SISTEM DETEKSI SERANGAN
ROGUE ACCESS POINT

Disusun Oleh:

N a m a : Washayatul Iman
NIM : 15523130

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
2020

HALAMAN PENGESAHAN DOSEN PEMBIMBING

SISTEM DETEKSI SERANGAN ROGUE ACCESS POINT

TUGAS AKHIR

Disusun Oleh:

N a m a : Washayatul Iman

NIM : 15523130

Yogyakarta, 1 Maret 2020

Pembimbing,

(Dr. Syarif Hidayat, S.Kom., M.I.T.)

HALAMAN PENGESAHAN DOSEN PENGUJI

SISTEM DETEKSI SERANGAN ROGUE ACCESS POINT

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika – Program Sarjana di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 11 Juni 2020

Tim Penguji

Dr. Syarif Hidayat, S.Kom., M.I.T. _____

Anggota 1

Fayruz Rahma, S.T., M.Eng. _____

Anggota 2

Taufiq Hidayat S.T., M.C.S. _____

Mengetahui,
Ketua Program Studi Informatika – Program Sarjana
Fakultas Teknologi Industri
Universitas Islam Indonesia

(Dr. Raden Teduh Dirgahayu, S.T., M.Sc.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Washayatul Iman

NIM : 15523130

Tugas akhir dengan judul:

SISTEM DETEKSI SERANGAN ROGUE ACCESS POINT

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung resiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 1 Maret 2020

(Washayatul Iman)

HALAMAN PERSEMBAHAN

Tugas Akhir ini saya persembahkan kepada kedua orang tua saya dan teman-teman yang mungkin akan menggunakannya sebagai referensi ataupun mengembangkan penelitian ini.

HALAMAN MOTO

"Bukan orang lain yang mematikan mimpi-mimpi mu, tapi rasa malasmu sendiri"

"Bukan masalah tepat waktu, tetapi segalanya di waktu yang tepat"

"jangan biarkan hari berlalu tanpa sesuatu yang berguna"

"teruslah mencari, karena dunia itu luas"

"karena waktu mudamu adalah gambaran untuk hari tuamu"

KATA PENGANTAR

بِـــــــسـمِ اللهِ الرَّحْمَنِ الرَّحِيْمِ

Assalamulaikum Wr. Wb.

Segala puji dan syukur penulis panjatkan ke hadirat Tuhan yang Maha Esa karena rahmat, hidayah serta karunia-Nya penyusunan tugas akhir ini dapat terlaksana dengan tepat waktu.

Proses penyusunan tugas akhir ini dapat penulis laksanakan dengan bantuan serta bimbingan dari berbagai pihak. Oleh karena itu, penyusun bermaksud untuk menyampaikan ucapan terima kasih yang sebesar besarnya kepada:

1. Allah SWT yang telah melimpahkan rahmat, kesehatan dan kekuatan sehingga dapat melaksanakan Tugas Akhir dan menyelesaikan penyusunan Laporan tanpa halangan apapun.
2. Kedua Orang tua, bapak Suhardin dan ibunda Juraidah. Terima kasih telah memberikan semua hal baik dari materi, kasih sayang, perhatian dan doa yang tak pernah putus kepada penulis.
3. Nenek penulis Hj. Siti Maani yang selalu memberikan semangat, perhatian, dan doa-doan yang tak pernah putus kepada penulis.
4. Bapak Dr. Syarif Hidayat, S.Kom., MIT, selaku Dosen Pembimbing Tugas Akhir ini yang selalu membantu serta memberikan arahan sehingga pelaksanaan TA dapat berjalan sesuai prosedur yang sudah ada.
5. Bapak Hendrik, S.T., M.Eng, selaku Ketua Jurusan Informatika Fakultas Teknologi Industri Universitas Islam Indonesia Yogyakarta.
6. Bapak Dr. Raden Teduh Dirgahayu, ST., M,Sc, selaku Ketua Program Studi Informatika – Program Sarjana Fakultas Teknologi Industri Universitas Islam Indonesia Yogyakarta.
7. Seluruh Dosen Informatika UII yang telah memberikan ilmu yang sangat bermanfaat.
8. Semua sahabat-sahabat penulis yang selalu setia menemani di saat kebuntuan dalam mengerjakan tugas akhir dan selalu memberikan energi-energi positif untuk selalu bersemangat.

Penulis menyadari terdapat banyak kekurangan dalam penyampaian dan penyusunan laporan Tugas Akhir (TA) ini. Oleh karena itu, saya membuka diri atas segala kritik, saran, dan

masukan yang bersifat membangun demi terciptanya kesempurnaan dalam penyusunan laporan ini.

Akhir kata semoga laporan ini dapat memberikan manfaat bagi semua pihak. Aamiin. Wassalaamu'alaikum Wr. Wb.

Yogyakarta, 1 Maret 2020

(Washayatul Iman)

SARI

Keamanan jaringan sangat berpengaruh terhadap ketersediaan sumber daya dalam jaringan yang selalu mengarah ke privasi pengguna jaringan. Perkembangan teknologi jaringan khususnya dalam penggunaan jaringan WiFi (*Wireless*) memiliki beberapa ancaman bagi pengguna awam, salah satunya adalah serangan *Rogue Access Point*.

Rogue Access Point (RAP) adalah *access point* (AP) ilegal yang tidak dilindungi dan tidak dibuat oleh administrator jaringan. Serangan RAP ini sendiri dikatakan berbahaya, karena ketika seorang korban sudah terjebak pada serangan RAP ini, segala sesuatu yang dilakukan oleh korban tersebut dalam mengakses sesuatu pada internet akan diketahui atau dimata-matai oleh pelaku penyerangan RAP tersebut. Karena banyaknya kejadian-kejadian yang memanfaatkan ancaman tersebut maka penulis mengembangkan suatu sistem yang dapat mendeteksi ketika suatu jaringan yang dilindungi akan dibuat pemalsunya oleh serangan RAP tersebut.

Dalam hal pendeteksian serangan RAP tersebut, sistem yang dibangun melakukan pengecekan secara terus menerus secara *realtime* berdasarkan *ssid, mac address, channel frequency* dan kualitas jaringan nirkabel terhadap serangan RAP yang ada di sekitar dari jaringan yang dilindungi dengan menggunakan perintah-perintah dalam mengontrol perangkat yang digunakan, yaitu Wemos D1 ESP8266 dan mengirimkan hasil pendeteksian tersebut berdasarkan kondisi-kondisi atau parameter ke web server *thinkspeak.com* untuk melihat hasil pendeteksian.

Dari hasil perancangan dan pengujian sistem yang dibangun ini. Memiliki 2 web server, web server lokal dan web server *thinkspeak*. Web server lokal ini digunakan untuk mendaftarkan perangkat mana yang akan dilindungi oleh sistem yang dibangun dengan ditandai lampu indikator pada modul wifi perangkat yang digunakan akan menyala dan Web server *thinkspeak* yang akan menampilkan hasil pendeteksian dari sistem yang dibangun.

Hasil pengujian yang dilakukan, menggunakan metode simulasi penyarangan RAP menggunakan sistem operasi Kali Linux dengan tools WiFi-Pumpkin, ketika menggunakan tools ini dapat membuat suatu penyerangan RAP dengan SSID dan MAC Address yang dapat dibuat dengan bebas oleh orang yang menggunakannya. Ketika dilakukan pengujian tersebut sistem yang dibangun mampu mendeteksi serta mengelompokan tingkat keseriusan serangan RAP yang dilakukan.

Kata kunci: keamanan jaringan, WLAN, pendeteksi, RAP, *realtime,* sistem.

GLOSARIUM

Access Point	sebuah perangkat jaringan yang berisi sebuah pemancar jaringan dan antena untuk transmisi dan menerima sinyal ke dan dari perangkat pengguna.
IDS	Intrusion Detection System adalah sebuah metode yang dapat digunakan untuk mendeteksi aktivitas yang mencurigakan dalam sebuah sistem atau jaringan.
IoT	Internet of Things, sebuah konsep di mana suatu objek yang memiliki kemampuan untuk mentransfer data melalui jaringan internet tanpa melakukan interaksi manusia ke manusia atau manusia ke komputer.
Jaringan	Jaringan telekomunikasi yang memungkinkan antar komputer untuk saling berkomunikasi dengan bertukar data.
MITM	*Man in The Middle* merupakan satu teknik peretasan di mana penyerang menempatkan dirinya berada di tengah-tengah dua perangkat yang saling berkomunikasi.
RAP	*Rogue Access Point, access point* (AP) illegal yang tidak terotentifikasi dan tidak dibuat oleh administrator WLAN.
WEP	*Wired Equivalent Privacy*, keamanan nirkabel pertama dan dianggap paling tidak aman dari semua protokol.
WLAN	Suatu jenis jaringan komputer yang menggunakan gelombang radio sebagai alat atau media transmisi data.
WPA	*Wi-Fi Protected Access*, penerus WEP akibat kelemahan dari sistem tersebut. WPA didesain dengan komponen enkripsi *Temporal Key Integrity Protocol* (TKIP) yang kemudian digantikan oleh *Advanced Encryption Standard* (AES).
WPA2	Penerus untuk menggantikan WPA. Salah satu perubahan paling signifikan antara WPA dan WPA2 adalah penggunaan algoritma AES dan CCMP (*Counter Cipher Mode with Block Chaining Message Authentication Code Protocol*) sebagai pengganti TKIP.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Seiring dengan pesatnya perkembangan teknologi jaringan nirkabel atau WLAN (*Wireless Local Area Network*) sehingga membuat penggunaan jaringan yang dikenal sebagai wifi ini sangat banyak digunakan karena mobilitas serta skalabilitasnya yang cukup tinggi. Banyaknya pengguna teknologi ini, membuat perusahaan-perusahaan banyak yang mengembangkan teknologi tersebut, sehingga membuat banyak pelaku *cyber crime* tertarik untuk memanfaatkan kelemahan dari jaringan nirkabel ini. Salah satunya serangan *Rogue Access Point*.

Rogue Access Point (RAP) adalah *access point* (AP) ilegal yang tidak terotentifikasi dan tidak dibuat oleh administrator WLAN (Han, Sheng, Tan, Li, & Lu, 2018). RAP seringkali dikonfigurasi agar menyerupai AP yang sah dengan tujuan mengelabuhi pengguna jaringan nirkabel agar menyambungkan perangkatnya pada RAP, sehingga masuk ke dalam kelompok *cyber crime*. RAP ini biasanya dikonfigurasi dengan SSID yang dibuat mirip atau bahkan sama dengan AP yang asli (SSID dan MAC Address). Kebanyakan pengguna awam mengira kedua AP dengan SSID yang sama merupakan AP yang sah (Syahrulah, Bhawiyuga, & Data, 2018). Pengguna jaringan wifi yang terjebak pada RAP akan dengan mudah di mata-matai informasi pribadinya atau semua aktivitas pengguna ketika menggunakan internet akan diketahui oleh pembuat serangan RAP tersebut, sehingga menyebabkan data milik pengguna bocor dan dapat dimanfaatkan oleh pelaku tersebut.

Pada tahun 2016, *cyber crime* di Indonesia menjadi yang tertinggi kedua di dunia setelah Jepang. Total serangan *cyber* ini sampai mendekati angka 90 juta serangan. Angka tersebut merupakan serangan *cyber* yang tidak sedikit dilakukan oleh sesama warga negara Indonesia, yang tidak sedikit juga termasuk penyerangan berupa RAP (Syafruddin, 2016). Dalam beberapa teknik peretasan yang ada, serangan RAP ini termasuk ke dalam teknik *social engineering*, dengan memanfaatkan titik lemah target untuk mendapatkan data-data sensitif yang kemudian akan digunakan sebagai langkah awal melakukan kejahatan yang lebih besar lagi.

Akibat banyaknya serangan RAP yang terjadi, sudah banyak yang mencoba mengantisipasi permasalahan tersebut, salah satunya dengan membuat suatu sistem pendeteksi RAP menggunakan metode perhitungan nilai *round trip time*. *Round Trip Time* (RTT) didefinisikan

sebagai waktu yang dibutuhkan ketika pengguna mengirim sebuah perintah hingga server memberikan jawaban kembali ke penggunanya. RAP akan menghasilkan nilai RTT yang lebih besar dari pada AP sebenarnya, akibat dari pertambahan jumlah lompatan yang dilaluinya (Syahrulah et al., 2018). Dengan metode tersebut akan membuat hasil pendeteksiannya kurang tepat jika AP yang sebenarnya memiliki kondisi kecepatan jaringan yang lambat. Oleh karena itu dibutuhkan penelitian mengenai sistem deteksi serangan RAP, untuk meminimalisir kesalahan pendeteksian serta mengurangi korban dari serangan RAP.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, permasalahan yang diangkat dalam penelitian ini adalah karena masih kurangnya ketepatan pendeteksian sistem RAP yang ada saat ini.

1.3 Batasan Masalah

Batasan masalah merupakan batas-batas kegunaan sistem yang dimiliki oleh sistem deteksi serangan AP palsu yang dirancang, antara lain sebagai berikut:

a. Dalam masalah tersebut penelitian ini mendeteksi ketika terjadinya serangan AP Rogue dan tidak mencegah secara langsung.
b. Sistem deteksi serangan RAP ini tidak dapat melindungi AP yang menggunakan keamanan selain WPA2-Enterprise, karena jenis keamanan ini menggunakan SSID yang sama dan MAC Address yang berbeda-beda pada suatu tempat yang jaraknya berdekatan, sedangkan sistem yang dibangun ini hanya melindungi satu AP pada satu tempat.
c. Jaringan nirkabel yang akan dilindungi harus mempunyai koneksi atau terhubung ke internet.
d. Sistem atau perangkat keras pendeteksi serangan RAP harus diletakkan bersampingan dengan AP yang akan dilindungi, karena salah satu pendeteksiannya menggunakan kekuatan sinyal.
e. Perangkat yang digunakan adalah Mikrokontroller Wemos D1 dengan menggunakan modul WiFi ESP8266

1.4 Tujuan Penelitian

Berdasarkan rumusan masalah yang telah dibuat sebelumnya peneliti memiliki tujuan melakukan penelitian ini untuk meningkatkan keamanan jaringan khususnya pada teknologi WiFi dengan cara mendeteksi serangan RAP pada jaringan WiFi yang akan dilindungi.

1.5 Metodologi Penelitian

Metodologi peneltian yang akan digunakan dalam menyusun tugas akhir ini meliputi beberapa tahap sebagai berikut:

a. Perancangan Wemos D1 ESP8266

Perancangan Wemos D1 ESP8266 di sini adalah mempersiapkan suatu modul wifi dengan code-code bahasa pemrograman agar bisa berjalan seperti yang sudah direncanakan pada flowchart yang dibuat.

b. Perancangan Web Server Lokal

Perancangan Web Server di sini adalah persiapan dalam membangun suatu webserver lokal yang akan digunakan untuk mengkonfigurasi IDS atau sistem pendeteksi RAP agar melindungi AP sesuai yang dikonfigurasi pada webserver lokal. Kemudian akan diimpor ke dalam modul wifi yang ada, yaitu ESP8266 menggunakan metode SPIFFS yang sudah disediakan oleh developer ESP8266. Selain itu dapat dilakukan *scanning* terhadap AP apa saja yang ada di sekitar perangkat IDS.

c. Registrasi Server IoT

Mendaftar Server IoT yang nantinya akan digunakan untuk memberikan notifikasi pada administrator jaringan yang dikirim dari hasil pendeteksian Modul ESP8266.

d. Konfigurasi IDS atau Sistem Pendeteksi RAP

Konfigurasi IDS ini nantinya digunakan untuk mendaftarkan bagaimana Spesifikasi dari AP yang akan dilindingi (SSID, MAC, *channel frequency*, kualitas sinyal) dan menyambungkan modul ESP8266 ke AP yang tersambung ke internet yang nantinya akan digunakan sebagai langkah untuk mengirim notifikasi ke server IoT.

e. Implementasi dan Pengujian Sistem

Bagian pengujian ini adalah tahap implementasi sistem dan pengujian pada AP yang akan dilindungi dengan membuat simulasi penyerangan AP Rogue terhadap jaringan nirkabel yang ada, kemudian IDS akan beroperasi dengan mengirimkan pesan pemberitahuan ke administrator jaringan bahwa pada waktu tersebut sedang dilakukan penyerangan terhadap jaringan wifi yang dilindungi.

1.6 Sistematika Penulisan

Untuk mempermudah pembacaan serta dapat memberikan gambaran secara menyeluruh terhadap masalah yang akan dibahas, berikut sistematika penulisan laporan tugas akhir ini dibagi dalam lima bab. Sistematika penulisan laporan tugas akhir ini adalah sebagai berikut:

a. BAB I PENDAHULUAN

Pendahuluan membahas permasalahan umum tentang latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian serta sistematika penulisan penelitian sistem deteksi serangan RAP.

b. BAB II LANDASAN TEORI

Bagian ini berisi tentang landasan teori yang digunakan dalam pembuatan sistem deteksi serangan RAP Menggunakan Mikrokontroler Wemos D1 ESP8266 Berbasis Internet of Things. Landasan teori yang digunakan adalah bagaimana suatu sistem dapat mendeteksi serangan RAP menggunakan RTT *(round trip time)* dan menggunakan metode *block/un-block* dengan mengenali *mac address* perangkat yang tersambung pada jaringan yang dilindungi.

c. BAB III METODOLOGI

Metodologi memuat uraian tentang perancangan sistem dan kebutuhan perangkat keras serta kebutuhan masukan, keluaran dalam pembuatan sistem deteksi serangan RAP menggunakan Mikrokontroler Wemos D1 ESP8266 berbasis *Internet of Things.*

d. BAB IV IMPLEMENTASI DAN PENGUJIAN

Bagian ini merupakan bagian yang berisi pembahasan mengenai hasil dan implementasi pembuatan sistem deteksi serangan RAP Menggunakan Mikrokontroler Wemos D1 ESP8266 Berbasis Internet of Things.

e. BAB V KESIMPULAN DAN SARAN

Dalam bab ini memuat kesimpulan dari penelitian sistem deteksi serangan RAP menggunakan Mikrokontroler Wemos D1 ESP8266 berbasis *Internet of Things* dan merupakan rangkuman dari analisis kinerja yang akan menemukan beberapa saran untuk dilaksanakan lebih lanjut untuk pengembangan selanjutnya dari penelitian ini

BAB II
LANDASAN TEORI

2.1 Tinjuan Pustaka

Berikut adalah penelitian sebelumnya yang berkaitan dengan sistem deteksi serangan AP palsu (RAP) yang berhasil dirangkum:

a. Penelititian yang berjudul "Pengembangan dan Uji Kinerja Sistem Pendeteksi *Rogue Access Point* Menggunakan Aplikasi Berbasis Web dengan Metoda Pengukuran Waktu *Round Trip Time*". Membahas tentang implementasi suatu sistem untuk membantu pengguna terutama pengguna awam dalam melakukan pendeteksian *Rogue Access Point* menggunakan aplikasi berbasis web berdasarkan metoda pengukuran waktu pengiriman paket data atau *Round Trip Time* (RTT), sehingga keamanan informasi dan privasi dari user dapat terjaga, juga terdapat suatu penelitian yang hampir mirip dengan penelitian tersebut, yaitu "Implementasi Sistem Pendeteksi *Rogue Access Point* Dengan Metode Perhitungan Nilai RTT" (Syahrulah, Bhawiyuga, & Data, 2018). Dalam kedua penelitian tersebut terdapat potensi kesalahan dalam pendeteksian RAP. Jika kondisi jaringan sibuk atau traffiknya tinggi maka RTT juga akan tinggi sehingga akan mempengaruhi hasil dari pendeteksiannya, oleh karena itu dibutuhkannya penelitian untuk meningkatkan validnya informasi yang diberikan oleh sistem dalam pendeteksian RAP.

b. Penelitian yang berjudul "Implementasi Sistem Penanganan *Rogue Access Point* Model Bridging Connection Pada Fakultas Ilmu Terapan Universitas Telkom" (Bonardo, n.d.). Membahas tentang implementasi suatu sistem dalam mendeteksi serta mencegah serangan RAP model *bridging connection* pada fakultas Ilmu Terapan Universitas Telkom Bandung, maksudnya adalah jika ada serangan RAP yang menggunakan wifi milik Telkom Bandung maka MAC Address dari perangkat tersebut akan *di-block* sehingga tidak dapat tersambung lagi ke dalam jaringan yang dimiliki tersebut. Jika ada pengguna jaringan yang ingin membuat *hotspot* dari jaringan yang dimiliki maka harus mendaftarkan terlebih dahulu kepada MAC address perangkat yang akan dibuat hotspot ke administrator jaringan agar bisa memberikan hotspot kepada perangkat lainnya. Metode yang digunakan tersebut sudah cukup baik, akan tetapi ketika dilakukan serangan RAP yang menggunakan jaringan sendiri maka pada area tersebut tidak akan dapat dilakukan keamanan pada pengguna-pengguna yang ada di sekitar. Oleh karena itu dibutuhkan

penelitian untuk meningkatkan keamanan dari jaringan yang dimiliki terhadap serangan RAP.

Berikut tabel ringkasan dari tinjauan pustaka penelitian yang dirangkum berdasarkan poin-poin yang dijelaskan untuk mempermudah memahami pembahasan dari penelitian-penelitian sebelumnya yang dijadikan landasan teori dari penelitian yang dilakukan, ditunjukan pada Tabel 2.1.

Tabel 2.1. Ringkasan landasan teori

Landasan Teori	**Point-point**	
	1	**2**
Pengembangan Dan Uji Kinerja Sistem Pendeteksi *Rogue Access Point* Menggunakan Aplikasi Berbasis Web Dengan Metoda Pengukuran Waktu *Round Trip Time*.	Mendeteksi serangan RAP dengan menggunakan parameter round trip time.	Tingkat akurasi validnya pendeteksian kurang, karena menggunakan parameter pendeteksi berdasarkan round trip time yang akan berubah-ubah berdasarkan kecepatan jaringan.
Implementasi Sistem Penanganan *Rogue Access Point* Model *Bridging Connection* Pada Fakultas Ilmu Terapan Universitas Telkom.	Mendeteksi serangan RAP dengan mendeteksi ketika terdapat jaringan nirkabel yang menggunakan *resource* atau *bandwidth* dari jaringan yang dilindungi maka *mac address* dari perangkat yang membuat hotspot atau AP palsu akan di-*block*.	Kurangnya efektivitas karena ketika pengguna ingin membagikan jaringan (*hotspot*) harus mendaftarkan terlebih dahulu ke administrator jaringan agar tidak ter-*block* atau disangka serangan RAP.

