OTOMATISASI KARMA *ATTACK*

Shandika Dianaji Santoso[1)], M. Yusuf B. Setiadji[2)]
(1) *Rekayasa Keamanan Siber, Politeknik Siber dan Sandi Negara, shandika.dianaji@bssn.go.id*
(2) *Rekayasa Keamanan Siber, Politeknik Siber dan Sandi Negara, yusuf.setiadji@poltekssn.ac.id*

Abstrak

Wireless Fidelity (WiFi) merupakan salah satu bentuk telekomunikasi tanpa kabel yang memiliki perkembangan cukup pesat. WiFi dapat dijumpai di tempat-tempat umum seperti rumah makan, warung kopi, pusat perbelanjaan, dan tempat umum lainnya. Namun, kenyamanan WiFi juga disertai dengan kerawanan yang dimiliki. Komunikasi antara klien dan access point rentan terhadap man in the middle attack seperti sniffing atau Spoofing. Salah satu serangan yang menggunakan teknik Spoofing adalah KARMA attack. Dengan adanya kerawanan tersebut, maka perlu dilakukan pengujian menggunakan alat uji manual atau otomatis agar dapat menjadi rekomendasi untuk pengamanan WiFi. Penggunaan alat uji otomatis dinilai memberikan efektifitas terhadap proses uji tersebut. Pada penelitian ini dibuat purwarupa aplikasi otomatisasi KARMA attack dengan metode System Development Life Cycle (SDLC) menggunakan pendekatan waterfall. Pengujian dan analisis dilakukan untuk mengetahui kemampuan purwarupa otomatisasi KARMA attack pada kondisi tertentu. Hasil pengujian dan analisis menunjukkan bahwa purwarupa otomatisasi KARMA attack berjalan dengan baik.

Kata kunci: KARMA *attack, Spoofing, system development life cycle,* uji keamanan*, wireless fidelity*

1. PENDAHULUAN

Wireless Fidelity atau yang disebut sebagai jaringan WiFi atau 802.11 karena mencakup teknologi IEEE 802.11, merupakan teknologi nirkabel yang tersebar luas sehingga pengguna dapat terhubung hampir di seluruh tempat [1]. Salah satu kerentanan WiFi adalah rentan untuk dilakukan serangan *Spoofing*. *Spoofing* merupakan serangan yang menjadikan penyerang (*hacker*) dapat berlaku seolah-olah sebagai pihak yang sah pada sebuah sistem [2].

Salah satu serangan yang menggunakan metode *Spoofing* adalah KARMA. KARMA merupakan tools yang digunakan untuk menguji kerawanan klien *wireless* pada beberapa *layer* komunikasi [5]. KARMA memanfaatkan informasi SSID dari *probe request frame* untuk berlaku sebagai *access point* yang dicari oleh perangkat korban. Sehingga perangkat korban akan melakukan permintaan autentikasi agar terhubung dengan *access point* tersebut.

Berdasarkan ancaman keamanan yang terdapat pada jaringan *wireless*, maka diperlukan sistem keamanan yang memadai [6]. Terdapat aspek-aspek keamanan jaringan *wireless* yang harus dipenuhi, yaitu: privasi dan kerahasiaan, keutuhan data, autentikasi, ketersediaan data, dan kendali akses. Oleh karena itu, perlu dilakukan uji keamanan pada jaringan *wireless* agar dapat meningkatkan sistem keamanan yang dimiliki terutama dalam mendeteksi berbagai serangan yang menggunakan metode MITM *attack*.

Dalam melakukan uji keamanan dapat digunakan alat uji manual atau otomatis. Beberapa manfaat penggunaan alat uji otomatis adalah mengurangi beban kerja operator, kemampuan operator dapat dialihkan kepada tugas-tugas lain, pengurangan stres yang disebabkan oleh situasi, pengurangan faktor kelelahan, sistem otomatis memberikan stabilitas dalam penanganan tugas, dan sistem otomatis dapat mengurangi kemungkinan kesalahan manusia [9].

Berdasarkan permasalahan di atas, pada penelitian ini akan dibuat purwarupa aplikasi serangan KARMA yang dapat melakukan serangan secara otomatis. Aplikasi ini akan dijalankan pada *personal computer* (PC) berbasis sistem operasi Linux. Serangan yang dilakukan akan menargetkan perangkat-perangkat pengguna WiFi yang ada di sekitar aplikasi. Aplikasi akan menampilkan SSID berdasarkan informasi pada *probe request frame* yang berhasil ditangkap. Apabila telah terjadi asosiasi antara perangkat klien dengan PC yang menjalankan aplikasi KARMA, PC dapat mengetahui segala aktivitas yang dilakukan oleh perangkat klien dalam mengakses internet.

