

**PENGUJIAN SERANGAN BERBASIS ARP POISONING PADA  
JARINGAN LOKAL STUDI KASUS: PENGAKSESAN  
APLIKASI BERBASIS WEB UNIVERSITAS BRAWIJAYA**

**SKRIPSI**

Diajukan untuk memenuhi sebagian persyaratan untuk mencapai gelar

Sarjana Komputer



Disusun oleh:

**RYAN NANDA PERDANA**

**NIM. 0910683085**

**KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN**

**UNIVERSITAS BRAWIJAYA**

**PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER**

**MALANG**

**2014**

## KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Allah SWT atas limpahan rahmat, karunia serta hidayah-Nya sehingga penulis dapat menyelesaikan skripsi yang dengan judul “**Pengujian Serangan Berbasis ARP Poisoning pada Jaringan Lokal Studi Kasus: Pengaksesan Aplikasi Berbasis Web Universitas Brawijaya**”. Penyusunan naskah skripsi ini merupakan salah satu persyaratan dalam memperoleh gelar Sarjana Komputer di Program Studi Informatika, Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya.

Penulisan skripsi ini tentu saja tidak lepas dari bantuan beberapa pihak, oleh karena itu penulis bermaksud menyampaikan terima kasih kepada:

1. Bapak Aswin Suharsono, ST., MT. selaku dosen pembimbing I atas ilmu, bimbingan dan pengarahan yang diberikan kepada penulis selama penyusunan skripsi.
2. Bapak Sabriansyah Rizzika Akbar, ST., M.Eng. selaku dosen pembimbing II atas ilmu, bimbingan dan pengarahan yang diberikan kepada penulis selama penyusunan skripsi.
3. Rekan-rekan PPTI Universitas Brawijaya, sebagai penanggung jawab aplikasi berbasis *web* UB atas kerjasama dan fasilitas yang telah diberikan kepada penulis.
4. Dosen Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya atas bimbingan dan ilmu yang telah diberikan kepada penulis.
5. Staff Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya atas bantuan yang telah diberikan selama penulis menempuh studi.
6. Kedua orang tua penulis, saudara, seseorang yang istimewa dan keluarga besar yang telah memberikan dukungan, semangat serta kasih sayang sehingga penulis dapat menyelesaikan tugas akhir ini.
7. Sahabat serta rekan-rekan seperjuangan di Program Studi Informatika yang telah memberikan saran, semangat dan dukungan kepada penulis selama penyusunan naskah tugas akhir ini.

Penulis menyadari bahwa skripsi ini masih memiliki banyak kekurangan dan jauh dari sempurna, oleh karena itu penulis mengharapkan saran dan kritik yang membangun. Semoga naskah skripsi ini dapat memberikan sumbangan yang bermanfaat bagi perkembangan ilmu pengetahuan khususnya di bidang teknologi informasi.

Malang, April 2014

Penulis





## ABSTRAK

**Ryan Nanda Perdana. 2014. Pengujian Serangan Berbasis ARP Poisoning pada Jaringan Lokal Studi Kasus: Pengaksesan Aplikasi Berbasis Web Universitas Brawijaya.** Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya, Malang. Dosen pembimbing: Aswin Suharsono, ST., MT. dan Sabriansyah Rizqika Akbar, ST., M.Eng.

Protokol SSL (Secure Socket Layer) dibutuhkan sebagai sarana keamanan pada aplikasi berbasis *web* yang berguna untuk mencegah adanya usaha penyalahgunaan aset informasi dari berbagai pihak yang terhubung dengan *internet*. Meskipun SSL telah memberikan keamanan dalam transmisi data, SSL tidak menjamin keamanan dari sisi *user* yang dapat dikelabui dengan menggunakan serangan seperti *sniffing*. Serangan *sniffing* dapat membuat penyerang berada di antara *client* dan *server* serta mengambil informasi penting dari komunikasi yang dijalin. Penelitian ini menganalisa tingkat keberhasilan *sniffing* menggunakan serangan *ARP poisoning* dengan beberapa metode uji serang yaitu penangkapan *traffic*, *SSL hijacking*, *session hijacking* dan *DNS spoofing* serta menganalisa penyebab kegagalan dari metode uji serang tersebut. Dari hasil pengujian didapatkan faktor keberhasilan dan kegagalan dari masing-masing metode uji serang serta bagaimana melakukan tindakan pencegahan di sisi *client* agar terhindar dari *ARP poisoning*.

Persentase keberhasilan masing-masing metode

**Kata kunci:** *Sniffing*, *ARP Poisoning*, *SSL*

## ABSTRACT

**Ryan Nanda Perdana. 2014. Pengujian Serangan Berbasis ARP Poisoning pada Jaringan Lokal Studi Kasus: Pengaksesan Aplikasi Berbasis Web Universitas Brawijaya. Information Technology and Computer Science Program, Brawijaya University, Malang. Advisor: Aswin Suharsono, ST., MT. and Sabriansyah Rizqika Akbar, ST., M.Eng.**

*SSL (Secure Sockets Layer) protocol is needed as a means of security in a web-based application, and useful to prevent the abuse of information assets from various parties connected to the internet. In spite of security in data transmission has been provided by SSL protocol, the security of user has not assured with the existence of sniffing attacks. Sniffing attack inflicts the attacker exist between client and server which leads to the retrieval of important information. This study analyzed the success rate of sniffing using ARP poisoning attacks by using several attacking test method such as traffic capturing, SSL hijacking, session hijacking and DNS spoofing and also analyze the causes of the failure from those attacking test method. Based on the test result, the factors of success and failure from each attacking test method were obtained and also what precautions that the client needs to do to avoid ARP poisoning.*

**Keywords:** *Sniffing, ARP Poisoning, SSL*

## DAFTAR ISI

<b>KATA PENGANTAR</b> .....	<b>i</b>
<b>ABSTRAK</b> .....	<b>iii</b>
<b>ABSTRACT</b> .....	<b>iv</b>
<b>DAFTAR ISI</b> .....	<b>v</b>
<b>DAFTAR GAMBAR</b> .....	<b>viii</b>
<b>DAFTAR TABEL</b> .....	<b>x</b>
<b>BAB I PENDAHULUAN</b> .....	<b>1</b>
1.1 Latar Belakang .....	1
1.2 Rumusan Masalah .....	2
1.3 Batasan Masalah.....	2
1.4 Tujuan.....	3
1.5 Manfaat.....	3
1.6 Sistematika Penulisan.....	4
1.7 Timeline Penelitian .....	5
<b>BAB II DASAR TEORI</b> .....	<b>6</b>
2.1 Komunikasi Jaringan Internet .....	6
2.1.1 ARP (Address Resolution Protocol) .....	7
2.1.2 ARP Cache .....	8
2.2 Sniffing.....	9
2.2.1 Passive Sniffing.....	9
2.2.2 Active Sniffing .....	9
2.3 Serangan MITM (Man-in-the-middle).....	9
2.3.1 ARP Poisoning .....	10
2.3.2 SSL Hijacking .....	11
2.3.3 Session Hijacking.....	12
2.3.4 DNS Spoofing .....	13
2.4 Serangan Keamanan WLAN.....	14