Penelitian-penelitian di atas memiliki perbedaan dengan sistem yang akan dibangun, karena sistem ini akan melindungi AP dengan cara mendaftarkan beberapa paremeter, yaitu ssid, *mac address*, channel frekuensi dan kualitas sinyal. Setelah berhasil mengenali AP yang dilindung maka perangkat wemos d1 akan melakukan pemindaian secara terus menerus untuk mengenali AP yang ada di sekitar, oleh karena itu perlindungan terhadap serangan RAP akan lebih baik lagi.

2.2 Dasar Teori

Dasar teori dalam sistem pendeteksi access poin palsu ini ini adalah teori yang menjadi acuan dalam penelitian yang dilakukan. Seperti yang disebutkan dalam penjelasan di bawah ini:

2.2.1 Access Point

Access Point (AP) adalah sebuah perangkat jaringan yang berisi sebuah pemancar sinyal dan antena untuk meneruskan dan menerima sinyal ke dan dari perangkat pengguna. Dengan AP, pengguna jaringan nirkabel bisa dengan cepat dan mudah untuk terhubung pada jaringan nirkabel atau bisa dibilang sebuah alat yang digunakan untuk menghubungkan alat-alat dalam suatu jaringan, dari dan ke jaringan nirkabel (Firdana, Munadi, & Widodo, 2012).

Wireless Lokal Area Network (WLAN) menggunakan standardisasi IEEE 802.11 AP yang dikonfigurasikan untuk berkomunikasi antara perangkat komputer pada saluran *Radio Frequency* (RF) yang bekerja untuk menyediakan stasiun Wi-Fi (STA) dengan cakupan sinyal yang memadai dan konektivitas yang efisien (Raschellà et al., 2020). Titik akses yang merupakan dasar dari transiver radio dua arah yang tipikalnya bekerja di bandwith 2,4 GHz (802.11b, 802.11g) atau 5 GHz (802.11a). Kebanyakan peralatan mempunyai kualifikasi Wi-Fi, IEEE 802.11b atau akomodasi IEEE 802.11g dan menawarkan beberapa level keamanan seperti WEP,WPA dan WPA2 (Oktavianti, 2019).

2.2.2 Wireless Security

Wireless Lokal Area Network (WLAN) sedang berkembang dan banyak digunakan oleh pengguna IT. WLAN diinstalasi oleh beberapa perusahaan dan institusi dari berbagai bidang, seperti bidang pendidikan, pemerintahan, perdagangan dan lain-lain. Alasan WLAN menyediakan pengguna untuk dapat akses internet/mendapatkan informasi di lokasi manapun (mobilitas tinggi). Beberapa perusahaan dan institusi lebih banyak memilih dan menggunakan WLAN. WLAN terbatas oleh geografis atau batasan cakupan luas. WLAN menyediakan fleksibilitas untuk para pengguna termasuk untuk kolaborasi yang dapat diakomodasi oleh wireless yang dapat mencakup pengguna yang lebih banyak. Di samping itu, dari kelebihan dari WLAN, ada ancaman yang mengakibatkan WLAN tidak fungsional, tidak terlihat, dan tidak terlindungi dengan baik. Beberapa ancaman keamanan WLAN (*security attack*) sudah banyak dilakukan dari waktu ke waktu, di mana berbanding lurus dengan evolusi dari keamanan jaringan (s*ecurity network*).

Jaringan Wi-Fi sangat rentan terhadap risiko yang muncul dari kemungkinan penyadapan sinyal. Siapa pun yang berada dalam jangkauan titik akses memiliki kemungkinan untuk melakukan penyadapan. Tentu saja, penggunaan enkripsi tipe WEP, WPA dan WPA2 secara signifikan menghambat kemampuan untuk mengambil alih suatu paket, beberapa informasi yang dilakukan oleh pengguna jaringan akan termuat semua ke dalam *traffic-traffic* jaringan. Bagian penting dalam hal ini adalah kesadaran akan ancaman saat menggunakan Wi-Fi. Hal ini diperlukan untuk menghindari hilangnya privasi akibat aktivitas seperti *Evil Twin* atau *Man in the Middle* atau serangan RAP. Ancaman kedua yang sering diidentifikasi dengan kebocoran privasi oleh pengguna Wi-Fi adalah berdasarkan area layanan berbasis lokasi (Aarthy Devi, Mohan, & Sethumadhavan, 2017).

Ancaman terhadap jaringan Wi-Fi sampai batas tertentu sama dengan di jaringan lain. Sumber utama ancaman adalah tiga elemen: Internet, email, dan kemampuan untuk mencegat sesi komunikasi. Yang membedakan jaringan nirkabel dari kabel adalah kemungkinan intersepsi dan interferensi oleh pihak ketiga ke dalam sesi transmisi. Setiap orang harus memahami risiko sehingga akan mungkin untuk menghindari atau meminimalkan efek dari keadaan darurat atau akibatnya. Resiko umum untuk Wi-Fi atau WLAN dikaitkan dengan kemungkinan intersepsi pada sesi komunikasi dan paket yang di ambil (Masiukiewicz, Tarykin, & Podvornyi, 2016).

2.2.3 *Rogue Access Point*

Keamanan jaringan menjadi bahan pembicaraan yang cukup banyak dalam perkembangan teknologi jaringan. Satu dari banyaknya tantangan terhadap keamanan jaringan dalam IT saat ini ialah RAP. Sebagai standardisasi yang cukup banyak dan berkembang secara cepat, saat ini teknologi yang digunakan untuk standardisasi ialah 802.11 yang telah menjadi lebih popular, lebih murah dan mudah digunakan oleh user atau instalasi, di samping itu ancaman terhadap jaringan di perusahaan-perusahaan juga ikut meningkat (Nugroho, 2013).

RAP terkadang didefinisikan sangat sederhana, yaitu *access point* (AP) yang diletakkan secara bebas dan tidak terotorisasi dan menggunakan SSID yang sama atau mirip, padahal tidak sesederhana itu dalam kenyataannya. RAP dapat lebih kompleks dari itu dan dengan metode-metode yang baru dan sulit. RAP adalah AP yang terhubung ke dalam jaringan tanpa otorisasi (*Authorization*). AP itu di luar dari manajemen jaringan administrator dan kebijakan keamanan jaringan. Sebuah RAP mengizinkan siapapun pengguna yang memiliki teknologi nirkabel

untuk terhubung ke dalam suatu jaringan. IT *Resources* menjadi hal yang sangat rentan untuk dilakukan manipulasi termasuk kejahatan dalam jaringan (Proxim, 2004).

2.2.4 Tipe *Rogue Access Point*

Rogue Access Point merupakan salah satu ancaman terbesar dalam keamanan jaringan khususnya pada WLAN. Bahkan, dalam beberapa waktu hampir 20% dari perusahaan telah terinfeksi Rogue AP. Sama halnya, konfigurasi AP yang tidak benar dan *phising* AP dan dideteksi seperti ancaman keamanan yang sama sekali dimanipulasi oleh musuh. Bagaimanapun, hal seperti itu juga dapat dianggap sebagai Rogue AP, dan lebih pentingnya lagi sebenarnya ada beberapa tipe RAP yang disebut *compromised* AP, yang mana tipe itu yang paling bahaya karena dapat selalu ikut serta dalam komoditas jaringan nirkabel, karena tipe compromised AP ini ialah tipe perusak, atau punya niat yang tidak baik, sehingga akan selalu dilakukan cara apapun untuk merusak. Untuk lebih detail klasifikasi rogue AP dapat dilihat pada Tabel 2.2 berikut.

Tabel 2.2. Klasifikasi *Rogue Access Point*

Rogue AP Class	Kemungkinan cara yang dilakukan (Possible Scenarios)
Improperly Configured	Tidak cukup menguasai keamanan jaringan, kerusakan pada *driver*, cacat fisik perangkat, adanya multi NIC CARD.
Unauthorized	Penyerang terhubung ke jaringan internal dan membuat AP dengan SSID yang sama sehingga *traffic* yang ada dapat dimata-matai.
Phising	Pengelabuhan oleh penyerang atau membuat sesuatu yang secara fisik sama, namun isinya berbeda dengan memberikan AP tanpa keamanan.
Compromised	Keamanan jaringan yang dilakukan oleh penyerang sama agar target menghubungkan menggunakan ssid dan *password* yang sama.

Untuk lebih jelasnya lagi berikut penjelasan lebih mendetail dari klasifikasi RAP (Agrawal & Tapaswi, 2015):

a. *Improperly Configured*, kesalahan konfigurasi kecil, titik akses yang sah dapat tiba-tiba berubah menjadi perangkat yang tidak terkonfigurasi dengan benar. Alasan di balik jenis AP ini adalah: administrator jaringan dengan pengetahuan keamanan yang tidak memadai (misalnya, memilih teknik enkripsi dan otentikasi yang tidak tepat), penggunaan *driver* AP yang salah, dan kadang-kadang setelah pembaruan perangkat lunak, AP yang terkonfigurasi dengan baik menjadi rentan.
b. *Unauthorized*, untuk fleksibilitas dan skalabilitas, pengguna memasang AP tanpa sepengetahuan administrator. Sehingga penyerang dapat terhubung ke jaringan internal, mencuri data sensitif, mencuri bandwidth dan menggunakan jaringan untuk menyerang pengguna lain. Membuat AP yang tidak sah di sekitar WLAN yang sah. Karena pengguna selalu lebih suka untuk terhubung dengan AP yang memiliki kekuatan sinyal tinggi, terkadang pengguna dari satu organisasi dapat terhubung dengan AP di sekitarnya dengan sinyal yang tinggi.
c. *Phising*, di luar jaringan nirkabel, jika pengguna jahat memasang AP untuk memperoleh kredensial pengguna seperti nama pengguna dan kata sandi dengan menyamar sebagai pengguna yang sah, disebut sebagai phishing AP. Ini memungkinkan penyerang untuk melakukan serangan *Man in the Middle* (MITM) di jaringan nirkabel yang tidak memerlukan otentikasi bersama antara pengguna ke server dan server ke pengguna, karena jaringan yang diaktifkan WEP tidak menerapkan otentikasi timbal balik, sehingga mudah untuk meluncurkan serangan MITM di jaringan tersebut.
d. *Compromised*, jika penyerang memecahkan kunci yang sedang digunakan dalam jaringan berkemampuan bersama WEP dan WPA (WPA-PSK), maka itu akan dikompromikan. Setelah penyerang menemukan kunci, semua AP yang menggunakan kredensial yang sama akan menjadi AP jahat. Sangat mudah bagi penyerang untuk menyamar sebagai pengguna yang sah dalam jaringan yang disusupi.

2.2.5 Internet of Things

Internet of Things (IoT) adalah suatu konsep yang bertujuan untuk memanfaatkan teknologi internet yang terus berkembang agar dapat implementasi ke dalam benda fisik sehingga manusia dapat berinteraksi langsung dengan benda tersebut seperti mengirim data dan melakukan kendali jarak jauh secara *realtime*. Makna lain serupa, IoT adalah sebuah

konsep di mana suatu objek yang memiliki kemampuan untuk mentransfer data melalui jaringan internet tanpa melakukan interaksi manusia ke manusia atau manusia ke komputer (Sasmoko & Arie, 2017). Teknologi perangkat keras IoT yang digunakan pada umumnya adalah teknologi *Radio Frequency Identification* (RFID), *Wireless Sensor Network* (WSN), dan nano teknologi. Beberapa teknologi perangkat lunak adalah pemrosesan informasi dan teknologi keamanan. IoT memiliki arsitektur yang terdiri atas *perception layer, network layer,* dan *application layer.*

Perception layer adalah lapisan yang terdiri atas sensor dan perangkat yang digunakan untuk menerima data dari lingkungan yang diubah menjadi bentuk digital dan kemudian akan disalurkan ke *network layer*. Sensor yang dapat digunakan contohnya dapat berupa RFID chip, perangkat yang dapat menerima data dari lingkungan, maupun *gateway* yang diakses oleh suatu perangkat. Kamera pada *smartphone* juga dapat digunakan sebagai sensor. Network layer merupakan lapisan kedua yang berfungsi untuk menghubungkan lapisan sensor dengan lapisan aplikasi. Pada lapisan ini ditentukan informasi yang akan disalurkan pada lapisan aplikasi. Selain itu, pemrosesan data dilakukan pada lapisan ini. Kemampuan jaringan dan bagaimana data dikirim ditentukan pada lapisan ini. Application layer adalah lapisan terakhir pada arsitektur IoT. Lapisan ini merupakan antarmuka yang mudah digunakan oleh pengguna yang terhubung dengan lapisan jaringan. Pengguna dapat berkomunikasi dengan lapisan sensor untuk mendapatkan data sesuai denganyang dibutuhkan.

2.2.6 Monitoring

Monitoring adalah proses untuk mengumpulkan informasi atau data dari beberapa macam sumber daya. Sistem *monitoring* ini biasanya berupa data atau informasi yang akan diambil yaitu data secara *realtime* (Juwariyah, Prayitno, & Mardhiya, 2018). *Monitoring* dilakukan agar dapat menemukan kesalahan secepat mungkin atau pencegahan sehingga mengurangi risiko yang lebih besar. Untuk mendapatkan evaluasi dari tindakan apa yang harus dilakukan bersumber dari hasil informasi *monitoring*. *Monitoring* dibagi menjadi tiga proses yaitu diawali dengan proses mengumpulkan data *monitoring*, setelah itu dilanjutkan pada tahap proses menganalisis data *monitoring* dan proses terakhir adalah proses menampilkan data *monitoring* dapat berupa gambar, tabel, notifikasi dan lain-lain.

2.2.7 Mikrokontroler Wemos D1

Mikrokontroler adalah sebuah chip yang berfungsi sebagai pengontrol rangkaian elektronik. Umumnya mikrokontroler dapat menyimpan dan mengeksekusi perintah berdasarkan program yang diberikan. Wemos D1 adalah mikrokontroler yang sudah dilengkapi dengan modul ESP8266 sebagai salah satu solusi untuk membuat sistem berbasis Internet of Things (IoT). Dengan adanya modul wifi yang tertanam pada Wemos membuat Wemos lebih mudah terhubung dengan perangkat lain dalam implementasi sistem *monitoring* RAP. Terdapat beberapa keunggulan yang dimiliki oleh Wemos yaitu dapat diprogram menggunakan Arduino IDE dengan cara memanfaatkan sintaks program *library* yang sudah banyak terdapat di internet dan *pin out* yang *compatible* dengan Arduino Uno sehingga mudah untuk menghubungkan dengan Arduino *shield* lainnya serta mempunyai *memory* yang sangat besar yaitu 4MB jika dibandingkan dengan Arduino Uno yang memiliki *flash memory* 32KB. Wemos juga sesuai dengan beberapa bahasa pemograman lainnya seperi bahasa Python dan Lua, serta bahasa pemrograman web yang menggunakan *library* SPIFFS sehingga memudahkan untuk mengunggah program ke dalam Wemos apabila seorang *programmer* belum terlalu paham dengan cara program menggunakan Arduino IDE. Untuk memasukkan program ke dalam Wemos dapat menggunakan mikro usb hal itu juga menjadi salah satu kelebihan yang ada pada Wemos karena saat ini mikro usb sangat mudah ditemukan. Sedangkan untuk memberikan sumber daya pada Wemos dapat menggunakan DC dan mikro usb. Dalam Wemos terdapat pin digital dan analog:

a. Pin Analog

   Pin analog pada modul Wemos ini memiliki 10 bit resolusi dengan nilai maksimal 3.2 Volt. Pin Analog ini dapat digunakan persis dengan cara yang sama dengan pin digital.

b. Pin Digital

   Salah satu I/O *port* pada modul Wemos dikenal dengan pin digital. Pin ini dapat dikonfigurasikan baik sebagai *input* ataupun *output*. Untuk lebih jelasnya ada pada Gambar 2.1.

Gambar 2.1. Mikrokontroler Wemos D1

Dari setiap perangkat wemos memiliki bentuk dan spesifikasi yang berbeda-beda. Adapun perangkat yang digunakan, yaitu wemos d1 ESP8266 mempunyai spesifikasi pada Tabel 2.3.

Tabel 2.3. Spesifikasi Wemos D1

Mikrokontroler	ESP-8266EX
Tegangan operasi	3.3V
Pin Digital I/O	11 buah (Di dalamnya terdapat spesial pin untuk fungsi i2c, one-wire, PWM, SPI, interrupt)
Analog Input Pin	1 (Max input: 3.2V)
Clock speed	80MHz/160MHz
Flash memory	4MB
Panjang	68.6mm
Lebar	53.4mm
Berat	25gram
Power	jack
input	9-24 V Koneksi USB Micro USB

BAB III
METODOLOGI

3.1 Analisis Kebutuhan

Analisis kebutuhan merupakan tahapan dimana dilakukan proses pengumpulan data dan informasi yang akan digunakan untuk mendukung atau menunjang pembuatan sistem yang akan dibuat serta dapat memperoleh jawaban dari rumusan masalah yang telah dibentuk sebelumnya.

Metode pengumpulan data yang dilakukan menggunakan metode studi pustaka. Studi pustaka adalah sebuah metode dalam pengumpulan data dengan melakukan pencarian informasi melalui media seperti buku, jurnal atau internet. Studi pustaka dilakukan dengan membaca buku dan mencari literatur dari internet.

Mengenai sistem yang dibangun dalam penelitian ini, membahas tentang suatu sistem yang akan mendeteksi serangan RAP secara *realtime*, mencari informasi mengenai perangkat keras ataupun perangkat lunak yang sesuai agar sistem dapat berjalan dan berfungsi sesuai dengan yang diharapkan. Sistem yang dibangun ini dapat di implementasi menggunakan perangkat mikrokontroler wemos d1 dan ketika sistem tersebut sudah berhasil berjalan maka pengguna dapat melakukan *monitoring* melalui web server *thinkspeak*.

3.1.1 Analisis Kebutuhan Fungsi

Analisis kebutuhan fungsi adalah tahapan di mana dilakukan penetapan fungsi yang dapat dilakukan oleh sistem, sehingga dapat menjawab rumusan masalah yang ada. Sistem ini nantinya akan memiliki fungsi sebagai berikut:

a. Melindungi AP yang akan dilindungi di sekitar perangkat Wemos dan memberikan pesan secara *realtime* melalui server IoT (*thinkspeak*).
b. Meng-*scan* jaringan nirkabel yang ada di sekitar, sehingga dapat mengetahui SSID, MAC Address, Channel, dan kualitas sinyal dari perangkat WLAN.

3.1.2 Analisis Kebutuhan Masukan

Pada tahap analisis kebutuhan masukan ini dilakukan tahapan untuk menentukan masukan apa yang dibutuhkan dalam pembuatan sistem deteksi serangan RAP.

a. Data AP yang akan digunakan sebagai mediasi agar perangkat yang dirancang terhubung ke internet (SSID dan *Password* AP).
b. Data AP yang akan dilindungi dari serangan RAP (ssid, *mac address*, dan *channel frequency* AP yang akan dilindungi).
c. Data authentikasi *login* sebagai pengamanan akses perangkat yang secara *default* diatur, yaitu *username* = admin dan *password* = admin.
d. Data WLAN di sekitar yang didapatkan dari modul WiFi ESP8266 pada Wemos D1.

3.1.3 Analisis Kebutuhan Keluaran

Pada penelitian ini akan dibuat sebuah sistem yang akan memberikan informasi kepada administrator yang mengimplementasikan sistem yang dirancang ini. Kebutuhan keluaran dari sistem yang dibangun ini adalah:

a. Jaringan nirkabel yang dibuat sementara.
b. Web server lokal yang dapat diakses jika sudah terhubung ke jaringan nirkabel yang dibuat sementara untuk administrator jaringan mengkonfigurasi awal perangkat yang dirancang agar dapat berjalan seperti yang direncanakan.
c. Meng-*scan* perangkat-perangkat WLAN di sekitar dan menampilkannya di web server lokal ketika tombol scan ditekan.
d. Mengirimkan pesan ke server IoT yang digunakan.

3.1.4 Analisis Kebutuhan

Perangkat Lunak Penelitian ini menggunakan perangkat lunak dalam melakukan pengerjaan dan penunjang kinerja sistem. Perangkat lunak yang digunakan dalam penelitian ini adalah:

a. Sublime, sebagai media atau *software* yang digunakan untuk merancang web server lokal berdasarkan bahasa pemrograman web.
b. Arduino IDE, *software* yang digunakan untuk merancang proses atau alur kerja sistem yang dibangun dengan menggunakan bahasa pemrograman C.
c. Browser, sebagai media atau perangkat lunak yang digunakan pengguna untuk mengakses web server lokal dan web server *thinkspeak.*

d. Server IoT *thinkspeak*, diakses oleh pengguna atau administrator jaringan untuk melihat hasil pendeteksian sistem yang dibangun.

3.1.5 Analisis Kebutuhan Perangkat Keras

Penelitian ini menggunakan berbagai macam perangkat keras sebagai media yang akan mengendalikan dan mengolah data yang ada. Perangkat keras yang digunakan dalam penelitian ini adalah:

a. Mikrokontroler Wemos D1
b. Kabel Micro & adaptor.

3.2 Metode Perancangan

Metode perancangan merupakan suatu cara untuk menjelaskan perancangan dari sebuah penelitian. Tahap ini merupakan tahapan penting sebelum lanjut ke tahap pengerjaan dalam sebuah penelitian. Pada tahap ini akan didapatkan metode yang akan digunakan dalam penelitian, sehingga bisa mengidentifikasi dan mengevaluasi kendala-kendala dalam pengembangan sistem.