2. LANDASAN TEORI

Pada bab ini akan dijelaskan landasan teori yang berisikan tentang konsep dasar dan penelitian sebelumnya yang terkait seperti *wireless, management frame, Spoofing,* KARMA *attack.*

2.1 *Wireless Fidelity* (WiFi)

Komunikasi *wireless* telah mengalami berbagai peningkatan sejak awal diperkenalkan. Teknologi ini pertama kali diperkenalkan oleh The European Telecommunications Standard Institute (ETSI) pada tahun 1995 dengan HiperVLAN1 [1]. Kemudian pada tahun 1997, IEEE mengeluarkan standar 802.11 sebagai standar teknologi komunikasi *wireless.* Kemampuan awal teknologi 802.11 dapat menyokong

kecepatan pengiriman data 1-2 Mbps dan seiring berjalannya waktu telah mengalami berbagai peningkatan hingga kini dapat melebihi angka 300 Mbps dengan menggunakan mode *multiple input, multiple output* (MIMO). Peningkatan kemampuan ditandai dengan pengubahan standar 802.11. Berawal dari standar 802.11a pada tahun 1999 hingga 802.11ac yang dikembangkan pada tahun 2013.

Namun, peningkatan kemampuan *wireless* juga disertai dengan peningkatan permasalahannya, terutama dalam sisi keamanan [1]. Teknologi *wireless* memiliki kerawanan pada serangan *spoofing,* akses ilegal, dan *denial of service* (DOS). Selain itu, standar asli 802.11 tidak memiliki ketentuan keamanan dari sisi otentikasi pengguna, enkripsi ataupun integritas data [1]. Kemudian untuk mendukung keamanan, dikembangkan berbagai fitur keamanan seperti WEP, WPA, WPA2, dan WPA3 yang hingga kini masih digunakan.

2.2 *Management Frame*

Management frame adalah salah satu *frame* yang digunakan untuk membangun komunikasi nirkabel yang memiliki fungsi untuk mengatur pengguna untuk aktif dan non-aktif dalam jaringan [2]. Beberapa *frame* yang digunakan pada proses autentikasi dan asosiasi diantaranya adalah *beacon frame, probe request frame, probe response frame, authentication frame, association request frame, association response frame.*

2.3 *Spoofing*

Spoofing adalah metode serangan yang dapat dilakukan pada jaringan *wired* maupun *wireless* [3]. Penyerang mendapatkan akses ke dalam jaringan atau sumber daya lain dengan menyusun ulang *data frame* yang dipertukarkan antar perangkat dengan cara mengisi *field* alamat dan *identifier* milik pengguna yang sah. Sehingga penyerang dianggap sebagai pengguna yang sah pada jaringan tersebut. Alamat dan *identifier* yang dimaksudkan dapat berupa alamat IP atau MAC *address* milik pengguna.

Serangan *Spoofing* biasanya tidak dapat dideteksi oleh sistem keamanan perangkat *wireless* [4]. Sehingga metode ini sering digunakan oleh para *hacker* untuk mengawali serangan lain seperti pencurian data dan *password.* Serangan ini dilakukan dengan menggunakan *access point* (AP) palsu yang memiliki identitas (SSID) yang disamakan dengan AP legal yang lain. Hal ini memungkinkan para korban untuk terhubung dengan AP tersebut. Apabila korban telah berhasil terhubung dengan AP palsu tersebut, maka penyerang dapat memantau lalu lintas data yang melewatinya seperti pengiriman surat elektronik, riwayat penelusuran pada *browser* hingga kata sandi.

2.4 KARMA *Attack*

Karma Attacks Radio Machines Automatically (KARMA) adalah sebuah serangan *man in the middle* (MITM) yang memiliki cara kerja melalui AP palsu dan dapat melakukan *intercept* lalu lintas data yang melewatinya [5]. KARMA dapat digunakan sebagai alat penguji keamanan dari perangkat nirkabel pada berbagai lapisan. Dengan kata lain, target serangan ini adalah perangkat *client* pada komunikasi nirkabel [6].

