksi *signature-based* menggunakan *signature* statik untuk mencocokkan lalu lintas yang mencurigakan. Tipe pencocokan ini bekerja dengan baik untuk serangan-serangan yang telah dikenal sebelumnya dan cocok dengan pola tertentu. Sebagai contoh, untuk mendeteksi *rogue* AP, WIDS menggunakan daftar AP yang sah dan memberikan sinyal jika ada AP yang terdeteksi tidak cocok dengan daftar. Deteksi *knowledge-based* menggunakan acuan dasar historis dan memberikan sinyal ketika lalu lintas jaringan berbeda dari acuan dasar historis. Beberapa serangan *wireless* tidak cocok dengan *signature* tetapi serangan ini dapat menyebabkan anomali lalu lintas jaringan.

Analisis protokol *stateful* merupakan proses membandingkan profil yang sudah ditentukan sebelumnya. Analisis protokol *stateful* bergantung pada profil yang dikembangkan vendor secara universal yang menspesifikasikan bagaimana protokol tertentu seharusnya digunakan. Kata "*stateful*" berarti IDS mampu memahami dan mengikuti keadaan protokol *network*, *transport*, dan *application*. Sebagai contoh, ketika *user* memulai sesi *File Transfer Protocol* (FTP), sesi pertama-tama berada pada keadaan yang belum sah. User yang tidak sah seharusnya hanya dapat melakukan perintah-perintah tertentu pada keadaan ini, seperti melihat informasi bantuan. Bagian penting memahami keadaan adalah mencocokkan *request* dengan *Response*, jadi ketika autentikasi FTP terjadi, IDS dapat menentukan apakah sukses atau tidak dalam menemukan kode status dalam respon yang sesuai. Ketika *user* sudah berhasil diautentikasi, sesi ada pada

keadaan sah, dan *user* dapat melakukan beberapa perintah lainnya. Melakukan hampir semua perintah ketika masih dalam keadaan tidak sah dapat dipertimbangkan sebagai keadaan yang mencurigakan.

2.5.3 Kismet *Wireless* IDS

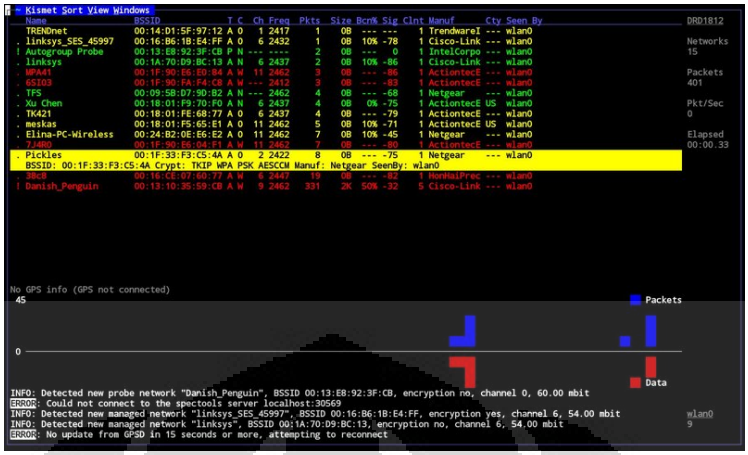
Kismet *wireless* merupakan sebuah aplikasi analisa jaringan. Kismet mengidentifikasi jaringan dengan cara mengumpulkan paket dan mendeteksi jaringan secara pasif. Kismet dapat mendeteksi nama SSID tersembunyi dan keberadaan jaringan *non-beaconing* melalui lalu lintas data.

Kismet termasuk dalam kategori *passive sniffer*. Tidak seperti NetStumbler dimana melakukan *broadcast* permintaan ke *Access Point* yang memiliki nama SSID "ANY", kismet tidak mengirimkan paket *broadcast* tersebut. Kismet bekerja dengan memberikan dengan cara mengubah *wireless client* adapter ke RF Monitor Mode. Biasanya dikenal dengan "rfmon" mode.

Kismet dapat dijalankan dengan *wireless card* apapun yang mendukung mode *raw monitoring*, dan dapat men-*sniff* lalu lintas 802.11b, 802.11a, 802.11g, dan 802.11n. Kismet juga mendukung arsitektur *plugin* yang memperbolehkan protokol non-802.11 untuk di-*decode*.

Kismet juga tersedia untuk versi Windows, yaitu Cygwin, tetapi fitur-fitur yang ditawarkan untuk Linux lebih banyak daripada Windows. Kismet dapat diintegrasikan dengan alat GPS untuk menggambarkan koordinat jaringan yang terdeteksi.

Berikut ini merupakan tampilan utama Kismet.



Gambar 2.14 Tampilan utama Kismet

Tampilan utama Kismet dibagi menjadi 3 bagian, yaitu:

1. *Network List,* yang memperlihatkan semua jaringan *wireless* yang terlihat.
2. Info, yang terletak pada sisi kanan, menunjukkan rangkuman mengenai paket-paket yang dilihat oleh Kismet.
3. Status, yang memperlihatkan status *real time* dari Kismet.

Warna-warna pada *Network List* mempunyai makna tertentu, yaitu:

Warna	**Arti**
Kuning	jaringan yang tidak dienkripsi
Merah	jaringan yang masih menggunakan konfigurasi *default* dari vendor
Hijau	menandakan jaringan yang relatif aman karena sudah menggunakan enkripsi, baik WEP, WPA, atau WPA2
Biru	menandakan jaringan yang tidak menyertakan informasi SSID dalam paket *beacon*-nya atau dapat disebut jaringan *wireless* yang dijalankan dalam modus tersembunyi

Tabel 2.3 Makna warna pada *Network* List Kismet

Simbol-simbol dalam T (*Type*) berarti:

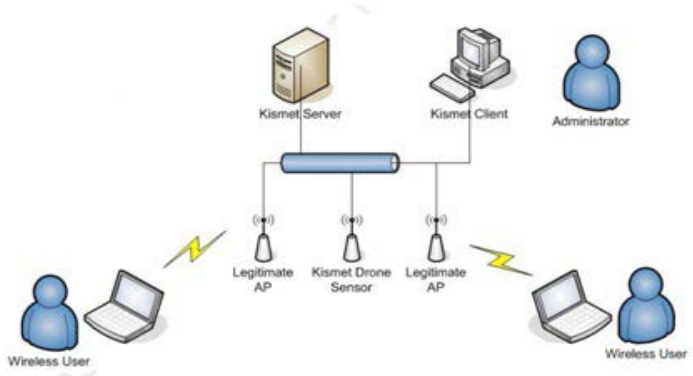
Simbol	Singkatan dari	Arti
P	Probe request	Belum ada asosiasi
A	Access Point	Jaringan wireless standar
H	Ad-hoc	Jaringan wireless point-to-point
T	Turbocell	Turbocell alias Karlnet atau Lucent Router
G	Group	Grup jaringan wireless
D	Data	Data hanya jaringan dengan tidak ada control packet

Tabel 2.4 Makna simbol dari Tipe pada *Network List* Kismet *Wireless*

Ch (*Channel*) berarti *channel* yang dipakai jaringan tersebut. Pkts (*Packets*) berarti jumlah paket yang ditangkap. Size berarti data yang ditangkap oleh Kismet.

Kismet *wireless* telah dilengkapi dengan Kismet *drone* yang membuatnya menjadi aplikasi WIDS. Arsitektur Kismet terdiri dari:

- Kismet *drone*: mengumpulkan data-data dalam jaringan dan mengirimkannya ke Kismet *server*
- Kismet *server*: mengolah data-data yang telah dikumpukan
- Kismet *client*: menampilkan hasil dari pengolahan data yang telah dikumpulkan



Gambar 2.15 Arsitektur Kismet WIDS [16]

Perangkat-perangkat yang dibutuhkan agar dapat menjalankan Kismet WIDS ini yaitu:

- Komputer (Kismet *server* dan Kismet *client* dapat menggunakan 1 komputer)
- *Network wireless* adapter
- *Wireless Router* (sebagai *drone*)

Konfigurasi Kismet dapat diubah untuk mengontrol *channel* dan pola loncatan untuk menangkap *source* di Kismet. Kismet *plugin* dapat melakukan hampir semua hal yang dapat dilakukan oleh Kismet asli. Hal ini termasuk menambah kapabilitas *logging*, menambah sinyal (*alert*) IDS, mendefinisikan *capture source* yang baru, dan menambah fitur baru ke Kismet *User Interface* (UI). Kismet dapat diintegrasikan dengan *divais* GPS untuk memberikan koordinat jaringan yang telah terdeteksi.

Kismet membolehkan 1 sinyal per detik dan maksimum 5 sinyal per menit. Kismet dapat membaca dengan metode *fingerprint* (serangan paket tunggal spesifik) dan *trend* (*Probe* yang tidak biasa, *disassociation* floods, dll). Kismet dapat diintegrasikan dengan aplikasi lain menggunakan tun/tap ex*port* untuk menyediakan *interface* virtual jaringan dari lalu lintas *wireless*. Kismet mendukung sinyal-sinyal di bawah ini:

1. AIRJACKSSID (*fingerprint*)

   Airjack adalah aplikasi *hacking* 802.11. Airjack telah lama tidak dilanjutkan pembuatannya, sehingga bisa dikatakan sinyal ini sudah tidak relevan lagi.

2. APSPOOF (*fingerprint*)

   Jika respon dari *beacon* atau *Probe* untuk SSID yang dilihat dari *MAC address*-nya tidak ada di daftar yang diperbolehkan mengakses AP, sinyal ini akan diaktifkan. Sinyal ini dapat digunakan untuk mendeteksi AP yang konflik, *spoofed* AP, atau serangan seperti Karma/Airbase dimana serangan ini merespon ke semua *Probe request*. Dengan Karma/Airbase, *client* dapat menjadi target dengan membuat *rogue* AP. Karma sekarang dikembangkan dengan nama Karmetasploit.

3. BSSTIMESTAMP (*trend/stateful*)

   Invalid/out-of-sequence BSS *timestamp* dapat mengindikasikan AP *spoofing*. AP dengan BSS *timestamp* yang berfluktuasi dapat mengindikasikan serangan *"evil twin" spoofing*, karena beberapa aplikasi tidak dapat mengsinkronkan BSS *timestamp* sama sekali. *Timestamp* bertugas mengsinkronkan antar *station* dalam BSS.

   Evil twin adalah *wireless* AP palsu yang menyamar menjadi AP asli dengan menyebarkan nama WLAN-nya, yaitu ESSID dan SSID. *Evil twin* dapat menggunakan Karma, sebuah aplikasi untuk memonitor *Probe station*, mengamati SSID yang biasa digunakan dan mengadopsi salah satunya. Penyerang juga dapat menggunakan SSID yang umum digunakan seperti SSID *default* dari vendor. Bahkan AP yang tidak mengirim SSID di *beacon*-nya dapat menjadi target, selama *user* asli dapat dimonitor dengan Wireshark, Kismet, atau WLAN *analyzer* yang lain.

4. CHANCHANGE (*trend/stateful*)

   AP yang sudah terdeteksi tiba-tiba berganti *channel*. Hal ini dapat mengindikasikan serangan *spoofing*. Dengan *spoofing* AP asli di *channel* yang berbeda, penyerang dapat menarik *client* ke *spoofed* AP. Sebuah AP

yang berganti *channel* selama operasi normal dapat mengindikasikan serangan ini sedang dalam proses.

5. CRYPTODROP (*trend/stateful*)

   Spoofing sebuah AP dengan enkripsi yang kurang aman dapat membodohi *client* agar membuat koneksi. Satu-satunya situasi dimana sebuah AP harus mengurangi keamanan enkripsinya adalah pada saat AP dikonfigurasi ulang.

6. DEAUTHFLOOD dan BCASTDISCON (*trend/stateful*)

   Dengan *spoofing disassociate* dan *deauthenticate packet*, seorang penyerang bisa memutuskan *client* dari jaringan, menyebabkan DoS dimana hanya berlangsung selama penyerang bisa mengirim paket.

7. DHCPCLIENTID (*fingerprint*)

   Client yang mengirim paket DHCP DISCOVER berisi *Client*-ID *tag* (*Tag* 61), dimana tidak cocok dengan sumber *MAC* dari paket, dapat melakukan DHCP DoS untuk menghabiskan *resource* DHCP *pool*.

8. DHCPCONFLICK (*trend/stateful*)

   Client yang sudah menerima DHCP *address* dan melanjutkan untuk menggunakan IP *address* yang berbeda dapat mengindikasikan kesalahan konfigurasi atau *spoofed client*.

9. DISASSOCTRAFFIC (*trend/stateful*)

   Client yang putus dari jaringan seharusnya tidak secepatnya melanjutkan pertukaran data. Hal ini dapat mengindikasikan *spoofed client* mencoba untuk memasukkan data ke jaringan dengan cara yang tidak benar, atau dapat mengindikasikan *client* sedang menjadi korban dari serangan DoS.

10. DISCONCODEINVALID dan DEAUTHCODEINVALID (*fingerprint*)

    Spesifikasi 802.11 mendefinisikan *valid reason codes* untuk kejadian *disconnect* dan *deauthenticate*. Berbagai *driver client* dan AP telah dilaporkan salah dalam memperlakukan *invalid/undefined reason code*.

11. DHCPNAMECHANGE dan DHCPOSCHANGE (*trend/stateful*)

    Protokol konfigurasi DHCP membolehkan *client* secara optimal memasukkan sistem operasi/vendor *host name* dan DHCP *client* dalam paket DHCP Discover. Nilai ini hanya berubah jika *client* tiba-tiba baru

saja mengganti sistem (sistem *dual-boot*). Mengganti nilai ini sering mengindikasikan serangan *spoofing*/*MAC cloning*.

12. LONGSSID (*fingerprint*)

    Spesifikasi 802.11 membolehkan maksimum 32 *byte* untuk SSID. Kelebihan nilai SSID mengindikasikan serangan dimana penyerang berusaha untuk mengeksploitasi vulnerabilitas dari beberapa *driver*.

13. LUCENTTEST (*fingerprint*)

    Lucent Orinoco *card* lama yang dalam mode *scanning* test tertentu menghasilkan paket yang bisa diidentifikasi.

14. MSFBCOMSSID (*fingerprint*)

    Beberapa versi *wireless driver* Broadcom Windows tidak dapat memperlakukan SSID *field* secara benar lebih lama daripada spesifikasi 802.11, menyebabkan *system compromise* dan *code execution*. *System compromise* adalah penggunaan sistem secara ilegal oleh yang bukan pemilik sistem tersebut. *Code execution* adalah mengeksekusi perintah apapun yang diinginkan penyerang pada targetnya. Vulnerabilitas ini dieksploitasi oleh Metasploit *frame*work.

15. MSFDLINKRATE (*fingerprint*)

    Beberapa versi *wireless driver* D-Link Windows tidak dapat secara benar mengurus 802.11 *valid rate field* yang sangat panjang, menyebabkan *system compromise* dan *code execution*. Vulnerabilitas ini dieksploitasi oleh Metasploit *frame*work.

16. MSFNETGEAR*BEACON* (*fingerprint*)

    Beberapa versi *wireless driver* Netgear Windows tidak dapat secara benar mengurus kelebihan *frame* dari *beacon*, menyebabkan *system compromise* dan *code execution*. Vulnerabilitas ini dieksploitasi oleh Metasploit *frame*work.

17. NETSTUMBLER (*fingerprint*)

    Versi lama dari Netstumbler (3.22, 3.23, 3.30) menghasilkan, dalam kondisi tertentu, paket khusus.

18. NULLPROBERESP (*fingerprint*)

    Paket *Probe Response* dengan komponen SSID IE tag dengan panjang 0 dapat menyebabkan *card* lama (prism2, orinoco, air*port*-classic) gagal.

19. PROBENOJOIN (*trend/stateful*)

    Aplikasi *scanning* aktif seperti Netstumbler terus-menerus mengirim *network discovery probes* tetapi tidak pernah bergabung dengan jaringan yang merespon. Sinyal ini dapat menyebabkan *false positive* yang terlalu banyak ketika *channel hopping*, dan sinyal ini di-nonaktifkan secara *default*.

Kismet *wireless* IDS dapat digunakan untuk mendeteksi *rogue* AP, serangan deautentikasi, *spoofing* AP, dan *Active scanning*. WIDS ini dapat menggunakan lebih dari satu sensor sehingga penggunaannya dapat dikembangkan sendiri tergantung kebutuhan administrator. Kismet *wireless* IDS mempunyai kekurangan yang sangat menyulitkan administrator, yaitu dalam hal pembacaan sinyal. Kismet tidak menulis sinyal IDS-nya ke file log, tetapi menampilkannya melalui *rolling* log dalam Kismet *client*. Hal ini membuat admonistrator harus terus berada di depan Kismet *client* untuk membaca arus sinyal yang masuk. Oleh karena itu, dibutuhkan perangkat lunak lain yang dapat menampilkan sinyal-sinyal dari Kismet *wireless* IDS ini.

2.6 Tipe frame 802.11

Dalam menganalisa suatu operasi pada jaringan *wireless* LAN, maka kita akan menggunakan *software* 802.11 packet analyzer (airopeek atau sniffer *wireless*) untuk memonitor komunikasi antara *interface wireless* (NIC) dengan *Access Point*. Setelah menangkap paket, kita harus memahami perbedaan antara 802.11 tipe frame.

Terdapat 3 tipe frame dalam *wireless* LAN yaitu Management Frame, Control Frame, dan Data Frame.

802.11 management frame memperbolehkan *client* untuk membuat dan menjaga komunikasi. Berikut adalah beberapa subtype dari 802.11 management frame :

a. Authentication Frame

802.11 authentication adalah proses dimana akses poin menerima atau tidak identitas yang dikirimkan oleh NIC. NIC akan memulai proses dengan mengirimkan sebuah authentication frame berisikan identitasnya ke akses poin. Dengan Open system authentication (*default* configuration pada AP), NIC *client* hanya akan mengirimkan satu authentication frame dan akses poin akan merespon dengan sebuah authentication frame sebagai respon yang menandakan penerimaan atau penolakan. Dengan menggunakan sistem enkripsi, NIC harus mengirimkan versi enkrip dari challenge text (menggunakan WEP key) pada authentication frame ke AP. AP akan memastikan NIC mengirimkan WEP key yang benar dengan membandingkannya.

b. Deauthentication Frame

Deauthentication Frame merupakan frame yang digunakan oleh *client* untuk memutuskan sebuah komunikasi dengan AP.

c. Association Request Frame

Association Request Frame adalah frame permintaan kepada AP untuk mengalokasikan resource untuk di sinkronisasikan dengan NIC *client*. Frame ini berisikan informasi mengenai NIC (contohnya data rate yang didukung). Setelah menerimanya AP akan melakukan asosiasi dengan NIC dan jika diterima maka AP akan menyediakan memory space dan membangun asosiasi ID terhadap NIC *client*.

d. Association *Response* Frame

Association *Response* Frame adalah frame yang dikirim oleh AP berisikan penerimaan atau penolakkan terhadap NIC.

e. Reassociation Request Frame

Jika sebuah NIC roaming dari AP yang berasosiasi dan mencari AP lainnya dengan sinyal beacon yang lebih kuat, NIC akan mengirimkan reassociation frame ke AP yang baru tersebut.

f. Reassociation *Response* Frame

Frame yang dikirim oleh AP ke *client* yang berisikan penerimaan atau penolakkan terhadap permintaan reassociation dari *client*.

g. Dissassociation Frame

Frame yang digunakan oleh *client* untuk memberitahukan bahwa NIC *client* telah dimatikan sehingga AP dapat membebaskan alokasi memory untuk *client* tersebut.

h. Beacon Frame

AP akan mengirimkan secara periodik sebuah beacon frame untuk memberitahukan keberadaannya dan informasi, seperti *timestamp*, SSID, dan parameter lain tentang AP ke NIC radio yang berada di jangkauannya. NIC *client* akan terus melakukan scanning terhadap semua 802.11 radio channel dan mendengarkan beaon sebagai basis untuk memilih akses poin mana yang paling tepat untuk berasosiasi.

i. Probe Request Frame

Sebuah *client* akan mengirimkan sebuah probe request frame untuk mendapatkan informasi dari stasiun lainnya. Sebagai contoh, sebuah *client* akan mengirimkan probe request untuk mendeterminasikan keberadaan AP dalam area tersebut.

j. Probe *Response* Frame

Sebuah stasiun akan merespon dengan sebuah probe response frame, berisikan kapabilitas informasi, supported data rates, dll.