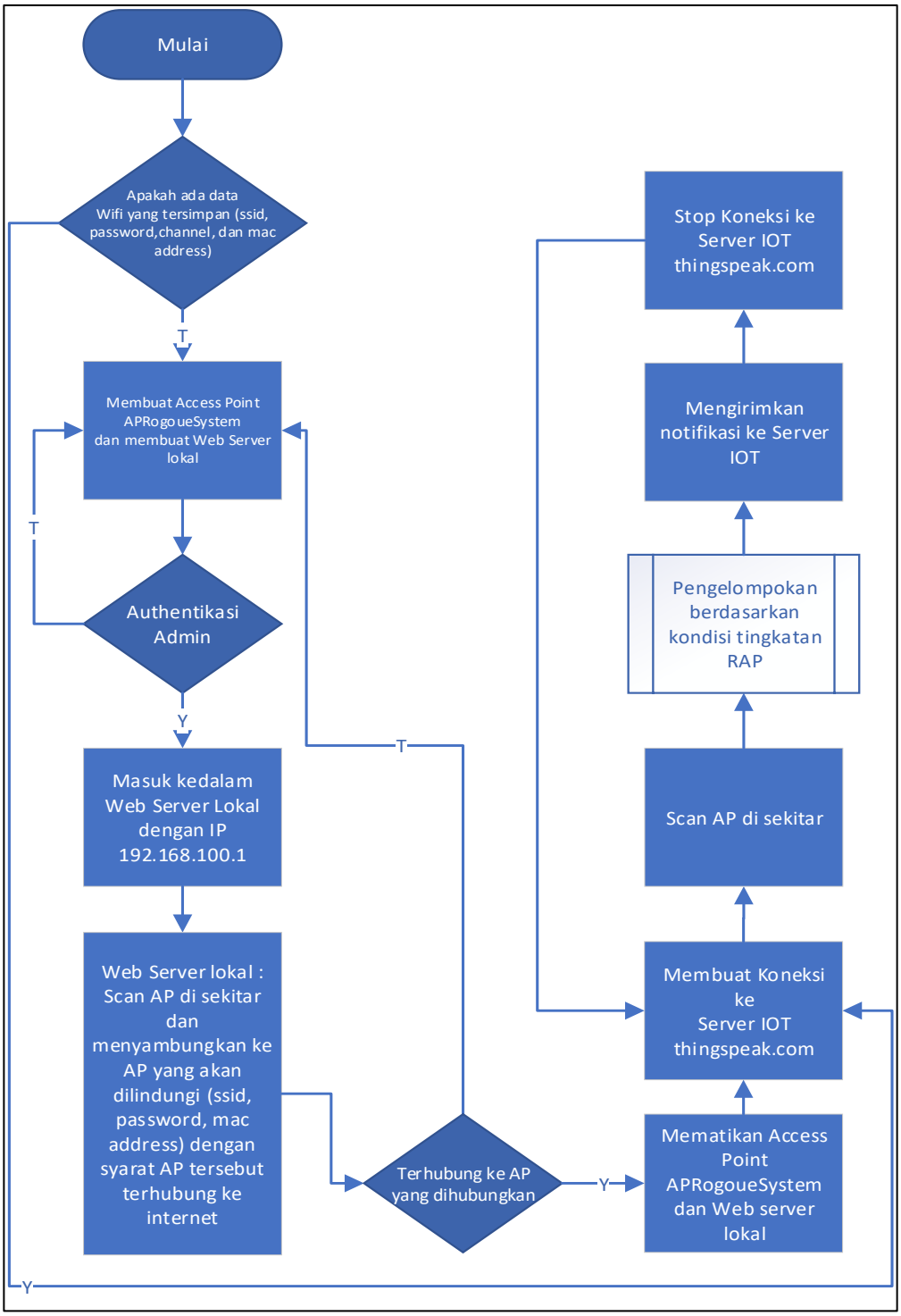
3.2.1 Perancangan *Flowchart*

Flowchart adalah bagan atau gambar yang menunjukkan alur proses dan hubungan dari suatu program. *Flowchart* digunakan untuk mempermudah orang lain dalam memahami alur sistem yang dibuat tersebut. Dalam pembuatan flowchart sistem pendeteksi AP palsu ini menggunakan Arduino terdapat beberapa kondisi guna menentukan tingkat klasifikasi pada serangan RAP. Tabel 3.1 menunjukkan kondisi klasifikasi RAP sebagai tinjauan untuk menentukan notifikasi yang akan dikirimkan ke server IoT yang akan dilihat oleh administrator jaringan. Untuk kondisi lebih jelasnya dapat dilihat pada Tabel 3.1.

Tabel 3.1. Tabel Kondisi pengiriman pesan ke server IoT

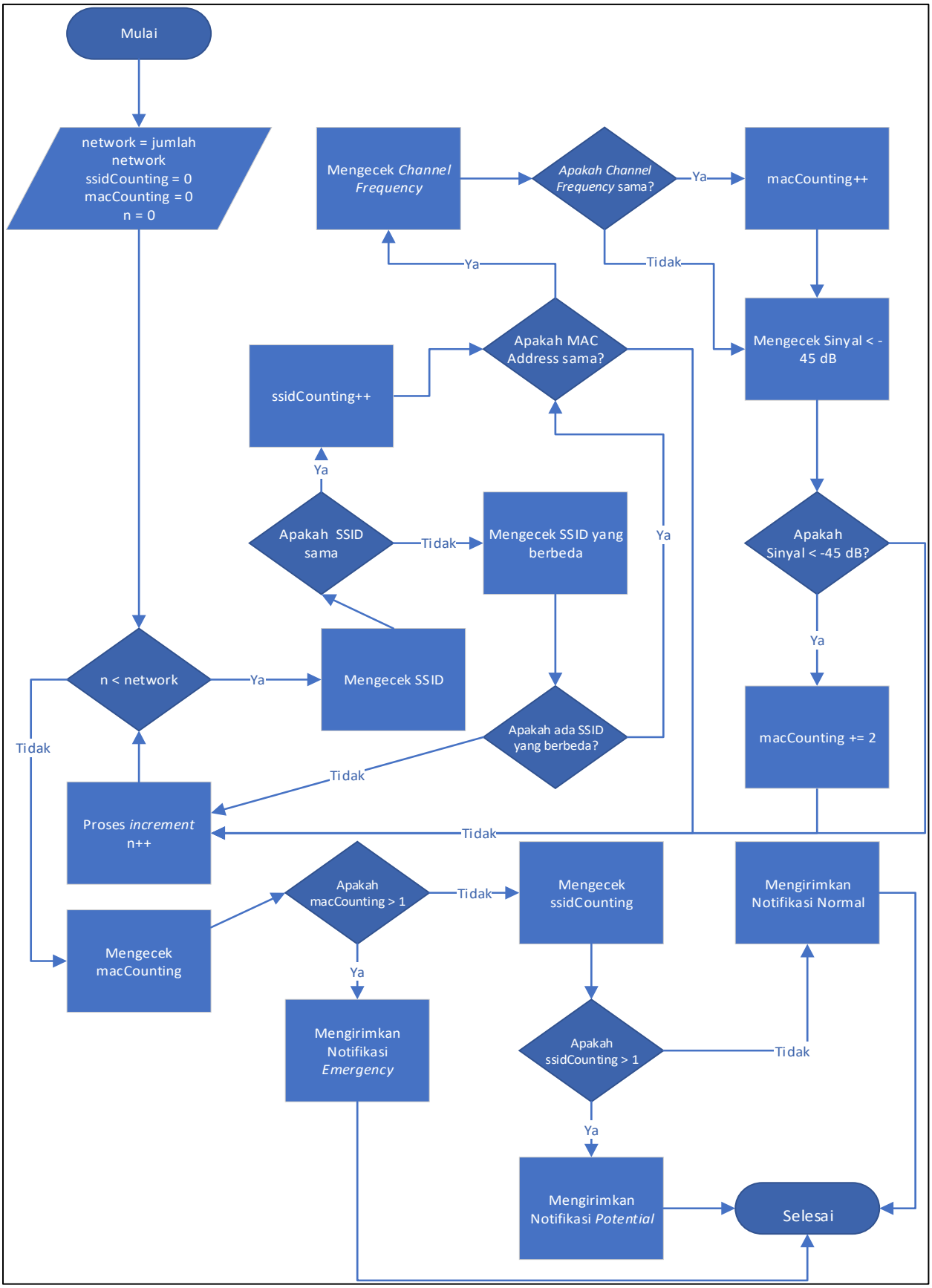
No.	Attribut AP yang di scan				Pesan yang dikirimkan (angka tingkatan berbahaya)	Keterangan Pesan
	SSID	MAC Address	Channel	Sinyal > -45 dB		
1.	-	-	-	-	1	Normal
2.	✔	-	-	-	1	Normal
3.	-	✔	-	-	1	Normal
4.	-	-	✔	-	1	Normal
5.	-	-	-	✔	1	Normal
6.	✔	✔	-	-	1	Normal
7.	✔	-	✔	-	1	Normal
8.	✔	-	-	✔	2	*Potential*
9.	-	✔	✔	-	1	Normal
10.	-	-	✔	✔	1	Normal
11.	-	✔	-	✔	3	*Emergency*
12.	✔	✔	✔	-	1	Normal
13.	-	✔	✔	✔	3	*Emergency*
14.	✔	-	✔	✔	2	*Potential*
15.	✔	✔	-	✔	3	*Emergency*
16.	✔	✔	✔	✔	3	*Emergency*

Adapun alur proses kerja sistem secara keseluruhan yang dapat digambarkan dengan diagram alir (*flowchart)* yang sesuai aturan pembuatannya. Berikut bentuk *flowchart* dapat dilihat pada Gambar 3.1.



Gambar 3.1. *Flowchart* Perancangan Sistem

Dari flowchart di atas terdapat sub proses yang mengenai bagaimana proses pengelompokan berdasarkan kondisi tingkatan pengiriman pesan RAP ke web server *thinkspeak.com*. Untuk lebih jelasnya dapat dilihat pada Gambar 3.2 berikut.



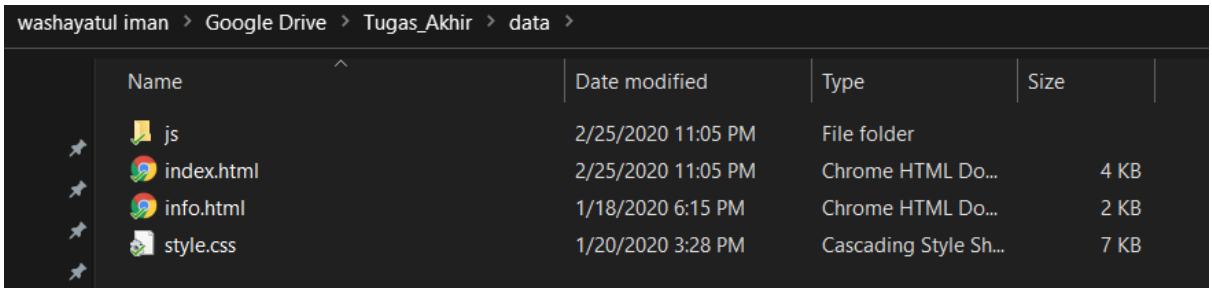
Gambar 3.2. *Flowchart* proses pengelompokan berdasarkan tingkatan RAP

3.2.2 Perancangan Sistem

Sistem pendeteksi RAP pada penelitian ini memiliki web server lokal untuk mengkonfigurasi perangkat WLAN mana yang ingin dilindungi dari serangan RAP dan perangkat jaringan mana yang akan digunakan agar sistem ini dapat terhubung ke internet.

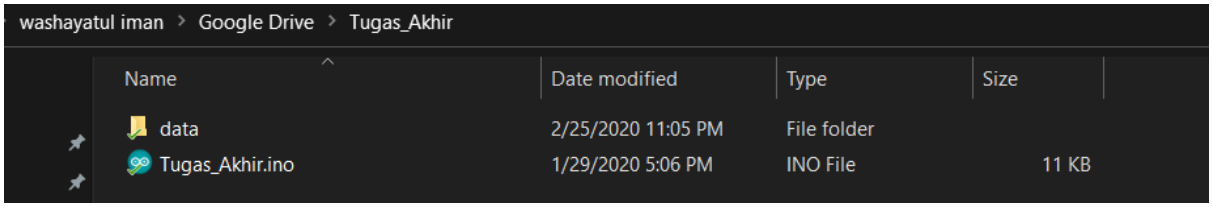
Perangkat ini hanya bisa melindungi satu perangkat WLAN terhadap serangan RAP. Jika sudah pernah dikonfigurasi sebelumnya dan ketika dijalankan lagi, kemudian tidak ditemukan perangkat yang sebelumnya dilindungi, maka akan secara otomatis sistem pendeteksi RAP ini membuat AP sementara untuk administrator konfigurasi kembali atribut-atribut yang akan dibutuhkan untuk melindung AP yang baru terhadap serangan RAP. Jika ditemukan maka akan secara otomatis perangkat jaringan nirkabel tersebut dilindungi kembali dan dapat dilihat di server IoT *thinkspeak* untuk memantau langsung hasil pendeteksiannya dalam bentuk grafik.

Pada perancangan sistem ini dibuat menggunakan bahasa pemrograman C sebagai bahasa yang dikenali oleh mikrokontroller wemos d1 yang digunakan, kemudian untuk web server lokal yang digunakan menggunakan bahasa pemrograman web seperti html, css, php, json, dan javascript. Ketika web server sudah berhasil dirancang sesuai kebutuhan atau sesuai *prototype* yang direncanakan, file tersebut akan dibuat atau disimpan dalam satu folder yang sama dan diberi nama “data” yang nantinya akan diupload menggunakan metode SPIFFS agar bahasa pemrograman web tersebut dapat dikenali oleh perangkat yang sebelumnya hanya mengenali bahasa pemrograman C. Berikut tampilan file-file yang akan di upload menggunakan metode SPIFFS pada Gambar 3.3.



Gambar 3.3. Isi folder data

Kemudian disimpan di dalam satu folder dengan file hasil pemrograman dengan bahasa C. berikut tampilan penyimpanan file tersebut pada Gambar 3.4.



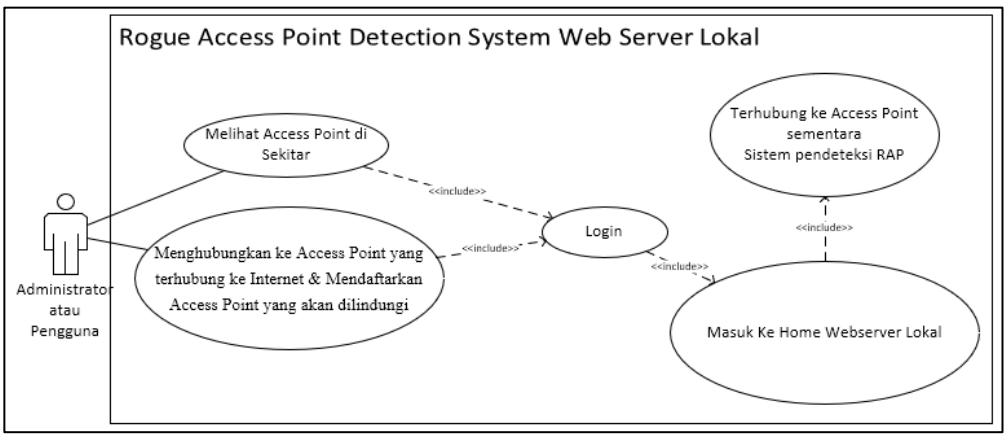
Gambar 3.4. Isi folder sistem yang dibangun

3.2.3 *Use case* Diagram

Use case Diagram adalah sebuah diagram yang menggambarkan dan menjelaskan fungsi-fungsi yang ada dalam suatu sistem. *Use case* Diagram seringkali digunakan untuk menggambarkan kinerja dari sisi pengguna sistem, sehingga pembuatan *Use case* Diagram lebih diarahkan kepada fungsi suatu sistem, bukan berdasarkan alur proses suatu sistem.

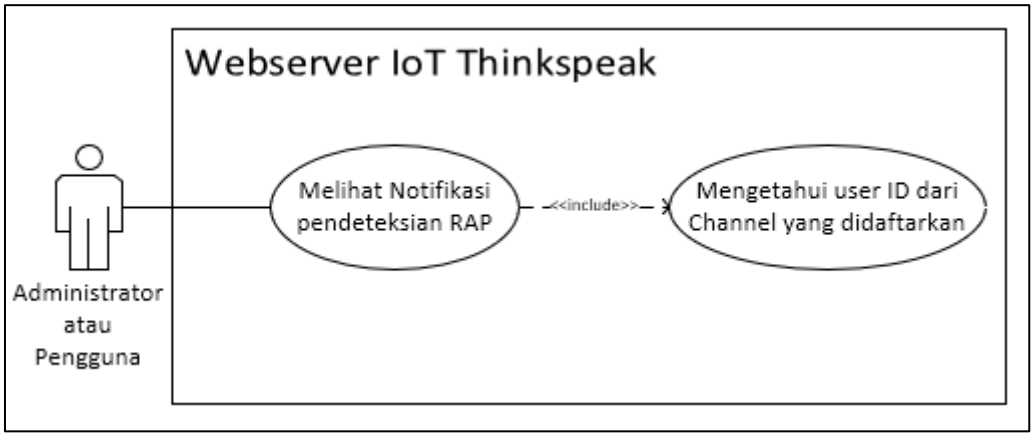
a. *Use case* Diagram Sistem deteksi serangan RAP

Pada sistem deteksi serangan RAP ini hanya terdapat satu aktor yang berinteraksi dengan sistem, yaitu pengguna atau administrator. Pengguna tersebut menggunakan 2 sistem yang berbeda. Sistem pertama digunakan untuk konfigurasi awal penggunaan perangkat pendeteksi serangan RAP, jika sudah mengkonfigurasinya maka otomatis webserver tersebut tidak dapat diakses lagi hingga AP yang dilindungi tidak terdeteksi lagi oleh perangkat yang dirancang atau perangkat tersebut tidak terhubung ke internet untuk mengirimkan notifikasi ke server yang kedua, yaitu salah satu server IoT yang akan digunakan untuk memberikan hasil pendeteksian ke administrator jaringan secara *realtime*. Pengguna dapat mengkonfigurasi perangkat pendeteksi untuk mendeteksi perangkat AP mana yang ingin dilindungi. *Use case* Diagramnya dapat dilihat pada Gambar 3.5.



Gambar 3.5. *Use case* Diagram Webserver Lokal

Setelah sistem sudah berhasil dikonfigurasi dan sudah terhubung ke AP yang dilindung dengan tanda lampu indikator pada modul wifi sudah mati dan jaringan wifi sementara yang dibuat sudah tidak ada lagi maka pengguna atau administrator sudah bisa melihat hasil pendeteksian RAP pada web server IoT *thinkspeak*. Berikut *use case* diagram hasil pendeteksiannya pada Gambar 3.6.



Gambar 3.6. *Use case* Diagram Hasil Pendeteksian

Pada gambar 3.2 dan 3.3 terdapat 1 aktor yaitu pengguna dan 7 *case* dari 2 sistem yang berbeda. Dari gambar di atas maka dapat dilihat interaksi antara aktor (pengguna) dengan case pada sistem deteksi serangan RAP.

b. Identifikasi *Use case* Diagram

Terdapat Tujuh *case* yang telah dipetakan oleh kebutuhan fungsionalitas perangkat lunak. Keterangan *use case* diagram dari sistem web server lokal pada tabel 3.2.

Tabel 3.2. Tabel Identifikasi *Use case* Diagram Web Server Lokal

Kode	Nama & Deskripsi
WL01	Melihat AP di Sekitar **Deskripsi:** Administrator jaringan bisa melihat daftar *access point* (AP) yang ada di sekitar (ssid, *mac address*, channel freq, sinyal)
WL02	Menghubungkan ke AP yang terhubung ke Internet & Mendaftarkan AP yang akan dilindungi **Deskripsi:** Administrator jaringan dapat menghubungkan perangkat Wemos ke AP yang terhubung ke Internet dan akan dilindungi

	terhadap serangan RAP dengan memasukkan ssid, *password* , *mac address*, dan channel.
WL03	*Login* **Deskripsi:** Administrator harus *login* terlebih dahulu jika ingin mengkonfigurasi sistem agar dapat berjalan seperti semestinya.
WL04	Masuk Ke Home Webserver Lokal **Deskripsi:** Administrator harus masuk ke Home dari Webserver lokal untuk *Login*.
WL05	Terhubung ke AP dari Sistem pendeteksi RAP **Deskripsi:** Administrator jaringan jika ingin masuk ke web server lokal dari perangkat pendeteksi RAP harus tersambung terlebih dahulu ke AP sementara dari Sistem pendeteksi RAP.

Fungsionalitas yang terdapat pada web server iot untuk melihat hasil pendeteksian dapat digambarkan dalam *use case* diagram pada Tabel 3.3 berikut.

Tabel 3.3. Tabel Identifikasi *Use case* Diagram Hasil Pendeteksian

Kode	Nama & Deskripsi
HP01	Melihat notifikasi pendeteksian RAP **Deskripsi:** Administrator jaringan bisa melihat pesan atau notifikasi hasil pendeteksian pada salah satu web server IoT, yaitu *thinkspeak.* Pada web server tersebut dapat dilihat hasil pendeteksian dalam bentuk grafik.
HP02	Mengetahui user ID dari Channel perangkat pendeteksi RAP **Deskripsi:** Administrator jaringan dapat melihat channel dengan user ID yang didaftarkan sebelumnya pada web *thinkspeak.*

c. Skenario *Use case* Diagram

1. WL01 Melihat AP di Sekitar

   Aktor: Pengguna

   Prasyarat: *Login* & terhubung ke AP perangkat pendeteksi RAP

   a) Pengguna menekan tombol Scan AP

   b) Sistem menampilkan semua AP yang ada di sekitar

2. WL02 Menghubungkan ke AP yang terhubung ke Internet & Mendaftarkan AP yang akan dilindungi

   Aktor: Pengguna

   Prasyarat: *Login* & terhubung ke AP perangkat pendeteksi RAP

   a) Sistem menampilkan semua AP yang ada di sekitar.

   b) Pengguna memilih AP yang ditampilkan untuk dilindungi dari serangan RAP.

   c) Pengguna memasukkan *Password* dari AP yang akan digunakan untuk terhubung ke internet pada *form* yang ditampilkan.

3. WL03 *Login*

   Aktor: Pengguna

   Prasyarat: Terhubung ke AP perangkat pendeteksi RAP

   a) Sistem menampilkan jendela awal atau *pop-up* yang berisi form untuk *login*.

   b) Pengguna memasukkan *username* dan *password* sesuai yang ditetapkan (*username*: admin & *password*: admin).

4. WL04 Masuk Ke Home Webserver Lokal

   Aktor: Pengguna

   Prasyarat: Terhubung ke AP perangkat pendeteksi RAP

   a) Pengguna memasukkan ip dari perangkat pada browser, yaitu 192.168.100.1

   b) Sistem menampilkan pop-up *login*.

5. WL05 Terhubung ke AP dari Sistem pendeteksi RAP

   Aktor: Pengguna

   Prasyarat: -

   a) Pengguna menghubungkan perangkat komputer ke AP sementara perangkat yang dibangun, dengan SSID APRogueDetectionSystem dan *Password* 12345678.

   b) Sistem memberikan alamat ip serta subnet pada perangkat menggunakan DHCP pada komputer pengguna

6. HP01 Melihat notifikasi pendeteksian RAP

   Aktor: Pengguna

   Prasyarat: Mengetahui user ID Channel perangkat pendeteksi RAP

   a) Sistem menampilkan jendela awal home web *thinkspeak*

   b) Pengguna memilih tab *channel*

   c) Sistem menampilkan jendela *public channel* dan tab pencarian *search by user* ID

   d) Pengguna memasukkan *user* ID dari channel perangkat pendeteksi RAP dan *submit*

   e) Sistem menampilkan channel-channel yang dibuat dengan user ID tersebut

   f) Pengguna memilih channel yang tepat berdasarkan Channel ID dari perangkat pendeteksi RAP

   g) Sistem menampilkan halaman pemberitahuan dalam bentuk grafik dari hasil pendeteksian yang dilakukan.