<b>BAB III METODOLOGI PENELITIAN .....</b>	<b>15</b>
3.1 Studi Literatur .....	16
3.2 Penyusunan Dasar Teori.....	16
3.3 Analisa dan Perancangan .....	16
3.3.1 Kebutuhan Jaringan Komputer.....	16
3.3.2 Kebutuhan Penyerang .....	17
3.3.3 Kebutuhan Target.....	17
3.3.4 Kebutuhan Aplikasi Berbasis Web .....	17
3.3.5 Kebutuhan Data.....	18
3.3.6 Skenario Penelitian.....	18
3.4 Implementasi .....	19
3.5 Pengujian.....	19
3.5.1 Reconnaissance .....	19
3.5.2 Active Attacks .....	20
3.5.2.1 Pengujian Sniffing.....	20
3.5.2.2 Pengujian Penangkapan Traffic .....	20
3.5.2.3 Pengujian SSL Hijacking .....	21
3.5.2.4 Pengujian Session Hijacking.....	22
3.5.2.5 Pengujian DNS Spoofing .....	22
3.6 Penulisan Laporan .....	23
<b>BAB IV IMPLEMENTASI DAN PENGUJIAN .....</b>	<b>24</b>
4.1 Implementasi .....	24
4.1.1 Implementasi Skenario Jaringan .....	24
4.1.2 Implementasi Reconnaissance.....	26
4.1.3 Implementasi Active Attacks .....	26
4.1.3.1 Sniffing.....	26
4.1.3.2 Penangkapan Traffic .....	28
4.1.3.3 SSL Hijacking .....	28
4.1.3.4 Session Hijacking.....	29
4.1.3.5 DNS Spoofing .....	30



4.2 Pengujian.....	31
4.2.1 Lingkungan Jaringan Konfigurasi Standar.....	32
4.2.1.1 Reconnaissance .....	32
4.2.1.2 Pengujian Sniffing.....	35
4.2.1.3 Pengujian Penangkapan Traffic .....	39
4.2.1.4 Pengujian SSL Hijacking .....	41
4.2.1.5 Pengujian Session Hijacking.....	42
4.2.1.6 Pengujian DNS Spoofing .....	44
4.2.2 Lingkungan Jaringan Konfigurasi Lanjut .....	46
4.2.2.1 Reconnaissance .....	46
4.2.2.2 Pengujian Sniffing.....	49
4.2.2.3 Pengujian Penangkapan Traffic .....	53
4.2.2.4 Pengujian SSL Hijacking .....	55
4.2.2.5 Pengujian Session Hijacking.....	56
4.2.2.6 Pengujian DNS Spoofing .....	57
<b>BAB V ANALISIS HASIL PENGUJIAN.....</b>	<b>58</b>
5.1 Analisa Uji Penangkapan Traffic .....	58
5.2 Analisa Uji SSL Hijacking .....	59
5.3 Analisa Uji Session Hijacking.....	60
5.4 Analisa Uji DNS Spoofing.....	61
5.5 Analisa Penanggulangan Sniffing .....	62
5.5.1 Pemeriksaan ARP Poisoning.....	62
5.5.2 Deteksi ARP Poisoning.....	64
5.5.3 Pencegahan ARP Poisoning.....	67
<b>BAB VI PENUTUP .....</b>	<b>69</b>
6.1 Kesimpulan.....	69
6.2 Saran.....	70
<b>DAFTAR PUSTAKA .....</b>	<b>DP-1</b>



## DAFTAR GAMBAR

Gambar 2.1 ARP <i>Request</i> .....	7
Gambar 2.2 ARP <i>Reply</i> .....	8
Gambar 2.3 <i>Traffic normal &amp; poisoned</i> .....	10
Gambar 2.4 SSL <i>Hijacking</i> .....	11
Gambar 2.5 <i>Session Hijacking</i> .....	12
Gambar 2.6 DNS <i>Spoofing</i> .....	13
Gambar 2.7 Klasifikasi Serangan WLAN .....	14
Gambar 3.1 <i>Flowchart</i> Runtutan Pengerjaan Penelitian .....	15
Gambar 3.2 Skenario Penelitian .....	18
Gambar 3.3 Tahap Pengujian Jaringan Lokal .....	19
Gambar 4.1 Skenario Jaringan .....	25
Gambar 4.2 Tabel ARP Normal .....	27
Gambar 4.3 Tabel ARP Setelah Proses <i>Poisoning</i> .....	27
Gambar 4.4 Implementasi Penangkapan <i>Traffic</i> .....	28
Gambar 4.5 Implementasi SSL <i>Hijacking</i> .....	29
Gambar 4.6 Implementasi <i>Session Hijacking</i> .....	30
Gambar 4.7 Implementasi DNS <i>Spoofing</i> .....	31
Gambar 4.8 Tampilan Airmon-ng .....	33
Gambar 4.9 Tampilan Airodump-ng .....	33
Gambar 4.10 Tampilan Informasi Host yang Terhubung di Jaringan .....	34
Gambar 4.11 ARPing pada <i>Router</i> .....	34
Gambar 4.12 ARPing pada Target .....	35
Gambar 4.13 Alamat IP Target .....	35
Gambar 4.14 Alamat IP Penyerang .....	36
Gambar 4.15 Penyerang Menentukan Target .....	37
Gambar 4.16 Penyerang Melakukan Cek ARP <i>Poisoning</i> .....	37
Gambar 4.17 Tabel ARP Normal .....	38
Gambar 4.18 Tabel ARP Ter- <i>poison</i> .....	38
Gambar 4.19 Tampilan <i>Traffic</i> e-complaint.ub.ac.id .....	40
Gambar 4.20 Tampilan <i>traffic</i> bais.ub.ac.id .....	40