Control frame membantu dalam menyampaikan frame data antar *client*. Berikut adalah beberapa 802.11 control frame subtype:

a. Request to Send (RTS) Frame

Fungsi RTS/CTS adalah opsional dan mengurangi tabrakan frame ketika hidden *client* telah berasosiasi pada satu AP. Sebuah *client* mengirimkan sebuah RTS frame ke *client* lainnya sebagai fase pertama dari two-way handshake yang diperlukan sebelum mengirimkan sebuah data frame.

b. Clear to Send (CTS) Frame

Client akan merespon ke RTS dengan CTS frame, memberitahukan keadaan bebas untuk melakukan pengiriman frame data. CTS berisikan sebuah nilai waktu yang mengakibarkan *client-client* yang lain menahan pengiriman frame terhadap sebuah periode waktu tertentu.

c. Acknowledgement Frame

Setelah menerima sebuah data frame, *client* yang menerima akan mengutilisasi sebuah pengecekan kesalahan proses untuk mendeteksi keberadaan error. Jika *client* pengirim tidak mendapatkan sebuah ACK setelah periode tertentu, makan *client* pengirim akan mengirimkan kembali frame data tersebut.

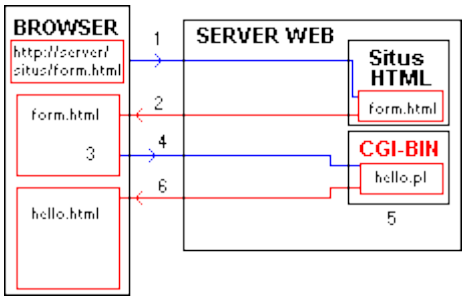
Data frame merupakan data yang akan dikirimkan oleh *client* melalui NIC *wireless*.

2.7 Bahasa Pemrograman *Server* Side - PHP

PHP (Hypertext Preprocessor), merupakan bahasa pemrograman web bersifat *server*-side, artinya bahasa berbentuk *script* yang disimpan dan dijalankan di komputer *server* (*WebServer*) sedang hasilnya yang dikirimkan ke komputer *Client* (*Web*Browser) dalam bentuk *script* HTML (*Hypertext Mark up Language*). Karakteristik *script* PHP dapat diuraikan sebagai berikut :

- File PHP disimpan dengan extensi filenya yaitu : *.php3, *.php4, *.php
- Script PHP biasanya diawali dengan tag ‘<?’ atau ‘<?php’ dan ditutup dengan tag ‘?>’
- *File* PHP dapat menginduk atau disisipkan pada bahasa *script* lainnya atau dapat berdiri sendiri. Contoh skrip PHP yang disisipkan pada HTML.

Pada level dasar, PHP dapat melakukan semua apa yang dapat dilakukan oleh pemrograman berbasis CGI (*Common Gateway Interface*) lainnya. CGI *Common Gateway Interface*, adalah suatu Antarmuka (*interface*) untuk menjalankan program dari luar, *software* atau *gateaway* di *Server*.



Gambar 2.16 Gambaran tentang konsep CGI [16]

PHP memiliki ratusan fungsi dasar dan ribuan lebih via tambahan ekstensi. Fungsi-fungsi tersebut terdokumentasi dengan sangat baik disitusnya http://php.net. PHP tidak memiliki fungsi untuk *thread programming* meskipun begitu PHP sendiri mendukung multiprocessing *programming* pada POSIX *System*.

PHP sendiri dapat mengakses fungsi-fungsi eksternal dari bahasa program lain dan bahkan menjalankan perintah-perintah sistem operasi diluar programnya sendiri.

Terdapat 4 opsi utama untuk menjalankan program eksternal dari PHP yaitu :

- **Fungsi *System*()**

  Fungsi *System*() mengambil argumen dengan perintah yang akan dijalankan. Fungsi ini menjalankan perintah yang spesifik dan mengeluarkan isinya ke *output stream* (HTTP *output* atau console pada command *line tool*). Hasil dari fungsi ini adalah baris terakhir dari *output* dari program, jika yang dihasilkan oleh program adalah teks.

- **Fungsi exec()**

  Fungsi exec() merupakan fungsi yang cukup berguna dan *power*ful, tapi masalah terbesarnya adalah semua hasil text dari program akan di kirimkan langsung ke *output stream*. Akan tetapi hal itu dapat di ubah dengan operator manipulasi *string* sehingga hanya menampilkan *output* apa yang diinginkan.

- **Fungsi shell_exec()**

  Program-program yang telah di eksekusi pada contoh-contoh sebelumnya kurang lebih adalah program sebenarnya. Namun, lingkungan pada *Windows* atau Unix memiliki lebih dari

program-program tersebut. *Windows* memiliki opsi dengan menggunakan *Windows Command* Prompt program yaitu cmd.exe. Program ini di kenal dengan *command shell*.

- **Fungsi passthru()**

  Fungsi yang sangat mengagumkan yang dapat diberikan oleh PHP untuk mengontrol program adalah fungsi passthru. Fungsi ini tidak seperti yang lain, mengeksekusi program yang kita sebutkan.

  Fungsi ini sangat berguna ketika kita bekerja pada gambar dan akan mengeksekusi program untuk menunjukkan atau memanipulasi *File* tersebut.

Fungsi-fungsi seperti *System*(), exec(), *shell*_exec() dan passthru() sangat berguna dalam melakukan aktifitas ping dan dns *lookup* oleh *Server* yang mana akan digunakan untuk menentukan *Round Trip Time* antara *Server* dan *Client* yang terkoneksi dan menentukan apakah *Client* terkoneksi pada jaringan yang benar atau tidak.

2.7 Linux *Server*

2.7.1 Pengenalan Linux

Linux merupakan sebuah Sistem Operasi yang memiliki sifat Unix-Like di rancang dibawah *Open Source Software Developtment* and Distribution. Komponen dari Linux *System* adalah Linux *Kernel*.

Beberapa Distribusi OS Linuxyang terkenal seperti Debian (Contoh turunannya adalah Ubuntu), Fedora dan openSuse. Linux *server* biasanya telah mengemas paket servis seperti *Web*, *FileSharing*, Administrasi Jaringan dan lain-lain.

Programming di linux sangatlah bervariatif, *original development tool*s yang digunakan untuk membangun aplikasi Linux dan program sistem operasi ditemukan dalan GNU *tool*chain, yang mana sudah termasuk GNU *Compiler* Collection (GCC) dan GNU build *System*. GCC menawarkan *compiler* untuk Ada, C, C++, Java dan Fortran.

Beberapa Linux juga mendukung secara *native* bahasa pemrograman lain seperti PHP, Perl, Ruby, Python dan bahasa pemrograman dinamis lainnya.

2.7.2 File*System* Hierarchy Linux

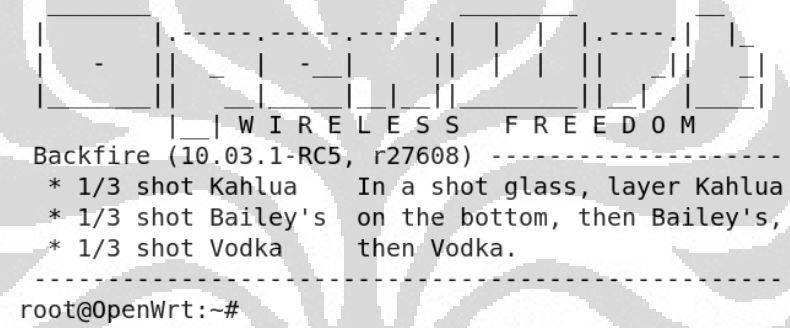
File System Hierarchy (FHS) mendefinisikan direktori utama dan isinya pada Sistem Operasi Linux. FHS di berdayakan oleh Linux Foundation. Versi terakhir pada Januari 2004 adalah 2.3

Pada distribusi Linux modern terdapat direktori /sys sebagai sebuah *virtual FileSystem*, dimana menyimpan dan memperbolehkan perubahan pada alat-alat yang terhubung ke sistem.

2.8 OpenWrt

OpenWrt merupakan sebuah distribusi sistem operasi Linux untuk embedded system.

sadsa



Gambar 2. 17 Tampilan menu awal OpenWrt

OpenWrt dapat berjalan hampir disemua embedded device seperti CPE router, smartphone, pocket computer, laptop, termasuk *Access Point*.

Gambar 2. 18 OpenWrt WebGUI - LuCI

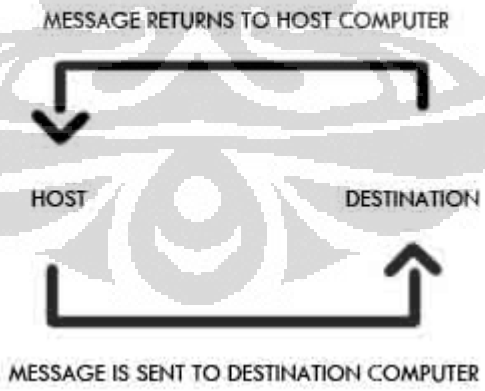
2.9 RTT

Round-trip Time (RTT) adalah waktu yang dibutuhkan sebuah *Client* untuk mengirimkan permintaan dan sebuah *server* untuk mengirimkan balas respon melalui jaringan, tidak termasuk waktu yang dibutuhkan untuk pengiriman data. Sederhanannya RTT adalah waktu ping. Pengguna Internet dapat menentukan RTT dengan menggunakan perintah ping.

Ketika sebuah RTT baru selesai di kalkulasikan, maka akan dimasukkan ke perhitungan diatas untuk mendapatkan rata-rata RTT pada koneksi tersebut dan prosedur akan terus dilakukan tiap kali ada kalkulasi RTT baru.

Contoh RTT, pada sebuah *browser* yang akan inisiasi koneksi pertama kali ke sebuah *Web Server*, *browser* tersebut harus melakukan setidaknya minimal 3 RTT yaitu RTT untuk DNS (*name resolution*), RTT untuk TCP (connection setup) dan RTT untuk HTTP *Request* dan *byte* pertama dari HTTP respon.

RTT menggunakan Ping seringkali digunakan dalam mengukur sebuah kinerja dan performansi sebuah jaringan komputer. Ping atau singkatan dari Packet InterNet Grouper memiliki karakteristik unik. Ping melakukan pengiriman sebuah paket yang berisikan ICMP ECHO_REQUEST ke sebuah komputer tujuan, yang mana komputer tujuan ini akan mengembalikan paket tersebut dengan paket yang berisikan ECHO_REPLY..



Gambar 2. 19 Ilustrasi ping

DNS (*Domain Name Service*) merupakan sebuah layanan dalam membagikan sebuah *database* terdistribusi yang memetakan nama *host* yang dapat dibaca dengan mudah oleh manusia (contoh: www.ee.ui.ac.id) ke sebuah IP

Address (contoh: 152.118.101.8). Mesin yang memberikan layanan ini disebut DNS *Server* yang akan menjawab permintaan DNS *lookup* dari *client*.

DNS *lookup* biasanya diklasifikasikan menjadi 2 tipe yaitu Recursive *Query* dan Non-Recursive *Query*. Pada recursive DNS *lookup*, sebuah *query* dari *client* ke sebuah *local server* untuk sebuah *host name*. Jika *server* tersebut tidak menjawab *query*, maka *Server* akan menghubungi *root* DNS *server* dimana akan secara rekursif menanyakan ke *server* lainnya untuk mendapatkan IP *Address* dari *host name* yang ditanyakan. Untuk *Non-Recursive query*, *local* DNS *Server* hanya akan mencari pada *cache record* secara lokal tanpa menghubungi *root* DNS *Server*. Jika tidak terdapat di database DNS maka *server* akan mengirimkan pesan "*no such host name*" ke *client.*

Waktu yang digunakan antara pengiriman permintaan *host name* dengan DNS *lookup* hingga didapatkan oleh *Client* ini disebut dengan *Round-Trip-Time* atau RTT DNS.

Untuk mendeteksi RTT (*Round-Trip-Time*) berdasarkan DNS *lookup* secara efisien digunakan *Non-recursive* DNS *Server*. *Client* akan mengirimkan *request* berupa DNS *lookup* untuk sebuah *host name* dan menunggu hingga respon dari *local* DNS *Server* dan menghitung RTT dengan cara mengurangi waktu pengiriman dan waktu penerimaan.

BAB 3
PERANCANGAN DAN IMPLEMENTASI PENDETEKSIAN ROGUE ACCESS POINT

3.1 Skema Pendeteksian *Rogue Access Point*

3.1.1 Deskripsi

Beberapa riset telah melakukan pendekatan terhadap pendeteksian *access point*. Salah satu solusi adalah dengan mengukur beberapa identitas/*fingerprint* dari sebuah *access point* seperti SSID (*password*), *MAC Address*, RSSI (Received Signal Strength Indication) dan *Clock Skew*. *Rogue Access Point* terdeteksi ketika identitas tersebut di bandingkan ke identitas *access point* yang *Legitimate*. Solusi lainnya adalah dengan melakukan analisa trafik jaringan di *Gateway* untuk mendeteksi keberadaan *Rogue Access Point*. Pendekatan dengan solusi ini tidak efektif mendeteksi *Rogue* AP dari sisi *client* dikarenakan ketika *Rogue Access Point* yang "cerdas" sudah berhati-hati dengan pendeteksian tersebut.

Berikut adalah beberapa kendala dalam mendesain sebuah skema pendeteksian :

Client tidak memiliki akses ke informasi tentang *access point* yang sah, terutama AP (*access point*) yang digunakan pada *hotspot*. Oleh karena itu hampir tidak mungkin untuk melakukan perbandingan untuk mengindentifikasikan infoormasi sebuah AP dengan informasi AP yang telah diotorisasi dan disimpan di dalam sebuah database.

Client memiliki hak yang kurang dibandingkan Adminstrator. *Client* dilimitasi oleh setting yang ada pada jaringan. Sebagai contoh, ketika *client* tidak dapat men-set *dedicated server* untuk mendeteksi. Tanpa bantuan dari Administarator sangat susah bagi *client* menangkap trafik jaringan di *Gateway*.

Rogue Access Point (RogueAP atau RAP) yang "pintar" dapat dengan mudah mengetahui skema deteksi dan lolos dari pendeteksian. RogueAP yang pintar tersebut dapat memalsukan identitasnya, menghalangi pesan yang tidak pentin dan langsung menjawab permintaan *client* tanpa meneruskan pesan ke AP yang sah. Oleh karena itu pertahanan yang sederhana pada AP dapat di hindar. Pada

pembahasan berikutnya akan dijelaskan bagaimana strategi dalam menghadapi "perlawanan" dari *Rogue Access Point* yang lebih rumit.

Client dapat menggunakan program seperti traceroute untuk menentukan *access point* (AP) yang terkoneksi merupakan sebuah *Rogue Access Point* (RougeAP). traceroute akan menampilkan jumlah *hop* ke sebuah *host* /*server*. Dari hasil tersebut, *client* dapat mempelajari sebuah *hop* bertambah atau dengan kata lain, RogueAP terdeteksi di dalam rute yang ditampilkan oleh traceroute. Akan tetapi, RogueAP akan sangat mudah menghindari pendeteksian tersebut dengan cara mengawasi *channel wireless* untuk mempelajari SSID dan *MAC Address* dari sebuah AP yang sah dan menentukan parameter yang sama dengan AP yang sah. RogueAP dapat menghindari penerusan *hop* dari balasan AP yang asli ke *client*, dan memberikan impresi bahwa *client* terkoneksi pada *gateway* yang sama dengan AP yang asli.

Traffic Monitoring merupakan teknik untuk membedakan antara trafik melalui *wireless* dan *wired*. Sebagai contoh, mengawasi semua trafik pada *gateway* dan mengolah interval antara dua TCP ACK paket yang berurutan. Semakin panjang interval mengindikasikan paket TCP menjelajah memalui melalui koneksi *wireless*. Akan tetapi teknik ini tidak dapat digunakan dikarenakan hanya dapat membedakan trafik yang berasal dari *wireless* dan *wired* sedangkan untuk kasus *Rogue*AP, keduanya berasal dari media yang sama.

Client dapat menggunakan informasi seperti *Round Trip Time* (RTT) untuk mendeteksi sebuah *Rogue*AP. Dikarenakan *Rogue*AP terdiri atas sebuah tambahan sambungan *wireless* ke AP yang asli, hal ini akan menyebabkan sebuah *delay* ketika akan meneruskan data. *Client* dapat mendeterminasikan RTT ketika akan mengirimkan sebuah pesan seperti permintaan ping atau paket data TCP dan menunggu sebuah *reply*. Akan tetapi, *Rogue*AP dapat dengan mudah membuat sebuah respon untuk mengembalikan ke *user*,hal ini menghindari tambahan waktu dari tambahan sambungan antara *Rogue*AP ke AP yang asli. Sebagai contoh, *Rogue*AP dapat menghasilkan sebuah respon Ping sebagai balasan respon ke *client* tanpa meneruskan ke AP yang asli dan sama halnya untuk paket data TCP.

Pendekatan dalam mendeteksi *rogue*AP terbagi menjadi 3 kategori yaitu : pendekatan *Wireless-side*, pendekatan *Hybrid* (*Wireless-side* + *Wired-side*), dan pendekatan *Wired-side*. Berikut adalah beberapa pendekatan yang digunakan untuk mendeteksi :

a. Pendekatan *Wireless-side*

Kebanyakan pendekatan *wireless side* berbasis industri dalam mendeteksi *Rogue Access Point* (*rogue*AP) memakai cara yang sederhana dan mudah dihindari oleh hacker. Beberapa organisasi melengkapi personil IT dengan *wireless packet* analyzer tools (seperti *sniffer*) di laptop dan *divais* lainnya (seperti AirMagnet dan NetStumbler), memaksa personil IT untuk berjalan di sekitar aula *enterprise* atau kampus untuk mencari keberadaan *rogue*AP. Metode ini sangat tidak efektif karena *scanning* secara manual sangat menyita waktu dan juga sangat mahal. Lalu, dengan *hardware* 802.11 yang beroperasi pada frekuensi yang berbeda (802.11a – 5GHz, 802.11b/g – 2.4Ghz), pesonil IT tersebut mesti melakukan *upgrade divais Detection* untuk mengakomodir frekuensi yang berbeda tersebut. Akan tetapi *Rogue*AP dapat saja di matikan ketika patroli dilakukan dan menyalakan kembali ketika patroli berakhir.