7. HP02 Mengetahui user ID dari Channel perangkat pendeteksi RAP

   Aktor: Pengguna

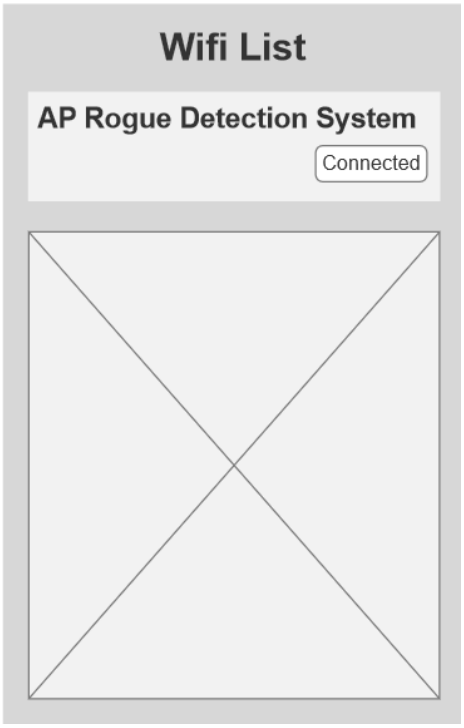
   Prasyarat: -

   a) Pengguna ketika sedang melakukan konfigurasi perangkat RAP dapat melihatnya pada bagian info dari Web Server Lokal perangkat pendeteksi RAP.

3.2.4 Perancangan Antarmuka

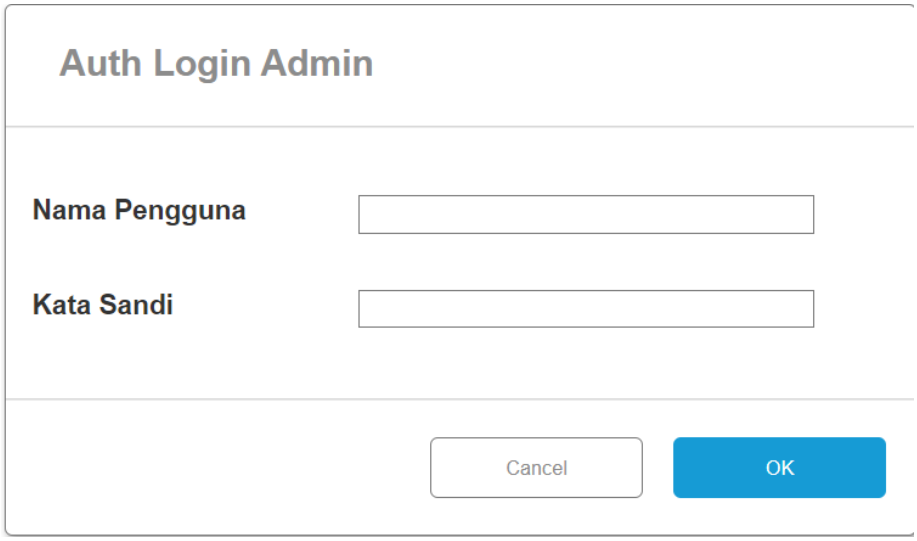
a. Halaman *Login*

   Halaman *login* merupakan halaman yang muncul pertama kali pada saat pengguna menghubungkan komputernya ke AP sementara yang dibuat oleh perangkat pendeteksi serangan RAP kemudian memasukkan alamat ip 192.168.100.1 pada *browser*-nya maka akan tampil halaman *login* tersebut sebagai authentikasi user dengan tujuan keamanan agar tidak disalahgunakan oleh orang-orang yang tidak semestinya. Untuk ilustrasi AP sementara dapat dilihat pada Gambar 3.7.



Gambar 3.7. AP Sementara Device Wemos D1

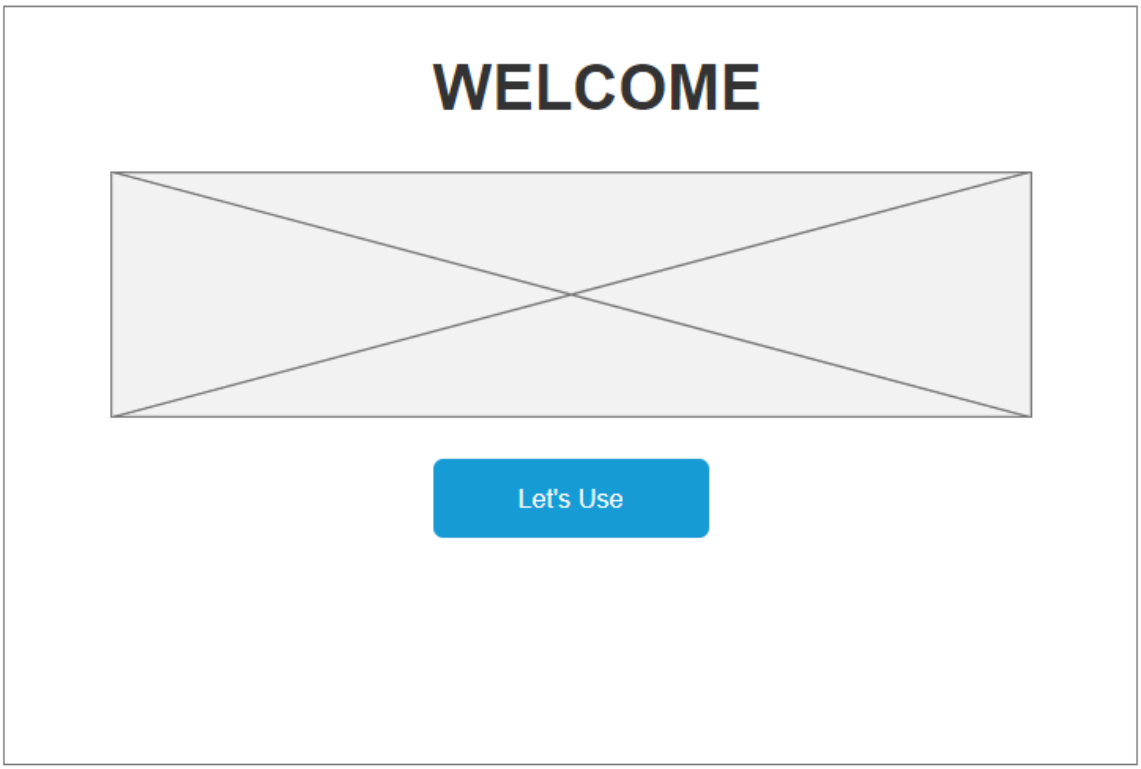
Ketika PC pengguna sudah berhasil terhubung ke AP sementara yang dibuat perangkat wemos d1 maka pengguna sudah dapat mengakses web server lokal untuk mengkonfigurasi sistem yang dibuat. Berikut ilustrasinya dapat dilihat pada Gambar 3.8.



Gambar 3.8. Halaman *Login* Web Server Lokal

b. Halaman Pembuka

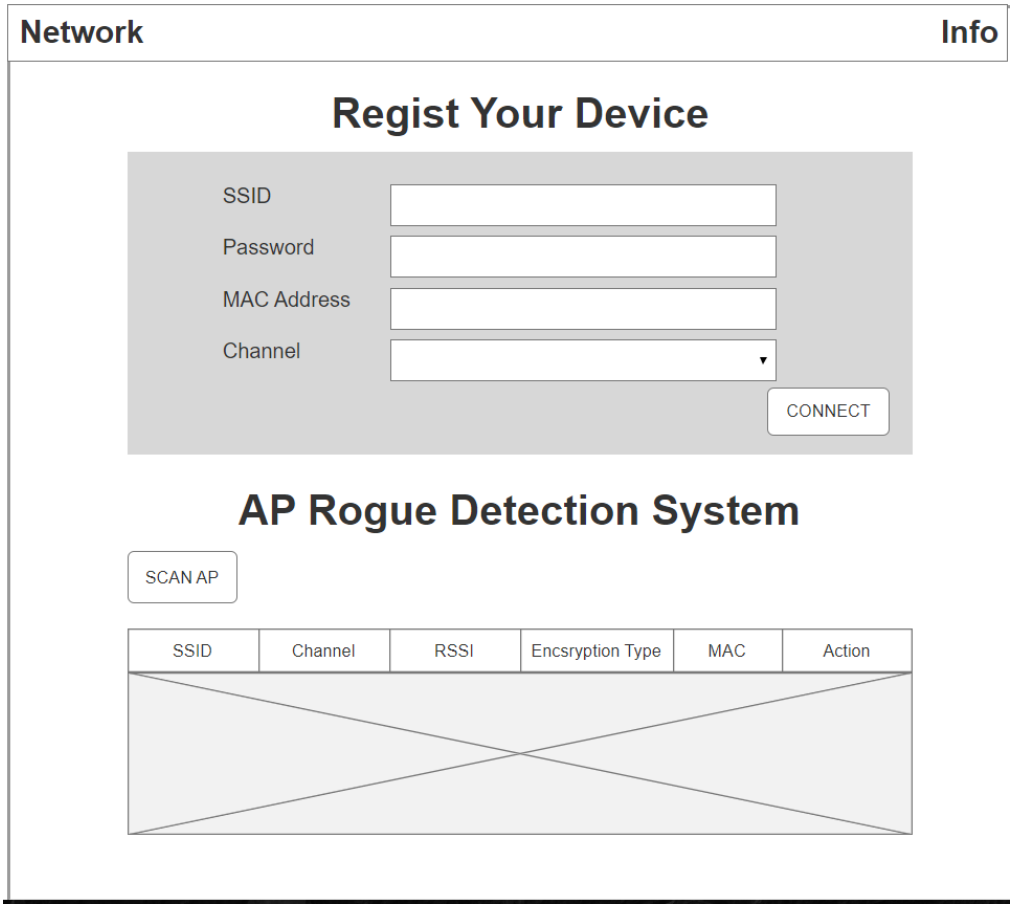
Setelah melakukan *login* pengguna akan secara otomatis dialihkan ke halaman pembuka sebelum masuk ke halaman home. Halaman ini memberikan pesan atau info kepada pengguna fungsi dari sistem ini secara singkat. Tampilannya dapat dilihat Gambar 3.9.



Gambar 3.9. Rancangan Halaman Pembuka

c. Halaman Utama

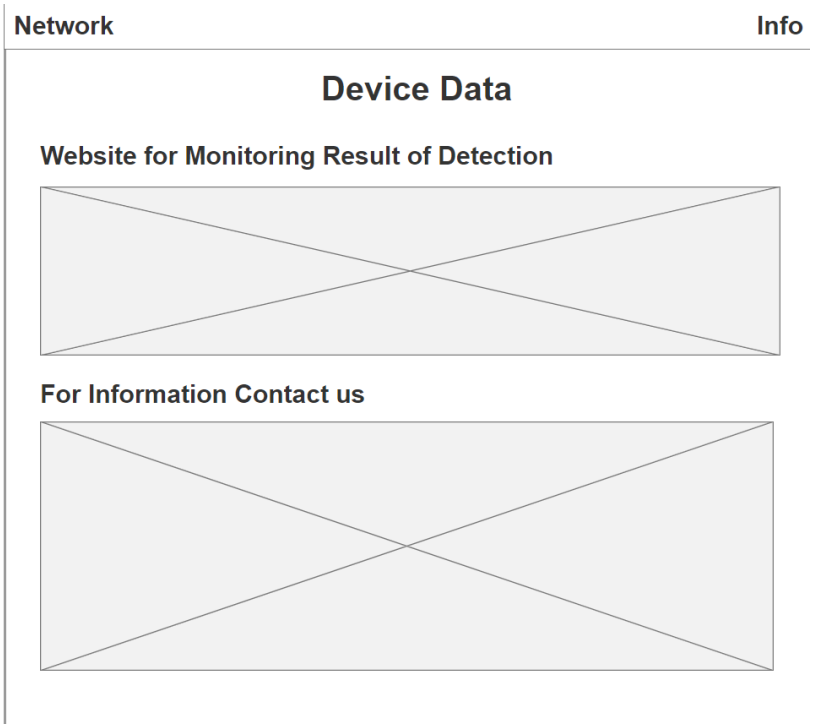
Pada halaman utama sistem deteksi serangan RAP ini terdapat 2 fitur utama, yaitu fitur untuk melihat jaringan AP mana saja yang ada di sekitar yang dapat di-*scan* oleh perangkat yang digunakan dan fitur yang kedua adalah fitur utama pada sistem ini, yang mana digunakan sebagai media menghubungkan parangkat yang dirancang dengan mendaftarkan perangkat mana yang akan dilindungi oleh perangkat yang dirancang tersebut dengan syarat terhubung ke internet dan jenis keamanannya selain dari WPA2-Enterprise. Rancangan antarmuka halaman utama dapat dilihat pada Gambar 3.10.



Gambar 3.10. Rancangan Halaman Utama

d. Halaman Info

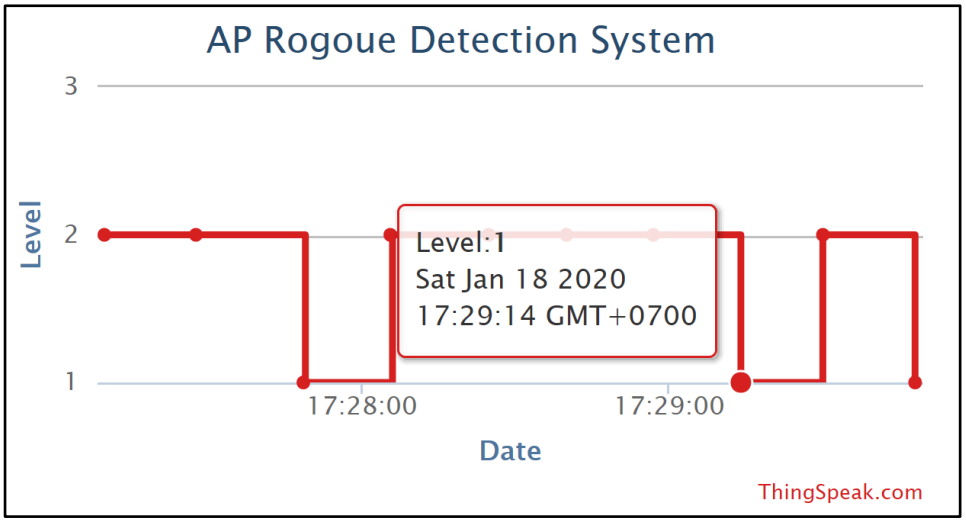
Pada halaman Info berisi info-info yang akan dibutuhkan untuk mengetahui jika peralatan pengguna ingin melihat hasil pendeteksian perangkat yang dibangun, dengan meng-*click* bagian "Click Here" atau membuka secara manual pada website *thinkspeak.com* dan memasukkan User ID dan Channel ID yang terdapat pada halaman info tersebut. Selain itu terdapat juga kontak untuk menanyakan jika terdapat kendala atau ada hal yang ingin disampaikan. Rancangan halaman info dapat dilihat pada Gambar 3.11.



Gambar 3.11. Rancangan Halaman Peralatan

e. Halaman Hasil Pendeteksian

Pada halaman ini administrator dapat melihat hasil pendeteksian yang dilakukan secara *realtime* oleh perangkat yang dirancang tersebut, jika sudah melakukan konfigurasi awal perangkat, yaitu mendafarkan perangkat AP yang akan dilindung dan sudah terhubung ke internet maka grafik tersebut akan secara otomatis ter-*update* setiap waktu. Tampilannya dapat dilihat pada Gambar 3.12.

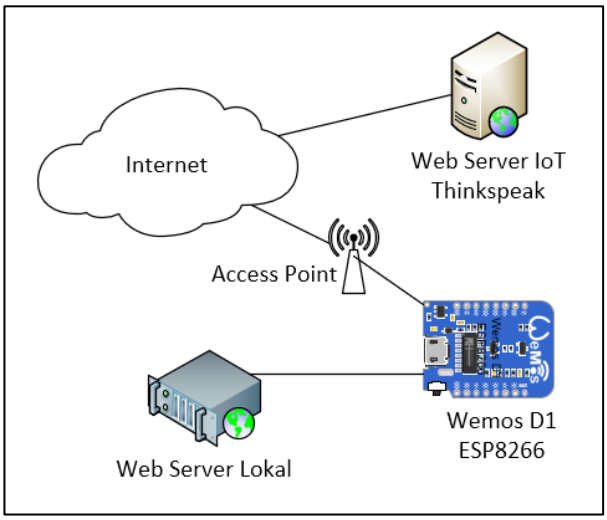


Gambar 3.12. Halaman Hasil Pendeteksian

BAB IV
IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi

Implementasi adalah tahap selanjutnya untuk melakukan apa yang telah dirancang pada tahapan di bab sebelumnya. Implementasi akan menunjukkan apakah perancangan yang telah dilakukan sebelumnya dapat berjalan dan dapat digunakan dengan baik. Pada tahapan implementasi ini terbagi menjadi dua bagian utama yaitu implementasi perangkat lunak dan implementasi perangkat keras. Dalam implementasi perangkat lunak terdapat implementasi web server lokal yang akan menggambarkan secara jelas bagaimana sistem dalam mengkonfigurasi perangkat yang dibangun, dan server IoT juga akan dijelaskan bagaimana kedua sistem dapat saling terhubung. Kemudian pada implementasi perangkat keras akan dijelaskan bagaimana penggunaan atau cara penyimanan perangkat agar pendeteksian lebih akurat.



Gambar 4.1. Ilustrasi Implementasi Sistem Pendeteksi RAP

4.1.1 Implementasi Pembuatan Perangkat Lunak

Pada implementasi perangkat lunak dapat dijelaskan bagaimana sistem dibangun agar dapat berjalan seperti yang sudah direncanakan pada bab sebelumnya. Pada implementasi ini memuat dari pembangunan web server lokal hingga pengguna dapat melihat hasil pendeteksian yang dilakukan oleh sistem.

A. Kode Program Arduino IDE

Tahap implementasi selanjutnya adalah pembuatan kode program untuk Arduino. Pada tahap pembuatan kode program ini, digunakan aplikasi Arduino IDE yang dapat di unduh secara gratis. Aplikasi ini menggunakan bahasa pemrograman C.

Dalam pembuatan program Arduino diperlukan *library-library* yang nantinya digunakan untuk mempermudah dalam membuat program. Dalam pembuatan perangkat pendeteksi serangan RAP menggunakan wemos d1, digunakan beberapa *library* yang ditunjukkan oleh Tabel 4.1. Pada Tabel 4.1 di bawah digunakan beberapa *library*, di antaranya adalah ESP8266WiFi digunakan untuk memanggil perintah-perintah yang berhubungan dengan teknologi WiFi yang mana pada kasus ini menggunakan modul, ESP8266WebServer digunakan untuk membangun suatu web server lokal dari jaringan wifi yang dibangun menggunakan modul WiFi ESP8266, FS atau SPIFFS (Serial Peripheral Interface Flash File System) yang digunakan untuk memasukkan atau *mengimport* web server yang dibangun menggunakan bahasa pemrograman Web seperti html, ccs, php, java script, json, jquery dan lainnya. Dengan menggunakan *library* tersebut akan sangat memudahkan dalam membangun web server. Selanjutnya adalah ArduinoJson digunakan untuk merepresentasikan struktur data sederhana dalam pengiriman data yang terjadi pada sistem. Source code *library* yang digunakan dapat dilihat pada Tabel 4.1.