Cara kerja KARMA adalah menangkap paket *probe request frame* yang disebarkan oleh perangkat klien untuk menemukan AP. *Probe request frame* adalah salah satu dari *management frame* yang dipertukarkan antar perangkat *client* dengan AP sebelum terjadinya proses autentikasi dan asosiasi. AP palsu yang berhasil menangkap paket ini kemudian mengubah SSID sehingga sesuai dengan informasi dari *probe request frame.* AP akan memberikan *probe response frame* dengan SSID yang pernah terhubung dengan perangkat *client.* Hal ini mengakibatkan perangkat *client* dapat terhubung dengan AP palsu karena pernah terasosiasi dengan AP yang benar.

Serangan KARMA mengakibatkan penyerang dapat memantau aktivitas internet korban melalui AP palsu tanpa disadari oleh pemilik dari perangkat tersebut. Korban tidak sadar karena perangkat korban seolah-olah sedang terhubung dengan AP yang pernah terasosiasi sebelumnya.

2.5 Airmon-ng

Airmon-ng adalah sebuah *tools* yang termasuk di dalam Aircrack-ng yang digunakan untuk memfasilitasi mode monitor untuk seluruh *driver*/*kernel* adapter jaringan pada komputer [7]. Airmon-ng merupakan *tools* berbasis *command line* yang berjalan pada sistem operasi Linux. Dengan menggunakan Airmon-ng, pengguna dapat mengubah konfigurasi adapter jaringan menjadi mode monitor agar dapat melakukan pemantauan lalu lintas data di dalam jaringan.

2.6 Dnsmasq

Dnsmasq adalah sebuah *tools* yang menyediakan layanan DHCP, TFTP, DNS, dan PXE *server* [8]. Dnsmasq menerima permintaan DNS lalu meneruskannya kepada DNS *server.* Dnsmasq juga dapat menyimpan nama *host* lokal yang tidak tersimpan pada DNS *server* global, sehingga apabila terdapat permintaan DNS lokal dapat langsung direspon oleh Dnsmasq tanpa harus meneruskan ke DNS *server* global. Selain melayani permintaan DNS, Dnsmasq juga dapat memberikan layanan DHCP *server* untuk memberikan alamat IP statis kepada perangkat jaringan. Pemberian alamat IP ini dapat dikonfigurasi melalui file konfigurasi *dnsmasq.conf* yang didapatkan setelah melakukan instalasi.

2.7 Hostapd-Mana

Hostapd-mana adalah *tool* yang dapat digunakan untuk membuat sebuah akses poin palsu, *tool* ini juga dapat digunakan untuk melakukan pelacakan, mengumpulkan kredensial dari perangkat korban dan melakukan serangan *Man in The Middle* [9]. Hostapd-mana merupakan pengembangan dari versi

sebelumnya yaitu Hostapd dengan menambahkan fitur Mana yang berfungsi untuk melakukan serangan KARMA. Hostapd-mana merupakan *tool* yang berbasis *command line* dan berjalan pada sistem operasi Linux.

2.8 Iptables

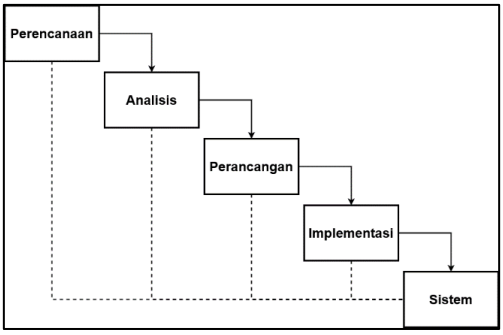
Iptables merupakan salah satu aplikasi yang digunakan untuk mengatur pemrosesan aliran data yang terhubung dengan Linux *kernel* [10]. Iptables menggunakan serangkaian aturan atau yang disebut dengan *rules* untuk mengatur pemrosesan tersebut. Iptables juga dapat digunakan untuk membuat dan mengelola aturan yang mampu untuk manipulasi paket data, pelacakan koneksi perangkat, jaringan NAT, dan pembatasan tingkat koneksi. Penggunaan Iptables untuk membuat jaringan NAT digunakan beberapa *rules* seperti *prerouting, input, output,* dan *postrouting.*

2.9 Tshark

Tshark adalah sebuah *tool* yang digunakan untuk melakukan analisis protokol jaringan [11]. Cara kerja dari Tshark adalah dengan melakukan penangkapan data dari jaringan secara langsung, atau dengan membaca paket-paket data dari file hasil penangkapan data sebelumnya. Tshark menyimpan data dari jaringan dengan format *pcap,* sehingga file dari Tshark dapat dibaca juga melalui Wireshark atau berbagai *tool* analis protokol jaringan yang lain.