Pendekatan yang lain adalah dengan menginisiasi sebuah pemindai dengan jarak jangkauan yang besar dari sebuah lokasi sentral dengan menggunakan *divais hardware* (contohnya sensor) dan mengirimkan informasi tersebut kembali ke sebuah *platform* management central yang berisikan *wireless network policy* untuk analisis. Metode ini menjadi menguras biaya, dimana dibutuhkan alat yang dipasang diseluruh tempat yang menjadi *hotspot* untuk mengawasi pergerakan data di medium *wireless. Administrator* yang pintar akan mengurangi kekuatan sinyal atau menggunakan sebuah antena *directional* untuk mengurangi jangkauan dari AP.

b. Pendekatan *Hybrid Wireless-side* and *Wired-side*

Pendekatan yang menggabungkan teknik sebelumnya untuk mendeteksi *Rogue*AP dengan cara mendengar di layer 2 (datalink) dan layer 3 (*network*) dan *query*ing switch dan router untuk mendeteksi *divais* apa yang terkoneksi dengan *router* atau *switch*. Kombinasi ini dijadikan patokan dalam memberikan pendekatan deteksi *wired* dan *wireless side*.

c.* Pendekatan *Wired-side

Pada pendekatan *Wired side*, sangat berguna dalam mendeteksi secara terpusat dimana tidak memerlukan sensor terdistribusi. Didalam pendekatan *wired-side* sangat tergantung terhadap pada konektivitas *Rogue*AP terhadap jaringan.

Pendeteksian dapat berupa analisis terhadap trafik pada jaringan dimana dilihat dari asal dan tujuan paket tersebut. Parameter juga akan di jadikan patokan dalam menentukan legitimasi dari AP tersebut. Parameter tersebut dapat berupa physical *address*, port number, IP *address*, OS *fingerprint*, dan parameter-parameter identitas sebuah divais.

Parameter lainnya akan digunakan adalah analisa RTT. Pada skenario *Rogue*AP yang terhubung melalui jaringan *wireless* ke AP yang *legitimate* dan menyebarkannya melalui NIC yang lain, maka kan terjadi penambahan *delay* dikarena bertambahnya *hop*. RTT dapat berupa Probe, Ping, dan DNS Lookup.

3.2 Perencanaan topologi skenario

Dalam buku ini akan dilakukan pengujian dengan beberapa skenario berdasarkan tipe-tipe kemungkinan serangan dari *Rogue*AP. Berikut adalah beberapa skenario dengan topologinya :

Untuk perencanaan pengukuran menggunakan metode RTT dan *functional test* metode WIDS terdapat 3 skenario yaitu :

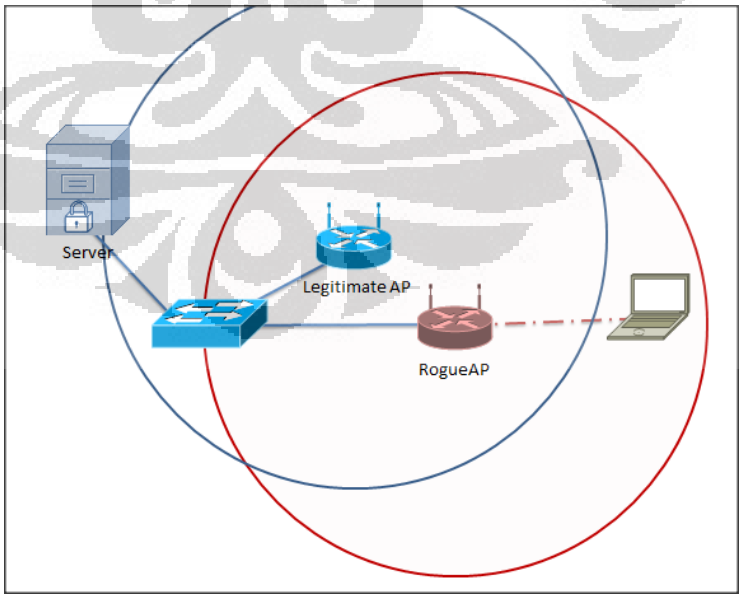
a. Skenario 1 : *Rogue*AP terkoneksi langsung ke jaringan *legitimate* melalui media kabel.

Deskripsi Topologi:

Pada gambar 3.1, dijelaskan terdapat 2 AP dimana memiliki fungsi masing-masing yaitu sebagai *Legitimate* AP dan *Rogue*AP.

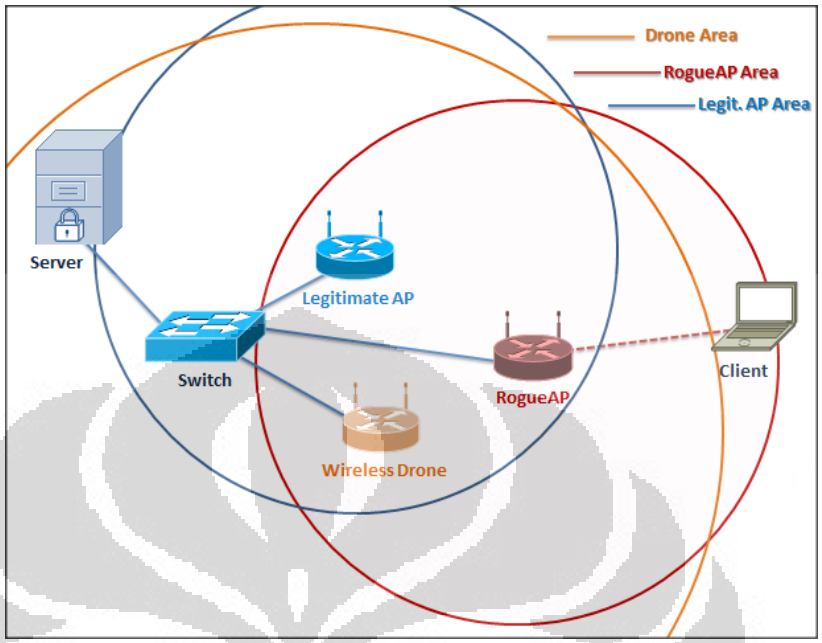
Kedua AP memiliki kemampuan yang sama dengan frekuensi yang sama dan berjalan di bawah IEEE 802.11G. Dalam skenario ini, *user* akan terkoneksi secara otomatis ke AP yang ilegal. Hal ini mungkin terjadi dikarenakan SSID dan paramater yang ada pada AP legal telah di kloning. Dengan sinyal yang kuat menyebabkan AP legal kalah dengan *Rogue*AP.

*Rogue*AP terkoneksi langsung ke switch sehingga mendapatkan jalur internet untuk disebarkan melalui hotspot ilegal.

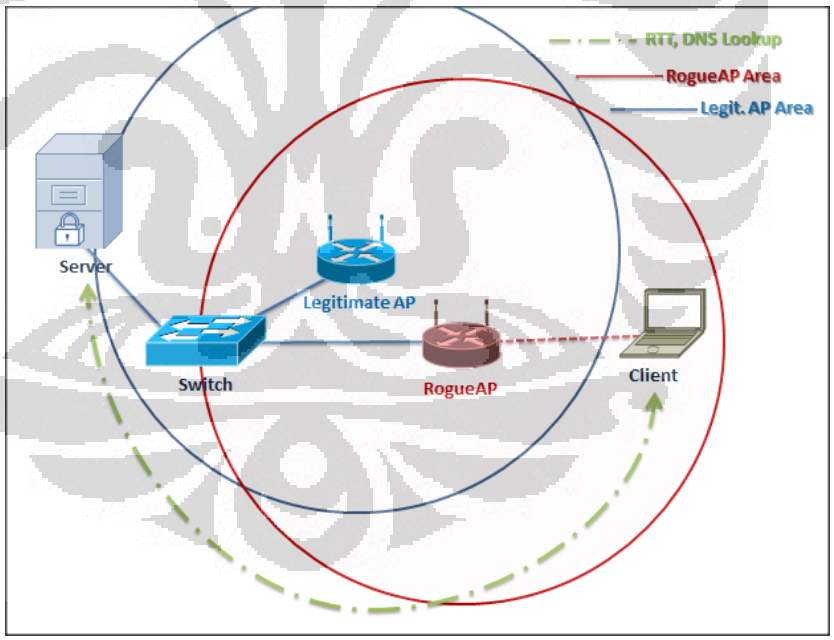


Gambar 3. 1 Skenario 1 *Rogue*AP terkoneksi langsung via kabel

Topologi ini akan di tes dengan menggunakan metode pendekatan *wireless*-side (sensor drone) dan *wired-side* (analisa trafik).



Gambar 3. 2 Skenario 1 Topologi 1 - pendekatan *wireless*-side (sensor drone)



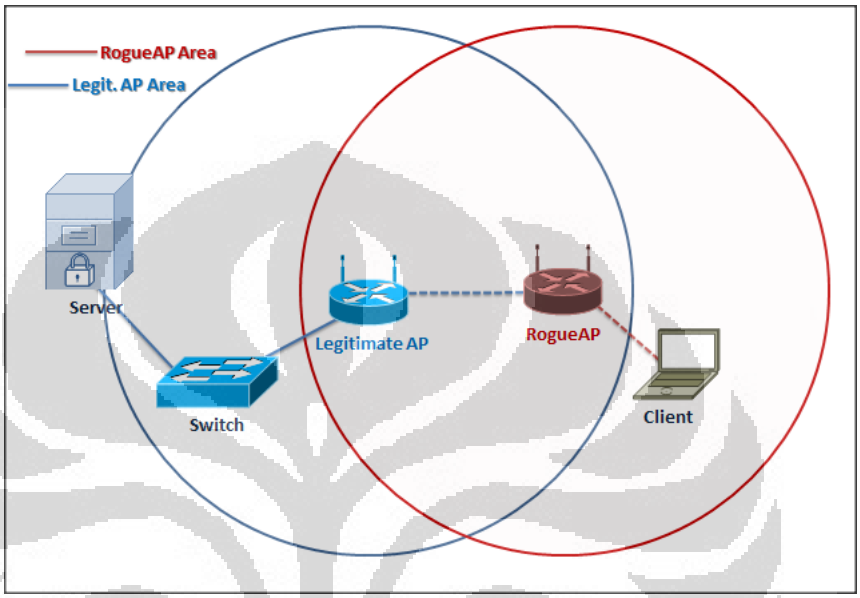
Gambar 3. 3 Skenario 1 Topologi 2 - pendekatan *wired-side* (analisa trafik RTT)

b. Skenario 2 *Rogue*AP terkoneksi tidak langsung ke jaringan *legitimate* melalui media nirkabel.

Deskripsi topologi :

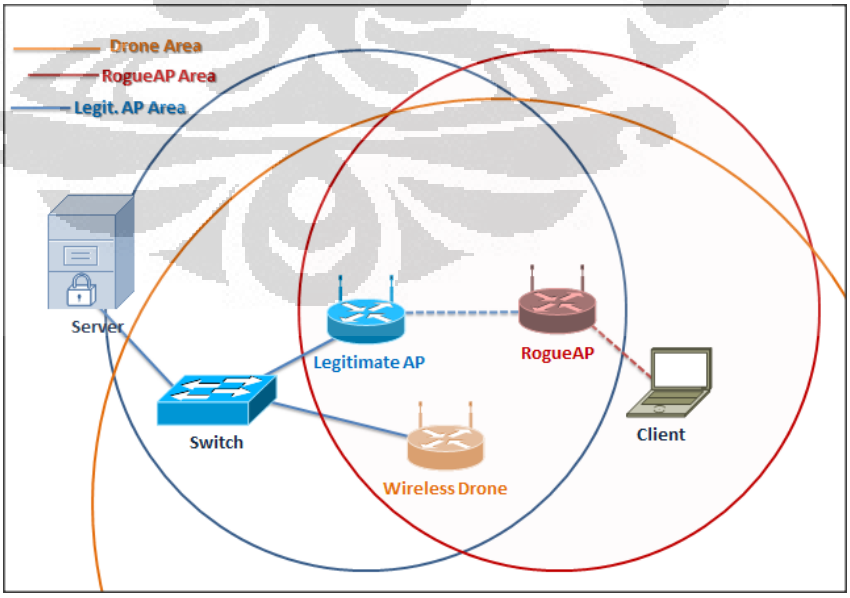
Pada gambar 3.4 terdapat 2 AP dimana merupakan *Legitimate* AP dan *Rogue*AP.

Rogue AP mendapatkan koneksi melalui AP yang ada melalui jalur *wireless*. Dengan begitu jalur internet menjadi mudah untuk didapatkan. Tipe *Rogue*AP yang dapat terjadi disini adalah tipe *Software* dimana menggunakan laptop dengan dua buah *Wireless* Card/USB.

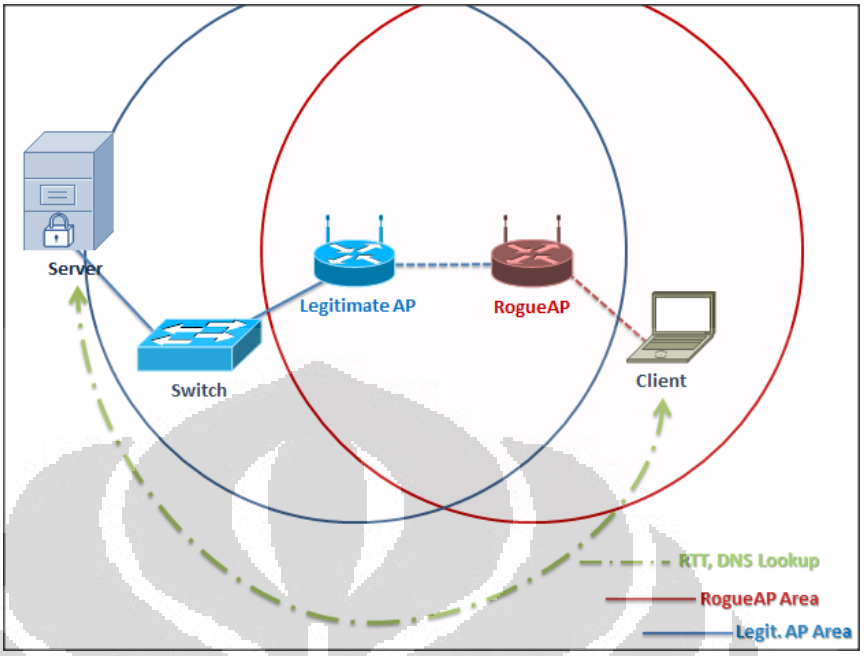


Gambar 3. 4 Skenario 2 dengan *Rogue*AP terkoneksi tidak langsung via nirkabel

Topologi 2 ini akan di tes menggunakan pendekatan *wireless*-side dan *wired-side*. Berikut adalah gambaran topologinya :



Gambar 3. 5 Skenario 2 Topologi 1 dengan pendekatan *wireless*-side (sensor drone)



Gambar 3. 6 Skenario 2 Topologi 2 dengan pendekatan *wired-side*(analisa trafik RTT)

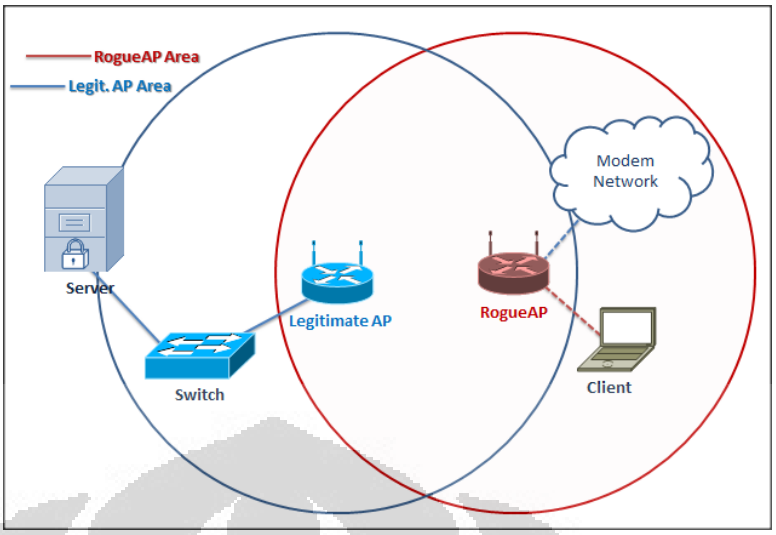
c. Skenario 3 : *Rogue*AP tidak terkoneksi ke jaringan *legitimate* (menggunakan jaringan lain).

Deskripsi topologi :

Pada gambar 3.7 terdapat 2 AP dimana merupakan *Legitimate* AP dan *Rogue*AP.

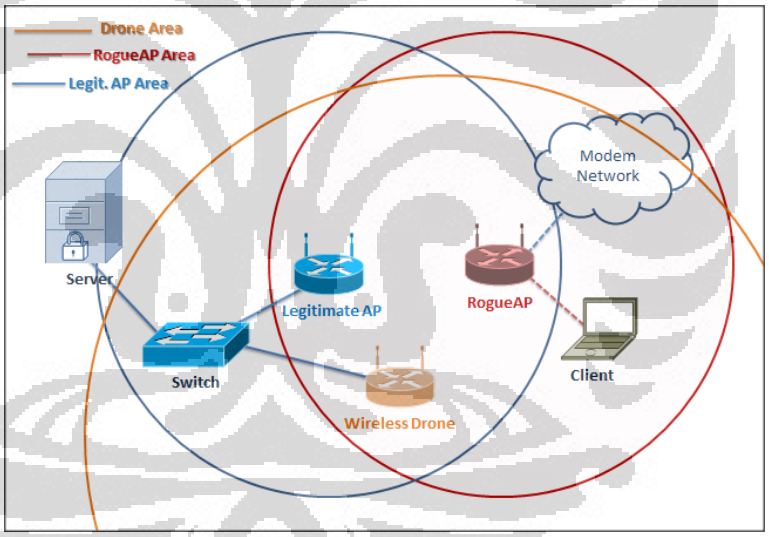
Rogue AP mendapatkan koneksi melalui AP yang ada melalui jalur modem. Dengan begitu jalur internet menjadi mudah untuk didapatkan. Tipe *Rogue*AP yang dapat terjadi disini adalah tipe *Software* dimana menggunakan laptop dengan dua buah *Wireless* Card/USB.

Batasan untuk topologi ketiga ini adalah *Rogue*AP masih terdapat pada area sensor drone untuk skenario pendekatan *wireless*-side.