Gambar 4.21 Tampilan <i>log</i> dari SSLStrip.....	41
Gambar 4.22 Tampilan TCP <i>stream</i> bais.ub.ac.id .....	43
Gambar 4.23 Tampilan Penyerang Mengakses Halaman dengan Akun Target ...	44
Gambar 4.24 Tampilan SEToolkit dengan Informasi <i>Login</i> .....	45
Gambar 4.25 Tampilan Airmon-ng.....	47
Gambar 4.26 Tampilan Airodump-ng.....	47
Gambar 4.27 Tampilan Informasi Host yang Terhubung di Jaringan .....	48
Gambar 4.28 ARPing pada <i>Router</i> .....	49
Gambar 4.29 ARPing pada Target.....	49
Gambar 4.30 Alamat IP Target.....	50
Gambar 4.31 Alamat IP Penyerang.....	50
Gambar 4.32 Penyerang Menentukan Target.....	51
Gambar 4.33 Penyerang Melakukan Cek ARP <i>Poisoning</i> .....	51
Gambar 4.34 Tabel ARP Normal.....	52
Gambar 4.35 Tabel ARP Ter- <i>poison</i> .....	53
Gambar 4.36 Tampilan <i>Traffic</i> nas.ub.ac.id.....	54
Gambar 4.37 Tampilan <i>Traffic</i> siam.ub.ac.id .....	55
Gambar 4.38 Tampilan <i>log</i> dari SSLStrip.....	56
Gambar 5.1 Tabel ARP Normal .....	63
Gambar 5.2 Tabel ARP Ter- <i>poison</i> .....	63
Gambar 5.3 DecaffeineatID Berjalan di <i>Taskbar</i> .....	64
Gambar 5.4 <i>Monitor</i> pada DecaffeineatID.....	64
Gambar 5.5 DecaffeineatID Mendeteksi Adanya Perubahan Tabel ARP .....	65
Gambar 5.6 <i>Log File</i> pada DecaffeineatID .....	65
Gambar 5.7 Tampilan XARP Dengan Rincian Tabel ARP .....	66
Gambar 5.8 XARP Mendeteksi Adanya Perubahan Tabel ARP .....	66
Gambar 5.9 Perintah <i>Static IP Gateway</i> pada Windows.....	67
Gambar 5.10 Alamat IP <i>Gateway</i> Menjadi <i>Static</i> pada Tabel ARP Target.....	68
Gambar 5.11 ARP <i>Poisoning</i> Gagal pada Penyerang.....	68

## DAFTAR TABEL

Tabel 1.1 <i>Timeline</i> Penelitian.....	5
Tabel 3.1 Tabel Pengujian Pola Serangan Penangkapan <i>Traffic</i> .....	21
Tabel 3.2 Tabel Pengujian Pola Serangan SSL <i>Hijacking</i> .....	21
Tabel 3.3 Tabel Pengujian Pola Serangan <i>Session Hijacking</i> .....	22
Tabel 3.4 Tabel Pengujian Pola Serangan DNS <i>Spoofing</i> .....	23
Tabel 4.1 Kebutuhan Informasi .....	26
Tabel 4.2 Hasil Uji Penangkapan <i>Traffic</i> .....	39
Tabel 4.3 Hasil Uji SSL <i>Hijacking</i> .....	41
Tabel 4.4 Hasil Uji <i>Session Hijacking</i> .....	42
Tabel 4.5 Hasil Uji DNS <i>Spoofing</i> .....	45
Tabel 4.6 Hasil Uji Penangkapan <i>Traffic</i> .....	54
Tabel 4.7 Hasil Uji SSL <i>Hijacking</i> .....	56
Tabel 4.8 Hasil Uji <i>Session Hijacking</i> .....	57
Tabel 4.9 Hasil Uji DNS <i>Spoofing</i> .....	57
Tabel 5.1 Analisa Uji Penangkapan <i>Traffic</i> .....	59
Tabel 5.2 Analisa Uji SSL <i>Hijacking</i> .....	60
Tabel 5.3 Analisa Uji <i>Session Hijacking</i> .....	61
Tabel 5.4 Analisa Uji DNS <i>Spoofing</i> .....	62
Tabel 6.1 Persentase Tingkat Keberhasilan Pengujian .....	69



## BAB I

### PENDAHULUAN

#### 1.1 Latar Belakang

*Internet* merupakan salah satu wujud dari kemajuan TIK (Teknologi Informasi dan Komunikasi) yang sangat populer saat ini. Sistem informasi yang berbentuk *website* menjadi pilihan untuk menyimpan dan bertukar informasi, dikarenakan kemudahan akses oleh siapa saja, darimana saja dan kapan saja.

Dalam konteks keamanan informasi, informasi dianggap sebagai aset yang bernilai dan harus dilindungi [ABS-13:2]. Keinginan untuk mendapatkan akses dan mengendalikan sebuah informasi akan meningkat seiring dengan meningkatnya nilai aset informasi. Hal tersebut menyebabkan munculnya individu atau kelompok yang menggunakan aset informasi demi berbagai tujuan dan dengan berbagai cara.

Suatu protokol dibutuhkan dalam menyediakan layanan keamanan pada komunikasi antar jaringan. Salah satu protokol yang dapat memberikan koneksi yang aman dan terenkripsi antara *server* dan *client* adalah SSL (*Secure Socket Layer*) [HRY-10:1]. Penggunaan protokol keamanan ini bertujuan untuk mencegah adanya usaha penyalahgunaan aset informasi dari berbagai pihak yang terhubung dengan *internet*.

ARP *poisoning* adalah salah satu jenis serangan dengan metode *sniffing* yang digunakan untuk mendapatkan informasi secara ilegal. Meskipun SSL sudah memberikan keamanan dalam transmisi data, namun tidak menjamin keamanan dari sisi *user*. *User* masih dapat dikelabui di sisi HCI (*Human Computer Interface*) tanpa memperdulikan fungsi SSL sebagai protokol keamanan [FQI-09:1]. Hal ini merupakan salah satu ancaman bagi keamanan informasi yang memanfaatkan aplikasi berbasis *web* sebagai sarana menyimpan informasi.



Berdasarkan hal tersebut, penulis merancang sistem serangan dengan metode *sniffing* untuk menguji keamanan pada aplikasi berbasis *web* yang menerapkan implementasi SSL sebagai protokol keamanannya. Penulis ingin meneliti lebih lanjut mengenai tingkat keberhasilan serangan *ARP poisoning* yang diterapkan pada jaringan lokal bersama *client* dengan aplikasi berbasis *web* UB (Universitas Brawijaya) sebagai *server* tujuan.

### 1.2 Rumusan Masalah

Berdasarkan uraian latar belakang diatas, dapat dirumuskan permasalahan pada tugas akhir ini yaitu sebagai berikut:

1. Bagaimana merancang pengujian serangan berbasis *ARP poisoning* pada jaringan lokal?
2. Berapa besar tingkat keberhasilan pada pengujian serangan berbasis *ARP poisoning* pada jaringan lokal?

### 1.3 Batasan Masalah

Agar permasalahan yang dirumuskan dapat lebih fokus, penelitian tugas akhir ini dibatasi dalam hal:

1. Metode uji serang yang digunakan dalam penelitian ini adalah *sniffing* dengan memanfaatkan teknik *ARP (Address Resolution Protocol) Poisoning* dan diterapkan pada jaringan lokal bersama *client*.
2. Aplikasi berbasis *web* yang menjadi *server* tujuan adalah beberapa aplikasi berbasis *web* UB yang dikelola oleh unit PPTI (Pengkajian dan Pengembangan Teknologi Informasi) UB yang menerapkan SSL sebagai protokol keamanannya.

#### 1.4 Tujuan

1. Mengetahui bagaimana merancang pengujian serangan berbasis ARP *poisoning* pada jaringan lokal.
2. Mengetahui hasil analisa tingkat keberhasilan pada pengujian serangan berbasis ARP *poisoning* pada jaringan lokal.

#### 1.5 Manfaat

Penelitian ini diharapkan dapat bermanfaat untuk berbagai pihak. Manfaat dari penelitian ini adalah sebagai berikut:

Bagi Penulis:

1. Mengetahui celah keamanan yang terdapat pada aplikasi berbasis *web* UB.
2. Menerapkan ilmu yang telah didapat dari perkuliahan di PTIIK (Program Teknologi Informasi dan Ilmu Komputer) UB.