Tabel 4.1. Library yang digunakan pada perangkat Wemos D1 ESP8266

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
#include <FS.h>
#include <ArduinoJson.h>
```

Dalam pembangunan perangkat lunak ini terdapat beberapa prosedur dan fungsi-fungsi yang dideklarasikan pada Arduino IDE untuk mencapai rencana perancangan sistem yang dibahas pada bab sebelum ini, yaitu String index, void handleRoot, bool loadFromSpiffs, String scanWifi, void handleWebRequest, void setup, void loop, void handleSettingUpdate, dan wifiConnect.

Fungsi String index berfungsi untuk membuat halaman pembuka setelah pengguna berhasil melakukan authentikasi *login* yang dirancang sedemikian rupa. Fungsi ini dibuat berdasarkan format html yang dideklarasikan dengan *variabel name* dan kemudian akan

mengembalikan nilai name sebagai output dari fungsi tersebut. Berikut potongan *source code* fungsi String index pada Tabel 4.2.

Tabel 4.2. Potongan *source code* String index pada Arduino ide

```
String index(){
  String name = "";
  name += "<html>";
  name += "<head>";
  name += "<title>AP Rogoue Detection System</title>";
  name += "<link rel='stylesheet' href='style.css'>";
  name += "</head>";
  name += "<body>";
  name += "<div class='container'>";
  name += "<div class='row'>";
  name += "<div class='col-12'>";
  name += "<div id='error' class='hide'></div>";
  name += "<h1 class='header' data-translate='warning'>WELCOME</h1>";
  name += "<p class='centered bold' data-translate='disclaimer'>";
  name += "This is a system for scan AP Rogue in your room and
configuration your device to send you a notification on the IoT server
thinkspeak. Check tab info for get the link to thinkspeak website";
  name += "</p>";
  name += "<p class='centered bold'>";
  name += "<a class='button' href='index.html' data-
translate='disclaimer-button'>Let's Use</a>";
  name += "</p>";
  name += "</div></div></div>";
  name += "</body>";
  name += "</html>";

  return name;
}
```

Prosedur void handleRoot berfungsi sebagai pengatur ketika pengguna mengakses *ip address* dari web server lokal setelah terkoneksi dengan wifi sementara yang dibuat maka akan secara otomatis dialihkan ke halaman *login*. Ketika *session login* sudah menjadi *true* akan secara otomatis dialihkan ke halaman pembuka, jika pada halaman *login* pengguna memasukkan *username* dan *password* yang tidak valid maka halaman atau *form login* akan

selalu tampil hingga mendapatkan *username* dan *password* yang valid, dan jika pada halaman *login* pengguna menekan tombol *cancel* atau *close* atau *esc* maka akan dialihkan ke halaman kosong. Berikut potongan *source code*-nya pada Tabel 4.3.

Tabel 4.3. Source code void handleroot pada arduino ide

```
void handleRoot(){
  server.sendHeader("Location", "/login",true);   //Redirect to our html
web page
  server.send(302, "text/plane","");
}
```

Fungsi bool loadFromSpiffs berfungsi sebagai pengatur ketika pengguna me-*request* atau ingin membuka halaman info atau home maka fungsi ini akan membaca file yang diimpor sebelumnya pada *library* SPIFFS, fungsi ini dibuat karena sebagian besar file web server lokal tersebut dirancang dengan bahasa pemrograman web dengan tujuan akan lebih mempermudah dalam proses perancangan. Berikut potongan *source code*-nya pada Tabel 4.4.

Tabel 4.4. Potongan *source code* bool loadFormSpiffs pada Arduino ide

```
bool loadFromSpiffs(String path){
  String dataType = "text/plain";
  if(path.endsWith("/")) path += "/index.html";

  if(path.endsWith(".src")) path = path.substring(0,
path.lastIndexOf("."));
  else if(path.endsWith(".html")) dataType = "text/html";
  else if(path.endsWith(".css")) dataType = "text/css";
  else if(path.endsWith(".js")) dataType = "application/javascript";
  else if(path.endsWith(".xml")) dataType = "text/xml";
  File dataFile = SPIFFS.open(path.c_str(), "r");
  if (server.hasArg("download")) dataType = "application/octet-stream";
  if (server.streamFile(dataFile, dataType) != dataFile.size()) {
  }
  dataFile.close();
  return true;
 }
```

Fungsi String scanWifi berfungsi untuk mengscan wifi yang ada di sekitar kemudian mengirimkannya dalam format html ke web server lokal yang dibuat. Sebab fungsi ini dibuat pada Arduino IDE dan tidak langsung dirancang pada file pemrograman web karena pemrograman web tidak mengenal perintah yang dipanggil pada fungsi-fungsi yang terdapat pada *library* yang digunakan. Oleh sebab itu fungsi ini dibuat sedemikian rupa untuk menjalankan perintah-perintah scan wifi dan mengirimkannya ke web server lokal yang dibangun. Berikut potongan *source code*-nya pada Tabel 4.5.

Tabel 4.5. Potongan *source code* String scanWifi pada Arduino ide

```
String scanWifi(){

    WiFi.scanNetworks(false,true);
    int n = WiFi.scanComplete();
    String name = "";
    name += "<h2><span>Access Points : ";
    name += n;
    name += "</span></h2>";
    name += "<table id='apTable'>";
    name += "<tr>";
    name +=  "<th class='ssid'>SSID</th>";
    name +=  "<th class='channel'>Channel</th>";
    name +=  "<th class='rssi'>RSSI</th>";
    name +=  "<th class='enc'>Encscryption Type</th>";
    name +=  "<th class='mac'>MAC</th>";
    name +=  "<th class='mac'>ACTION</th>";
    name +=  "</tr>";

    for (int i = 0; i < n; ++i)
    {
      name += "<tr>";
      name += "<td class='ssid'>";
      name += WiFi.SSID(i);
      name +="</td>"; // SSID
      name += "<td class='channel'>";
      name += WiFi.channel(i);
      name +="</td>";
      name += "<td class='rssi'><div class='meter_background'> <div
class='meter_forground'><div class='meter_value'>";
```

```
      name += WiFi.RSSI(i);
      name += "</div></div> </div></td>";  // RSSI
      name += "<td class='enc'>";
      name += WiFi.encryptionType(i) == ENC_TYPE_NONE?" Unsecured":"
Secured";
      name += "</td>"; // ENC
      name += "<td class='mac'>";
      name += WiFi.BSSIDstr(i);
      name += "</td>"; // MAC
      name += "<td class='action'>";
      name += "<button onclick='tambah()' data-
translate='reload'>ADD</button>";
      name +="</td>"; // ADD to Form
      name += "</tr>";
    }
    name += "</table>";
    return name;
}
```

Prosedur void handleWebRequest berfungsi sebagai pengatur awal dari fungsi loadFromSpiffs sebelumnya, pengatur maksudnya adalah ketika file yang di-*request* oleh pengguna atau halaman yang ingin dibuka oleh pengguna tidak ada di *library* SPIFFS maka akan diberikan respon balik ke pengguna bahwa halaman yang di-*request* tidak ada pada *library* SPIFFS yang diimpor sebelumnya. Berikut potongan *source code*-nya pada Tabel 4.6.

Tabel 4.6. Potongan *source code* void handleWebRequest pada Arduino ide

```
void handleWebRequests(){
  if(loadFromSpiffs(server.uri())) return;
  String message = "File Not Detected\n\n";
  message += "URI: ";
  message += server.uri();
  message += "\nMethod: ";
  message += (server.method() == HTTP_GET)?"GET":"POST";
  message += "\nArguments: ";
  message += server.args();
  message += "\n";
  for (uint8_t i=0; i<server.args(); i++){
```

```
    message += " NAME:"+server.argName(i) + "\n VALUE:" + server.arg(i) +
"\n";
  }
  server.send(404, "text/plain", message);
  Serial.println(message);
}
```

Prosedur void setup berfungsi untuk inisialisasi program ketika dijalankan, karena prosedur ini adalah prosedur yang hanya sekali saja dijalankan yaitu di awal program berjalan. Pada prosedur ini akan menjalankan prosedur wifiConnect. Berikut potongan *source code*-nya pada Tabel 4.7.

Tabel 4.7. Potongan *source code* void setup pada Arduino ide

```
void setup() {
  delay(1000);
  Serial.begin(115200);
  Serial.println();

  //Initialize File System
  SPIFFS.begin();
  Serial.println("File System Initialized");

  wifiConnect();

  server.on("/settings", HTTP_POST, handleSettingsUpdate);
}
```

Prosedur void loop merupakan prosedur yang selalu berjalan ketika perangkat menyala. Dalam prosedur ini memiliki 2 kondisi utama, yaitu dengan mengecek apakah perangkat wemos d1 terhubung ke AP atau tidak. Jika terhubung maka akan dilakukan pengecekan lagi, apakah AP yang disambungkan tersebut terhubung ke internet atau tidak, jika terhubung ke internet maka perangkat tersebut akan melakukan proses *scanning* wifi di sekitar untuk menentukan apakah ada serangan RAP atau tidak, kemudian hasil pendeteksian tersebut dikirimkan ke server *thinkspeak* dengan *API* yang didapatkan ketika melakukan registrasi awal pada server tersebut dan jika AP tersebut tidak terhubung ke internet maka akan secara otomatis perangkat wemos akan membuat AP sementaranya untuk dikonfigurasi kembali oleh pengguna

ke perangkat AP yang memiliki koneksi internet. Jika perangkat wemos tidak sedang terhubung ke AP manapun maka akan secara otomatis perangkat Wemos akan membuat AP sementaranya. Berikut potongan *source code*-nya pada Tabel 4.8.

Tabel 4.8. Potongan *source code* void loop pada Arduino ide

```
void loop() {
  if (WiFi.status() == WL_CONNECTED) {
    String postStr = apiKey;
    int network;
    int i = 0;
    bool normal = true;
    if (client.connect(serveriot, 80))
    {
      if (SPIFFS.exists("/config.json")) {
        const char * _ssid = "", * _mac = "", * _channel = "";
        File configFile = SPIFFS.open("/config.json", "r");
        if (configFile) {
          size_t size = configFile.size();
          std::unique_ptr<char[]> buf(new char[size]);
          configFile.readBytes(buf.get(), size);
          configFile.close();

          DynamicJsonBuffer jsonBuffer;
          JsonObject& jObject = jsonBuffer.parseObject(buf.get());
          if (jObject.success())
          {
            _ssid = jObject["ssid"];
            _mac = jObject["mac"];
            _channel = jObject["channel"];
            WiFi.scanNetworks(false, true);
            network = WiFi.scanComplete();
            if (network <= 0) {
              WiFi.scanNetworks();
              network = WiFi.scanComplete();
            } else {
              int ssidCounting = 0;
              int macCounting = 0;
              for (int i = 0; i < network; i++) {
                if (WiFi.SSID(i) == _ssid) {
```

```
                    ssidCounting++;
                    if (WiFi.BSSIDstr(i) == _mac) {
                      if (String(WiFi.channel(i)) != _channel) {
                        macCounting++;
                      }
                      if (int(WiFi.RSSI(i)) < -48) {
                        macCounting +=  2;
                      }
                    }
                  } else if (WiFi.SSID(i) != _ssid) {
                    if (WiFi.BSSIDstr(i) == _mac) {
                      if (String(WiFi.channel(i)) != _channel) {
                        macCounting++;
                      }
                      if (int(WiFi.RSSI(i)) < -48) {
                        macCounting +=  2;
                      }
                    }
                  }
                }
                if (macCounting > 1) {
                  postStr += "&field1=";
                  postStr += "3";
                } else if (ssidCounting > 1) {
                  postStr += "&field1=";
                  postStr += "2";
                } else {
                  postStr += "&field1=";
                  postStr += "1";
                }
                postStr += "\r\n\r\n";
                client.print("POST /update HTTP/1.1\n");
                client.print("Host: api.thingspeak.com\n");
                client.print("Connection: close\n");
                client.print("X-THINGSPEAKAPIKEY: " + apiKey + "\n");
                client.print("Content-Type: application/x-www-form-
urlencoded\n");
                client.print("Content-Length: ");
                client.print(postStr.length());
                client.print("\n\n");
```

```
              client.print(postStr);
              Serial.println(postStr);
            }
            client.stop();
          }
        }
      }
    } else {
    server.handleClient();
  }
}
```

Prosedur void handleSettingUpdate berfungsi sebagai pembaca apakah pada memori perangkat terdapat data config.json yang sebelumnya disimpan pada saat pengguna melakukan konfigurasi perangkat mana yang akan dilindungi. Ketika sudah dilakukan pengecekan maka akan menjalankan prosedur wifiConnect. Berikut potongan *source code*-nya pada Tabel 4.9.

Tabel 4.9. Potongan *source code* void handleSettingUpdate pada Arduino ide

```
void handleSettingsUpdate()
{
  String data = server.arg("plain");
  DynamicJsonBuffer jBuffer;
  JsonObject& jObject = jBuffer.parseObject(data);

  File configFile = SPIFFS.open("/config.json", "w");
  jObject.printTo(configFile);
  configFile.close();

  server.send(200, "application/json", "{\"status\" : \"ok\"}");
  delay(500);

  wifiConnect();
}
```

Prosedur wifiConnect dapat dikatakan merupakan prosedur jantung pada sistem ini. Dikatakan seperti itu karena prosedur ini yang akan menentukan perangkat wemos akan berada di posisi pengguna atau sebagai server. Prosedur ini akan mengecek terlebih dahulu apakah

pada file config.json terdapat data ssid dan *password* dari AP sebelumnya atau tidak, jika ada maka perangkat wemos akan menjadi pengguna dari AP yang datanya disimpan. Jika tidak maka perangkat wemos akan membuat AP sementara dengan ssid “APRogueDetecionSystem” dan *password* “12345678” kemudian menyalakan lampu led modul ESP8266 dan membuat session *login* ketika *ip address* dari perangkat wemos yang diakses oleh pengguna dari perangkat wemos dengan *ip address*“192.168.100.1” melalui browser pengguna. Berikut potongan *source code*-nya pada Tabel 4.10.

Tabel 4.10. Potongan *source code* void wifiConnect pada Arduino ide

```
void wifiConnect()
{
  //reset networking
  WiFi.softAPdisconnect(true);
  WiFi.disconnect();
  if(SPIFFS.exists("/config.json")){
      if(jObject.success())
      {
        _ssid = jObject["ssid"];
        _pass = jObject["password"];
        WiFi.mode(WIFI_STA);
        WiFi.begin(_ssid, _pass);
        unsigned long startTime = millis();
        while (WiFi.status() != WL_CONNECTED)
        {
          delay(500);
          Serial.print(".");
          if ((unsigned long)(millis() - startTime) >= 10000) break;
        }
        pinMode(ledPin, LOW);
      }
    }
  }
  if (WiFi.status() == WL_CONNECTED)
  {
  } else
  {
    pinMode(ledPin, HIGH);
    WiFi.mode(WIFI_AP);
```

```
    WiFi.softAPConfig(lokal_ip, gateway, netmask);
    WiFi.softAP(mySsid, password);

    server.on("/",handleRoot);
    server.on("/login", []() {
      if (!server.authenticate(www_username, www_password)){
        return server.requestAuthentication(DIGEST_AUTH, www_realm,
authFailResponse);
        }
        else{
          server.onNotFound(handleWebRequests);
          server.send(301, "text/html",index());

server.on("/scan",[](){server.send(200,"text/plain",scanWifi());});
        }
      });
    server.begin();
  }
}
```

B. Implementasi Pembuatan Web Server Lokal

Pada implementasi pembuatan sistem, juga dibangun suatu web server lokal yang digunakan sebagai media pengguna atau administrator jaringan dalam mengkonfigurasi perangkat keras atau sistem deteksi serangan RAP agar berjalan sesuai dengan yang dibutuhkan atau untuk mendaftarkan AP yang akan dilindungi dan melakukan *scanning* pada perangkat yang ada di sekitar perangkat wemos d1. Web server lokal ini dibangun dengan bahasa pemrograman web yang biasa digunakan untuk membangun halaman-halaman serta fitur-fitur yang dibutuhkan. Dalam webserver yang dibangun terdapat beberapa halaman, yaitu halaman pembuka, halaman home, dan halaman info. Ketika pengguna ingin mengakses ke web server ini harus *login* atau melakukan authentikasi pada saat pengguna pertama kali mengaksesnya dengan memasukkan *username* dan *password* yang sudah ditetapkan pada saat membangun sistem.

Halaman pembuka dirancang saat program dibuat pada Arduino IDE, karena proses authentikasinya menggunakan *library* yang terdapat pada Wemos D1 dan kemudian ketika pengguna berhasil melakukan authentikasi maka akan secara otomatis dialihkan ke halaman

pengguna. Berikut potongan *source code* dari halaman pengguna yang dibuat di Arduino IDE pada Tabel 4.11.

Tabel 4.11. Souce code halaman pembuka

```
String index() {
  String name = "";
  name += "<html>";
  name += "<head>";
  name += "<title>AP Rogoue Detection System</title>";
  name += "<link rel='stylesheet' href='style.css'>";
  name += "</head>";
  name += "<body>";
  name += "<div class='container'>";
  name += "<div class='row'>";
  name += "<div class='col-12'>";
  name += "<div id='error' class='hide'></div>";
  name += "<h1 class='header' data-translate='warning'>WELCOME</h1>";
  name += "<p class='centered bold' data-translate='disclaimer'>";
  name += "This is a system for scan AP Rogue in your room and configuration
your device to send you a notification on the IoT server thinkspeak. Check
tab info for get the link to thinkspeak website";
  name += "</p>";
  name += "<p class='centered bold'>";
  name += "<a class='button' href='index.html' data-translate='disclaimer-
button'>Let's Use</a>";
  name += "</p>";
  name += "</div></div></div>";
  name += "</body>";
  name += "</html>";

  return name;
}
```

Halaman home dirancang sebagai media pengguna mengkonfigurasi perangkat wemos d1 yang dibangun menggunakan bahasa pemrograman web dengan beberapa fungsi, yaitu menghubungkan perangkat wemos ke AP yang akan dilindungi dan melakukan *scanning* perangkat WLAN yang ada di sekitar. Pada fungsi yang terdapat pada halaman home ini saling tersinkron dengan Arduino IDE karena perintang untuk me-*scanning* wemos dan

menghubungkan wemos ke perangkat AP menggunakan *library* yang ada pada Arduino IDE, data hasil konfigurasi dikirimkan dalam JSON yang disimpan pada memori wemos d1 dan data hasil *scanning* dikirmkan langsung dengan format tabel html ke halaman home yang dibuat. Berikut *source code* dari halaman home dengan nama file index.html pada Tabel 4.12.

Tabel 4.12. Potongan *source code* halaman home

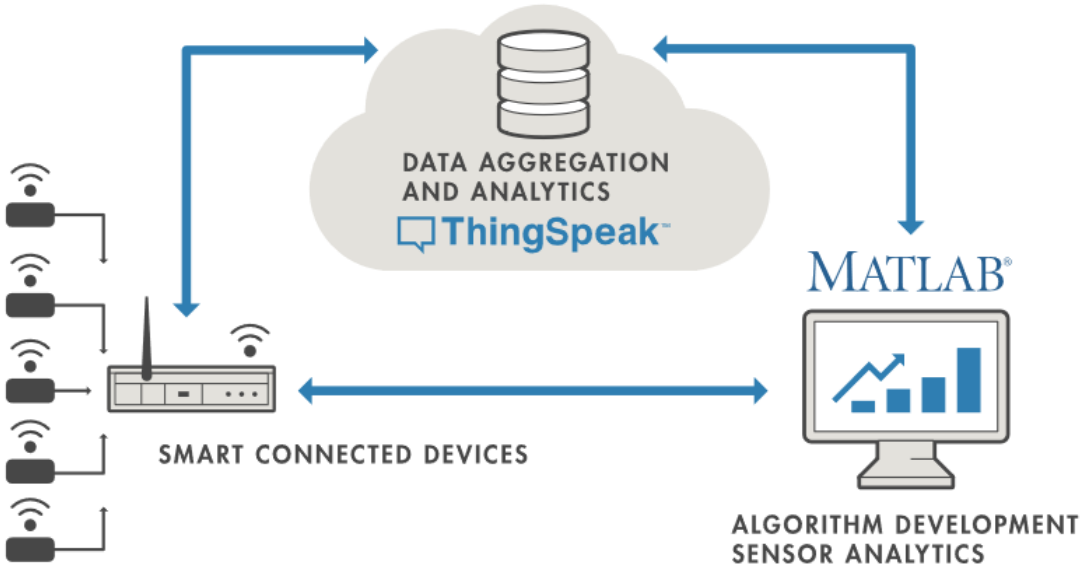
```
<script>
function myFunction()
{
  console.log("button was clicked!");
  var ssid = document.getElementById("ssid").value;
  var password = document.getElementById("password").value;
  var mac = document.getElementById("mac").value;
  var channel = document.getElementById("channel").value;
  var data = {ssid:ssid, password:password, mac:mac, channel:channel};
  var xhr = new XMLHttpRequest();
  var url = "/settings";
  xhr.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      if(xhr.responseText != null){
        console.log(xhr.responseText);
      }
    }
  };
  xhr.open("POST", url, true);
  xhr.send(JSON.stringify(data));
};