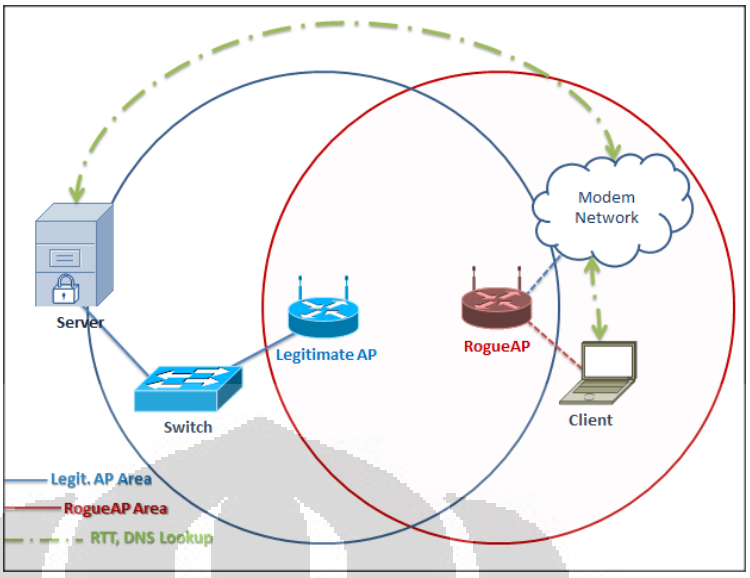


Gambar 3. 7 Skenario 3 dengan *Rogue*AP terkoneksi melalui jaringan modem

Topologi 3 ini akan di tes menggunakan pendekatan *wireless*-side dan *wired-side*. Berikut adalah gambaran topologinya :



Gambar 3. 8 Skenario 3 Topologi 1 dengan pendekatan *wireless*-side (sensor drone)

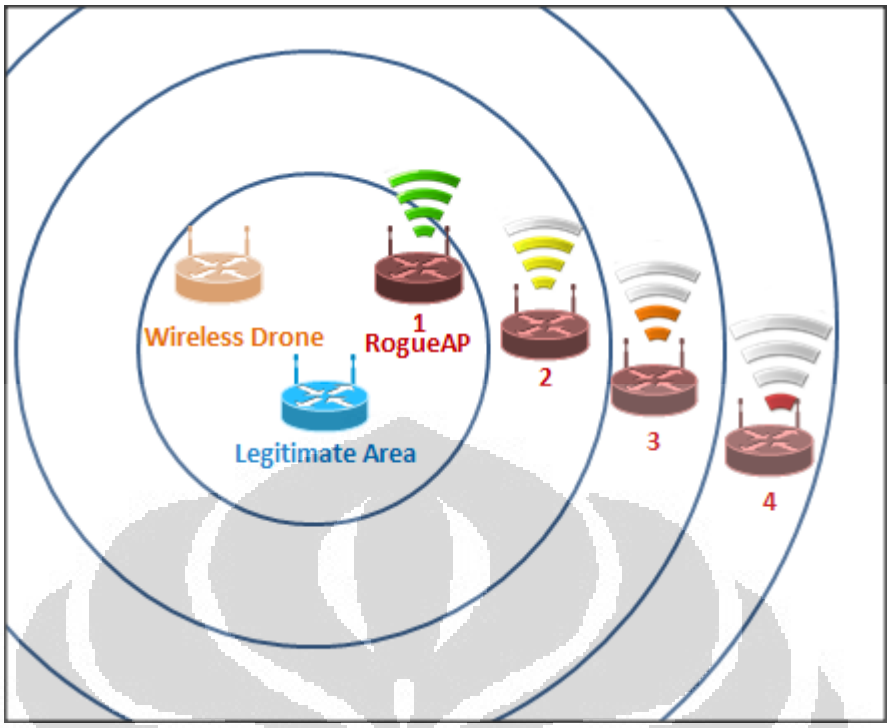


Gambar 3. 9 Skenario 3 Topologi 2 dengan pendekatan *wired-side*(analisa trafik)

Untuk perencanaan pengukuran *response time* WIDS terdapat beberapa skenario :

a. Kekuatan sinyal yang berbeda dari Legit. AP

Pada skenario ini kekuatan sinyal Legit. AP menjadi parameter pertanda jarak dari *Rogue* AP ke Sensor WIDS. Sensor WIDS akan diletakkan di dekat Legit. AP.



Gambar 3. 10 Skenario 1 WIDS response time testing

Pada perancangan diatas, RogueAP akan diposisikan berada pada area dengan kekuatan sinyal berbeda. Pembagian kekuatan sinyal dikategorikan menjadi 5 area yaitu area 1 (75%-100% signal strength), area 2 (50%-74% signal strength), area 3(25%-49% signal strength) dan area 4 (1%-24% signal strength). WIDS berjalan pada standar IEEE 802.11n sehingga cakupannya luas.

3.3 Pendeteksian *Rogue Access Point* dengan pendekatan *Wireless-side*

Dalam pendeteksian dengan menggunakan pendekatan *wireless-side* membutuhkan sebuah sensor *wireless* yang akan memonitor kondisi jaringan *wireless* dan akan mengirimkan informasi tentang trafik yang ada pada jaringan ke sebuah *server* sentral dan mengolah data-data tersebut.

Berikut adalah skema pendeteksian dengan menggunakan kismet drone sebagai *WIDS Sensor* dan kismet *server* sebagai pusat analisis.

3.3.1 Kebutuhan *Hardware/Software*

Berikut adalah perlengkapan *Hardware* maupun *Software* dalam perancangan :

a. 1 unit TP-Link MR3420

Spesifikasi *Hardware* :

- *Wireless Router*
- EncryptionUp to 128-bit Encryption
- 2x 3dBi Detachable Omni Directional *Antenna*
- 4-port full-duplex 10/100 Switch
- Mendukung IEEE 802.11a, 802.11b, 802.11b, 802.11n
- RAM : 32 MB, FLASH: 4 MB with USB Storage Support >32GB, USB 2.0
- *Wireless* : Atheros AR9287 (2x2 MIMO 300Mbps)
- RAM Chip : Zentel A3S56D40FTP -G5
- *Antenna* : 2 Removables x 3 dBi
- Ethernet : AG71xx 4 LAN, 1 WAN 100/10
- USB : 1 x 2.0

Berfungsi sebagai Kismet Drone (*WIDS Sensor*)

b. 1 unit Linksys WRT45G v7.0

Spesifikasi *Hardware* :

- Versi 7.0
- CPU : Atheros AR2317 @ 240 MHz
- RAM : 8 MB
- FLASH MEMORY: 2 MB

Berfungsi sebagai *Legitimate* AP

c. 1 unit Switch

d. 1 unit *Personal Computer*

- Processor : Intel Core2duo
- Memory: 2048 MB RAM
- *Operating System*: Backtrack 5 R2 “Revolution”
- Ethernet NIC : Broadcom

Berfungsi sebagai Kismet *Server* dan HTTP *Server*

e. 1 unit Laptop Toshiba NB255

Spesifikasi Detail :

- Processor : Intel(R) Atom(TM) CPU N455
- Memory: 1024 MB RAM
- *Operating System*: Backtrack 5
- *Wireless* Network *Adapter* : Atheros AR9285
- Total Space: 160 GB

  Berfungsi sebagai *Rogue*AP

f. 1 unit Laptop Acer S3

- Processor : Intel Core i5 2467M
- Memory: 4096 MB RAM
- *Operating System*: Windows 7 Ultimate 64-*bit*
- *Wireless* Network *Adapter* : Atheros AR5BWB225 *Wireless* Network *Adapter*
- Total Space: 512 GB

  Berfungsi sebagai *Client* atau User

g. 2 buah USB *Wireless* TP-Link WN722N

- Encryption : 64/128 bits WEP
- Operating Frequency : 2.4-2.4835 GHz
- Modulasi : OFDM/CCK/16-QAM/64-QAM
- Output Power : 20 dBm
- Standar Protocol : IEEE 802.11n, IEEE 802.11g, 802.11b

h. Kismet

- IEEE 802.11 layer 2 *wireless network* detector, sniffer, dan *Intrusion Detection System* (IDS)
- Mendukung hampir semua *wireless* card yang memiliki fungsi *raw monitoring* (rfmon)
- Mendeteksi *hidden network*
- Mengetahui keberadaan jaringan *non-beaconing* dengan menganalisa trafik data.

i. Modem SpeedUp™

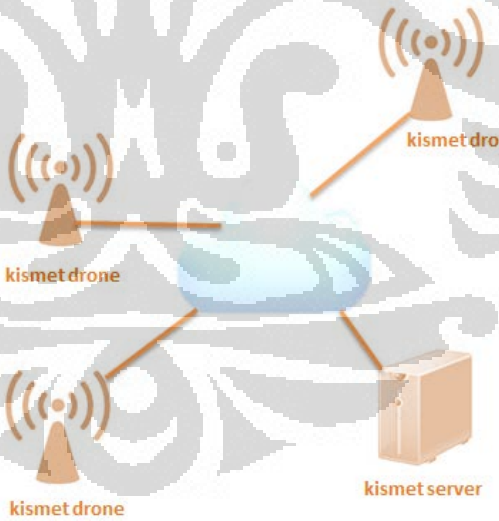
- Downlink HSPA+ 14.4 Mbps)
- Uplink HSUPA 5 Mbps
- Support OS : Windows 2000, XP, Vista, 7, Mac OS X
- Plug & Play
- Micro SD Slot

j. Backtrack 5 R2

- *Security-purpose* O.S (*Operating System*)
- Dilengkapi *software* yang mendukung aktifitas *hacking*
- Memiliki Kismet *Server* dan *Client* dalam satu O.S

3.3.2 Implementasi Sistem

Implementasi dari sistem ini terdiri beberapa tahap yaitu implementasi linux-based *firmware* dengan varian OpenWrt ke dalam AP, implementasi kismet drone ke dalam OpenWrt, konfigurasi Kismet *Server*/*Client* serta Kismet Drone dan pengamatan dan analisa data penelitian.



Gambar 3. 11 gambaran sederhana sistem

Pada gambar 3.2 dijelaskan kismet drone akan terkoneksi dan mengirimkan hasil tangkapannya ke kismet *server*.

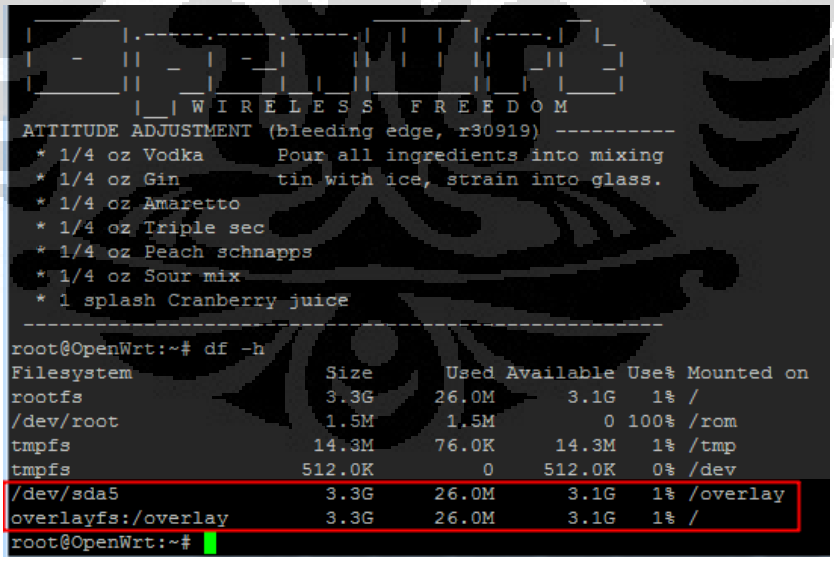
3.3.3 Instalasi dan Konfigurasi OpenWrt

Pada buku ini, *firmware* yang digunakan untuk *Access Point* menggunakan *firmware* OpenWrt dibandingkan menggunakan *firmware* aslinya. OpenWrt

merupakan distro linux untuk divais embedded dan dapat didownload di website resminya yaitu http://download.openwrt.org/ . Berdasarkan *platform* yang digunakan *firmware* OpenWrt terbagi atas 3 yaitu Snapshot-release (*platform* ar71xx), OpenWrt 10.03 "Backfire" (*platform* brcm47xx) dan OpenWrt 8.09.2 'kamikaze' (*platform* ar7).

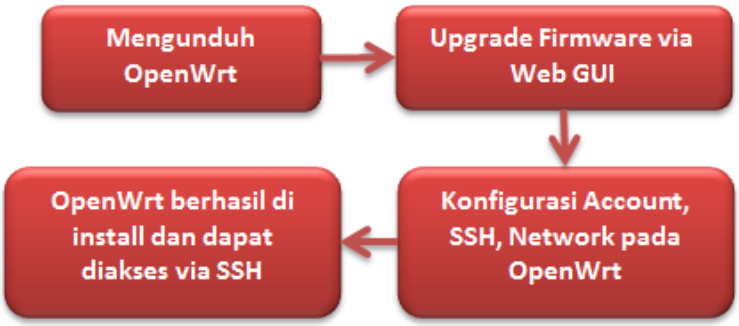
Di skripsi ini menggunakan divais AP dengan merk TP-Link MR3420 berbasiskan *wireless* driver atheros dengan *platform* ar71xx. Yang perlu diperhatikan dalam instalasi OpenWrt adalah *platform* apa yang digunakan oleh divais AP dan berapa besar kapasitas flash di AP tersebut.

Untuk instalasi dengan *plugin* dan modifikasi kernel yang banyak direkomendasikan menggunakan *firmware* yang tidak memiliki web *interface* atau biasanya dikenal dengan 'trunk'. Firmware tanpa web *interface* GUI tidak menghabiskan banyak space yang ada di AP. Pada skripsi kali ini menggunakan penambahan *storage* luar yaitu menggunakan media flashdisk sebesar 4 GB. Dengan *storage* yang besar dapat menambah kemampuan dari AP tersebut karena dapat menjadi SWAP file dan ekstra space untuk penyimpanan. Metode yang digunakan adalah metode ExtRoot dimana memindahkan sistem file ke flashdisk.



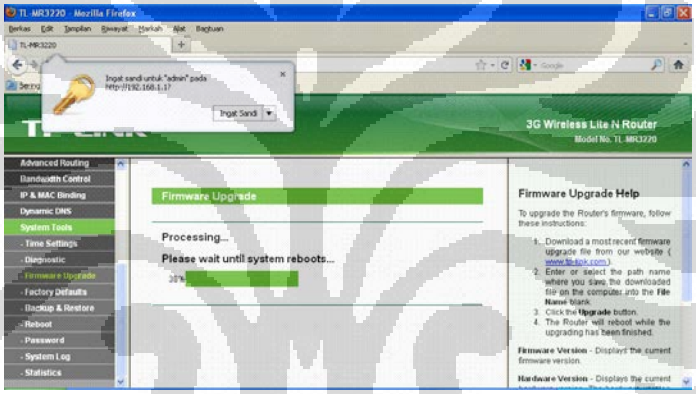
Gambar 3. 12 Storage bertambah setelah melakukan Extroot

Dibawah ini akan dijelaskan alur instalasi OpenWrt :



Gambar 3. 13 Alur instalasi OpenWrt

Pada gambar 3.3 dijelaskan langkah pertama adalah mengunduh OpenWrt di situs resminya. Pada kali ini penggunaannya menggunakan versi *platform* snapshot. Lalu *upgrade firmware* tersebut melalui *interface* web GUI. OpenWrt mendukung instalasi dan modifikasi kernel yang lebih.



Gambar 3. 14 Tampilan ketika Upgrade Firmware

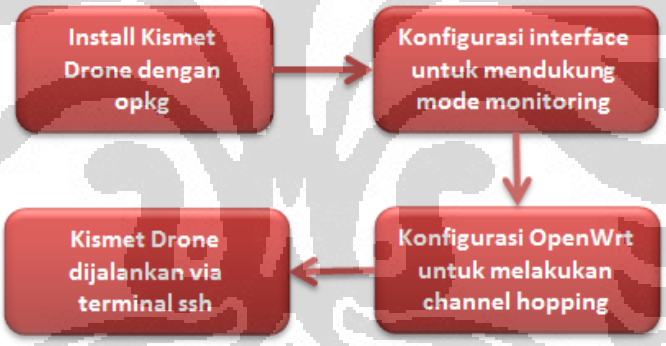
Gambar 3. 15 Tampilan OpenWrt setelah Upgrade Firmware dan Install Web GUI

Gambar 3. 16 alur instalasi ExtRoot

Pada gambar 3.6 dijelaskan alur konfigurasi dukungan flashdisk pada OpenWrt untuk mendapatkan kemampuan *storage* lebih. Dengan dukungan *storage* yang lebih memungkinkan untuk melakukan penambahan aplikasi.

3.3.4 Implementasi Kismet Drone

Kismet drone berfungsi sebagai penangkap semua paket yang ada di jaringan *wireless* dan akan mengirimkannya ke kismet *server* untuk nanti diolah.



Gambar 3. 17 Alur konfigurasi kismet drone

Perlu diperhatikan untuk channel hopping hanyalah sebuah persyaratan yang tidak mutlak sehingga sistem masih dapat berjalan meskipun tidak terdapat channel hopping. Kismet drone dijalankan langsung via terminal ssh.

3.3.5 Implementasi Kismet *Server*/*Client*

Pada bahasan ini penulis menggunakan Backtrack 5 Revolution sebagai basis Kismet *Server* yang sudah terdapat didalam OS sehingga tidak diperlukan lagi instalasi lebih lanjut.

Dalam konfigurasi kismet *server* untuk dapat mengolah data yang didapatkan oleh kismet drone maka diperlukan konfigurasi jaringan yang benar. Berikut adalah alur gambaran konfigurasi Kismet *Server*.

3.4 Pendeteksian *Rogue Access Point* dengan pendekatan Wired-side

Dalam pendeteksian dengan menggunakan pendekatan *wired-side* dibutuhkan beberapa *software* yang dapat menangkap dan menganalisa trafik jaringan. Berikut adalah skema pendeteksian dengan menggunakan pendekatan *wired-side* berdasarkan karakteristik trafik dan analisa link.

3.4.1 Kebutuhan *Hardware* dan *Software*

Berikut adalah perlengkapan *Hardware* maupun *Software* dalam perancangan :

a. 1 unit Linksys WRT45G v7.0

Spesifikasi *Hardware* :

- Versi 7.0
- CPU : Atheros AR2317 @ 240 MHz
- RAM : 8 MB
- FLASH MEMORY: 2 MB

Berfungsi sebagai *Legitimate* AP

b. 1 unit Switch

c. 1 unit Personal Computer

- Processor : Intel Core2duo
- Memory: 2048 MB RAM
- *Operating System*: Backtrack 5 R2 “Revolution”
- Ethernet NIC : Broadcom

Berfungsi sebagai DNS *Server*

d. 1 unit Laptop Acer S3

- Processor : Intel Core i5 2467M
- Memory: 4096 MB RAM
- *Operating System*: Backtrack 5 R2 Revolution
- *Wireless* Network *Adapter* : Atheros AR5BWB225
- Total Space: 512 GB

Berfungsi sebagai *Rogue Access Point*

e. 1 unit Toshiba NB255

Spesifikasi Detail :

- Processor : Intel(R) Atom(TM) CPU N455

- Memory: 1024 MB RAM
- *Operating System*: Windows 7 Ultimate
- *Wireless* Network *Adapter* : Atheros AR9285
- Total Space: 160 GB

  Berfungsi sebagai *Rogue*AP

3.4.2 Implementasi metode analisa RTT

Pada analisa trafik menggunakan metode pendekatan *wired-side* dimana tidak menggunakan *wireless* divais spesifik untuk melakukan pendeteksian. Terdapat banyak parameter yang dapat dijadikan patokan dalam melakukan pengujian menggunakan pendekatan analisa RTT. Parameter trafik paling sederhana adalah *Round Trip Time* (RTT). RTT dapat terbagi atas 2 jenis yaitu RTT Ping dan RTT DNS. Kedua hal ini akan dimasukkan dalam pendeteksian. Selain hal tersebut masih banyak yang dapat menjadi patokan parameter yaitu salah satunya adalah port web *service* yang terbuka (port 80).

RTT Ping dilakukan dengan cara mengirimkan ICMP Echo_Request dan menunggu ICMP Echo_Reply. Ping merupakan singkatan dari Packet InterNet Grouper.

```
% ping sgi.com
PING sgi.com (192.48.153.65) from 192.12.65.23 : 56(84) bytes of data.
64 bytes from SGI.COM (192.48.153.65): icmp_seq=0 ttl=241 time=66.092 msec
64 bytes from SGI.COM (192.48.153.65): icmp_seq=1 ttl=241 time=54.753 msec
64 bytes from SGI.COM (192.48.153.65): icmp_seq=2 ttl=241 time=55.036 msec

--- sgi.com ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/mdev = 54.753/58.627/66.092/5.279 ms
```

Gambar 4. 1 Contoh hasil ping

Dalam melakukan analisa trafik menggunakan RTT ping maka terdapat 3 permasalahan yaitu hal pertama adalah letak *server* yang akan dihubungi. Sebuah *server* yang akan dikontak lebih baik di *local network* karena jika di Internet akan menyebabkan *delay* yang dihasilkan mejadi sangat besar dimana RTT ping sangat sensitif terhadap *delay*. Permasalahan kedua adalah besarnya load yang terjadi disaat itu ketika *user* bertambah banyak, hal ini relatif dan bisa memberikan dampak secara drastis untuk persentase kesalahan pendeteksian.