Bagi Pembaca:

1. Menjadi peduli betapa pentingnya protokol keamanan pada sebuah aplikasi berbasis *web* maupun sistem informasi.
2. Mengetahui sistem kerja protokol keamanan (SSL) pada sebuah aplikasi berbasis *web* serta menyadari bahaya apa yang dapat mengancam protokol keamanan tersebut.

Bagi PPTI UB:

1. Mengetahui hasil analisa dari pengujian serangan ARP *poisoning* pada beberapa aplikasi berbasis *web* UB yang mengandung informasi penting.
2. Mengetahui letak celah keamanan pada aplikasi berbasis *web* UB sehingga dapat meningkatkan keamanan pada aplikasi berbasis *web* tersebut.



## 1.6 Sistematika Penulisan

Sistematika penulisan laporan tugas akhir ini adalah sebagai berikut:

### **BAB 1 Pendahuluan**

Memuat latar belakang permasalahan, identifikasi dan pembatasan masalah, rumusan masalah, tujuan dan manfaat serta sistematika penulisan.

### **BAB II Dasar Teori**

Menjelaskan teori yang terkait dengan komunikasi ARP antar komputer serta tahap dan metode serangan yang bisa dilakukan dengan memanfaatkan celah ARP.

### **BAB III Metodologi Penelitian**

Membahas metode yang digunakan dalam penelitian yang terdiri dari studi literatur, penyusunan dasar teori, analisa dan perancangan, implementasi, pengujian serta pengambilan kesimpulan dan saran.

### **BAB IV Implementasi dan Pengujian**

Membahas tentang implementasi serta pengujian dari serangan ARP *poisoning (sniffing)* pada sebuah target di jaringan lokal yang sedang mengakses *website* yang menerapkan protokol keamanan SSL.

### **BAB V Analisa Hasil Pengujian**

Memuat hasil analisa terhadap pengujian pada beberapa metode yang telah diimplementasikan dan diujikan.

### **BAB VI Penutup**

Memuat kesimpulan dan saran yang diperoleh dari implementasi dan pengujian pada beberapa metode untuk pengkajian dan pengembangan lebih lanjut.

### 1.7 Timeline Penelitian

Berikut adalah *timeline* pengerjaan penelitian beserta laporan:

**Tabel 1.1** *Timeline* Penelitian

No.	Proses Kegiatan:	Bulan dan Minggu ke:															
		Bulan I				Bulan II				Bulan III				Bulan IV			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	Studi literatur	■	■	■	■												
2	Penyusunan dasar teori					■	■	■	■								
3	Analisa dan perancangan sistem						■	■	■								
4	Implementasi sistem							■	■	■	■	■	■				
5	Pengujian sistem									■	■	■	■				
6	Penulisan laporan penelitian	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■

## BAB II

### DASAR TEORI

#### 2.1 Komunikasi Jaringan Internet

Sebuah perangkat yang tersambung di sebuah jaringan internet sudah seharusnya memiliki dua alamat, yaitu alamat MAC (*Media Access Control*) dan alamat IP (*Internet Protocol*). MAC yang disebut juga *physical address* merupakan satu pengalamatan khusus yang sudah tertera secara fisik dalam sebuah NIC (*Network Interface Card*) dan bersifat unik pada masing-masing device di seluruh dunia. Sedangkan IP yang disebut juga *logical address* adalah sebuah alamat yang diberikan secara *virtual* pada *software* dalam sebuah perangkat agar dapat berkomunikasi dengan perangkat lain di sebuah jaringan internet [VAM-05:3].

IP dan MAC sama-sama dibutuhkan dalam pengiriman paket, alamat IP berfungsi untuk menentukan tujuan (*destination*) dari sebuah paket. Paket yang akan dikirimkan melalui sebuah *interface* harus dirubah menjadi *frame* yang membutuhkan alamat MAC, kemudian pengiriman tersebut akan berjalan dengan membandingkan *destination address* dari IP tujuan dengan *destination address* dari MAC pada *interface* yang akan dilewati. Penentuan jalur akan dilakukan berdasarkan *routing table* yang berisikan korelasi antara IP dan MAC pada masing-masing *router* yang dilewati.

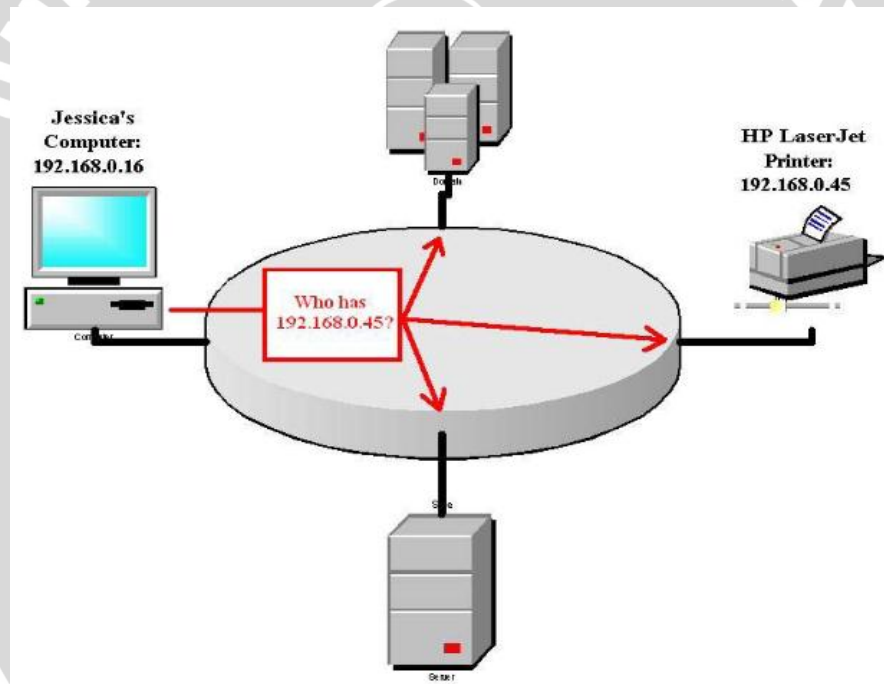
Sebuah paket membutuhkan alamat IP untuk dikirimkan ke tujuan (*destination*), namun alamat MAC juga diperlukan agar pengiriman dapat diteruskan sampai tujuan melalui beberapa *interface*. Protokol ARP (*Address Resolution Protocol*) digunakan untuk menemukan alamat MAC dari alamat IP tujuan [VAM-05:3].