3. METODE PENELITIAN

Metode penelitian yang digunakan pada penelitian ini adalah metode penelitian rancang bangun atau *System Development Life Cycle* (SDLC) yang menggunakan pendekatan *waterfall.* SDLC adalah sebuah proses yang menentukan bagaimana sebuah sistem informasi dapat mendukung kebutuhan bisnis, perancangan sistem, membangunnya, dan mengirimkannya kepada pengguna [12]. Sementara itu, pendekatan *waterfall* adalah sebuah pendekatan dalam SDLC yang memiliki empat tahap sebelum terbentuk sebuah sistem yang dibangun, keempat tahap tersebut adalah: perencanaan (*planning*), analisis (*analyze*), perancangan (*design*), dan implementasi (*implementation*). Perhatikan Gambar 1 berikut.



Gambar 1. Pendekatan *Waterfall*

3.1 Perencanaan

Pada tahap ini terdapat dua kegiatan yang akan dilakukan, yang pertama yaitu identifikasi proyek berupa pengumpulan segala kebutuhan dari aplikasi dan menentukan gambaran tentang aplikasi yang akan dibuat. Yang kedua yaitu manajemen proyek berupa pembuatan jadwal rencana kegiatan penelitian sehingga penelitian ini berjalan tepat waktu dan urut. Adapun rincian kedua kegiatan tersebut adalah sebagai berikut:

a. Identifikasi Penelitian
   Pada penelitian ini akan dibuat purwarupa aplikasi otomatisasi serangan KARMA. Aplikasi ini dibuat dengan tujuan untuk mengimplementasikan serangan KARMA pada kerawanan perangkat *WiFi* dari serangan *Spoofing*.
b. Penjadwalan Kegiatan
   Penjadwalan kegiatan dilakukan setelah penelitian diterima. Hal ini dibuat agar proyek dilaksanakan tepat waktu dan urut.

3.2 Analisis

Pada tahap analisis akan dilakukan analisa kebutuhan fungsional dan non-fungsional. Kebutuhan fungsional adalah syarat-syarat yang diperlukan untuk berfungsinya aplikasi yang akan dibuat. Sementara itu, kebutuhan non-fungsional adalah kebutuhan di luar sistem yang mendukung sistem agar tetap dapat berjalan sesuai dengan fungsinya. Kebutuhan fungsional dan non-fungsional didapatkan dari tinjauan yang dilakukan terhadap aplikasi serupa dan dilakukan perbandingan. Adapun aplikasi serupa yang dilakukan pembahasan adala FruityWiFi, Karmetasploit, dan WiFi Pinneapple. Hasil analisis tersebut ialah sebagai berikut:

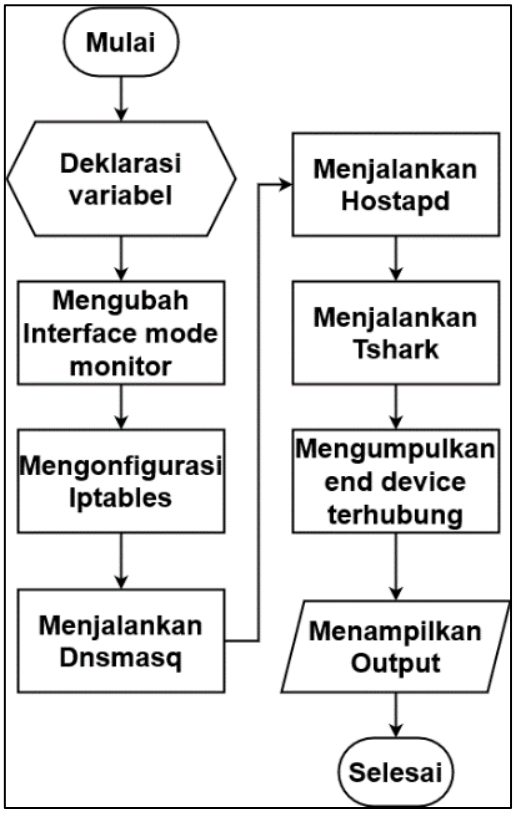
1. Kebutuhan Fungsional
   a. Aplikasi dapat menampilkan status serangan KARMA
   b. Aplikasi menggunakan modul KARMA
   c. Aplikasi dapat mengonfigurasi interface yang digunakan untuk melakukan serangan
   d. Aplikasi dapat melakukan pengaturan DHCP server
   e. Aplikasi dapat melakukan logging aktivitas client
   f. Aplikasi dapat dilakukan konfigurasi
2. Kebutuhan Non-fungsional
   a. Pemrograman GUI menggunakan Bahasa Pemrograman Python
   b. Aplikasi mampu berjalan pada sistem operasi Linux
   c. *Wireless adapter* Alfa dengan dukungan *packet injection*