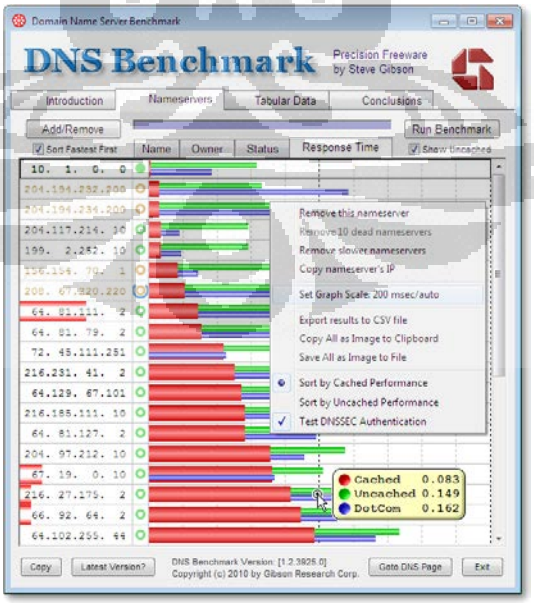
3.4.3 Pengukuran RTT DNS dengan DNSBench

DNS adalah kepanjangan dari *Domain* Name *Server*. DNS berfungsi sebagai penerjemah nama dari sebuah domain menjadi deretan sebuah angka yang disebut *Internet Protocol* (IP). Contohnya adalah bila kita akan membuka atau request URL *Domain* tertentu, biasanya kita menggunakan deretan nama atau huruf karena lebih mudah dihafal seperti google.com, yahoo.com, facebook.com, dsb. Nah, disinilah DNS ini bekerja.

DNS ini melakukan *encode* atau menerjemahkan dari domain google.com ke dalam bentuk deretan angka unik yaitu berupa IP misal google.com Ip nya adalah 208.67.219.231. Jadi bila kita masukan 208.67.219.231 pada browser maka juga akan membuka domain google.com tersebut. Deretan angka IP seperti 174.36.138.32. IP inilah yang digunakan mesin internet untuk saling berkomunikasi seperti *Server Domain*, *Server* Hosting, *Server* Proxy dan sebagainya.

Pengukuran DNS dapat menggunakan berbagai macam *software* bawaan sistem operasi seperti nslookup dan dig. Akan tetapi kedua program tersebut tidak memberikan waktu yang presisi.

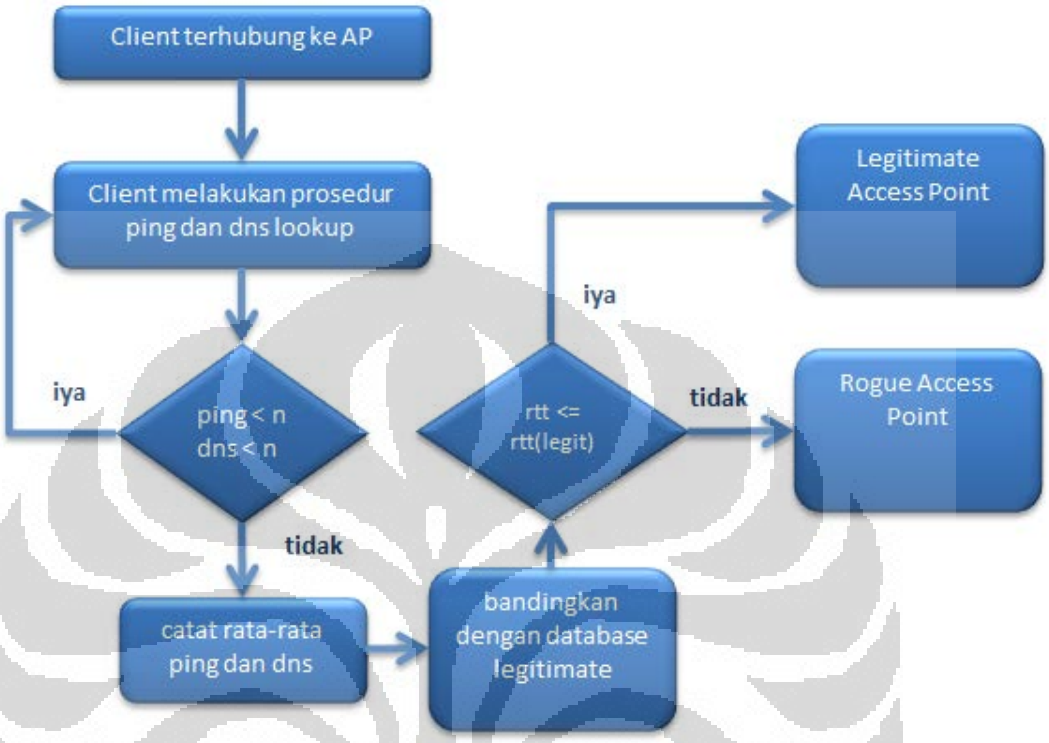
DNSBench merupakan salah satu alternatif pilihan untuk mendapatkan hasil presisi dan akurat dari DNS *query*.



Gambar 3. 18 Tampilan *default* DNSBench

Dalam pengukuran di 3 skenario menggunakan IP DNS *Server* lokal yaitu 152.118.24.10 atau DNS UI.

3.4.4 Diagram alir metode



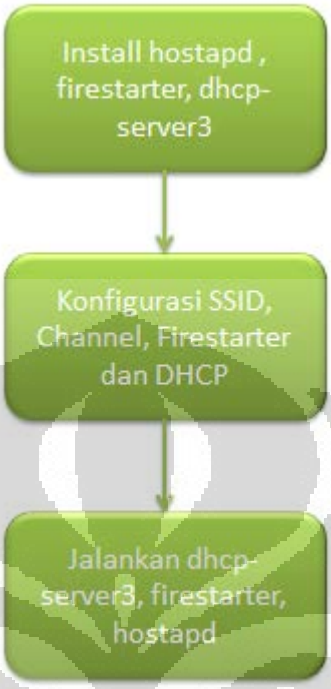
Gambar 3. 19 Diagram alir analisa trafik RTT

3.5 Implementasi *Rogue Access Point*

Dalam membuat sebuah *Rogue Access Point* sangatlah mudah. Di windows bahkan *software* seperti Connectify atau bahkan di Windows 7 sendiri sudah dapat membuat sebuah SSID sendiri dengan enkripsi atau *open*. Di Linux sudah pasti sangat banyak dan lebih dinamis.

Rogue Access Point yang dibuat di dalam skenario ini berjalan di sistem operasi Linux. *Software* yang digunakan adalah Hostapd. Dalam memberikan konektivitas ke *client* akan digunakan iptables. *Software* GUI untuk iptables ini bernama firestarter. DHCP *server* diperlukan untuk memberikan IP dinamis dimana untuk hal ini di perhitungkan si *attacker* tidak mendapatkan DHCP *relay* dan harus menyediakan DHCP sendiri.

Berikut adalah alur dalam melakukan instalasi dan konfigurasi :

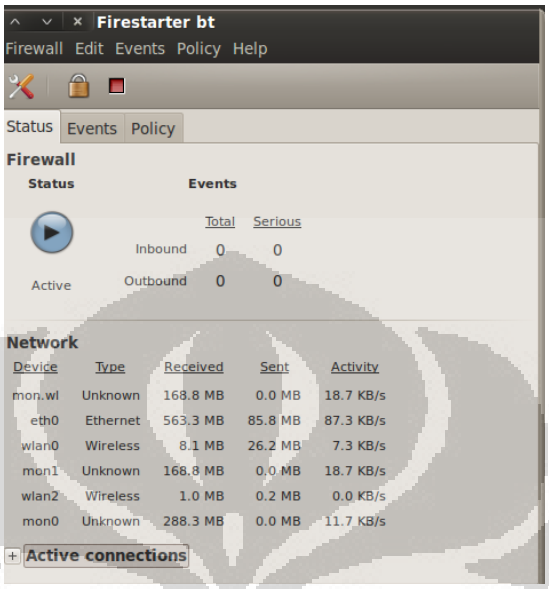


Gambar 3. 20 Diagram alir instalasi dan konfigurasi *Rogue*AP

Dengan hal-hal tersebut maka dapat terjadi *Rogue* AP bersifat *software*. Proses alur paket pada koneksi *Rogue* AP :

Gambar 3. 21 Diagram alir proses *Rogue*AP

Dalam hal ini sudah jelas setiap paket akan melalui komputer si *attacker*. Tidak ada keamanan lagi apalagi privasi terhadap *client*. Kondisi ini dinamakan zero security.



Gambar 3. 22 Firestarter GUI

BAB 4
PENGUJIAN DAN ANALISIS

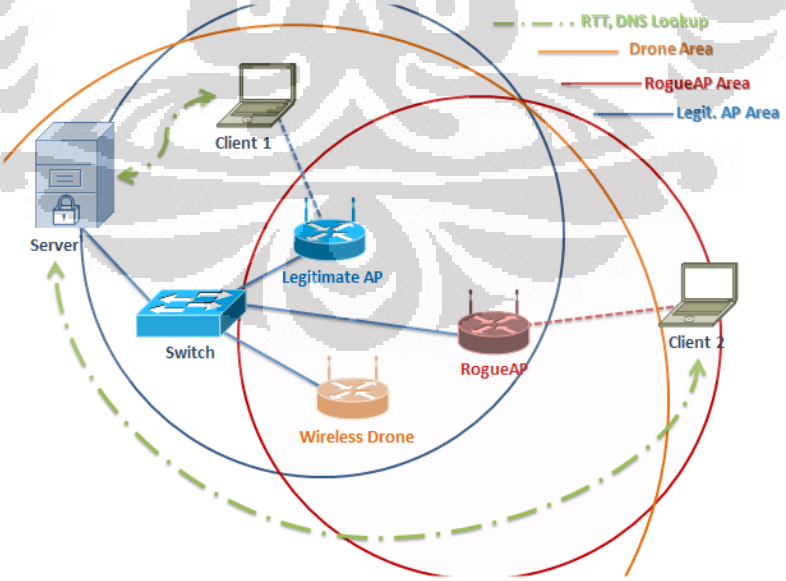
4.1 Pengujian Sistem

Seperti yang telah dijelaskan pada bab 3 bahwa terdapat beberapa skenario dan topologi yang akan dijalankan. Dari skenario tersebut akan dianalisis performansi, keakuratan serta keefektifan metode-metode yang diberikan terhadap topologi yang dirancang. Untuk membantu dalam melakukan pengujian terhadap metode analisa trafik RTT digunakan beberapa *software* seperti DNS *benchmark* dan Ping.

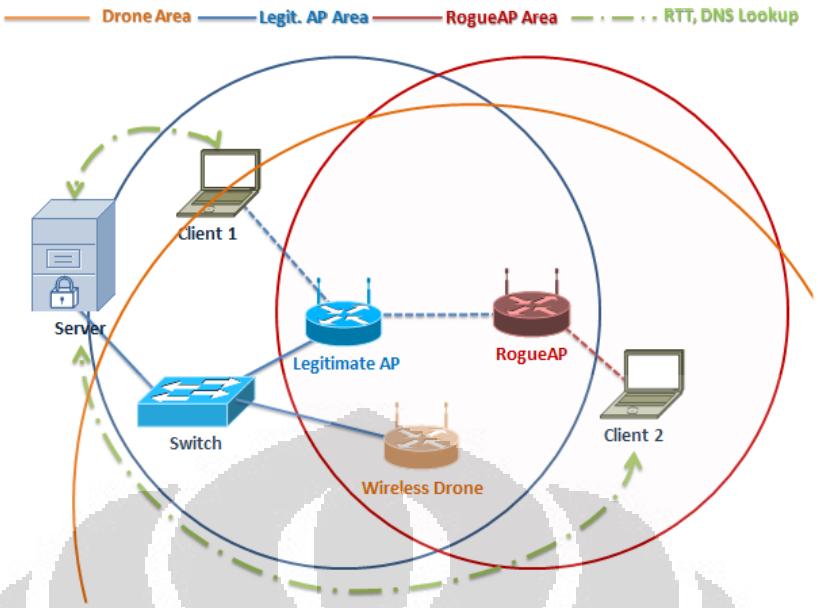
Dalam pengujian jaringan dilakukan beberapa skenario dan tiap skenario akan dianalisa parameter *delay* maksimum, minimum dan rata-rata untuk metode analisa trafik RTT dan kecepatan deteksi serta keakuratan menggunakan sensor WIDS.

Pengujian dilakukan dengan kondisi yang sama untuk skenario yang sama. Ketiga topologi akan di uji dengan menggunakan *software* dan parameter yang sama, namun menggunakan metode yang berbeda yaitu pendekatan *wired-side* dan *wireless*-side.

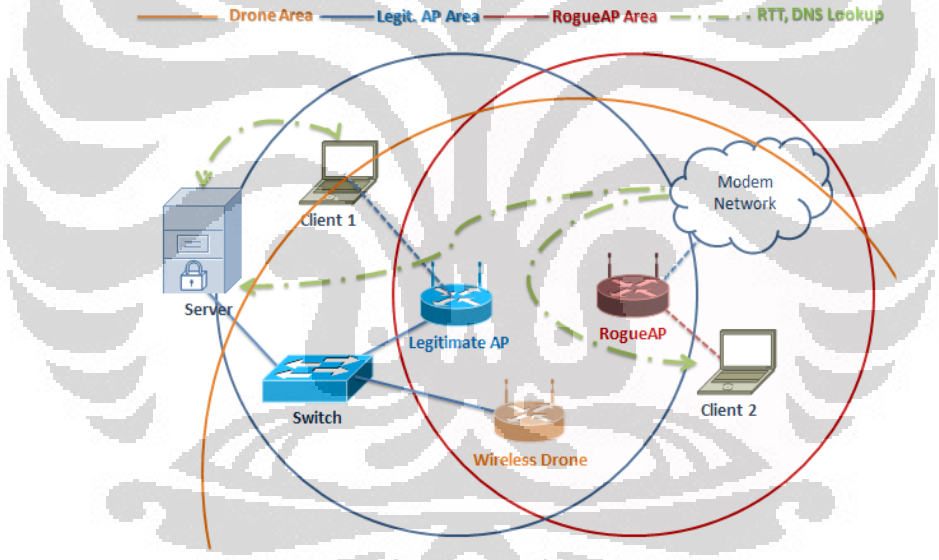
Gambar tiap topologi dapat dilihat pada gambar 4.1, 4.2, dan 4.3



Gambar 4. 2 Topologi Skenario 1 *Rogue*AP terkoneksi melalui perantara kabel



Gambar 4. 3 Topologi Skenario 2 *Rogue*AP terkoneksi melalui Legit. AP



Gambar 4. 4 Topologi Skenario 3 *Rogue*AP terkoneksi melalui jaringan modem

4.2 Pengukuran dan Analisa

Pada bagian ini akan dilakukan analisa RTT ping dan DNS serta respon time Kismet. Analisa dilakukan dengan membandingkan *delay* waktu RTT pada protokol ping dan DNS pada jaringan *legitimate* dan *rogue*. Analisa berikutnya adalah menggunakan metode sensor WIDS, dimana mengukur kecepatan deteksi dengan parameter jarak *Rogue* AP ke *Legitimate*.

Pada pengukuran digunakan 2 metode dengan parameter-parameter yang berbeda yaitu :

1. **Metode analisa trafik RTT**

   Metode dilakukan terhadap data-data parameter RTT berupa *delay* antara paket terkirim dan dikirim balik yang diperoleh dari tiap topologi yang ada.

   Analisa RTT dilakukan berdasarkan data yang diambil terdapat pada bagian lampiran.

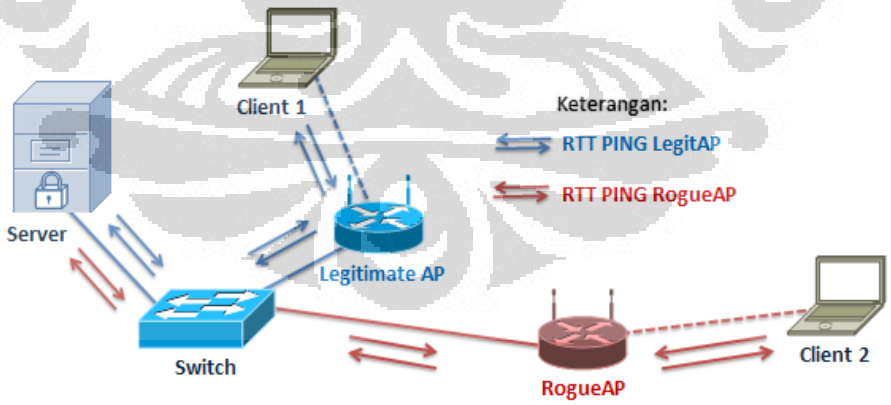
2. **Metode sensor WIDS**

   Metode ini dilakukan dengan menggunakan *wireless* sensor drone dimana melakukan monitoring terhadap lingkungan sekitar *Legitimate* AP dan mengirimkan hasil monitoring ke kismet *server* untuk diolah. Parameter pengujian berupa ketepatan waktu kismet drone ke kismet *server* hingga menghasilkan sebuah alert.

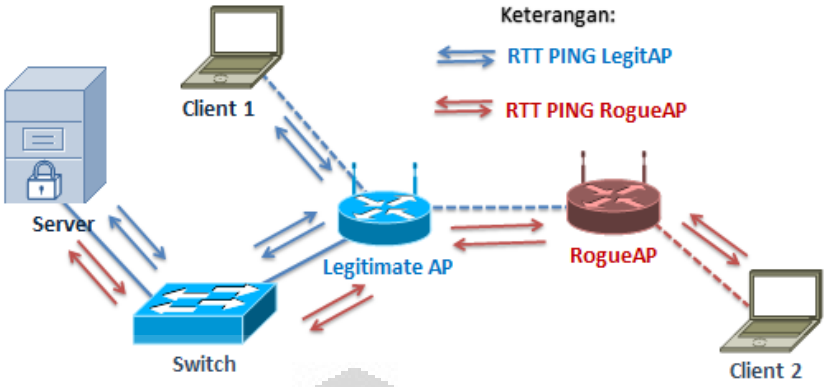
4.2.1 Pengukuran RTT Ping

RTT (*Round Trip Time*) merupakan sebuah cara mendeteksi lamanya waktu yang dibutuhkan sebuah paket dapat terkirim dan diterima kembali dengan keadaan baik. RTT Ping sendiri menggunakan protokol ping dimana dikirimkan dari *client* ke *server*.

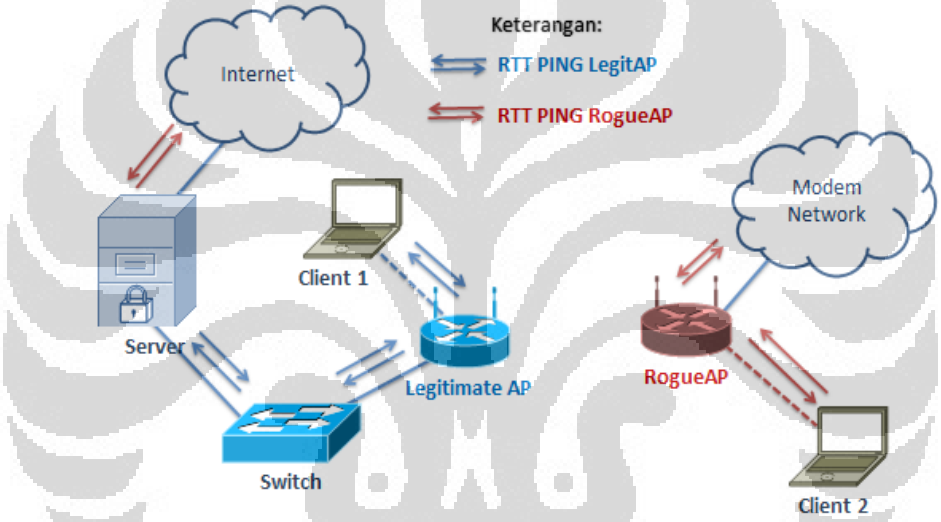
Metode ini sendiri dilakukan terhadap 3 skenario yang telah disebutkan pada bab 3.