### 2.1.1 ARP (Address Resolution Protocol)

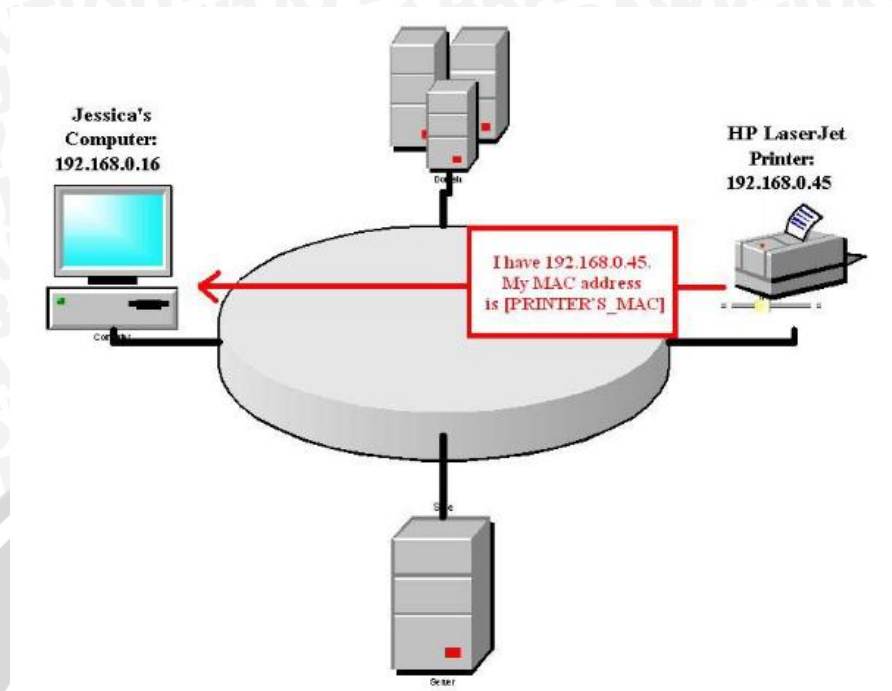
ARP adalah sebuah protokol yang digunakan untuk melakukan pengalamatan dari alamat IP menuju ke alamat MAC pada sebuah jaringan lokal. Cara kerja ARP adalah dengan mengirimkan *broadcast* pada sebuah jaringan hingga ada perangkat dengan alamat IP yang bersangkutan membalas dengan memberikan informasi alamat MAC pada perangkat yang melakukan *request*.

Pada gambar 2.1 akan ditunjukkan bagaimana sebuah perangkat (Jessica's Computer) akan melakukan *broadcast ARP request* untuk mencari alamat MAC yang dipunyai oleh alamat IP 192.168.0.45 dan pada gambar 2.2 akan ditunjukkan sebuah perangkat (HP LaserJet Printer) yang memberikan informasi alamat MAC dengan ARP *reply* pada perangkat yang melakukan *request*.



Gambar 2.1 ARP Request

Sumber: [VAM-05:5]



Gambar 2.2 ARP Reply

Sumber: [VAM-05:6]

### 2.1.2 ARP Cache

Untuk meningkatkan efisiensi pada sebuah jaringan serta tidak menghabiskan *bandwidth* dengan melakukan *ARP broadcast*, tiap perangkat mempunyai *ARP cache*. *ARP cache* adalah sebuah tabel yang berisi alamat IP beserta alamat MAC yang sesuai dan disimpan dalam memori, sebelum melakukan *broadcast*, perangkat akan memeriksa *ARP cache* apakah berisi informasi dari alamat tujuan atau tidak, jika terdapat informasi maka paket data akan dikirim tanpa melakukan *ARP broadcast* [VAM-05:4].

Informasi alamat IP dan MAC pada *ARP cache* di sebuah perangkat akan diperbarui seiring adanya *ARP reply* dari perangkat lain dengan alamat IP yang dimilikinya.

## 2.2 Sniffing

*Sniffing* merupakan sebuah aksi penyadapan paket data yang dikirimkan dari sebuah perangkat ke perangkat lainnya, misalnya komputer ke *server*. Terdapat dua jenis aksi *sniffing*, yaitu *passive* dan *active* [FAM-07:3]. Aksi *sniffing* ini dilakukan dengan jalan MITM *attack*.

### 2.2.1 Passive Sniffing

*Passive Sniffing* adalah aksi penyadapan paket data dengan tidak melakukan perubahan pada paket data tersebut, yang dilakukan hanyalah melakukan *monitoring* pada sebuah jaringan, misal: penggunaan wireshark pada sebuah jaringan lokal.

### 2.2.2 Active Sniffing

*Active Sniffing* adalah aksi penyadapan paket data dengan melakukan perubahan pada paket data yang melewati pelaku *sniffing*, informasi yang dapat dirubah pada paket data adalah *source* dan *destination address* pada IP dan MAC dalam sebuah paket data. Contoh dari *active sniffing* adalah ARP *poisoning* yang dilakukan seseorang dengan jalan MITM *attack* dengan cara berpura-pura sebagai *router* bagi sebuah *user* dan *user* bagi sebuah *router*. Hal ini tentu saja dapat dilakukan dengan cara merubah alamat MAC asli dengan alamat MAC dari penyerang.

## 2.3 Serangan MITM (Man-in-the-middle)

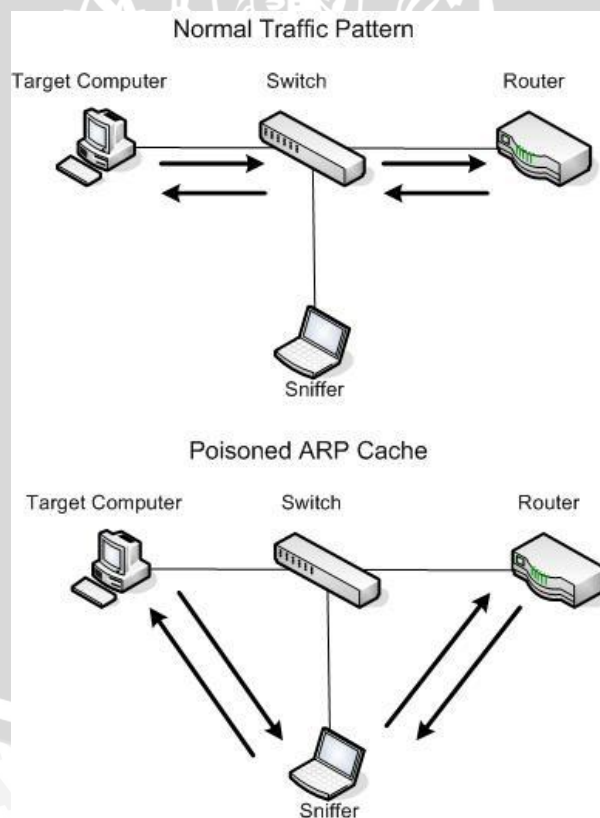
Serangan ini adalah salah satu metode dari *sniffing* dimana terdapat orang ketiga yang terlibat dalam jalur komunikasi antar dua komputer. Pada serangan ini tidak terjadi adanya interupsi dari *traffic* antar dua komputer karena orang ketiga tersebut melakukan *redirect* paket data ke komputer tujuan [VAM-05:9].