3.3 Perancangan

Purwarupa aplikasi otomatisasi serangan KARMA merupakan aplikasi berbasis *graphical user interface* (GUI) yang berjalan pada sistem operasi Kali Linux. Aplikasi ini membutuhkan Aircrack-ng,

Hostapd, Tshark, dan Dnsmasq agar dapat berjalan. Empat aplikasi tersebut harus sudah terpasang pada komputer pengguna sebelum purwarupa aplikasi otomatisasi serangan KARMA dijalankan.

Aplikasi akan mengubah *network interface* menjadi mode monitor agar dapat menangkap paket data yang dipancarkan oleh perangkat korban di sekitarnya. Kemudian aplikasi akan menjalankan Hostapd-mana untuk membuat akses poin palsu, aplikasi akan menangkap *probe request frame* yang dipancarkan oleh perangkat di sekitarnya dan memberikan respon berupa *probe response frame* yang berisi SSID dari akses poin yang dicari oleh perangkat korban sehingga memancing perangkat korban untuk terhubung dengan akses poin palsu tersebut. *Iptables* akan dikonfigurasi untuk memberikan akses internet kepada perangkat yang terhubung. Segala aktifitas jaringan yang dilakukan akan dipantau menggunakan Tshark. Alur kerja aplikasi otomatisasi serangan KARMA dapat dilihat pada Gambar 2.



Gambar 2. Rancangan Otomatisasi KARMA *Attack*

3.4 Implementasi

Pada subbab ini akan dijelaskan mengenai implementasi serangan KARMA yang terotomatisasi dengan menggunakan bahasa pemrograman Python dan disesuaikan dengan kebutuhan yang telah dijelaskan pada analisis kebutuhan sistem.

Pada tahap ini, digunakan dua perangkat untuk pengembangan dan implementasi aplikasi. Adapun spesifikasi perangkat yang digunakan dapat dilihat pada Tabel 1 dan 2.

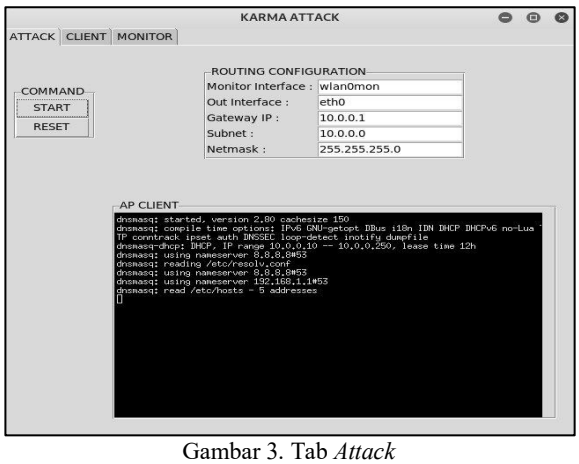
Tabel 1. Spesifikasi Perangkat Pengembang

Spesifikasi	Deskripsi
Laptop	ASUS P2430U
Perangkat Keras	
Processor	Intel® Core™ i3-6006U CPU @2.00 GHz (4 CPUs), ~2.0GHz
RAM	8 GB
Harddisk	500 GB
Perangkat Lunak	
Sistem Operasi	Ubuntu 18.04
Aplikasi Pengembangan	1. PyCharm Community Edition 2. Draw.io 3. Virtualbox 5.2
Python Modules	1. OS 2. Netifaces 3. Sys 4. Tkinter 5. Time 6. Getmac