Gambar 4. 5 Topologi Pengukuran RTT Ping Legit. dan *Rogue* AP untuk Skenario 1



Gambar 4. 6 Topologi Pengukuran RTT Ping Legit. dan *Rogue* AP untuk Skenario 2



Gambar 4. 7 Topologi Pengukuran RTT Ping Legit. dan *Rogue* AP untuk Skenario 3

Pada skenario 1 *client* 2 merupakan komputer *client* dimana terkoneksi ke *Rogue*AP dan melakukan pengiriman data ke *server* dimana setelah sampai dari *server* akan mengirimkan kembali ke *client* 2 dan didapatkan RTT untuk jaringan melalui jalur *rogue*. Untuk berikutnya adalah *client* 1 dimana sama halnya dengan *client* 2 melakukan ping ke *server* dan mendapatkan RTT. Proses ini dilakukan selama 100 kali per iterasinya. Besar paket ICMP yang dikirimkan adalah sebesar 64 bytes menuju IP s*erver* yaitu 152.118.101.11. Ping memiliki *time-to-live* sebesar 128.

Tabel 4.1 Hasil pengukuran RTT Ping pada Skenario 1, 2 dan 3

Iterasi ke-	TOPOLOGI 1		TOPOLOGI 2		TOPOLOGI 3	
	Cl1-AP-Server	Cl2-RAP-Server	Cl1-AP-Server	Cl2-RAP-Server	Cl1-AP-Server	Cl2-RAP-Server
	Rata-rata waktu RTT PING (ms)					
1	5.31	5.56	5.21	5.47	5.21	RTO
2	2.52	2.77	2.52	2.76	2.52	RTO
3	2.35	2.58	2.35	2.59	2.35	RTO
4	1.74	1.98	1.74	2	1.74	RTO
5	4.37	4.63	6.9	7.16	6.9	RTO
6	7.22	7.46	7.22	7.48	7.22	RTO
7	3.32	3.57	3.32	3.58	3.32	RTO
8	2.12	2.39	2.12	2.4	2.12	RTO
9	6.21	6.48	6.21	6.44	6.21	RTO
10	6.75	6.97	6.75	6.99	6.75	RTO

4.2.2 Analisa RTT Ping

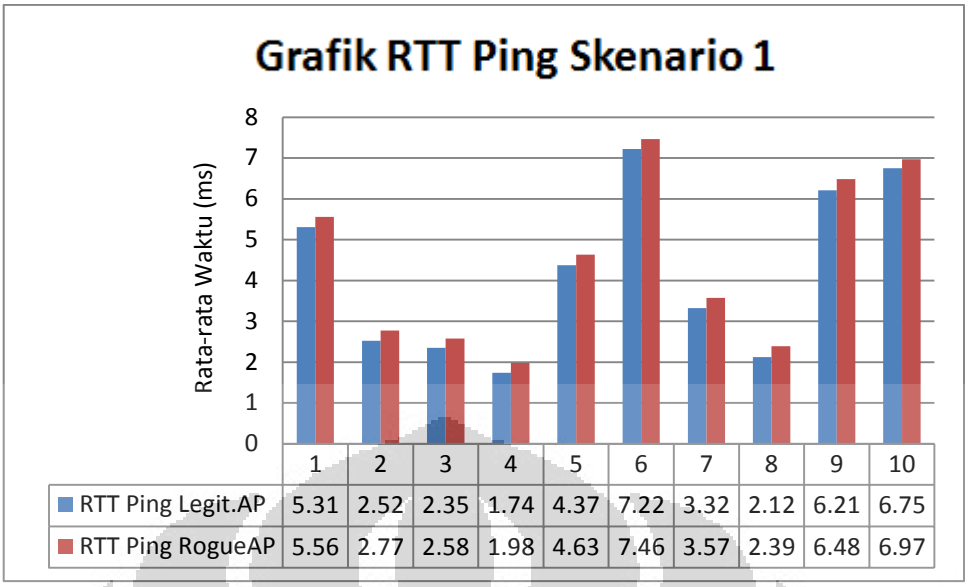
a. Analisa pengukuran RTT Ping Skenario 1

Tabel pengukuran RTT Ping Skenario 1 beserta persentase kenaikan *delay* dapat dilihat pada tabel 4.2 dibawah ini.

Tabel 4.2 Persentase kenaikan *delay* pada Skenario 1

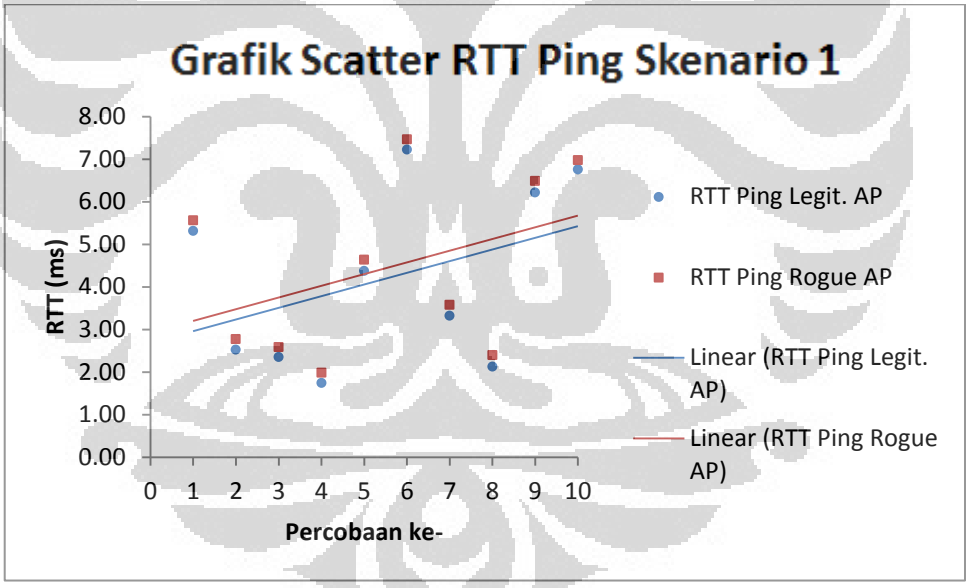
Iterasi ke-	SKENARIO 1			
	Cl1-AP-Server	Cl2-RAP-Server	Selisih Waktu (ms)	Persentase
	Rata-rata waktu RTT PING (ms)			
1	5.31	5.56	0.25	4.71%
2	2.52	2.77	0.25	9.92%
3	2.35	2.58	0.23	9.79%
4	1.74	1.98	0.24	13.79%
5	4.37	4.63	0.26	5.95%
6	7.22	7.46	0.24	3.32%
7	3.32	3.57	0.25	7.53%
8	2.12	2.39	0.27	12.74%
9	6.21	6.48	0.27	4.35%
10	6.75	6.97	0.22	3.26%

Grafik RTT Ping Skenario 1 terhadap waktu dapat dilihat pada gambar 4.8



Gambar 4. 8 Grafik Hasil Pengukuran RTT Ping Skenario 1 terhadap waktu

Grafik Scatter Ping Skenario 1 terhadap waktu dapat dilihat pada gambar 4.9 dibawah ini.



Gambar 4. 9 Grafik Scatter RTT Ping Skenario 1

Skenario 1 adalah topologi dimana *Rogue* AP terh ubung ke jaringan *legitimate* melalui perantara kabel.

Berdasarkan tabel 4.1 dan grafik pada Gambar 4.7 diatas, hasil pengukuran menunjukkan terdapat perbedaan waktu pada pengiriman paket melalui jalur *rogue* dan jalur *legitimate*. Pengambilan data dilakukan sebanyak 100 kali ping dan mendapatkan rata-rata ping dari tiap iterasi tersebut. Dari tabel 4.2 dapat dilihat bahwa terdapat perbedaan *delay* yang cukup lebar dan dibeberapa

iterasi terlihat cukup kecil. Perbedaan *delay* tersebut tidak mencapai lebih dari 15% dari *delay* pada jalur *legitimate* AP. Rata-rata RTT untuk 10 iterasi pada jalur *legitimate* adalah sebesar 4.191 ms dan *rogue* sebesar 4.439 ms. Perbedaan waktu tempuh yang terjadi cukup besar yaitu sebesar 0.248 ms atau penambahan sebesar 5.91% waktu tempuh jalur *legitimate*.

Penambahan waktu untuk jalur *Rogue* ini disebabkan perbedaan divais yang digunakan untuk menjadi *rogue* merupakan laptop dan menggunakan metode NAT (*Network Address Translation*). Untuk menjadi sebuah *rogue* AP maka laptop harus menjadi *wireless* router dimana menggunakan sistem NAT. Sistem NAT ini sendiri menjadi penyumbang *delay* dalam pengiriman paket sehingga paket-paket tersebut mengalami keterlambatan pengiriman ketika di proses di komputer *attacker*. Paket ICMP yang telah mencapai switch mendapatkan kualitas yang sama dengan paket ICMP dari *legitimate* AP sehingga perbedaan *delay* atau waktu tempuh hanya terjadi di komputer *attacker*.

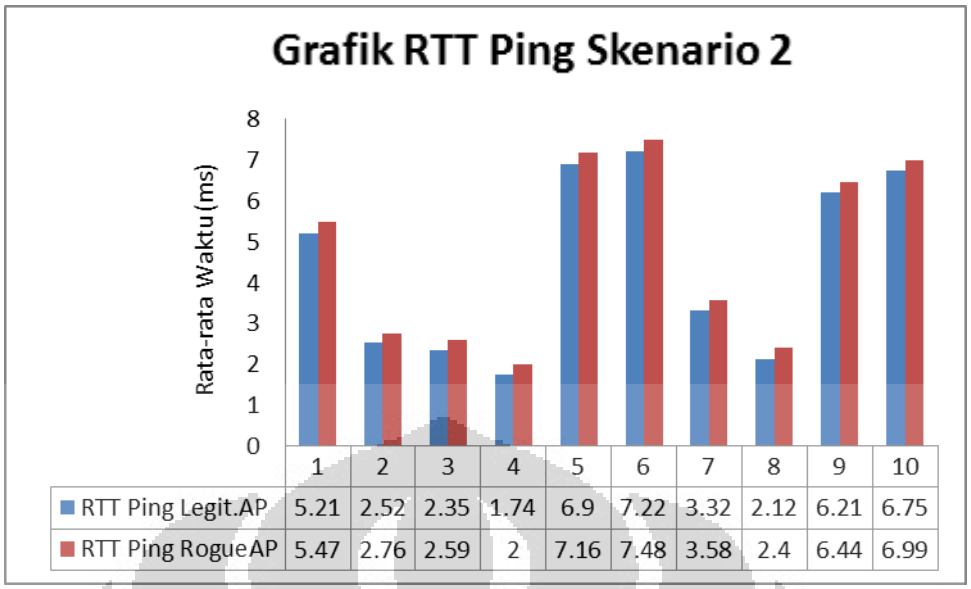
b. Analisa pengukuran RTT Ping Skenario 2

Tabel pengukuran RTT Ping Skenario 2 beserta persentase kenaikan *delay* dapat dilihat pada tabel 4.3 dibawah ini.

Tabel 4.3 Persentase kenaikan *delay* RTT Ping pada Skenario 2

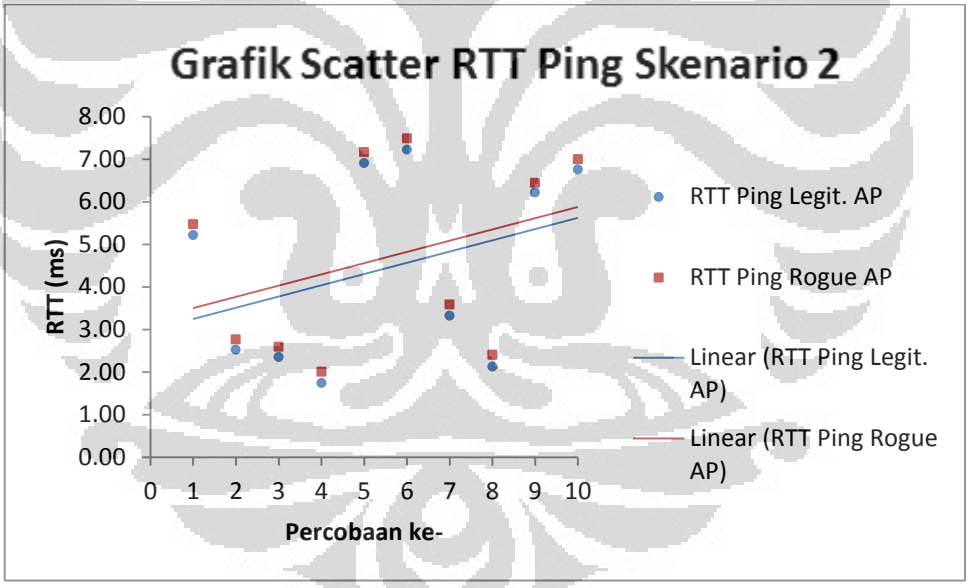
Iterasi ke-	TOPOLOGI 2			
	Cl1-AP-Server	Cl2-RAP-Server	Selisih Waktu (ms)	Persentase
	Rata-rata waktu RTT PING (ms)			
1	5.21	5.47	0.26	4.99%
2	2.52	2.76	0.24	9.52%
3	2.35	2.59	0.24	10.21%
4	1.74	2	0.26	14.94%
5	6.9	7.16	0.26	3.77%
6	7.22	7.48	0.26	3.60%
7	3.32	3.58	0.26	7.83%
8	2.12	2.4	0.28	13.21%
9	6.21	6.44	0.23	3.70%
10	6.75	6.99	0.24	3.56%

Grafik RTT Ping Skenario 2 terhadap waktu dapat dilihat pada gambar 4.10 dibawah ini.



Gambar 4. 10 Grafik Hasil Pengukuran RTT Ping Topologi 2 terhadap waktu

Grafik Scatter Ping Skenario 2 terhadap waktu dapat dilihat pada gambar 4.11 dibawah ini.



Gambar 4. 11 Grafik Scatter RTT Ping Skenario 2

Topologi 2 adalah topologi dimana *Rogue* AP mendapatkan koneksi melalui perantara jaringan *wireless* dari *legitimate* AP.

Berdasarkan tabel 4.2 dan grafik pada Gambar 4.8 diatas, hasil pengukuran menunjukkan terdapat peningkatan perbedaan waktu yang sangat signifikan dibandingkan dengan topologi 1. Pengambilan data dilakukan sebanyak 100 kali ping dan mendapatkan rata-rata ping dari tiap iterasi tersebut. Dari tabel 4.2 dapat dilihat persebaran *delay* RTT topologi 2 dengan topologi 1 dimana memiliki

persebaran acak, tidak terdapat pola pada perbedaan *delay* satu iterasi dengan iterasi lainnya. Hal ini dikarenakan banyaknya parameter eksternal yang tidak dapat dikendalikan seperti frekuensi pengganggu, *delay* pada kabel, switch yang digunakan dan parameter-parameter lainnya.

c. Analisa pengukuran RTT Ping Skenario 3

Topologi 3 adalah topologi dimana *Rogue* AP mendapatkan koneksi internet melalui jaringan modem.

Tabel pengukuran RTT Ping Skenario 3 beserta persentase kenaikan *delay* dapat dilihat pada tabel 4.4 dibawah ini.

Tabel 4.4 Persentase kenaikan *delay* pada Skenario 3

Iterasi ke-	SKENARIO 3			
	Cl1-AP-Server	Cl2-RAP-Server	Selisih Waktu (ms)	Persentase
	Rata-rata waktu RTT PING (ms)			
1	5.21	RTO	-	-
2	2.52	RTO	-	-
3	2.35	RTO	-	-
4	1.74	RTO	-	-
5	6.9	RTO	-	-
6	7.22	RTO	-	-
7	3.32	RTO	-	-
8	2.12	RTO	-	-
9	6.21	RTO	-	-
10	6.75	RTO	-	-

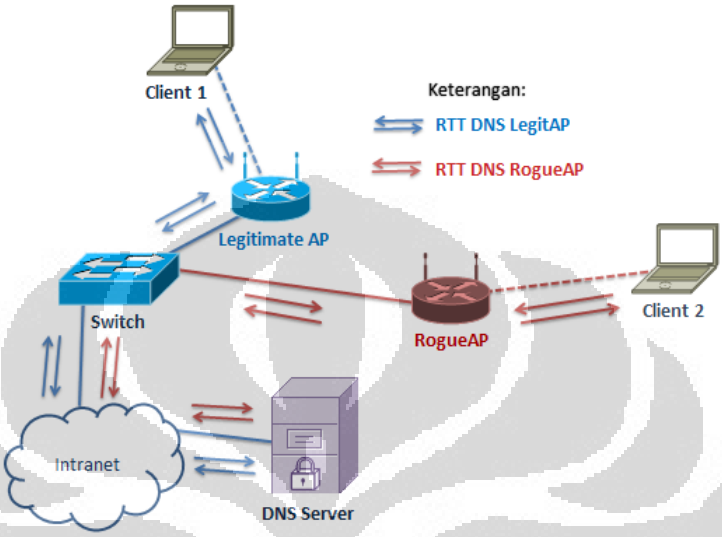
Untuk topologi ini akses ICMP di blok di *server* dari arah luar (Internet) sehingga paket yang dikirimkan akan ditolak dan menyebabkan Request *Time* Out (RTO). Ketika hal itu terjadi, indikasi jalur *rogue* dengan mudah dikenali dan dideteksi. Sebuah *server* yang hanya dapat diakses melalui jaringan intranet diperlukan untuk dapat memberikan nilai rata-rata waktu namun apabila tidak didapatkan waktu tempuh atau dengan kata lain RTO maka dengan mudah hal tersebut di kenali sebagai jalur *rogue*.

4.2.3 Pengukuran RTT DNS

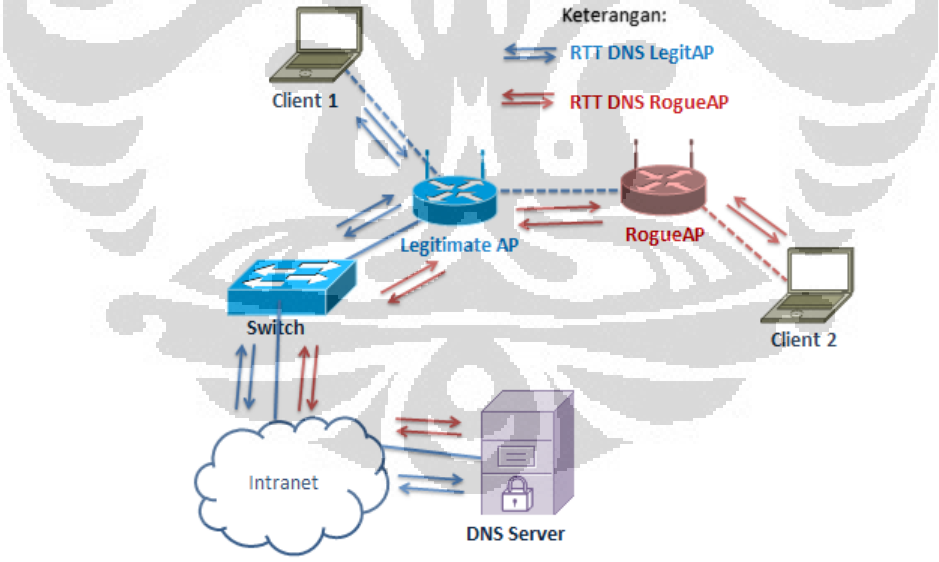
DNS *query* digunakan sebagai perintah untuk mengirimkan paket permintaan penerjemaahan suatu domain ke suatu IP Address dari sebuah DNS *Server*. Pengukuran DNS *query* ini menggunakan DNSBenchmark (http://www.grc.com/files/DNSBench.exe). Sifat dari DNS adalah *connectionless* dikarenakan tergolong dalam UDP (User Datagram Protocol). Oleh karena

connectionless maka protokol ini digunakan untuk menghitung nilai RTT dan kualitas jaringan.