### 2.3.1 ARP Poisoning

ARP *cache* pada sebuah perangkat akan selalu menerima *update* tiap adanya ARP *reply*. Teknik ARP *poisoning* yang dilakukan oleh seorang penyerang memanfaatkan fungsi *update* dari protokol ARP, yaitu adalah dengan secara berkala mengirimkan paket ARP *reply* ke sebuah *host* dan *host* tersebut akan dipaksa untuk melakukan *update* dari ARP *cache* yang sebelumnya sudah disimpan [SAN-10]. Sehingga teknik ini akan membuat sebuah *host* berpikir bahwa penyerang adalah pemilik sebenarnya dari alamat IP perangkat tujuan dan teknik inilah yang mendasari berbagai serangan MITM seperti SSL *hijacking*, *session hijacking* dan DNS *spoofing*.

Pada gambar 2.3 akan digambarkan bagaimana *traffic* normal dari komunikasi antar *host* serta bagaimana *traffic* sudah dibawah pengaruh ARP *poisoning* yang dilakukan oleh penyerang.



**Gambar 2.3** Traffic normal & poisoned

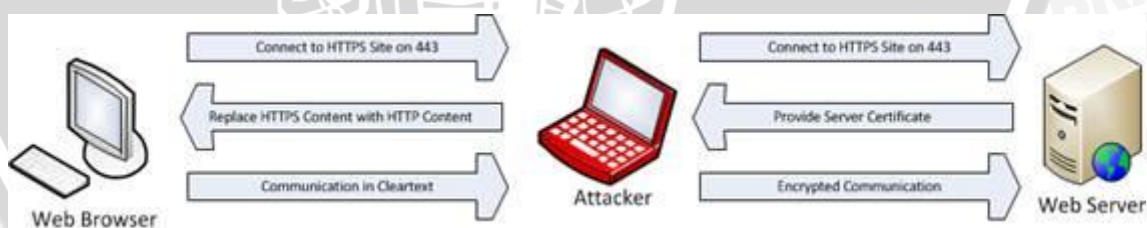
Sumber: [SAN-10]

### 2.3.2 SSL Hijacking

SSL (*Secure Socket Layer*) adalah protokol yang digunakan untuk melindungi komunikasi antar *host* dengan menggunakan *enkripsi* pada pertukaran paket data sehingga tercipta jalur yang aman pada komunikasi tersebut. Keamanan dari protokol SSL sangat tinggi, namun terdapat celah keamanan dengan memanfaatkan “jembatan” antara komunikasi yang belum terenkripsi dan komunikasi yang terenkripsi.

Seorang peneliti keamanan jaringan komputer bernama Moxie Marlinspike menyatakan bahwa di sebagian besar kasus, SSL tidak pernah diakses langsung, koneksi SSL akan diakses melalui *redirect* HTTPS (*Hypertext Transfer Protocol Secure*) dari sebuah kode respon HTTP 302 atau seseorang yang melakukan klik pada sebuah *link* yang melakukan *redirect* ke sebuah situs HTTPS, contohnya tombol *login* [SAN-10]. Itulah yang mendasari ide SSL *hijacking* dimana serangan akan dilakukan pada saat sebelum terjadi koneksi SSL, yaitu pada saat terjadi *redirect* dari HTTP yang belum aman ke HTTPS yang aman.

Pada gambar 2.4. akan digambarkan bagaimana penyerang akan menengahi koneksi HTTPS yang dilakukan oleh sebuah *host* menuju sebuah HTTPS *web server*.



Gambar 2.4 SSL Hijacking

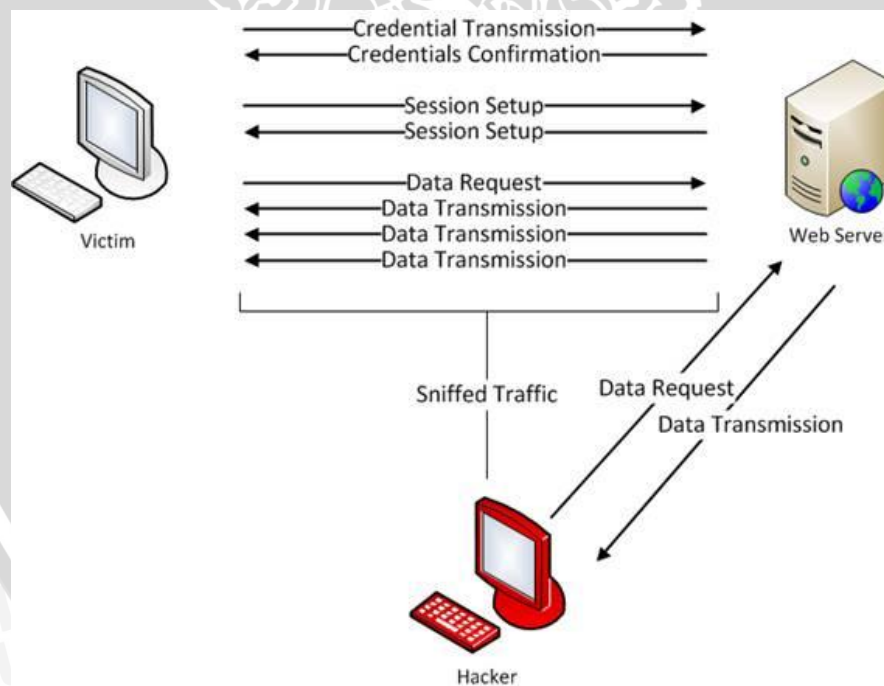
Sumber: [SAN-10]



### 2.3.3 Session Hijacking

*Session hijacking* adalah serangan yang melibatkan eksploitasi dari sebuah *session* yang terjadi antar *device*, *session hijacking* disini dilakukan melalui pencurian *cookie* yang melibatkan HTTP *session*. *Session* diperlukan pada *website* yang memerlukan *login* dalam mengakses informasi didalamnya, untuk membuat *session* diperlukan adanya otentikasi *username* dan *password*. Sebuah *website* akan melakukan *session tracking* untuk memastikan sebuah *host* dalam keadaan *logged in* dan dapat mengakses sumber daya pada *website* tersebut [SAN-10].

Penyerang akan mengambil *session* dari pertukaran data antar *client* dan *web server* dan menggunakannya untuk membuat koneksi ke *web server* sehingga penyerang bisa mengakses informasi menggunakan *session* dari seorang *client* yang sudah diserang. Pada gambar 2.5 akan digambarkan bagaimana seorang dapat menggunakan *session* dari seorang *client* untuk dapat mengakses informasi yang dimiliki *client* tersebut ke sebuah *web server*.