function tambah(){
   var ssid = "";
   var password = "";
   var mac = "";
   var channel = "";
}
</script>
```

Dari perancangan perangkat lunak khususnya pada potongan code-code yang dimasukan atau upload ke dalam wemos d1 menggunakan Arduino ide dan SPIFFS sudah dijelaskan di atas. Untuk *source code* lengkapnya dapat dilihat pada bagian lampiran.

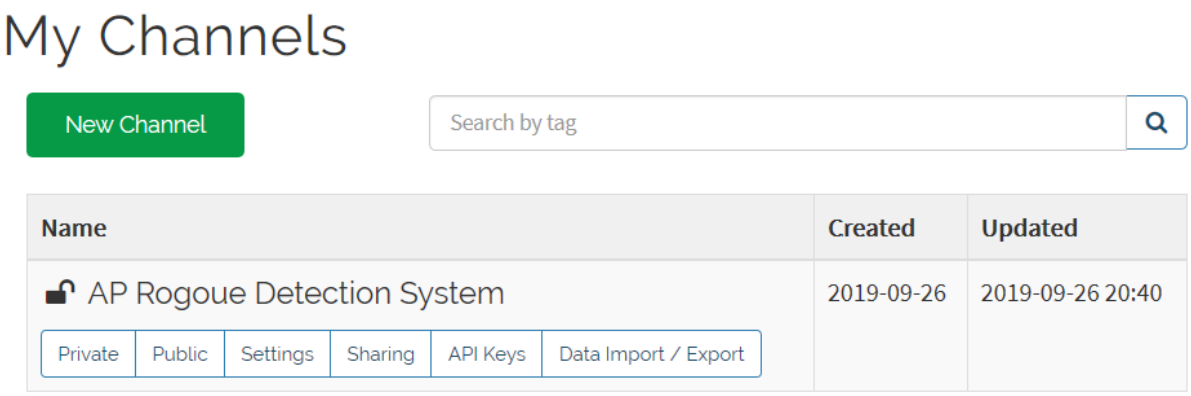
C. Impelementasi persiapan Web Server IoT

Pada implementasi persiapan web server IoT sebagai media pengguna memantau perangkat AP yang dilindungi dapat menggunakan server IoT *thinkspeak* yang sebelum digunakan harus registrasi terlebih dahulu untuk mendapatkan *API Key* agar data dari perangkat atau sensor dapat ditampilkan oleh web server *thinkspeak*. Ilustrasi cara kerja web server dapat dilihat pada Gambar 4.2.



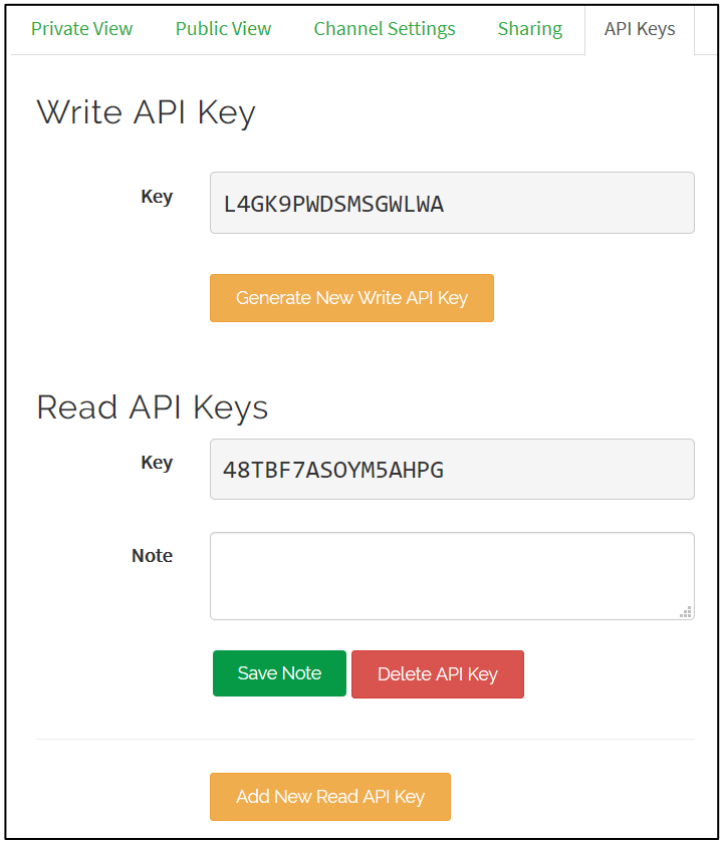
Gambar 4.2. Ilustrasi cara kerja server IoT

Dalam proses registrasi akan membutuhkan email, location, first name, dan last name pengguna sebagai data pengembang. Setelah berhasil melakukan registrasi dan berhasil *login* akan secara otomatis dialihkan ke halaman home, pada halaman ini pengembang bisa membuat channel sesuai kebutuhannya. Setiap channel akan digunakan dengan fungsi yang berbeda-beda. Berikut tampilan halaman *home* pada Gambar 4.3.



Gambar 4.3. Halaman home web server yang digunakan

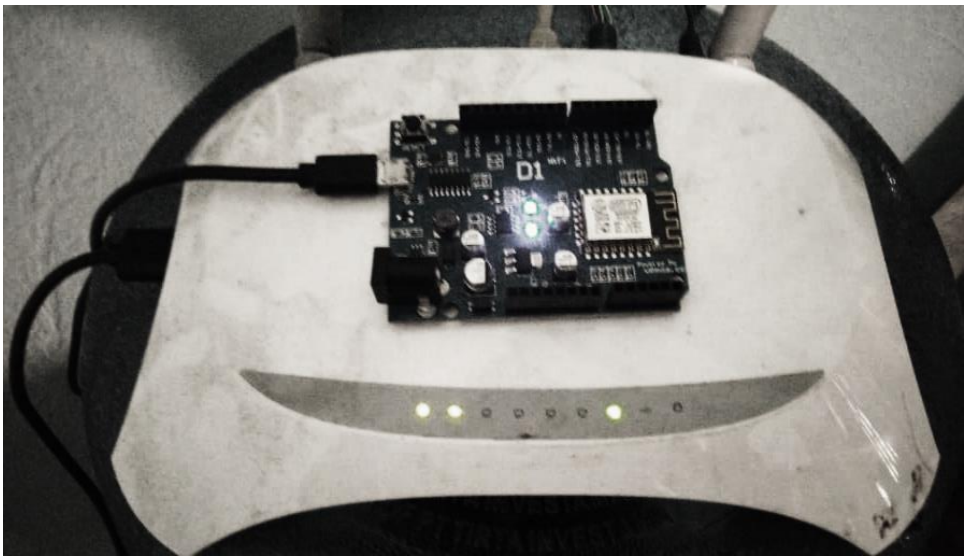
Untuk melihat *API Key* yang akan digunakan untuk menghubungkan perangkat dengan web server tinggal memilih tab *API Keys* dan akan masuk ke halaman *api key* yang mana terdapat beberapa *api key*, yaitu untuk menulis data (yang digunakan pada perangkat wemos) dan untuk membaca data jika data tersebut dibutuhkan untuk memantau melalui web sendiri. Tampilannya dapat dilihat pada Gambar 4.4.



Gambar 4.4. Tampilan halaman *API Keys*

4.1.2 Implementasi Persiapan Perangkat Keras

Penelitian sistem deteksi serangan RAP ini menggunakan Wemos sebagai media pengendali utama kelistrikan pada peralatan elektronik. Pada mikrokontroler Wemos sudah tertanam modul wireless Esp8266, yang sudah bisa digunakan tanpa harus menggunakan modul tambahan seperti yang ada pada Arduino. Karena Wemos sudah terintegrasi dengan jaringan wireless, maka komunikasi antar Wemos dan komunikasi dengan server IoT pun dilakukan dengan menggunakan koneksi wireless yang terhubung ke internet. Menghidupkan mikrokontroler Wemos digunakan daya sebesar 3.3 volt – 5 volt yang bisa didapatkan dengan menggunakan kabel micro usb. Dalam implementasi perangkat Wemos ini harus di tempelkan pada perangkat AP yang akan dilindungi. Tampilan rangkaian koneksi komunikasi pada Mikrokontroler dapat dilihat pada Gambar 4.5.



Gambar 4.5. Rangkaian komunikasi Mikrokontroler Wemos

4.2 Pengujian Sistem deteksi serangan RAP

Tahap pengujian penelitian dilakukan untuk mengetahui apakah sistem yang dibuat dapat berjalan dengan baik dan lancar. Dengan dilakukannya pengujian ini, dapat ditemukannya kekurangan dan kelebihan dari sistem pendeteksi AP palsu yang telah dibuat. Pengujian dilakukan dengan cara:

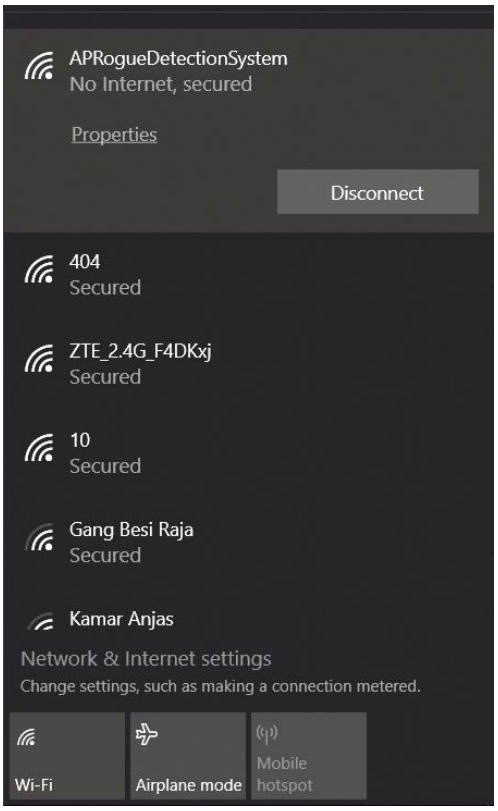
a. Melakukan uji proses menghubungkan ke AP sementara yang dibuat oleh wemos D1.
b. Melakukan uji proses *login* ke web server lokal.
c. Melakukan uji proses scan wifi yang ada di sekitar.
d. Melakukan uji proses menghubungkan ke AP yang akan dilindungi.

e. Melakukan uji *monitoring* pesan yang dikirimkan pada server IoT.

f. Melakukan simulasi penyerangan RAP pada AP yang dilindungi.

4.2.1 Pengujian menghubungkan ke AP sementara wemos d1

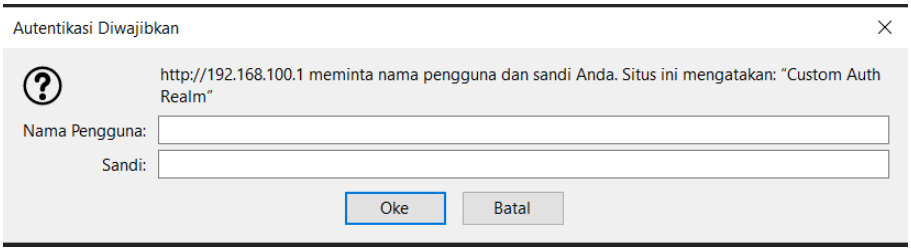
Pengguna atau administrator jaringan jika ingin membuka web server lokal pada browser sebelumnya harus tergabung ke satu jaringan yang sama dengan perangkat wemos d1, oleh karena itu wemos d1 membuat AP sementara untuk pengguna melakukan instalasi awal perangkat sebelum melindunginya dengan *password* "12345678". Tampilan AP sementara Wemos D1 pada perangkat komputer windows 10 pengguna pada Gambar 4.6.



Gambar 4.6. AP sementara wemos D1

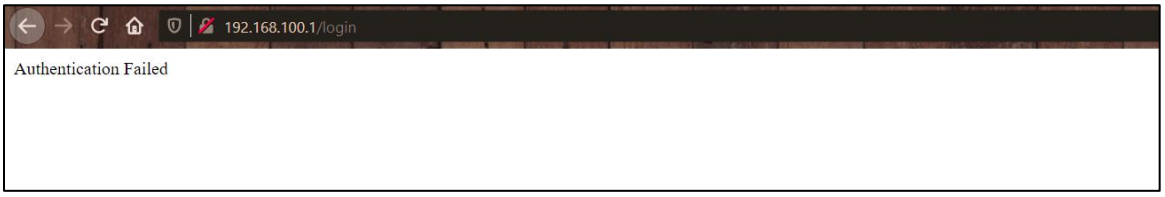
4.2.2 Pengujian login pada web server lokal

Ketika pengguna atau administrator jaringan sudah berhasil tersambung ke AP sementara yang dibuat oleh perangkat wemos d1 jika pengguna mengakses *ip address* "192.168.1.100" maka akan secara otomatis muncul tampilan *login* sebagai authentikasi pengguna dalam penggunaan perangkat. Untuk halaman *login* dapat dilihat pada Gambar 4.7.



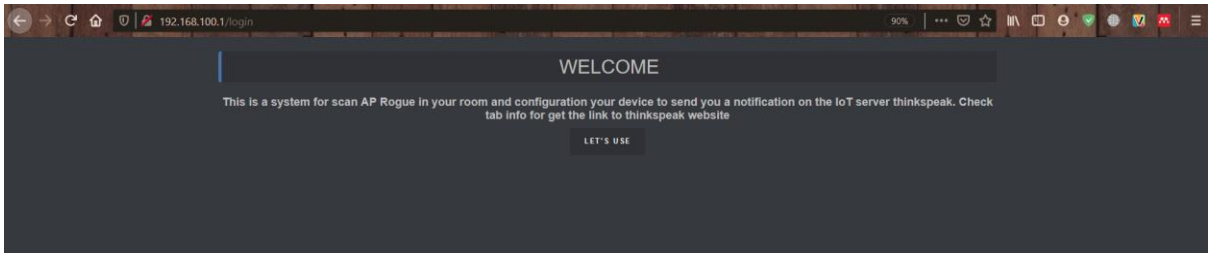
Gambar 4.7. Halaman *login* pengguna

Jika pengguna salah memasukkan *username* dan *password* maka halaman *login* tersebut akan selalu muncul untuk meminta *username* dan *password* yang valid dan jika pengguna me-*close* halaman tersebut maka akan dialihkan ke halaman kosong seperti pada Gambar 4.8.



Gambar 4.8. Halaman pengalih *login*

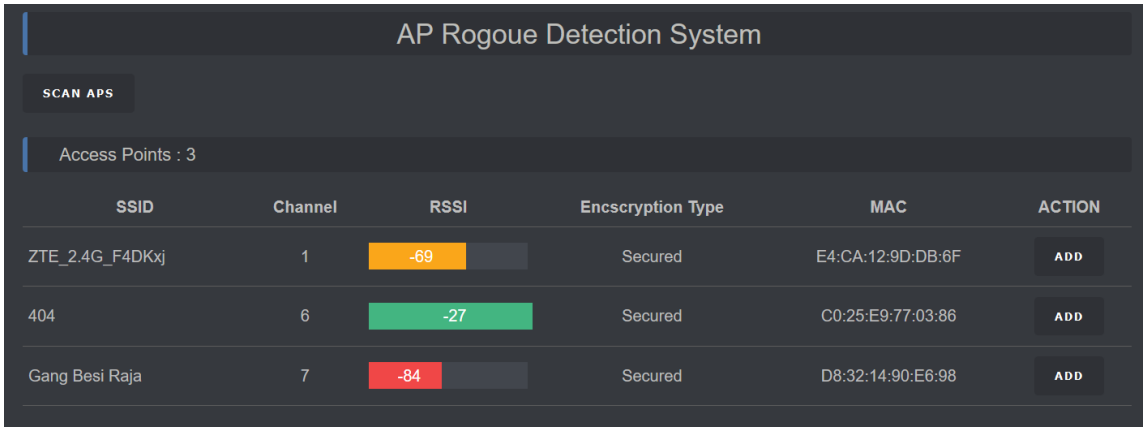
Jika pengguna berhasil *login* maka akan secara otomatis dialihkan ke halaman home atau pembuka yang dapat dilihat pada gambar 4.9.



Gambar 4.9. Halaman home atau pembuka

4.2.3 Pengujian *scanning* AP di sekitar

Pada sistem yang dirancang ini, *scanning* AP merupakan suatu proses yang dilakukan untuk mengetahui AP mana saja yang ada di sekitar ketika pengguna menekan tombol Scan AP. Dalam hasil *scanning* akan muncul beberapa AP dengan SSID, channel *frequensi*, kualitas sinyal yang didapat oleh perangkat wemos d1, secured/unsecured, dan MAC address dari AP yang di *scan*. Berikut hasil pengujian *scanning* AP di sekitar pada Gambar 4.10.

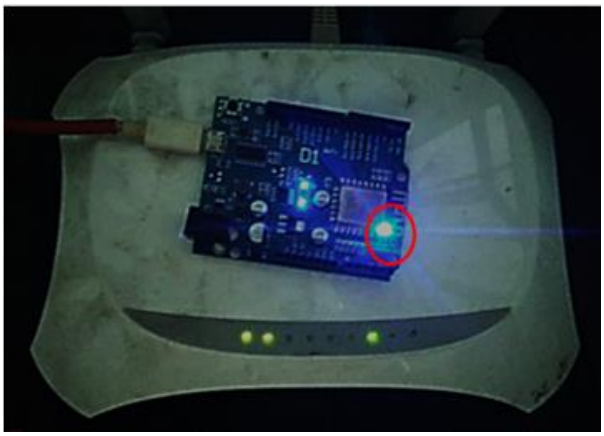


Gambar 4.10. Hasil *Scanning* AP

Dari hasil *scanning* AP diatas terdapat 3 AP yang berhasil di dapatkan pada proses *scanning* atau yang berada di sekitar. Dari hasil-hasil *scanning* yang ada dapat dilakukan action dengan menekan tombol ADD untuk menambahkan atau mengoper data-datanya pada form AP yang akan dilindungi.

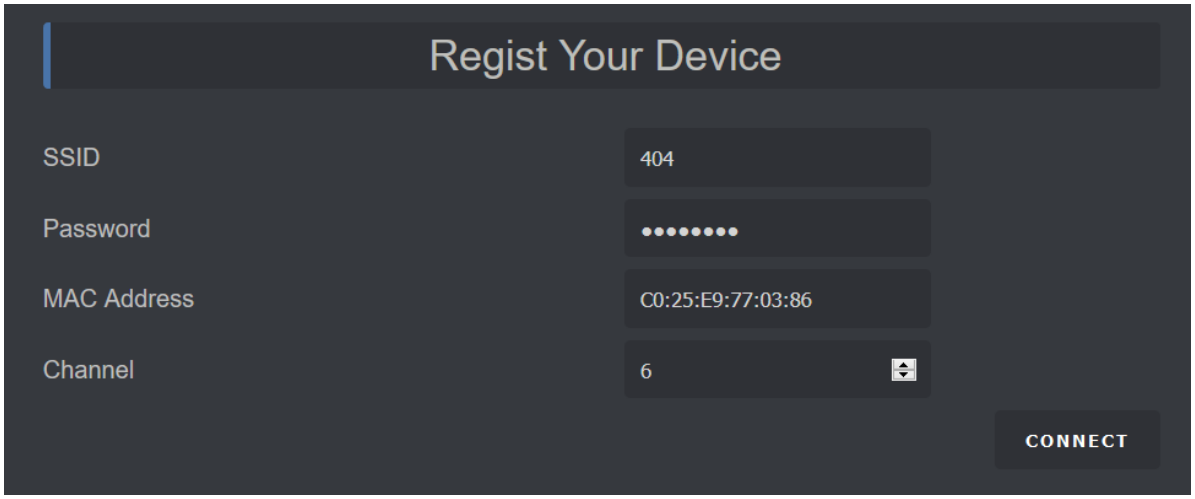
4.2.4 Pengujian menghubungkan AP pada Wemos D1

Pada proses ini akan dilakukan pengujian perangkat atau sistem dengan mendaftarkan AP yang akan dilindungi dari hasil *scanning* sebelumnya dengan menekan tombol action ADD untuk mengambil data AP hasil *scanning* ke form agar tidak perlu mengetikan ulang data-datanya dan tinggal memasukkan *password* dari AP yang akan dilindungi untuk menghubungkan ke AP tersebut. Perangkat wemos jika sedang dalam mode listening atau tidak melindungi AP akan menyalakan lampu indikator agar pengguna dapat mengetahuinya. Berikut tampilan perangkat jika dalam mode listening pada Gambar 4.11.



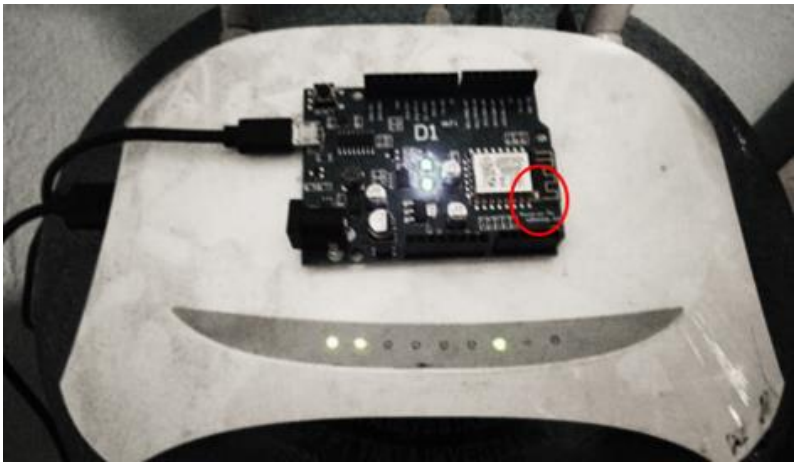
Gambar 4.11. Perangkat wemos mode listening

Pada saat perangkat wemos berada pada mode listening maka tidak ada ditemukannya perangkat yang dilindungi sebelumnya di sekitar ketika wemos dinyalakan. Oleh karena itu perangkat wemos harus dikonfigurasi lagi atau dilakukan proses instalasi dengan menghubungkan perangkat ke AP yang akan dilindungi lagi. Berikut tampilan instalasi perangkat wemos D1 pada gambar 4.12.



Gambar 4.12. Instalasi perangkat wemos D1

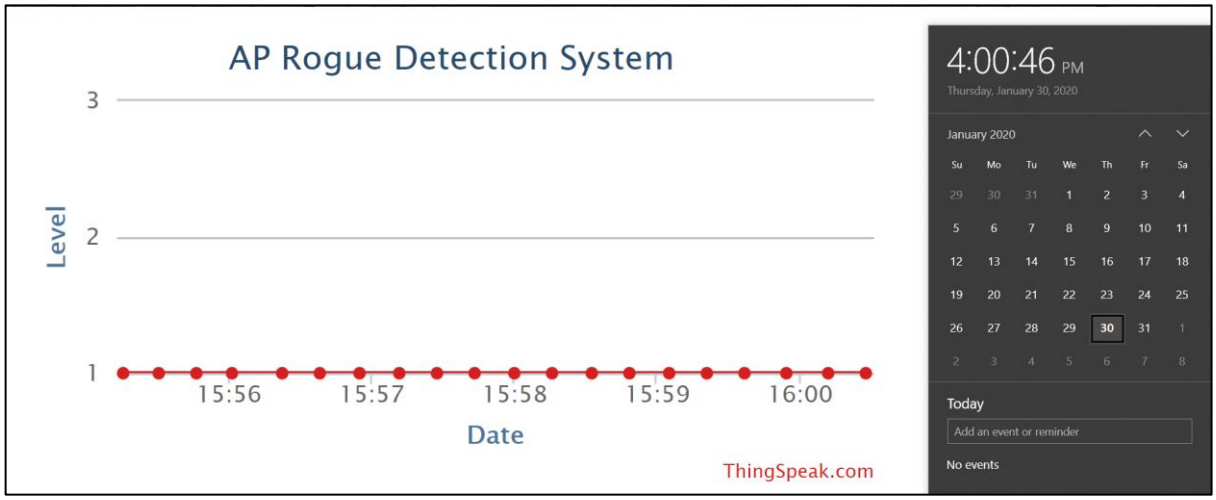
Sebelum tombol CONNECT ditekan pastikan terlebih dahulu bahwa perangkat wemos dan AP yang akan dilindungi sudah bersebelahan atau menempel, seperti Gambar 4.13. Ketika tombol ditekan maka perangkat wemos d1 akan mencoba terhubung ke AP yang data-datanya diisi pada form. Jika proses tersebut berhasil, maka akan ditandakan dengan lampu indikator pada modul ESP8266 mati dan masuk pada mode *detecting*. Pada mode *detecting* ini perangkat wemos d1 akan melakukan *scanning* secara otomatis dan mengecek pada kondisi-kondisi yang sudah di jabarkan pada proses perancangan sebelumnya, kemudian mengirimkan data tersebut ke server IoT untuk dilihat oleh pengguna atau administrator jaringan. Kondisi-kondisi yang ada dapat dilihat pada Tabel 3.1. dan berikut tampak perangkat wemos d1 jika sudah berhasil dalam proses penginstalasiannya atau masuk pada mode *detecting*.



Gambar 4.13. Perangkat wemos mode *detecting*

4.2.5 Pengujian *Monitoring* pesan yang dikirimkan

Setelah berhasil mendaftarkan AP yang akan dilindung maka proses *scanning* dilakukan selama perangkat wemos menyala dan mengirimkan data hasil pendeteksiannya ke server IoT *thinkspeak.com*. Berikut tampilan hasil pendeteksian yang dilakukan pada Gambar 4.14.



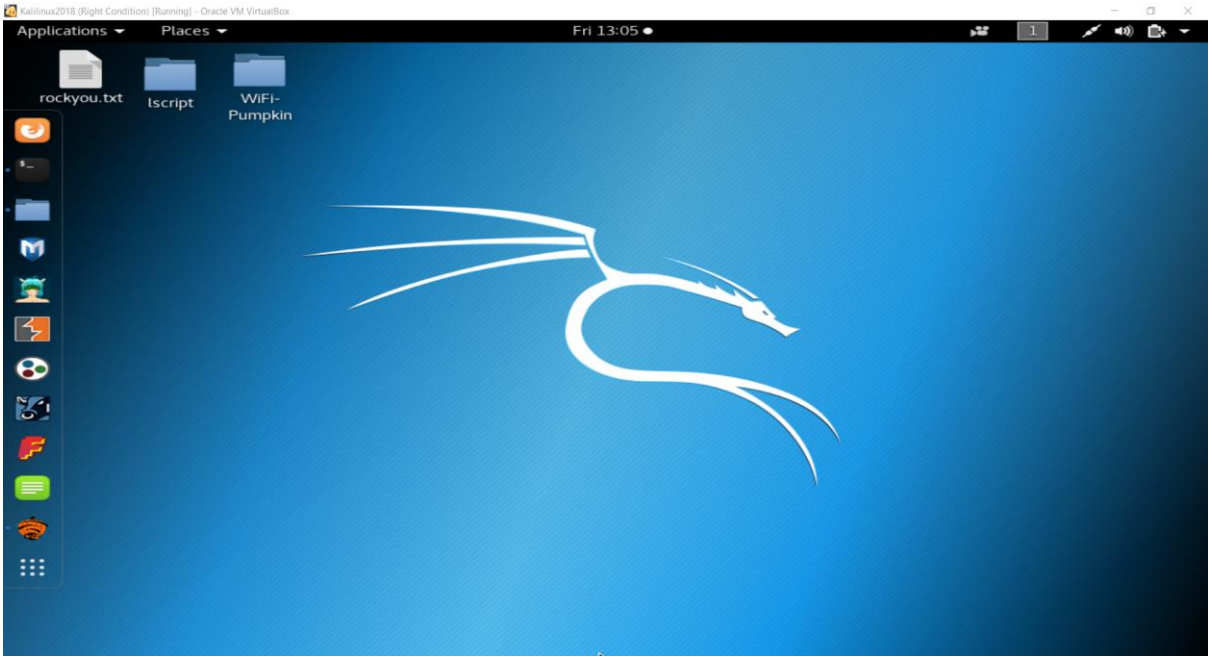
Gambar 4.14. Hasil pendeteksian *realtime*

4.2.6 Pengujian Simulasi Penyerangan RAP

Dalam percobaan perangkat atau sistem deteksi serangan RAP, perlu adanya dilakukan simulasi serangan RAP untuk mengetahui seberapa akuratnya sistem yang dibangun dalam mendeteksi serangan RAP yang dilakukan berdasarkan kondisi-kondisi pada Tabel 3.1. Dalam pengujian simulasi ini akan dilakukan suatu serangan RAP dengan 2 metode yang berbeda. Dalam simulasi penyerangan menggunakan *tool* yang terdapat pada sistem operasi Kali Linux, yaitu WiFi-Pumpkin. Berikut persiapan yang dilakukan dalam simulasi penyerangan RAP:

a. Persiapan Kali Linux

Untuk menggunakan tool yang akan digunakan yaitu WiFi-Pumpkin, siapkan terlebih dahulu sistem operasi Kali Linux, dalam pengujian yang dilakukan menggunakan virtual box untuk menjalankan sistem operasi Kali Linux. Berikut tampilan Kali Linux yang digunakan pada Gambar 4.15.

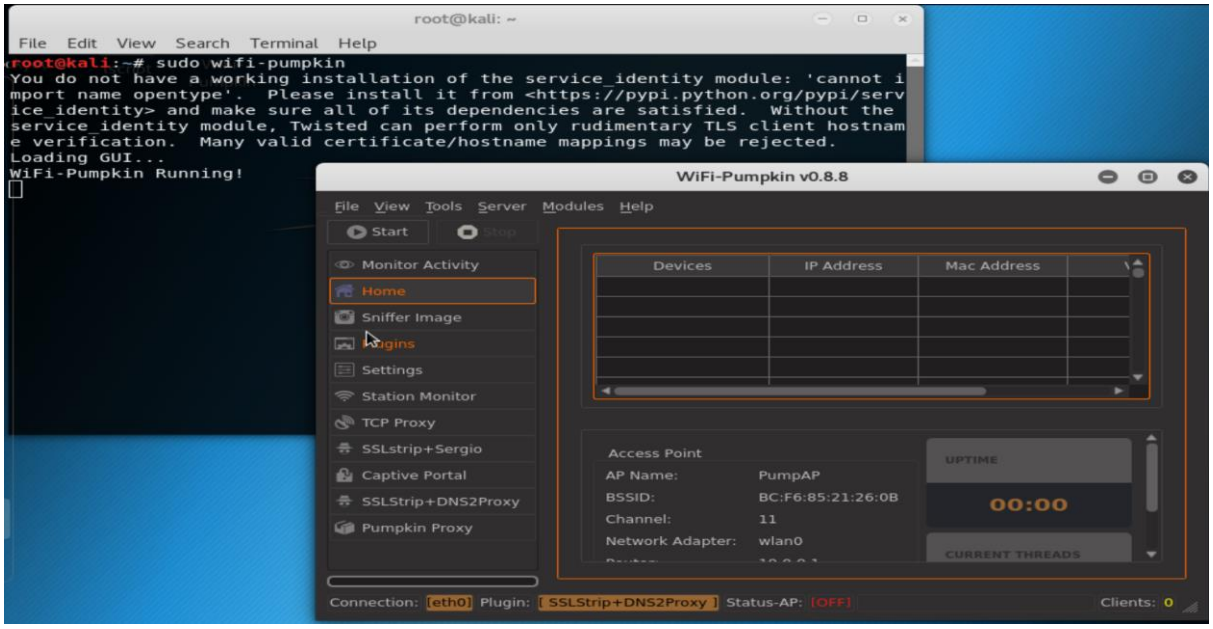


Gambar 4.15. Tampilan sistem operasi Kali Linux

b. Persiapan *tool* WiFi-Pumpkin

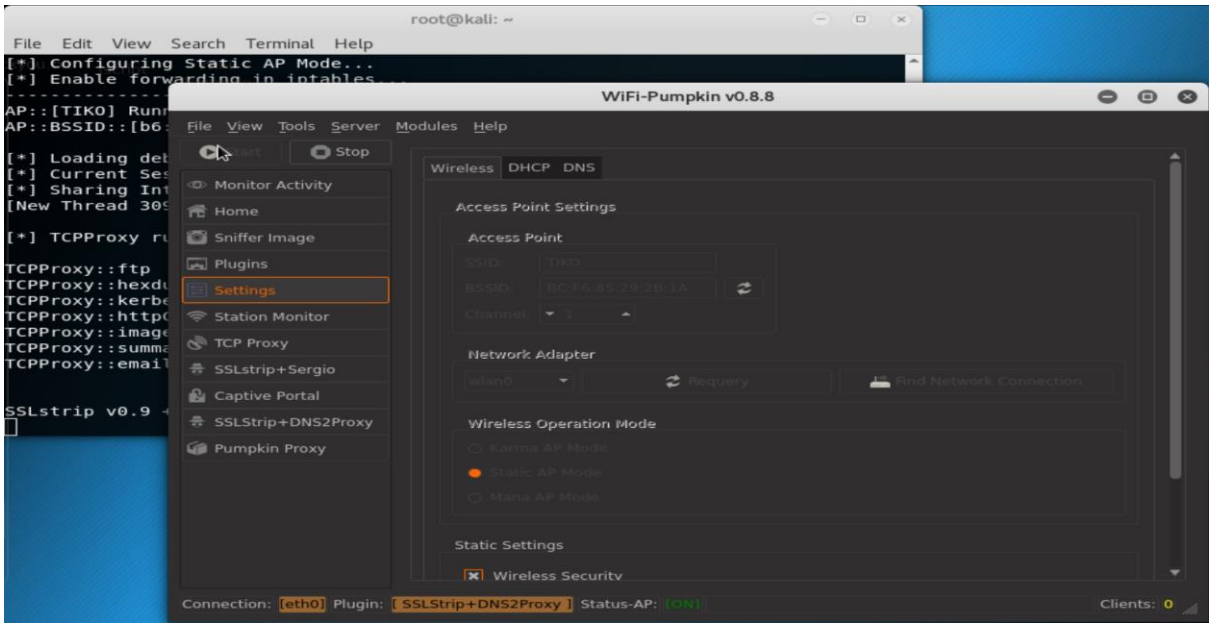
Dalam pengujian simulasi penyerangan RAP ini membutuhkan suatu *tool* yang ada pada sistem operasi Kali Linux sebagai media membuat AP palsu untuk menguji apakah sistem yang dibangun bekerja sesuai tujuan perancang atau tidak. WiFi-Pumpkin ini dapat digunakan untuk membuat AP palsu yang diinginkan oleh pengguna dengan membuat SSID, MAC Address, dan *channel frequensi* yang diinginkan. Berikut tahapan yang dapat dilakukan untuk membuat AP palsu menggunakan *tool* WiFi-Pumpkin pada Kali Linux.

1. Membuka terminal pada Kali Linux
2. Setelah terminal berhasil dibuka masukan perintah “sudo WiFi-Pumpkin”. Ketika sudah memasukkan perintah tersebut maka *tool* tersebut akan akan terbuka seperti tampilan pada Gambar 4.16.



Gambar 4.16. Tampilan *home tool* WiFi-Pumpkin

3. Untuk membuat AP palsu, masuk pada bagian *settings* kemudian masukan SSID, MAC *Address* dan *channel frequency* dari AP palsu yang akan dibuat kemudian klik tombol *start*, jika berhasil membuat AP palsu maka akan terlihat seperti pada Gambar 4.17 berikut.



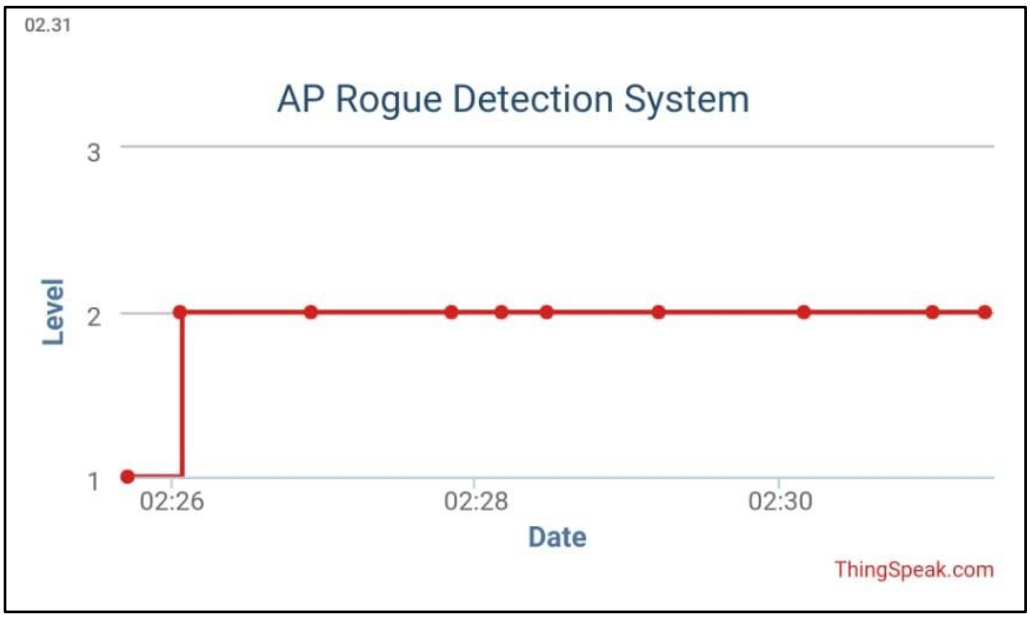
Gambar 4.17. Tampilan pembuatan AP palsu pada *tool* WiFi-Pumpkin

4. Ketika AP palsu sudah berhasil dibuat pengguna sistem yang dibangun ini dapat melihat hasil pendeteksian pada web *thinkspeak* yang di implementasi pada saat

perancangan sistem. Berikut hasil pendeteksian dari simulasi serangan RAP yang dilakukan.

- Serangan RAP dengan SSID sama dengan target.

Dalam serangan RAP metode ini sangat mudah dilakukan tanpa menggunakan tools-tools yang ada. Berikut hasil pengujian yang dilakukan pada Gambar 4.18.



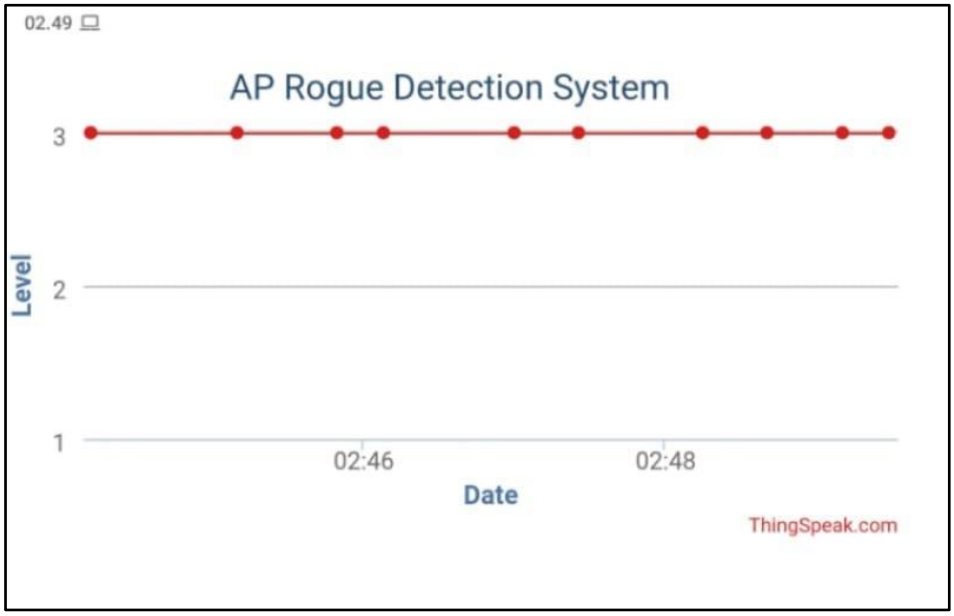
Gambar 4.18. Hasil pendeteksian serangan RAP potential

Pada tingkat potential atau Level 2 ini menggambarkan bahwa terdapat lebih dari satu SSID yang dilindungi pada area sekitar AP. Sebab dikategorikan sebagai potensial karena belum tentu serangan RAP, bisa saja AP tersebut merupakan AP yang sengaja dibuat untuk kebutuhan personal atau *password*nya berbeda dan bisa saja memang benar serangan RAP yang mencari korban dengan memalsukan AP yang aslinya.

- Serangan RAP dengan MAC Address sama tetapi SSID yang berbeda dengan target atau SSID dan MAC Address sama

Dalam serangan RAP ini sangat-sangat berbahaya, karena memanfaatkan celah atau kelemahan dari teknologi WLAN yang jika terdapat MAC Address dari AP ada yang sama maka pengguna yang ingin *connect* atau menyambungkannya akan bingung, terkadang yang terdeteksi pada PC adalah koneksi AP yang asli dan terkadang AP yang palsu. Ketika terjadi serangan dengan kategori ini maka akan dideteksi oleh perangkat atau sistem yang dibangun ini dengan status Emergency atau Level 3. Berikut tampilan

grafik hasil pendeteksian simulasi serangan RAP yang memalsukan MAC Address pada Gambar 4.19.



Gambar 4.19. Hasil pendeteksi serangan RAP emergency

4.3 Analisis Penelitian

Pada tahap analisis penelitian merupakan tahapan di mana sebuah penelitian dinilai kelebihan dan kekurangannya setelah dilakukan tahap pengujian. Setiap kelebihan dan kekurangan yang didapatkan akan menjadi bahan pertimbangan bagi penulis dan pembaca untuk menilai keberhasilan dari penelitian ini. Kekurangan yang didapat dari penelitian ini juga diharapkan dapat menjadi pertimbangan untuk mengembangkan penelitian yang lebih sempurna di masa mendatang. Kelebihan dari penelitian yang akan menjadi nilai tambah dari penelitian ini adalah:

4.3.1 Kelebihan Sistem

a. Sistem dapat dikonfigurasi pada berbagai macam AP.
b. Proses instalasi yang mudah dan aman.
c. Sistem dapat mendeteksi serangan RAP secara *realtime* dari serangan RAP yang meniru SSID hingga serangan RAP yang meniru MAC *Address* dari AP targetnya yang dapat dilihat pada Gambar 4.18 dan 4.19 di atas.
d. Setelah dilakukan pengujian sistem ini mempunyai keakuratan pendeteksian yang cukup tinggi karena pada saat proses pengujian yang dilakukan sistem yang dibangun mampu

memberikan hasil pendeteksian yang ditampilkan pada web *thinkspeak* setiap menitnya yang dapat dilihat pada Gambar 4.18 dan Gambar 4.19 di atas.

4.3.2 Kekurangan Sistem

a. Sistem atau perangkat Wemos D1 harus diletakan menempel atau bersampingan dengan AP yang akan dilindungi.
b. Sistem ini hanya dapat melindungi AP yang terhubung dengan internet.
c. Sistem ini hanya dapat melindung AP dengan jenis keamanannya selain dari WPA2-Enterprise.
d. Sistem ini hanya dapat melindung 1 AP pada suatu waktu yang sama.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan perancangan dan juga pengujian yang sudah dilakukan pada penelitian ini, didapatkan kesimpulan bahwa penelitian ini yaitu:

a. Sistem deteksi serangan RAP mampu mendeteksi serangan RAP secara *realtime* dengan hasil pendeteksian sesuai dengan tingkatan klasifikasi serangan RAP.
b. Sistem deteksi serangan RAP diimplementasikan pada perangkat mikrokontroller wemos d1 yang dilengkapi dengan modul esp8266.