Untuk perhitungan RTT dengan menggunakan DNS *query* dilakukan pada topologi 1, 2, dan 3 seperti pada gambar berikut.

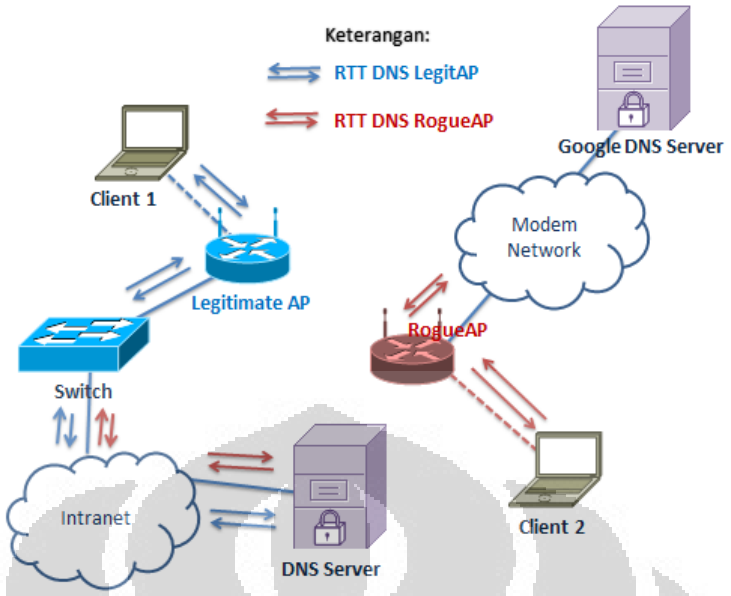


Gambar 4. 12 Topologi Pengujian RTT DNS pada Skenario 1



Gambar 4. 13 Topologi Pengujian RTT DNS pada Skenario 2

Pengukuran RTT DNS pada Skenario 1 dan 2 dilakukan *benchmarking* dengan mengirimkan *query* dns ke DNS *Server* UI (152.118.24.10) pada *Client* 1 dan *Client* 2. Proses dilakukan sebanyak 10 kali dan dilakukan bergantian antara *client* 1 dan *client* 2.



Gambar 4. 14 Topologi Pengujian RTT DNS pada Skenario 3

Pada Skenario 3, DNS *Benchmarking* dengan membandingkan performa antara menggunakan DNS *Server* Intranet kampus UI (152.118.24.10) dengan DNS *Server* Google (8.8.8.8). Hal ini dikarenakan lingkungan internet di Universitas Indonesia tidak memperbolehkan *query* DNS ke DNS *server*nya. Asumsi yang diberikan disini adalah *attacker* memberikan alamat DNS modem ke *client* 2 dimana menggunakan alamat DNS *Server* Google.

4.2.4 Analisa RTT DNS

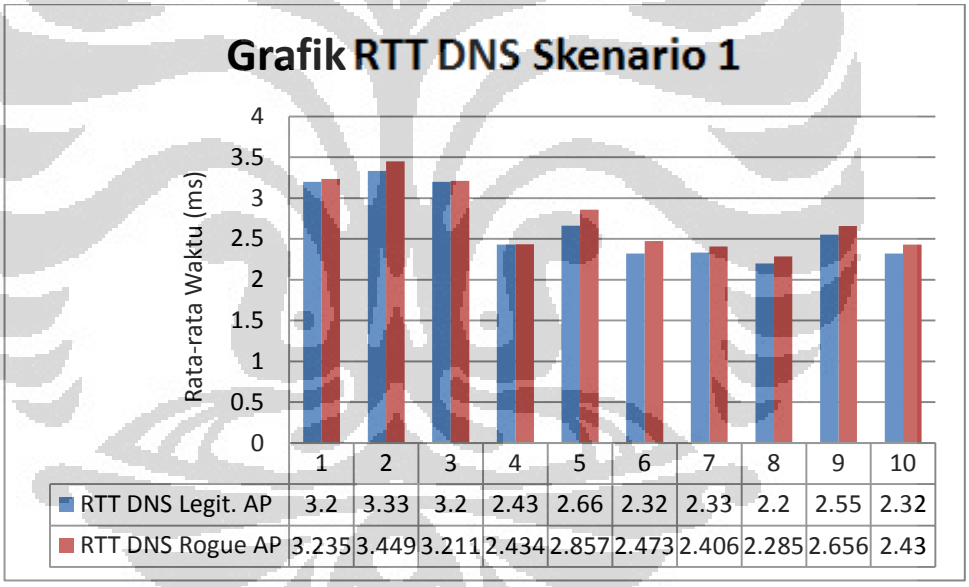
a. Analisa pengukuran RTT DNS Skenario 1

Tabel pengukuran RTT DNS Skenario 1 beserta persentase kenaikan *delay* dapat dilihat pada tabel 4.5 dibawah ini.

Tabel 4.5 Persentase kenaikan *delay* RTT DNS pada Skenario 1

Iterasi ke-	SKENARIO 1			
	Cl-AP-DNS Server	Cl-RAP-DNS Server	Selisih Waktu (ms)	Persentase
	Rata-rata waktu PING (ms)			
1	3.2	3.235	0.035	1.09%
2	3.33	3.449	0.119	3.57%
3	3.2	3.211	0.011	0.34%
4	2.43	2.434	0.004	0.16%
5	2.66	2.857	0.197	7.41%
6	2.32	2.473	0.153	6.59%
7	2.33	2.406	0.076	3.26%
8	2.2	2.285	0.085	3.86%
9	2.55	2.656	0.106	4.16%
10	2.32	2.43	0.11	4.74%

Grafik RTT DNS Skenario 1 terhadap waktu dapat dilihat pada gambar 4.15 dibawah ini.



Gambar 4. 15 Grafik hasil pengukuran RTT DNS Topologi 1 terhadap waktu

Pada pengukuran RTT DNS di skenario 1 kedua *client* memiliki konfigurasi DNS *Server* yang sama dan *Rogue*AP terhubung secara langsung sehingga *delay* yang dihasilkan dari Legit. AP dan *Rogue* AP pada *wired-connection* sama. Perbedaan mendasar dari jalur *rogue* dan *legitimate* terletak di konfigurasi NAT oleh *attacker*. Akan tetapi perbedaan *delay* tersebut tidak terlalu terlihat karena sesungguhnya DNS merupakan *connectionless* sehingga dalam pengirimannya tidak menunggu *reply* untuk mengirimkan paket selanjutnya.

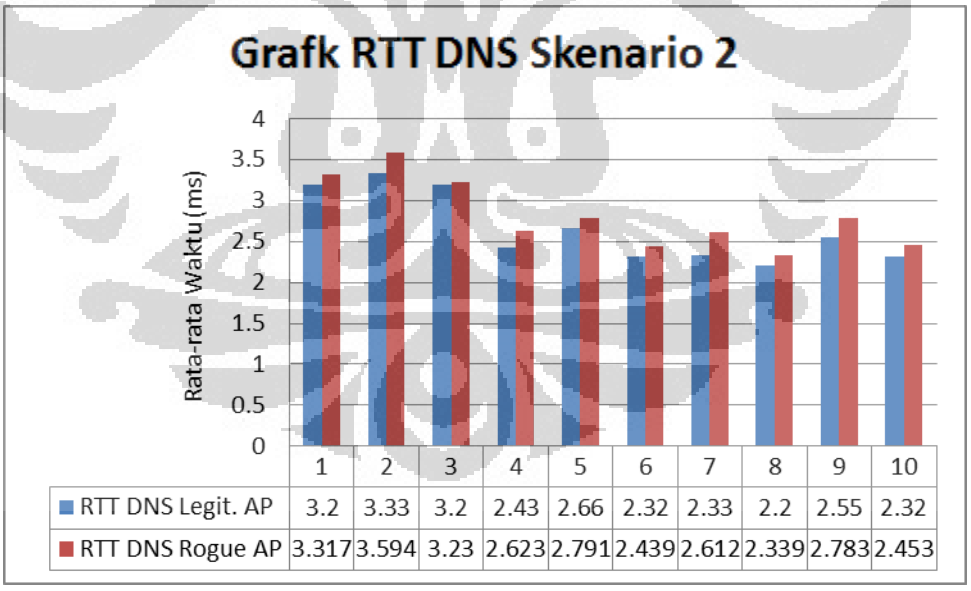
b. Analisa pengukuran RTT DNS pada Skenario 2

Tabel pengukuran RTT DNS Skenario 2 beserta persentase kenaikan *delay* dapat dilihat pada tabel 4.6 dibawah ini.

Tabel 4.6 Persentase kenaikan *delay* RTT DNS pada Skenario 2

Iterasi ke-	PENGUKURAN RTT DNS PADA Skenario 2			
	Cl-AP-DNS Server	Cl-RAP-DNS Server	Selisih Waktu (ms)	Persentase
	Rata-rata waktu PING (ms)			
1	3.2	3.317095046	0.117095046	3.66%
2	3.33	3.593798532	0.263798532	7.92%
3	3.2	3.229753049	0.029753049	0.93%
4	2.43	2.623133261	0.193133261	7.95%
5	2.66	2.790815752	0.130815752	4.92%
6	2.32	2.438940375	0.118940375	5.13%
7	2.33	2.612337616	0.282337616	12.12%
8	2.2	2.339191831	0.139191831	6.33%
9	2.55	2.783138209	0.233138209	9.14%
10	2.32	2.453077572	0.133077572	5.74%

Grafik RTT DNS Skenario 2 terhadap waktu dapat dilihat pada gambar 4.10 dibawah ini.



Gambar 4. 16 Grafik hasil pengukuran RTT DNS skenario 2 terhadap waktu

Pada pengukuran skenario 2 ketika di lakukan rata-rata terhadap total persentase kenaikan RTT dari nilai *default* jalur *legitimate* sehingga didapatkan rata-rata kenaikan menyeluruh sebesar 6.38%. Besarnya *delay* pada DNS *query* yang dikirimkan merupakan hasil dari penggunaan media *wireless* dan NAT pada

komputer *attacker*. Akan tetapi kenaikan ini masih tergolong kecil karena tidak mencapai 10% kenaikan nilai *default* atau nilai RTT pada jalur *legitimate*. Besarnya perbedaan *delay* RTT DNS untuk pengukuran skenario 2 melebihi

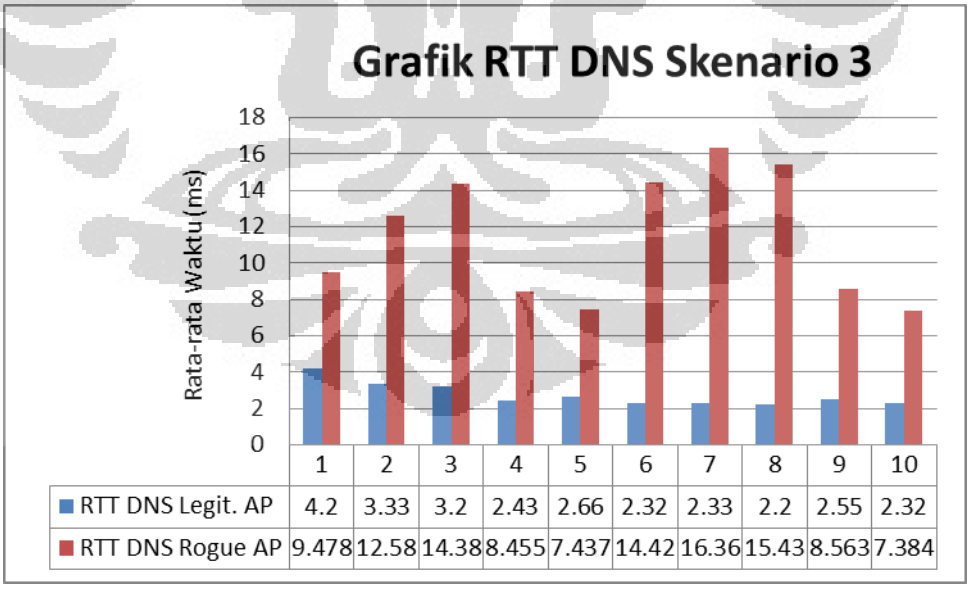
c. Analisa Pengukuran RTT DNS pada Skenario 3

Tabel pengukuran RTT DNS Skenario 3 beserta persentase kenaikan *delay* dapat dilihat pada tabel 4.7dibawah ini.

Tabel 4.7 Persentase kenaikan *delay* RTT DNS pada Skenario 3

Iterasi ke-	PENGUKURAN RTT DNS PADA TOPOLOGI 3			
	Cl-AP-DNS Server	Cl-RAP-DNS Server	Selisih Waktu (ms)	Persentase
	Rata-rata waktu PING (ms)			
1	4.2	9.478398369	5.278398369	125.68%
2	3.33	12.5780906	9.248090596	277.72%
3	3.2	14.38470975	11.18470975	349.52%
4	2.43	8.454625014	6.024625014	247.93%
5	2.66	7.437390092	4.777390092	179.60%
6	2.32	14.42361798	12.10361798	521.71%
7	2.33	16.35858943	14.02858943	602.09%
8	2.2	15.43010993	13.23010993	601.37%
9	2.55	8.562765167	6.012765167	235.79%
10	2.32	7.384146581	5.064146581	218.28%

Grafik RTT DNS Skenario 2 terhadap waktu dapat dilihat pada gambar 4.17 dibawah ini.



Gambar 4. 17 Grafik hasil pengukuran RTT DNS Skenario 3 terhadap waktu

Pada pengukursan RTT DNS di skenario 3 kedua *client* melakukan inisiasi DNS *query* ke DNS *Server* yang berbeda. Jarak lompatan (*hop*) antara DNS *Server* modem dan DNS *Server* UI sangat berbeda. Sehingga terlihat jelas

perbedaan *delay* RTT antara keduanya. Persentase selisih waktu antara RTT pada DNS *query* melalui jalur *legitimate* dan jalur *rogue* sangat besar. Lebih dari 100% sehingga dapat dengan mudah melakukan pendeteksian terhadap skenario tersebut dikarenakan ukuran perbedaan RTT sangat besar pada jaringan *rogue*.

4.3 Pengukuran dan Analisa WIDS

Pada bagian ini akan dilakukan pengujian sistem yang sudah dirancang sebelumnya. Pengujian sistem dilakukan dengan dua metode pengujian apakah sistem dapat berfungsi dengan baik dan juga memiliki tingkat *reliability* yang sesuai. Dua metode tersebut adalah :

1. *Functionality test*
2. *Response time*

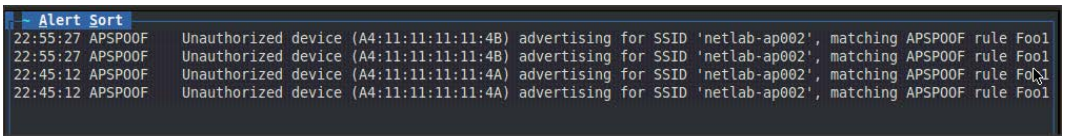
Pengujian pada skripsi ini yaitu menggunakan skenario yang diinginkan yang dapat mendeteksi keberadaan *Rogue*AP dengan konektivitas menggunakan perantara kabel, nirkabel dan modem. *Functionality test* ini nantinya akan menguji apakah sistem WIDS ini dapat berfungsi dengan baik yang sesuai dengan topologi pada skenario yang diinginkan, untuk menghitung *response time* maka dapat menggunakan wireshark dan *timestamp* pada *server* maupun *client*.

4.3.1 *Functionality Test*

Functionality test bertujuan untuk menguji apakah sistem WIDS (*Wireless Intrusion Detection System*) ini dapat berfungsi dengan baik dan juga sesuai dengan kriteria yang diinginkan. Kriteria yang diinginkan tentu saja dapat mendeteksi ketika terdapat adanya *Rogue*AP dalam jaringan maka memberikan *alterting* yang kemudian mengirimkannya ke *server* untuk diolah. WIDS yang digunakan adalah Kismet *Server*, *Client* dan Drone. Kismet *Server* dan *Client* terdapat di *Server* sedangkan Kismet Drone terdapat di AP yang dijadikan *wireless* sensor.

Pada *functionality test* akan dilakukan beberapa pengujian dengan menggunakan 3 skenario yang digunakan pada pengujian RTT Ping dan DNS yaitu pada gambar 4.5, gambar 4.6 dan gambar 4.7. Pengukuran dilakukan sebanyak 10 kali untuk tiap skenario. Pada tiap pengukuran skenario, jarak WIDS dengan *Legitimate* AP dan *Rogue* AP berdekatan atau masih dalam satu area sensor dengan kualitas kekuatan sinyal 90-100%. SSID Legit. AP yang digunakan

adalah netlab-ap002 dengan standar IEEE 802.11g. *Rogue* AP menggunakan standar IEEE 802.11g.



Gambar 4. 18 Contoh pendeteksian *Rogue* AP yang berhasil

Pengukuran *functionality test* dilakukan terhadap 3 skenario tersebut yaitu Skenario 1 *Rogue* AP terhubung ke jaringan *legitimate* melalui perantara kabel, Skenario 2 *Rogue* AP terhubung ke jaringan *legitimate* melalui perantara nirkabel, Skenario 3 *Rogue* AP terhubung ke jaringan modem.

Tabel 4.8 Hasil pengujian *functionality* test pada 3 skenario

Parameter-parameter *Rogue* AP	**Tabel *Functionality Test***		
	Skenario 1	**Skenario 2**	**Skenario 3**
MAC Address Berbeda	Terdeteksi	Terdeteksi	Terdeteksi
MAC Address Sama	-	Tidak terdeteksi	Tidak terdeteksi
IEEE 802.11g	Terdeteksi	Terdeteksi	Terdeteksi
IEEE 802.11b	Terdeteksi	Terdeteksi	Terdeteksi

Berdasarkan tabel 4.8 diatas, terlihat pada parameter *Rogue* AP yang defaultnya tidak mengubah mac *address* dari *wireless interface* dapat terdeteksi oleh Kismet. Akan tetapi ketika *Rogue* AP bermain di SoftAP dimana dapat dengan mudah mengubah mac addressnya dengan aplikasi macchanger pada linux maka untuk skenario 2 dan 3 tidak dapat ter deteksi oleh kismet. Pada skenario 1 untuk parameter mac *address* tidak dapat diubah karena secara *default* hal tersebut di batasin oleh *firmware* bawaan dari *hardware* AP tersebut.

Menurut hasil metode *functionality test* menunjukkan untuk tiap standar yang berbeda tidak mempengaruhi kinerja pendeteksian WIDS ini. Terlihat juga ketika perubahan *wireless* standar yang digunakan oleh *Rogue*AP dari 802.11a ke 802.11g dan masih menunjukkan keberadaaan *Rogue*AP tersebut.