Gambar 2.5 Session Hijacking

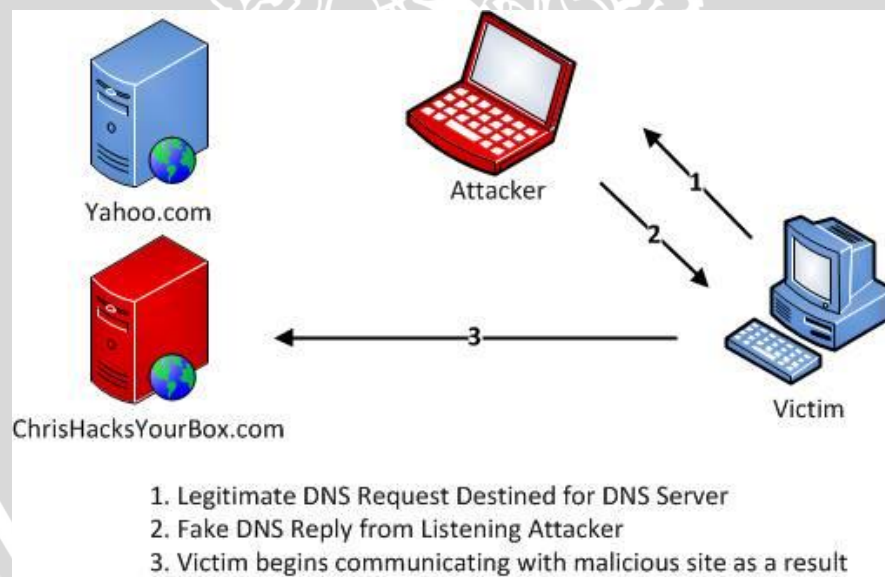
Sumber: [SAN-10]



### 2.3.4 DNS Spoofing

DNS (*Domain Name System*) adalah sebuah protokol yang digunakan untuk menerjemahkan sebuah *website domain* ke alamat IP dari *server* yang digunakan oleh *website* tersebut agar dapat diakses oleh sebuah *client* dengan hanya mengetikkan *website domain*. DNS *server* bekerja dengan melakukan entri kedalam *database* yang berisi alamat IP dari *website domain* yang ada, meneruskan informasi tersebut ke *client* dan ke DNS *server* lainnya [SAN-10].

DNS *spoofing* dilakukan dengan memberikan informasi alamat IP palsu dari sebuah *website domain* kepada *client* yang melakukan DNS *request* kepada penyerang yang sedang menghalangi *client* untuk berkomunikasi dengan DNS *server* yang sebenarnya. Jadi, *client* akan mengakses ke *server* palsu yang sudah disiapkan oleh penyerang dengan tujuan mengambil informasi yang bersifat *credentials* (pribadi). Pada gambar 2.6 akan digambarkan bagaimana penyerang bertindak sebagai DNS *server* palsu sehingga *client* menuju ke server yang salah.



Gambar 2.6 DNS Spoofing

Sumber: [SAN-10]

## 2.4 Serangan Keamanan WLAN

Serangan pada jaringan nirkabel seperti WLAN (*Wireless Local Area Network*) bersifat sama seperti serangan pada jaringan lokal yang menggunakan kabel (*wired*), penyerangan ditujukan pada ketersediaan jaringan dan informasi dari *client* yang bersifat rahasia [NAS-08]. Jaringan lokal mempunyai tahap serangan yang berbeda dengan *remote server*, tahap serangan di jaringan ini lebih sederhana yaitu:

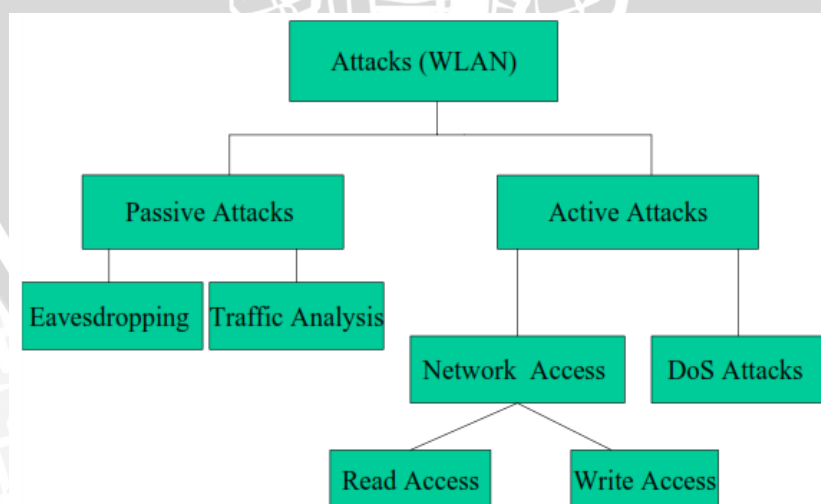
1. Serangan pasif (*passive attacks*)

Terdiri dari aksi penyadapan dan analisa *traffic* pada jaringan lokal namun tidak melakukan modifikasi pada paket. Tahap ini bertujuan untuk mengambil informasi yang berkaitan dengan jaringan lokal dan semua komponen yang ada didalamnya termasuk *router* dan *client*.

2. Serangan aktif (*active attacks*)

Tahap ini merupakan aksi penyerangan yang dilakukan setelah pengumpulan informasi, yang selanjutnya dimanfaatkan untuk melakukan serangan. Tahap ini bertujuan untuk melakukan modifikasi paket, mengambil file atau melumpuhkan jaringan yang sedang berjalan.

Gambar 2.7 berikut adalah diagram yang menunjukkan klasifikasi umum dari serangan keamanan WLAN.



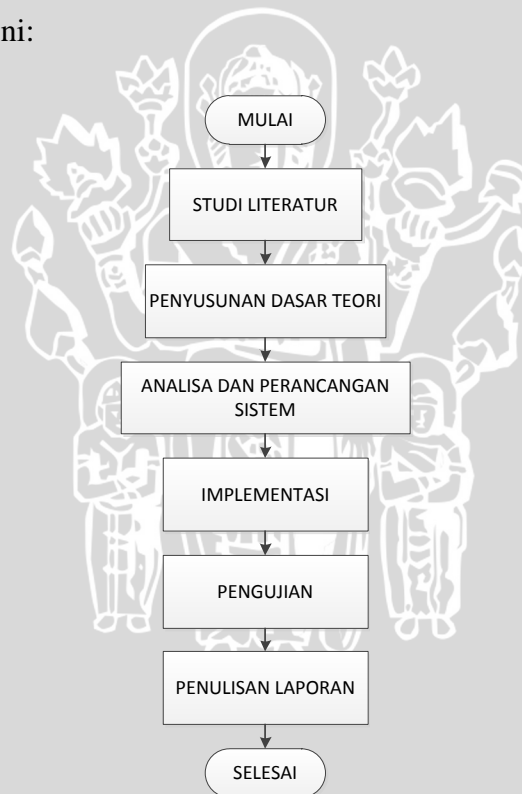
**Gambar 2.7** Klasifikasi Serangan WLAN

Sumber: [NAS-08]

## BAB III

### METODOLOGI PENELITIAN

Pada bab ini dijelaskan langkah-langkah yang akan dilakukan dalam pengerjaan tugas akhir yaitu studi literatur, penyusunan dasar teori, analisa dan perancangan, implementasi, analisis dan pengujian dari pola serangan yang akan dilakukan, hingga penulisan laporan. Kesimpulan dan saran disertakan sebagai catatan atas hasil uji serangan dan kemungkinan untuk perlindungan terhadap pola serangan yang diujikan. Gambar 3.1 mengilustrasikan diagraf alir runtutan pengerjaan penelitian ini:



**Gambar 3.1** Flowchart Runtutan Pengerjaan Penelitian



### 3.1 Studi Literatur

Studi literatur mempelajari mengenai penjelasan dasar teori yang digunakan untuk menunjang penulisan skripsi. Teori-teori pendukung tersebut diperoleh dari buku, jurnal, *e-book* dan dokumentasi *project*.