Tabel 2. Spesifikasi Perangkat Implementasi

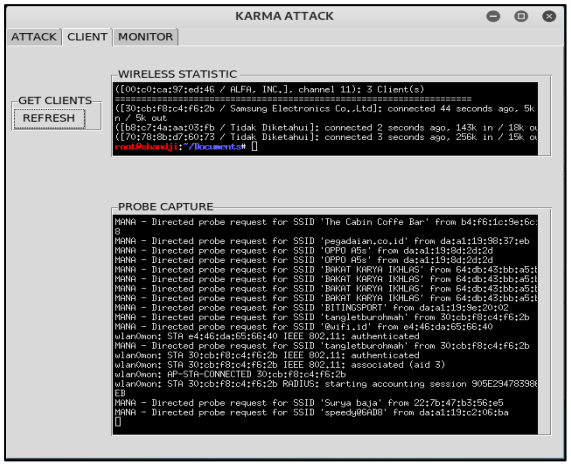
Spesifikasi	Deskripsi
Laptop	ASUS P2430U
Perangkat Keras	
Processor	Intel® Core™ i3-6006U CPU @2.00 GHz (4 CPUs), ~2.0GHz
RAM	4 GB
Harddisk	80 GB
Wireless Adapter	ALFA AWUS30NH
Perangkat Lunak	
Sistem Operasi	Kali-Linux 2019.2-amd64

Berdasarkan implementasi yang dilakukan, diperoleh hasil seperti berikut.



Gambar 3. Tab *Attack*

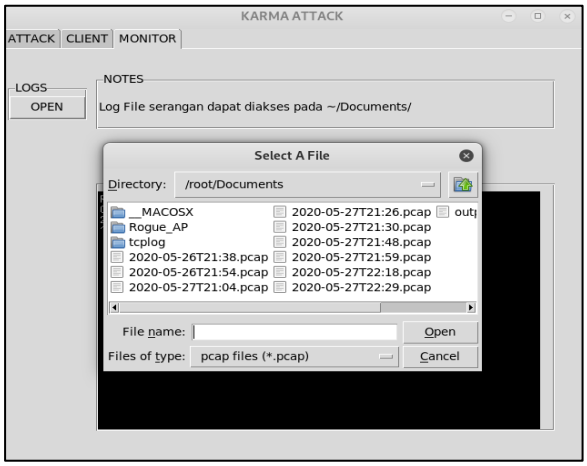
Berdasarkan pada Gambar 3, pengguna dapat memberikan parameter *input* pada "*routing configuration"* dan menjalankan aplikasi dengan tombol "*START*". Sementara frame *AP CLIENT* digunakan untuk menampilkan proses DHCP yang sedang berjalan.



Gambar 4. Tampilan Tab *Client*

Pada Gambar 4, *Tab Client* digunakan untuk menampilkan klien yang berhasil terasosiasi dan terpantau oleh perangkat.

Gambar 5. Tampilan Tab *Monitor*



Gambar 6. Tampilan Berkas Log

Tab Monitor seperti yang terlihat pada Gambar 5 digunakan untuk memantau lalu lintas jaringan yang melewati *interface* wlan0mon. Aplikasi akan menyimpan *log* lalu lintas jaringan tersebut dalam bentuk *pcap* di dalam direktori ~/Documents/ perangkat pengguna.

Pada Gambar 6, *tab Monitor* memiliki tombol "*OPEN*" yang digunakan untuk membuka berkas *log* lalu lintas jaringan. Apabila pengguna menekan tombol tersebut, aplikasi akan menampilkan isi direktori berkas *log* sehingga pengguna dapat memilih berkas. *pcap* yang akan dianalisis. Kemudian berkas *log* dapat dianalisis menggunakan tool *Wireshark.*

4. HASIL DAN PEMBAHASAN

Aplikasi yang dibuat dilakukan pengujian untuk memastikan bahwa aplikasi berjalan sesuai dengan perancangan yang telah dibuat. Pada penelitian ini dilakukan tiga jenis pengujian, yaitu: *unit testing, intergration testing,* dan *system testing.*

4.1 *Unit Testing*

Unit testing dilakukan untuk memastikan bahwa fungsi-fungsi pada aplikasi telah berjalan sesuai dengan semestinya. Pengujian ini digunakan pendekatan *black box testing* yang berfokus pada fungsionalitas aplikasi yang tidak mengacu pada struktur internal aplikasi. Hasil *unit testing* purwarupa aplikasi otomatisasi serangan KARMA secara keseluruhan adalah seperti pada Tabel 3.