Mon Jun 11 22:45:12 2012 APSPOOF 11 **4a:44:44:44:44:4A** 4a:44:44:44:44:4A FF:FF:FF:FF:FF:FF 00:00:00:00:00:00 Unauthorized device (4a:44:44:44:44:4A) advertising for SSID 'netlab-ap002', matching APSPOOF rule Foo1 with SSID which may indicate spoofing or impersonation.

Mon Jun 11 22:55:27 2012 APSPOOF 11 **4a:44:44:44:44:4B** 4a:44:44:44:44:4B FF:FF:FF:FF:FF:FF 00:00:00:00:00:00 Unauthorized device (4a:44:44:44:44:4B) advertising for SSID 'netlab-ap002', matching APSPOOF rule Foo1 with SSID which may indicate spoofing or impersonation.

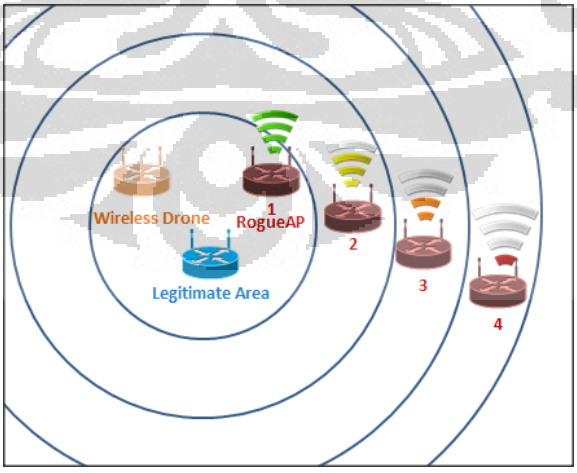
Berdasarkan hasil log diatas di ketahui terdapat SSID yang sama dengan MAC Address yang berbeda dan dengan begitu mengindikasikan telah terjadi spoofing AP atau *Rogue*AP terdeteksi.

4.3.2 *Response Time*

Untuk mengetahui tingkat kehandalan dari suatu sistem IDS maka perlu melihat dari beberapa parameter dan salah satunya adalah parameter *response time*. Pada pengujian ini *response time* adalah waktu yang dibutuhkan WIDS untuk merespon terhadap keberadaan RogueAP dimana dicatat waktu mulai RogueAP dibuat hingga terdeteksi oleh WIDS.

Pengujian ini dilakukan dengan mengubah beberapa parameter yang akan dijadikan patokan perbedaan kondisi. Parameter tersebut yaitu perbedaan jarak antara Legitimate AP dengan Rogue AP berdasarkan kekuatan sinyal.

Berikut adalah gambar topologi untuk pengukuran *response time* berdasarkan jarak RogueAP ke LegitAP berdasarkan kekuatan sinyal LegitAP.



Gambar 4. 19 Topologi pengukuran response time berdasarkan kekuatan sinyal

Untuk mengukur response time digunakan timestamp pada log alert kismet server dan timestamp pada perintah di komputer attacker. Untuk mendapatkan

nilai response time ini akan dilakukan pengurangan antara waktu ketika program hostapd (program RogueAP) dengan timestamp alert pada kismet server. Pada program hostapd, RogueAP akan mendapatkan identitas hardware (MAC Address) yang berbeda tiap menjalankannya dan akan terdeteksi. Hal ini bertujuan untuk mempermudah pembacaan alert pada kismet server. Dikedua tempat yaitu komputer RogueAP dengan Server di sinkronisasikan waktu dengan menggunakan perintah

```
# ntpdate ntp.ubuntu.com
```

Berikut adalah contoh alert yang berhasil pada mac address 4a:44:44:44:44:41 :

```
Wed Jun 13 00:47:20 2012 APSPOOF 11 4a:44:44:44:44:41 4a:44:44:44:44:41
FF:FF:FF:FF:FF:FF 00:00:00:00:00:00 Unauthorized device (4a:44:44:44:44:41)advertising for
SSID 'netlab-ap002', matching APSPOOF rule Foo1 with SSID which may indicate spoofing or
impersonation.
```

Berikut adalah contoh dari timestamp pada program hostapd :

```
1438  Jun13 - 01:54:57 hostapd -dd /etc/hostapd/hostapd.conf
```

Berikut adalah hasil percobaan pada kondisi signal strength 75%-100%

Tabel 4.9 Hasil response time pada signal strength 75%-100%

MAC Address	Kismet Alert Log	Command History	Delay (second)
4a:44:44:44:a1	1:15:00	1:15:01	1
4a:44:44:44:a2	1:15:08	1:15:09	1
4a:44:44:44:a3	1:15:18	1:15:19	1
4a:44:44:44:a4	1:15:29	1:15:30	1
4a:44:44:44:a5	1:15:40	1:15:10	1
4a:44:44:44:a6	1:15:48	1:15:49	1
4a:44:44:44:a7	1:16:00	1:16:01	1
4a:44:44:44:a8	1:16:13	1:16:14	1
4a:44:44:44:a9	1:16:20	1:16:21	1
4a:44:44:44:a4	1:16:26	1:16:27	1

Berikut adalah hasil percobaan pada kondisi signal strength 50%-74%

Tabel 4.10 Hasil response time pada signal strength 50%-74%

MAC Address	Kismet Alert Log	Command History	Delay (second)
4a:33:33:33:a1	3:07:54	3:07:54	1
4a:33:33:33:a2	3:08:12	3:08:12	1
4a:33:33:33:a3	3:08:27	3:08:27	1
4a:33:33:33:a4	3:08:39	3:08:39	1
4a:33:33:33:a5	3:08:52	3:08:50	2
4a:33:33:33:a6	3:09:06	3:09:05	1
4a:33:33:33:a7	3:09:20	3:09:18	2
4a:33:33:33:a8	3:09:31	3:09:31	1
4a:33:33:33:a9	3:09:43	3:09:41	2
4a:33:33:33:a4	3:09:56	3:09:56	1

Berikut adalah hasil percobaan pada kondisi signal strength 25%-49%

Tabel 4.11 Hasil response time pada signal strength 25%-49%

MAC Address	Kismet Alert Log	Command History	Delay (second)
4a:22:22:22:22:21	3:13:30	3:13:25	5
4a:22:22:22:22:22	3:13:41	3:13:38	3
4a:22:22:22:22:23	3:13:51	3:13:46	5
4a:22:22:22:22:24	3:14:04	3:14:59	5
4a:22:22:22:22:25	3:14:52	3:14:50	2
4a:22:22:22:22:26	3:14:27	3:14:25	2
4a:22:22:22:22:27	3:14:36	3:14:34	2
4a:22:22:22:22:28	3:14:53	3:14:47	6
4a:22:22:22:22:29	3:15:02	3:15:00	2
4a:22:22:22:22:2A	3:15:15	3:15:10	5

Berikut adalah hasil percobaan pada kondisi signal strength 1%-24%

Tabel 4.12 Hasil response time pada signal strength 1%-24%

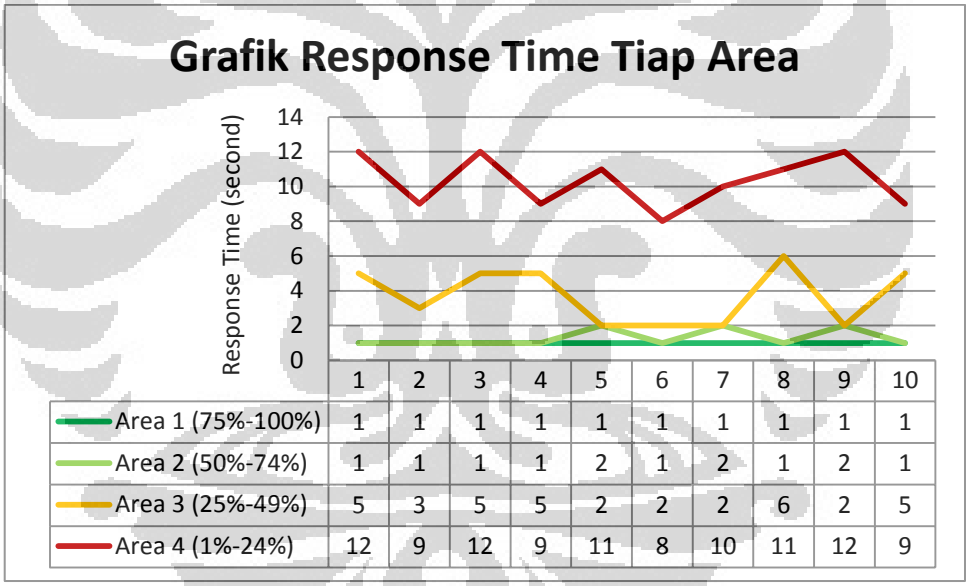
MAC Address	Kismet Alert Log	Command History	Delay (second)
4a:11:11:11:11:21	3:25:30	3:25:18	12
4a:11:11:11:11:11	3:25:41	3:25:32	9
4a:11:11:11:11:23	3:25:58	3:25:46	12
4a:11:11:11:11:24	3:27:08	3:26:59	9
4a:11:11:11:11:25	3:28:01	3:27:50	11
4a:11:11:11:11:26	3:28:11	3:28:03	8
4a:11:11:11:11:27	3:28:30	3:28:20	10

4a:11:11:11:11:28	3:29:11	3:29:00	11
4a:11:11:11:11:29	3:29:34	3:29:12	12
4a:11:11:11:11:2A	3:29:49	3:29:40	9

Tabel 4.13 Hasil response time pada tiap-tiap area

Kekuatan Sinyal	Percobaan respons time ke-									
	1	2	3	4	5	6	7	8	9	10
	Response Time (second)									
Area 1 (75%-100%)	1	1	1	1	1	1	1	1	1	1
Area 2 (50%-74%)	1	1	1	1	2	1	2	1	2	1
Area 3 (25%-49%)	5	3	5	5	2	2	2	6	2	5
Area 4 (1%-24%)	12	9	12	9	11	8	10	11	12	9

Berikut adalah grafik response time pada tiap area



Gambar 4. 20 Grafik *Response Time* tiap Area

Berdasarkan percobaan *response time* terlihat bahwa ketika jaraknya masih berdekatan dengan parameter kekuatan sinyal sekitar 75%-100% maka hanya butuh waktu 1 detik bagi WIDS untuk mendeteksi keberadaannya.

Delay naik ketika parameter kekuatan sinyal berkisar antara 50%-74%. Besarnya rata-rata kenaikan delay pada area 2 ini sebesar 1 detik dibandingkan pada area 1, terlihat pada tabel 4.10. Kenaikan delay dari area 1 dikarenakan sudah terdapat jarak yang berarti untuk WIDS Sensor mendeteksi lingkungan sekitarnnya

Tahapan berikutnya adalah untuk area 3 yaitu 25%-49% dimana terlihat pada tabel 4.11. Waktu delay sudah mencapai minimal 5 detik dan terdapat peningkatan delay sebesar 4 detik dari area 2. Hal ini diakibatkan jarak yang telah diperpanjang sehingga delay bertambah.

Tahapan terakhir pengujian adalah bagi area 4 yaitu 1%-24% dimana terlihat pada tabel 4.12. Ditabel ini sangat besar delay yang didapatkan yaitu minimal 8 detik dimana terjadi peningkatan signifikan bila dibandingkan area 1. Hal ini di sebabkan jarak yang sangat jauh dari WIDS dan terdapat beberapa noise atau hambatan bagi WIDS untuk menangkap beacon yang dikirimkan oleh RogueAP.

BAB 5
PENUTUP

5.1 Kesimpulan

Pada bab penutup, berikut adalah beberapa kesimpulan dari penulisan skripsi ini :

1. *Rogue Access Point* atau *Access Point* yang tidak *legitimate* merupakan salah satu kasus serangan di jaringan *wireless* yang sangat mudah dilakukan dan sulit untuk dideteksi dengan akurat.
2. Salah satu bentuk dari pendekatan *Wireless-side* adalah penggunaan Sensor berupa drone yang akan mengamati lalu lintas jaringan *wireless* dan mengirimkan informasi tentang kemungkinan keberadaan *Rogue Access Point*.
3. *Wireless-side* lebih menghabiskan dana untuk pembelian sebuah sensor drone hanya pada satu jaringan lokal untuk tiap daerah *Access Point.*
4. Untuk pendekatan *Wired-side*, contohnya menganalisa trafik pada jaringan mulai dari *wireless* hingga wired. Trafik tersebut akan dibandingkan dengan signature yang telah ditetapkan. Apabila terdapat kecocokan atau tidak maka akan diketahui keberadaan *Rogue Access Point* tersebut.
5. Response time pendeteksian RAP di WIDS sensor tergantung pada jarak dan kekuatan sinyal antara WIDS dengan RogueAP.
6. Pada Area 1 dan 2 rata-rata response time berkisar 1-2 detik sedangkan pada Area 3 sebesar 3.7 detik dan Area 4 (1%-24%) sekitar 10.4 detik.

Diperlukan sebuah identitas unik yang tidak dapat dipalsukan dan pendekatan *hardware* untuk mendapatkan keakuratan dalam analisa trafik RTT serta pendeteksian berbasiskan *fingerprint*. Sistem yang di implementasikan membuktikan RTT dapat dijadikan parameter dalam mendeteksi Rogue Access Point.

DAFTAR ACUAN

[1] (2011). Diakses dari en.wikipedia.org/wiki/*Wireless*_access_point.

[2] (2011). Diakses dari homesupport.cisco.com/en-us/*wireless*/lbc/wrt160n.

[3] (2011). Diakses dari en.wikipedia.org/wiki/Wi-Fi.

[4] Personal Area Network. (2011). Diakses dari en.wikipedia.org/wiki/Personal_area_*network*.

[5] A hybrid *rogue* access LAN.point protection framework for commodity Wi-Fi networks. (2011).

[6] Consulting, G. S. (2011). *Rogue Access Point* Detectoin and Mitigation MUM 2011.

[7] discovering-*rogue*-*wireless*-access-points-kismet-disposable-*hardware*. (2010).

[8] Guangzhi Qu, M. M. (2011). RAPiD: An Indirect *Rogue Access* Points *Detection System* .

[9] Hao Han, B. S. (2011). A Timing-Based Scheme for *Rogue* AP. *IEEE*, 1-14.

[11] Diakses dari http://www.cs.wright.edu/~pmateti/InternetSecurity/Lectures/WirelessHacks.

[12] L. Ma, A. Y. (2008). A hybrid *rogue* access LAN.point protection framework for commodity Wi-Fi networks. *IEEE*.

[13] Liran Ma, A. Y. (2011). A Hybrid *Rogue Access Point* Protection Framework for Commodity Wi-Fi Networks.

[14] *Rogue Access Point*. (2011). Diakses dari http://www.*rogue*ap.com.

[15] Rogue Access Point Detection: Automatically Detect And Manage Wireless Threats To Your Network . (2010).

[16] Diakses dari http://images.google.com.

[17] Diakses dari http://www.cisco.com.

[18] Diakses dari http://blog.tenablesecurity.com/2009/08/using-nessus-to-discover-*rogue*-access-points.html

[19] Diakses dari http://www.kismet.com

LAMPIRAN

Semua dokumentasi mengenai skripsi ini dapat dilihat di

http://www.ee.ui.ac.id/netlab/roguedetection

Berikut ini adalah instalasi Kismet di dalam platform Ubuntu 9.04 :

1. install kismet

apt-get install kismet

2. kemudian konfigurasi kismet dengan memasukkan tipe wireless card, dan network interface anda :

gedit gedit /etc/kismet/kismet.conf

kemudian cari tulisan seperti ini :

YOU MUST CHANGE THIS TO BE THE SOURCE YOU WANT TO USE

source=type, interface, addname

selanjutnya ganti dengan :

source=ipw2200,wlan0,encrypted

setelah selesai Save. kemudian masuk sebagai root jalankan Kismet, dengan mengetikkan

$sudo Kismet

*ket :

-type : tipe wireless card anda,pada kasus ini saya menggunakan intel pro 5300agn sebagai wireless card.

-interface : network interface yg anda gunakan, kalo saya menggunakan wlan0 sebagai interface.untuk mengetahui interface yang anda gunakan ketik iwconfig pada terminal.

- addname : nama yang akan dikenali oleh kismet

Cara Instalasi Kismet Drone

firmware yang dipakai:

http://downloads.openwrt.org/backfire/10.03.1/ar71xx/openwrt-ar71xx-tl-mr3420-v1-squashfs-factory.bin

Install firmware ke accesspoint.

Setting IP

install paket berikut:

block-mount

kmod-usb-storage

kmod-fs-ext4 -> untuk yg menggunakan EXT2, EXT3, EXT4

coba buat direktori untuk memastikan folder mounting ada

mkdir /mnt/sda5

edit /etc/opkg.conf

```
config 'global' 'automount'
        option 'from_fstab' '1'
        option 'anon_mount' '1'

config 'global' 'autoswap'
        option 'from_fstab' '1'
        option 'anon_swap' '0'

config 'mount'
        option 'fstype' 'ext2'
        option 'options' 'rw,sync'
        option 'enabled' '1'
        option 'device' '/dev/sda5'
        option 'target' '/mnt/sda5'

config 'swap'
        option 'device' '/dev/sda6'
        option 'enabled' '1'
/etc/init.d/fstab enable
/etc/init.d/fstab start
```

buat extroot

```
tar -C /overlay -cvf - . | tar -C /mnt/sda5 -xf -
```

```
mkdir -p /tmp/cproot
mount --bind / /tmp/cproot
tar -C /tmp/cproot -cvf - . | tar -C /mnt/sda5 -xf -
umount /tmp/cproot
```

lalu ubah fstabnya di /etc/config/fstab

```
config 'global' 'automount'
	option 'from_fstab' '1'
	option 'anon_mount' '1'

config 'global' 'autoswap'
	option 'from_fstab' '1'
	option 'anon_swap' '0'

config 'mount'
	option 'fstype' 'ext2'
	option 'options' 'rw,sync'
	option 'enabled' '1'
	option 'device' '/dev/sda5'
	option 'target' '/mnt/sda5'
	option 'is_rootfs' '1'

config 'swap'
	option 'device' '/dev/sda6'
	option 'enabled' '1'
```

untuk berjaga-jaga, coba buat is_rootfs nya dalam keadaan 0 lalu reboot, kalau bisa dimount sda5 maka jadikan 1 dan reboot lagi

kalau ingin yang install ke usb :

```
edit /etc/opkg.conf
berikan: dest usb /opt
```

jalankan

ln -s /mnt/sda5 /opt

edit /etc/profile

```
export PATH=<current default path>:/opt/bin:/opt/sbin:/opt/usr/bin:/opt/usr/sbin

export LD_LIBRARY_PATH=<current default LD library path>:/opt/lib:/opt/usr/lib
```

referensi:

http://livebeta.kaskus.us/thread/000000000000000010124215/nubie-berbagi-complete-step-by-step-openwrt--3g-router-tp-link-mr3420-mr3220

http://wiki.openwrt.org/doc/howto/usb.storage

http://wiki.openwrt.org/doc/howto/extroot

```
config 'global' 'automount'
        option 'from_fstab' '1'
        option 'anon_mount' '1'
config 'global' 'autoswap'
        option 'from_fstab' '1'
        option 'anon_swap' '0'
config 'mount'
        option 'fstype' 'ext2'
        option 'options' 'rw,sync'
        option 'enabled' '1'
        option 'device' '/dev/sda5'
        option 'target' '/mnt/sda5'
config 'swap'
        option 'device' '/dev/sda6'
        option 'enabled' '1'
```

cara upgrade openwrt : sysupgrade *bin