c. Sistem pendeteksi membuat AP sementara sebagai mediator instalasi sistem agar sistem yang dibangun lebih efektif dalam melakukan pendaftaran AP yang akan dilindungi.
d. Hasil pendeteksian dapat dipantau melalui server IoT *thinkspeak* dalam bentuk grafik.

5.2 Saran

Akibat adanya kekurangan dan kelebihan pada sistem ini, maka diharapkan pada penelitian selanjutnya dapat menghilangkan atau menutupi kekurangan yang ada pada sistem ini. Bahkan sangat diharapkan penelitian selanjutnya dapat mengembangkan sistem ini lebih baik lagi. Saran untuk pengembangan sistem pada penelitian selanjutnya adalah sebagai berikut:

a. Pengembangan agar dapat melindungi AP dengan pengamanan WPA2-Enterprise.
b. Peningkatan perangkat keras agar dapat mendeteksi dengan cakupan yang lebih luas.

DAFTAR PUSTAKA

Aarthy Devi, A., Mohan, A. K., & Sethumadhavan, M. (2017). Wireless Security Auditing: Attack Vectors and Mitigation Strategies. Procedia Komputer Science, 115, 674–682. https://doi.org/10.1016/j.procs.2017.09.153

Abdillah, M. F. (2011). Pengembangan Dan Uji Kinerja Sistem Pendeteksi *Rogue Access Point* Menggunakan Aplikasi Berbasis Web.

Agrawal, N., & Tapaswi, S. (2015). Wireless *Rogue Access Point* Detection Using Shadow Honeynet. Wireless Personal Communications, 83(1), 551–570. https://doi.org/10.1007/s11277-015-2408-0

Bonardo, G. S. (n.d.). Implementasi Sistem Penanganan *Rogue Access Point* Model Bridging Connection Pada Fakultas Ilmu Terapan Universitas Telkom.

Dahiya, M., & Gill, S. (2017). Detection of rogue access point in WLAN using Hopfield Neural Network. International Journal of Electrical and Komputer Engineering, 7(2), 1060–1070. https://doi.org/10.11591/ijece.v7i2.pp1060-1070

Firdana, A. C., Munadi, R., & Widodo, R. B. C. (2012). Analisis Qos (Quality of Service) Layanan Video , Packet Data Dan Voice Pada Jaringan Ip Berbasis Wimax Studi Kasus Di Wilayah Bandung Timur.

Han, H., Sheng, B., Tan, C., Li, Q., & Lu, S. (2018). A Timing-Based Scheme for Rogue AP Detection. MobiCom, 11, 104–115. Retrieved from https://pdfs.semanticscholar.org/1dd9/786e51dd4fbe5df185f4a6ae3e1d70113207.pdf

Masiukiewicz, A., Tarykin, V., & Podvornyi, V. (2016). Security Threats in Wi-Fi Networks. International Research Journal of Advanced Engineering and Science, 1(3), 6–11.

Nugroho, A. A. (2013). Implementasi Aplikasi Berbasis Web Sebagai Sistem Pendeteksi *Rogue Access Point* Dengan Wired-Side Solution. 00, 1–12.

Oktavianti, G. (2019). DAMPAK PEMANFAATAN PERANGKAT TELEKOMUNIKASI, INTERNET, DAN TEKNOLOGI NIRKABEL PADA GITA BUSANA. (43217120060), 1–28.

Prakoso, I. P. (2018). Voice Controlled Home Automation System. Journal of Network Communications and Emerging Technologies (JNCET), 8(4), 317–319.

Proxim. (2004). ROGUE ACCESS POINT DETECTION : AUTOMATICALLY DETECT AND MANAGE Common Approaches to Rogue AP Detection. The Rouge Access Point Problem, 1–7.

Raschellà, A., Bouhafs, F., Mackay, M., Shi, Q., Ortín, J., Gállego, J. R., & Canales, M. (2020). A dynamic access point allocation algorithm for dense wireless LANs using potential game. Komputer Networks, 167. https://doi.org/10.1016/j.comnet.2019.106991

Sasmoko, D., & Arie, M. (2017). Rancang Bangun Sistem Pendeteksi Kebakaran Berbasis IoT dan SMS Gateway Menggunakan Arduino. Jurnal SIMETRIS, Vol 8, 470-475.

Shourbaji, I. Al, & AlAmeer, R. (2013). Wireless Intrusion Detection Systems(WIDS). Retrieved from http://arxiv.org/abs/1302.6274

Syafruddin. (2016). Indonesia tertinggi kedua kejahatan siber di dunia. Retrieved from kominfo: https://kominfo.go.id/content/detail/13487/polri-indonesia-tertinggi-kedua-kejahatan-siber-di-dunia/0/sorotan_media

Syahrulah, F., Bhawiyuga, A., & Data, M. (2018). Implementasi Sistem Pendeteksi *Rogue Access Point* Dengan Metode Perhitungan Nilai *Round Trip Time*. Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer (J-PTIIK) Universitas Brawijaya, 2(12), 7367–7373.

Juwariyah, T., Prayitno, S., & Mardhiya, A. (2018). Perancangan Sistem Deteksi Dini Pencegah Kebakaran Rumah Berbasis IoT. Seminar Nasional Informatika, Sistem Informasi Dan Keamanan Siber, 57-61.

LAMPIRAN

A. Source Code Arduino IDE

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
#include <FS.h>
#include <ArduinoJson.h>

char* mySsid = "APRogueDetectionSystem";
char* password = "12345678";

const char* www_username = "admin";
const char* www_password = "admin";
// allows you to set the realm of authentication Default:"Login Required"
const char* www_realm = "Custom Auth Realm";
// the Content of the HTML response in case of Unautherized Access Default:empty
String authFailResponse = "Authentication Failed";

const char* htmlfile = "/index.html";

const int ledPin =  LED_BUILTIN;

IPAddress lokal_ip(192, 168, 100, 1);
IPAddress gateway(192, 168, 100, 100);
IPAddress netmask(255, 255, 255, 0);

WiFiClient client;

ESP8266WebServer server(80);

const char* serveriot = "api.thingspeak.com";
String apiKey = "L4GK9PWDSMSGWLWA";

String index() {
  String name = "";
  name += "<html>";
  name += "<head>";
  name += "<title>AP Rogoue Detection System</title>";
```

```
  name += "<link rel='stylesheet' href='style.css'>";
  name += "</head>";
  name += "<body>";
  name += "<div class='container'>";
  name += "<div class='row'>";
  name += "<div class='col-12'>";
  name += "<div id='error' class='hide'></div>";
  name += "<h1 class='header' data-translate='warning'>WELCOME</h1>";
  name += "<p class='centered bold' data-translate='disclaimer'>";
  name += "This is a system for scan AP Rogue in your room and configuration
your device to send you a notification on the IoT server thinkspeak. Check
tab info for get the link to thinkspeak website";
  name += "</p>";
  name += "<p class='centered bold'>";
  name += "<a class='button' href='index.html' data-translate='disclaimer-
button'>Let's Use</a>";
  name += "</p>";
  name += "</div></div></div>";
  name += "</body>";
  name += "</html>";

  return name;
}

void handleRoot() {
  server.sendHeader("Location", "/login", true);  //Redirect to our html
web page
  server.send(302, "text/plane", "");
}

bool loadFromSpiffs(String path) {
  String dataType = "text/plain";
  if (path.endsWith("/")) path += "/index.html";

  if      (path.endsWith(".src"))      path      =      path.substring(0,
path.lastIndexOf("."));
  else if (path.endsWith(".html")) dataType = "text/html";
  else if (path.endsWith(".css")) dataType = "text/css";
  else if (path.endsWith(".js")) dataType = "application/javascript";
  else if (path.endsWith(".xml")) dataType = "text/xml";
```

```
  File dataFile = SPIFFS.open(path.c_str(), "r");
  if (server.hasArg("download")) dataType = "application/octet-stream";
  if (server.streamFile(dataFile, dataType) != dataFile.size()) {
  }
  dataFile.close();
  return true;
}

String scanWifi() {

  WiFi.scanNetworks(false, true);
  int n = WiFi.scanComplete();
  String name = "";
  name += "<h2><span>Access Points : ";
  name += n;
  name += "</span></h2>";
  name += "<table id='apTable'>";
  name += "<tr>";
  name +=  "<th class='ssid'>SSID</th>";
  name +=  "<th class='channel'>Channel</th>";
  name +=  "<th class='rssi'>RSSI</th>";
  name +=  "<th class='enc'>Encscryption Type</th>";
  name +=  "<th class='mac'>MAC</th>";
  name +=  "<th class='mac'>ACTION</th>";
  name +=  "</tr>";

  for (int i = 0; i < n; ++i)
  {
    name += "<tr>";
    name += "<td class='ssid'>";
    name += WiFi.SSID(i);
    name += "</td>"; // SSID
    name += "<td class='channel'>";
    name += WiFi.channel(i);
    name += "</td>";
    name   +=   "<td   class='rssi'><div   class='meter_background'>   <div
class='meter_forground'><div class='meter_value'>";
    name += WiFi.RSSI(i);
    name += "</div></div> </div></td>";  // RSSI
    name += "<td class='enc'>";
```

```
    name += WiFi.encryptionType(i) == ENC_TYPE_NONE ? " Unsecured" : "
Secured";
    name += "</td>"; // ENC
    name += "<td class='mac'>";
    name += WiFi.BSSIDstr(i);
    name += "</td>"; // MAC
    name += "<td class='action'>";
    name        +=        "<button        onclick='tambah()'        data-
translate='reload'>ADD</button>";
    name += "</td>"; // ADD to Form
    name += "</tr>";
  }
  name += "</table>";
  return name;
}

void handleWebRequests() {
  if (loadFromSpiffs(server.uri())) return;
  String message = "File Not Detected\n\n";
  message += "URI: ";
  message += server.uri();
  message += "\nMethod: ";
  message += (server.method() == HTTP_GET) ? "GET" : "POST";
  message += "\nArguments: ";
  message += server.args();
  message += "\n";
  for (uint8_t i = 0; i < server.args(); i++) {
    message += " NAME:" + server.argName(i) + "\n VALUE:" + server.arg(i)
+ "\n";
  }
  server.send(404, "text/plain", message);
  Serial.println(message);
}

void setup() {
  delay(1000);
  Serial.begin(115200);
  Serial.println();

  //Initialize File System
```

```
  SPIFFS.begin();
  Serial.println("File System Initialized");

  wifiConnect();

  server.on("/settings", HTTP_POST, handleSettingsUpdate);
}

void loop() {
  if (WiFi.status() == WL_CONNECTED) {
    String postStr = apiKey;
    int network;
    int i = 0;
    bool normal = true;
    if (client.connect(serveriot, 80))
    {
      if (SPIFFS.exists("/config.json")) {
        const char * _ssid = "", * _mac = "", * _channel = "";
        File configFile = SPIFFS.open("/config.json", "r");
        if (configFile) {
          size_t size = configFile.size();
          std::unique_ptr<char[]> buf(new char[size]);
          configFile.readBytes(buf.get(), size);
          configFile.close();

          DynamicJsonBuffer jsonBuffer;
          JsonObject& jObject = jsonBuffer.parseObject(buf.get());
          if (jObject.success())
          {
            _ssid = jObject["ssid"];
            _mac = jObject["mac"];
            _channel = jObject["channel"];
            WiFi.scanNetworks(false, true);
            network = WiFi.scanComplete();
            if (network <= 0) {
              WiFi.scanNetworks();
              network = WiFi.scanComplete();
            } else {
              int ssidCounting = 0;
              int macCounting = 0;
```

```
              for (int i = 0; i < network; i++) {
                if (WiFi.SSID(i) == _ssid) {
                  ssidCounting++;
                  if (WiFi.BSSIDstr(i) == _mac) {
                    if (String(WiFi.channel(i)) != _channel) {
                      macCounting++;
                    }
                    if (int(WiFi.RSSI(i)) < -48) {
                      macCounting +=  2;
                    }
                  }
                } else if (WiFi.SSID(i) != _ssid) {
                  if (WiFi.BSSIDstr(i) == _mac) {
                    if (String(WiFi.channel(i)) != _channel) {
                      macCounting++;
                    }
                    if (int(WiFi.RSSI(i)) < -48) {
                      macCounting +=  2;
                    }
                  }
                }
              }
              if (macCounting > 1) {
                postStr += "&field1=";
                postStr += "3";
              } else if (ssidCounting > 1) {
                postStr += "&field1=";
                postStr += "2";
              } else {
                postStr += "&field1=";
                postStr += "1";
              }
              postStr += "\r\n\r\n";
              client.print("POST /update HTTP/1.1\n");
              client.print("Host: api.thingspeak.com\n");
              client.print("Connection: close\n");
              client.print("X-THINGSPEAKAPIKEY: " + apiKey + "\n");
              client.print("Content-Type:         application/x-www-form-
urlencoded\n");
              client.print("Content-Length: ");
```

```
              client.print(postStr.length());
              client.print("\n\n");
              client.print(postStr);
              Serial.println(postStr);
            }
            client.stop();
          }
        }
      }
    }
  } else {
    server.handleClient();
  }
}

void handleSettingsUpdate()
{
  String data = server.arg("plain");
  DynamicJsonBuffer jBuffer;
  JsonObject& jObject = jBuffer.parseObject(data);

  File configFile = SPIFFS.open("/config.json", "w");
  jObject.printTo(configFile);
  configFile.close();

  server.send(200, "application/json", "{\"status\" : \"ok\"}");
  delay(500);

  wifiConnect();
}

void wifiConnect()
{
  //reset networking
  WiFi.softAPdisconnect(true);
  WiFi.disconnect();
  delay(1000);
  //check for stored credentials
  if (SPIFFS.exists("/config.json")) {
    const char * _ssid = "", *_pass = "";
```

```
    File configFile = SPIFFS.open("/config.json", "r");
    if (configFile) {
      size_t size = configFile.size();
      std::unique_ptr<char[]> buf(new char[size]);
      configFile.readBytes(buf.get(), size);
      configFile.close();

      DynamicJsonBuffer jsonBuffer;
      JsonObject& jObject = jsonBuffer.parseObject(buf.get());
      if (jObject.success())
      {
        _ssid = jObject["ssid"];
        _pass = jObject["password"];
        WiFi.mode(WIFI_STA);
        WiFi.begin(_ssid, _pass);
        unsigned long startTime = millis();
        while (WiFi.status() != WL_CONNECTED)
        {
          delay(500);
          Serial.print(".");
          if ((unsigned long)(millis() - startTime) >= 10000) break;
        }
        Serial.println("");
        WiFi.printDiag(Serial);
        Serial.print("Connected to ");
        Serial.println(_ssid);
        Serial.print("Ip address: ");
        Serial.println(WiFi.lokalIP());
        pinMode(ledPin, LOW);
      }
    }
  }

  if (WiFi.status() == WL_CONNECTED)
  {

  } else
  {
    pinMode(ledPin, HIGH);
    WiFi.mode(WIFI_AP);
```

```
    WiFi.softAPConfig(lokal_ip, gateway, netmask);
    WiFi.softAP(mySsid, password);
    Serial.println("");

    WiFi.printDiag(Serial);

    server.on("/", handleRoot);
    server.on("/login", []() {
      if (!server.authenticate(www_username, www_password)) {
        return    server.requestAuthentication(DIGEST_AUTH,    www_realm,
authFailResponse);
      }
      else {
        server.onNotFound(handleWebRequests); //Set server all paths are
not found so we can handle as per URI
        server.send(301, "text/html", index());
        server.on("/scan", []() {
          server.send(200, "text/plain", scanWifi());
        });
      }
    });
    server.begin();
  }
}
```