Tabel 3. Hasil *Unit Testing*

No	Fungsi	Keterangan
1.	daftar_iface()	Sesuai
2.	daftar_mac()	Sesuai
3.	mode_monitor()	Sesuai
4.	routing()	Sesuai
5.	reset()	Sesuai
6.	get_vendor()	Sesuai
7.	get_hostapd()	Sesuai

Berdasarkan pada Tabel 3, dapat diketahui bahwa fungsi-fungsi dalam aplikasi dapat berjalan sesuai dengan yang diharapkan.

4.2 *Testing*

Pengujian ini dilakukan untuk mengetahui apakah fitur-fitur dalam aplikasi telah saling terintegrasi dengan baik. Pada pengujian ini dilakukan 2 pendekatan yaitu: *user interface testing* dan *scenario testing* dengan hasil seperti berikut.

a. *User Interface Testing*

User Interface Testing digunakan untuk menguji fungsionalitas dari komponen-komponen antarmuka yang ada pada aplikasi. Pada pengujian ini dilakukan terhadap beberapa tombol pada aplikasi. Hasil pengujian dapat dilihat pada Tabel 4.

Tabel 4. Hasil *User Interface Testing*

No	Tampilan Antarmuka	Hasil yang diharapkan	Keterangan
1.	Tombol *Start*	Aplikasi memulai serangan	Sesuai
2.	Tombol *Open*	Membuka direktori *log*	Sesuai
3.	Tombol *Refresh*	Menampilkan daftar klien yang terhubung	Sesuai
4.	Tombol *Reset*	Mengubah *network interface* menjadi mode manage	Sesuai

b. Scenario Testing

Digunakan dua skenario untuk melakukan pengujian ini, yaitu skenario pertama dengan menggunakan tiga buah perangkat yang telah diketahui identitas berupa MAC *address* dan skenario kedua dengan menempatkan aplikasi pada tempat umum untuk mengetahui apakah aplikasi dapat merespon perangkat-perangkat yang acak atau belum diketahui. Hasil pengujian pertama dapat dilihat pada Tabel 5.

Tabel 5. Hasil Pengujian Skenario Pertama

No	MAC *Address*	*Vendor*	Keterangan
1.	0c:98:38:21:31:2d	Tidak Diketahui	Terhubung
2.	dc:f0:90:0a:af:3d	Private	Terhubung
3.	e0:b9:a5:9d:19:e0	Azurewave Technology	Terhubung

Sementara itu, penjelasan mengenai hasil pengujian menggunakan skenario kedua dapat dilihat pada Tabel 6.

Tabel 6. Hasil Pengujian Skenario Kedua

No	MAC *Address*	*Vendor*	Keterangan
1.	04:92:26:9e:da:d7	Tidak Diketahui	Terhubung
2.	9c:5f:5a:ad:1d:d1	Tidak Diketahui	Terhubung
3.	1e:4a:5b:25:4e:b6	Tidak Diketahui	Terhubung
4.	1c:48:ce:9b:c4:ef	OPPO	Terhubung
5.	ec:d0:9f:f6:fa:5c	Tidak Diketahui	Terhubung
6.	f4:d6:20:bd:29:de	Tidak Diketahui	Terhubung
7.	4c:49:e3:f3:88:77	Tidak Diketahui	Terhubung

Berdasarkan pada Tabel 5 dan 6 dapat diketahui bahwa aplikasi otomatisasi serangan KARMA dapat berjalan dengan skenario yang telah diterapkan.

4.3 *System Testing*

Pengujian ini dilakukan untuk mengetahui kesesuaian aplikasi terhadap kebutuhan fungsionalitas dan non-fungsionalitas. Hasil *system testing* terhadap kesesuaian kebutuhan fungsional dan non-fungsional dapat dilihat pada Tabel 7 dan 8.

Tabel 7. Pemenuhan Kebutuhan Fungsional

No	Kebutuhan Fungsional	Keterangan
1.	Aplikasi dapat menampilkan status serangan KARMA	Sesuai
2.	Aplikasi menggunakan modul KARMA	Sesuai
3.	Aplikasi dapat mengonfigurasi *interface* yang digunakan untuk melakukan serangan	Sesuai
4.	Aplikasi dapat melakukan pengaturan DHCP *server*	Sesuai
5.	Aplikasi dapat melakukan *logging* aktivitas *client*	Sesuai
6.	Aplikasi dapat dilakukan konfigurasi	Sesuai