B. Source code pada web server lokal menggunakan SPIFFS

1. File index.html

```
<!Doctype html>
<html>
   <head>
      <meta charset="utf-8">
      <meta    name="viewport"    content="width=device-width,    initial-
scale=0.8, minimal-ui">
      <meta name="description" content="ESP8266 Deauther">
      <meta name="author" content="Spacehuhn - Stefan Kremser">
      <meta name="theme-color" content="#36393E" />
      <title>AP Rogoue Detection System</title>
      <link rel="stylesheet" href="style.css">
      <script src="js/jquery.js"></script>
```

```
   </head>
   <body>
      <nav>
         <a href="index.html" data-translate="scan">Network</a>
         <a class="right" href="info.html" data-translate="info">Info</a>
      </nav>
      <form>
         <div class="container" style="width: 50%;">
            <div class="row">
               <div class="col-12">
                  <div id="error" class="hide"></div>
                  <h1 class="header" data-translate="ssids">Regist Your
Device</h1>
               </div>
            </div>
            <div class="row">
               <div class="col-6">
                  <label for="ssid">SSID</label>
               </div>
               <div class="col-6">
                  <input  type="text"  value=""  id="ssid"  name="ssid"
placeholder="SSID" maxlength="32">
               </div>
            </div>
            <div class="row">
               <div class="col-6">
                  <label>Password</label>
               </div>
               <div class="col-6">
                  <input  type="password"  id="password"  name="password"
placeholder="Password" maxlength="16"><br>
               </div>
            </div>
            <div class="row">
               <div class="col-6">
                  <label>MAC Address</label>
               </div>
               <div class="col-6">
```

```
                    <input      type="text"      id="mac"      name="mac"
placeholder="XX:XX:XX:XX:XX:XX" maxlength="32">
                </div>
            </div>
            <div class="row">
                <div class="col-6">
                    <label>Channel</label>
                </div>
                <div class="col-6">
                    <input   type="number"   id="channel"   name="channel"
placeholder="1 - 14"><br>
                </div>
            </div>
            <div class="row">
                <div class="col-12">
                    <button  class="primary"  id="savebtn"  type="button"
onclick="myFunction()" style="float: right;">CONNECT</button>
                </div>
            </div>
        </div>
  </form>
      <div class="container">
          <div class="row">
              <div class="col-11">
                  <div id="error" class="hide"></div>
                  <h1   class="header"   data-translate="scan">AP   Rogoue
Detection System</h1>
                  <button  onclick="scanAP()"  data-translate="reload">Scan
APs</button>
              </div>
          </div>
          <div class="row">
              <div class="col-13"></div>
          </div>
      </div>
      <script type="text/javascript">
          function scanAP(){
              $(".col-13").html("");
              $.get("/scan", function(data){
                  $(".col-13").html(data);
```

```
                $(".col-13 #apTable .meter_value").each(function(){
                    var color="";
                    var width=parseInt($(this).html())+130;
                    if(width < 50) color = "meter_red";
                    else if(width < 70) color = "meter_orange";
                    else color = "meter_green";
                    $(this).parent().addClass(color).css("width",
width+"%");
                });
            });
        }

        function myFunction()
        {
          console.log("button was clicked!");
          var ssid = document.getElementById("ssid").value;
          var password = document.getElementById("password").value;
          var mac = document.getElementById("mac").value;
          var channel = document.getElementById("channel").value;
          var   data   =   {ssid:ssid,   password:password,   mac:mac,
channel:channel};
          var xhr = new XMLHttpRequest();
          var url = "/settings";
          xhr.onreadystatechange = function() {
            if (this.readyState == 4 && this.status == 200) {
              // Typical action to be performed when the document is
ready:
              if(xhr.responseText != null){
                console.log(xhr.responseText);
              }
            }
          };
          xhr.open("POST", url, true);
          xhr.send(JSON.stringify(data));
        };

        $('body').on('click', '.action button',  function(){
            console.log($(this).parent().parent().html());
            var ssid = $(this).parent().parent().find('.ssid').html();
            var mac = $(this).parent().parent().find('.mac').html();
```

```
            var                          channel                          =
$(this).parent().parent().find('.channel').html();
            $('#ssid').val(ssid);
            $('#mac').val(mac);
            $('#channel').val(channel);
        });

    </script>
  </body>
</html>
```

2. File info.html

```
<!Doctype html>
<html>
  <head>
    <title>AP Rogoue Detection System</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <nav>
      <a href="index.html" data-translate="scan">Network</a>
      <a class="right" href="info.html" data-translate="info">Info</a>
    </nav>
    <div class="container">
      <div class="row">
        <div class="col-12">
          <div id="error" class="hide"></div>
          <h1 class="header" data-translate="info">Device Data</h1>

          <h2>Website for *Monitoring* Result of Detection</h2>
          <p>
            Website : <a target="_blank"
href="https://thingspeak.com/channels/873234/charts/1?bgcolor=%23ffffff&c
olor=%23d62020&days=1&dynamic=true&max=3&min=1&results=10&type=step&yaxis
max=3&yaxismin=1">Click Here!</a><br>
            Author or User ID : jmanz18 <br>
            Channel ID : 873234 <br>
          <p>
          <h2>For Information Contact us</h2>
```

```
                <p>
                    Instagram:             <a             target="_blank"
href="https://www.instagram.com/wiman18_/">@wiman18_</a><br/>
                    Twitter:              <a             target="_blank"
href="https://twitter.com/washayatuliman">Washayatul Iman</a><br/>
                </p>

   <br><br><br><br><br><br><br><br><br><br><br><br><br><br><br>
                <div id="copyright">
                    Version 1.0<br>
                    Copyright (c) 2019 Washayatul Iman<br>
                </div>
            </div>
         </div>
      </div>
   </body>
</html>
```

3. File style.css

```
/* Global */
body {
    background: #36393e;
    color: #bfbfbf;
    font-family: sans-serif;
    margin: 0;
}

h1 {
    font-size: 1.7rem;
     margin-top: 1rem;
     background: #2f3136;
     color: #bfbfbb;
     padding: 0.2em 1em;
     border-radius: 3px;
     border-left: solid #4974a9 5px;
     font-weight: 100;
     text-align: center;
}
```

```
h2 {
    font-size: 1.1rem;
     margin-top: 1rem;
     background: #2f3136;
     color: #bfbfbb;
     padding: 0.4em 1.8em;
     border-radius: 3px;
     border-left: solid #4974a9 5px;
     font-weight: 100;
}

table{
    border-collapse: collapse;
}

label{
    line-height: 38px;
}

p{
    margin: 0.5em 0;
}

.left {
    float: left;
}
.right {
    float: right;
}

.bold {
    font-weight: bold;
}
.red{
    color: #F04747;
}
.green{
    color:#43B581;
}
.clear {
```

```
    clear: both;
}
.centered{
    text-align: center;
}
.select{
    width: 98px !important;
    padding: 0 !important;
}
.selected{
    background: #4974a9;
}
.status{
    width: 120px;
    padding-left: 8px;
}
.labelFix {
    line-height: 44px;
}
.clickable{
    cursor: pointer;
}
.settingName{
    text-transform: uppercase;
    font-weight: bold;
    text-decoration: underline;
}

#error {
     text-align: center;
     color: #fff;
     background: #af3535;
     border-radius: 5px;
    padding: 10px;
    margin-top: 10px;
}

#closeError{
    float: right;
    color: #fff;
```

```
    padding: 0px 10px;
    cursor: pointer;
}

#copyright{
    font-size: 0.95em;
    text-align: center;
    margin-top: 3em;
    margin-bottom: 3em;
}

/* ===== CHECKBOX ===== */
/* Customize the label (the container) */
.checkBoxContainer {
  display: block;
  position: relative;
  padding-left: 35px;
  margin-bottom: 12px;
  cursor: pointer;
  font-size: 22px;
  -webkit-user-select: none;
  -moz-user-select: none;
  -ms-user-select: none;
  user-select: none;
  height: 32px;
}

/* Hide the browser's default checkbox */
.checkBoxContainer input {
  position: absolute;
  opacity: 0;
  cursor: pointer;
}

/* Create a custom checkbox */
.checkmark {
  position: absolute;
  top: 8px;
  left: 0;
  height: 28px;
```

```
  width: 28px;
  background-color: #2F3136;
  border-radius: 4px;
}

/* Create the checkmark/indicator (hidden when not checked) */
.checkmark:after {
  content: "";
  position: absolute;
  display: none;
}

/* Show the checkmark when checked */
.checkBoxContainer input:checked ~ .checkmark:after {
  display: block;
}


.checkBoxContainer .checkmark:after {
  left: 10px;
  top: 7px;
  width: 4px;
  height: 10px;
  border: solid white;
  border-width: 0 3px 3px 0;
  -webkit-transform: rotate(45deg);
  -ms-transform: rotate(45deg);
  transform: rotate(45deg);
}

/* ERROR */
.hide {
   display: none;
}

.show {
   display: block !important;
   animation-name: fadeIn;
   animation-duration: 1s;
}
```

```
@keyframes fadeIn {
    0%   {opacity: 0;}
    100% {opacity: 1;}
}


hr {
    background: #3e4146;
}

a {
    color: #5281bb;
     text-decoration: none;
}

a:hover {
    color: #95b8e4;
}

li{
    margin: 4px 0;
}

/* Meter */
.meter_background{
    background: #42464D;
    width: 100%;
    word-break: normal;
    min-width: 100px;
}
.meter_forground{
    color: #fff;
    padding: 4px 0;
    /* + one of the colors below
    (width will be set by the JS) */
}
.meter_green{
    background: #43B581;
}
```

```
.meter_orange{
   background: #FAA61A;
}
.meter_red{
   background: #F04747;
}
.meter_value{
   padding-left: 8px;
}



/* Tables */
table {
   width: 100%;
   min-width: 400px;
   margin-bottom: 2em;
}

td{
   word-break: break-all;
}

td.ssid{
   text-align: left;
}

td.mac{
   text-transform: uppercase;
}

th{
   word-break: break-word;

}

th, td {
    padding: 10px 6px;
    border-bottom: 1px solid #5d5d5d;
    text-align: center;
}
```

```
@media screen and (max-width: 820px) {
    #apTable .id,
    #apTable .enc,
    #apTable .mac,
    #apTable .vendor,
    #apTable .name,

    #stTable .id,
    #stTable .pkts,
    #stTable .lastseen,
    #stTable .mac,

    #nTable .id,
    #nTable .vendor,
    #nTable .AP,
    #nTable .mac,

    #ssidTable .id   {
      display: none;
    }

    .meter_background{
       min-width: 45px;
    }
}

/* Navigation bar */
nav {
    display: block;
    padding: 8px 10px;
    background: #2f3136;
}

nav a {
    color: #bfbfbf;
    padding: 0.5em;
    display: inline-block;
    text-decoration: none;
}
```

```
nav a:hover{
    background: #36393f;
    color:#cecece;
    border-radius: 4px;
}

/* Inputs and buttons */
.upload-script,        .button,        button,        input[type="submit"],
input[type="reset"], input[type="button"] {
    display: inline-block;
    height: 38px;
    padding: 0 20px;
    color:#fff;
    text-align: center;
    font-size: 11px;
    font-weight: 600;
    line-height: 38px;
    letter-spacing: .1rem;
    text-transform: uppercase;
    text-decoration: none;
    white-space: nowrap;
    background: #2f3136;
    border-radius: 4px;
    border: none;
    cursor: pointer;
    box-sizing: border-box;
}


button:hover,    input[type="submit"]:hover,    input[type="reset"]:hover,
input[type="button"]:hover {
    background: #42444a;
}

/* Forms */
input[type="email"],      input[type="number"],      input[type="search"],
input[type="text"],       input[type="tel"],       input[type="password"],
input[type="url"], textarea, select {
    height: 38px;
```

```
    padding: 6px 10px; /* The 6px vertically centers text on FF, ignored by
Webkit */
    background-color: #2f3136;
    border-radius: 4px;
    box-shadow: none;
    box-sizing: border-box;
    color: #d4d4d4;
    border: none;
    width: 15em;
}

input[type="text"], input[type="password"]{
    width: 15em;
}

.setting {
    width: 100% !important;
    max-width: 284px !important;
}

input[type="file"] {
     display: none;
}

/* ==== GRID SYSTEM ==== */
.container {
  width: 100%;
  margin-left: auto;
  margin-right: auto;
  max-width: 1140px;
}



.row {
  position: relative;
  width: 100%;
}

.row [class^="col"] {
  float: left;
```

```
  margin: 0.25rem 2%;
  min-height: 0.125rem;
}

.row::after {
    content: "";
    display: table;
    clear: both;
}

.hidden-sm {
  display: none;
}

@media only screen and (min-width: 24em) {
  .col-1 {
    width: 4.33%;
  }

  .col-2 {
    width: 12.66%;
  }

  .col-3 {
    width: 21%;
  }

  .col-4 {
    width: 29.33%;
  }

  .col-5 {
    width: 37.66%;
  }

  .col-6 {
    width: 46%;
  }

  .col-7 {
```

```
    width: 54.33%;
  }

  .col-8 {
    width: 62.66%;
  }

  .col-9 {
    width: 71%;
  }

  .col-10 {
    width: 79.33%;
  }

  .col-11, .col-12, .col-13 {
    width: 96%;
  }

  .hidden-sm {
    display: block;
  }
}
```