Tabel 8. Pemenuhan Kebutuhan Non-fungsional

No	Kebutuhan Non-Fungsional	Keterangan
1.	Pemrograman GUI menggunakan Bahasa pemrograman Python	Sesuai
2.	Aplikasi mampu berjalan pada sistem operasi berbasis Linux	Sesuai
3.	*Wireless adapter* Alfa dengan dukungan *packet injection*	Sesuai

Berdasarkan pada Tabel 7 dan 8 dapat diketahui bahwa purwarupa otomatisasi serangan KARMA telah memenuhi kebutuhan fungsional dan non-fungsionalnya. Oleh karena itu, berdasarkan pada pengujian yang telah dilakukan, didapatkan hasil sebagai berikut.

a. Fungsi-fungsi yang terdapat dalam purwarupa telah diuji dan berperan sesuai dengan kebutuhan.
b. Tampilan antarmuka pada purwarupa aplikasi telah terintegrasi dengan fungsi-fungsi yang lain sesuai dengan perancangan yang telah dilakukan berdasarkan *integration testing*.
c. Keseluruhan fungsi telah diuji menggunakan *system testing* yang menunjukkan bahwa purwarupa telah memenuhi kebutuhan fungsional dan non-fungsionalnya.

d. Aplikasi berhasil mengimplementasikan serangan KARMA pada perangkat-perangkat yang belum diketahui sebelumnya.

e. Aplikasi dapat berjalan secara otomatis sesuai dengan perancangan yang telah dibuat.

5. KESIMPULAN

Berdasarkan pengujian dan analisis terhadap purwarupa otomatisasi serangan KARMA dalam penelitian ini, maka dapat diperoleh kesimpulan bahwa pembuatan purwarupa aplikasi otomatisasi serangan KARMA telah berhasil dilakukan. Proses pembuatan purwarupa aplikasi otomatisasi serangan KARMA dilakukan dengan mengintegrasikan berbagai macam *tools* seperti Airmon-ng, Dnsmasq, Hostapd-mana, Iptables, dan Tshark dengan menggunakan modul-modul yang ada dalam pemrograman Python 3.7. Masing-masing *tools* berjalan sesuai dengan yang diharapkan. Airmon-ng digunakan untuk melakukan pengubahan *network interface* menjadi mode monitor, Dnsmasq digunakan untuk membuat DHCP *server* untuk memberikan alamat IP kepada perangkat-perangkat yang terhubung, Hostapd-mana digunakan untuk membuat akses poin palsu, Iptables digunakan untuk mengonfigurasikan proses *routing* sehingga perangkat yang terhubung mendapatkan akses internet, dan Tshark digunakan untuk memantau lalu lintas jaringan dan membuat file *pcap* untuk proses *logging*.

REFERENSI

[1] P. C. Z. Qiang, “Proof of Concept: Network Vulnerability through Wi-Fi *Spoofing*,” Universiti Tunku Abdul Rahman, Perak, 2017.

[2] W. S. Bhaya and S. A. AlAsady, "Prevention of *Spoofing Attack*s in the Infrastructure *Wireless* Networks," *Journal of Computer Science,* pp. 1769-1779, 2012.

[3] A. Redondi dan M. Cesana, “Building up knowledge through passive WiFi probes,” *Computer Communication,* pp. 1-12, 2018.

[4] M. Conti, N. Dragoni and V. Leysk, "A Survey of Man in The Middle *Attack*s," IEEE, 2016.

[5] D. A. D. Zovi, "KARMA *Attack*s Radioed Machines Automatically," Online. (http://theta44.org/karma), 2006.

[6] E. Ramadhani, "EKSPLORASI ISU KEAMANAN JARINGAN *WIRELESS* STUDI KASUS UNIVERSITAS GADJAH MADA," Media Informatika, Yogyakarta, 2014.

[7] J. Creasey, "A guide for running effective Penetration Testing programme," CREST, 2017.

[8] Kemendikbud, "Kamus Besar Bahasa Indonesia," Online (https://kemendikbud.go.id).

[9] R. Breton and E. Bosse, "The Cognitive Cost and Benefit of Automation," Warsaw, 2002.

[10] A. Holt and C.-Y. Huang, 802.11 *Wireless* Network: Security and Analysis, London: Springer, 2010.

[11] R. B. Abdelrahman, A. B. Mustafa and A. A. Oman, "A Comparison between IEEE 802.11a, b, g, n, and ac Standards," *IOSR Journal of Computer Engineering,* pp. 26-29, 2015.

[12] Cisco, "802.11 Association process explained," Cisco Meraki.