### 3.2 Penyusunan Dasar Teori

Penyusunan dasar teori dilakukan setelah mendapatkan referensi yang tepat untuk mendukung penulisan penelitian ini. Teori-teori pendukung tersebut meliputi:

1. Komunikasi Jaringan Internet
2. Sniffing
3. Serangan MITM (*Man-in-the-middle*)

### 3.3 Analisa dan Perancangan

Analisa kebutuhan bertujuan untuk mendapatkan semua kebutuhan yang diperlukan oleh pola serangan yang akan dirancang dan diuji. Analisa kebutuhan dilakukan dengan mengidentifikasi kebutuhan pola serangan dan apa saja serta siapa saja yang terlibat didalamnya. Berikut analisa kebutuhan dalam penelitian yang akan dilakukan:

#### 3.3.1 Kebutuhan Jaringan Komputer

Jaringan komputer yang dibutuhkan dalam pola serangan dalam penelitian ini adalah sebagai berikut:

1. Jaringan komputer yang digunakan adalah jaringan LAN (*Local Area Network*).
2. Jaringan komputer yang digunakan pada lingkungan nirkabel (*wireless*)
3. Dapat melakukan koneksi antar komputer pada satu jaringan LAN

### 3.3.2 Kebutuhan Penyerang

Kebutuhan sistem operasi serta fungsi yang harus dimiliki oleh penyerang dalam membuat pola serangan adalah sebagai berikut:

1. Sistem operasi yang digunakan berbasis Linux dan merupakan *distro* turunan Debian yaitu Kali Linux, yang memiliki berbagai *tools* untuk melakukan uji penetrasi (*penetration testing*) pada jaringan komputer.
2. Dapat menggunakan *ARP poisoning* untuk melakukan *sniffing* terhadap perangkat lain yang akan diuji serang.

### 3.3.3 Kebutuhan Target

Kebutuhan sistem operasi serta apa yang harus digunakan oleh target dalam melakukan akses aplikasi berbasis *web* adalah sebagai berikut:

1. Sistem operasi yang digunakan bebas, bisa berbasis Windows, Linux maupun Mac OS yang terhubung pada jaringan LAN dengan penyerang.
2. Akses HTTP atau HTTPS menggunakan *browser* HTML yang ada pada sistem operasi yang digunakan target.

### 3.3.4 Kebutuhan Aplikasi Berbasis Web

Kebutuhan sistem keamanan dari *website* serta pengaksesan informasi dari aplikasi berbasis *web* yang akan diakses oleh target adalah sebagai berikut:

1. Pengaksesan informasi pada aplikasi berbasis *web* adalah dengan memasukkan informasi *login* berupa *username* dan *password* serta penggunaan *session* agar tetap dapat mengakses informasi.
2. Pengamanan aplikasi berbasis *web* menggunakan SSL yang ditandai dengan *https://* pada alamat *web* dari aplikasi tersebut.

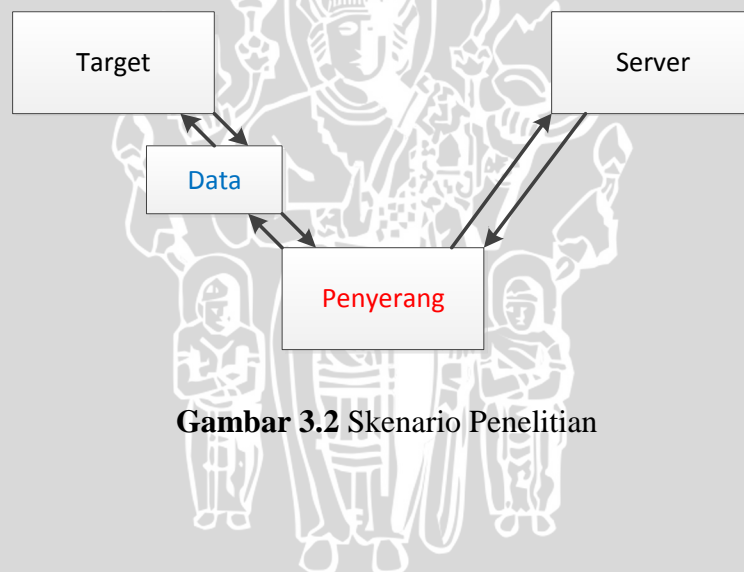
### 3.3.5 Kebutuhan Data

Data yang akan dimanfaatkan pada pola penyerangan yang akan dilakukan dalam penelitian ini adalah:

1. Data berupa teks berisi informasi *username* dan *password*.
2. Data berupa *session* yang digunakan oleh target untuk menjalin komunikasi dengan *server*.

### 3.3.6 Skenario Penelitian

Penelitian ini ditujukan untuk menguji keamanan pada aplikasi berbasis *web* menggunakan pola serangan berbasis *ARP poisoning*. Untuk pengujian dibutuhkan tiga jenis elemen, yaitu Target, Penyerang dan *Server*. Ilustrasi tentang skenario penelitian akan dijelaskan pada gambar 3.2 berikut ini.



Gambar 3.2 Skenario Penelitian

Data pada skenario penelitian diatas dapat berupa teks berisi *username* dan *password* atau *session*. Proses dimulai dari komunikasi antar Target dan Server yang dilanjutkan dengan pengiriman data dari Target ke Server. Pada proses antar Target dan Server tersebut sudah terdapat Penyerang di tengah-tengah yang dapat mengetahui apa saja yang dikirim melalui komunikasi dua elemen tersebut. Data yang dikirim oleh Target dapat ditangkap dan dimanfaatkan oleh Penyerang untuk berkomunikasi dengan Server.



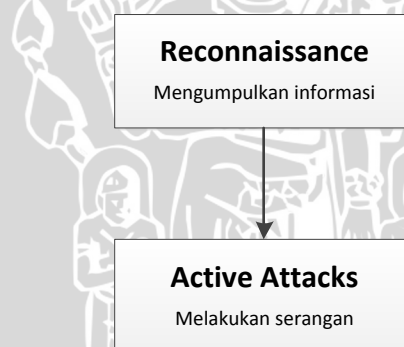
### 3.4 Implementasi

Implementasi pola penyerangan akan dilakukan dengan mengacu pada skenario penelitian, *tools* yang dibutuhkan terdapat pada sistem operasi Kali Linux. Implementasi ini meliputi:

1. Melakukan pengumpulan informasi menggunakan *reconnaissance*
2. Melakukan serangan menggunakan *active attacks* yang terdiri dari ARP *poisoning* untuk melakukan *sniffing* yang disertai penggunaan beberapa metode serangan yaitu penangkapan *traffic*, SSL *hijacking*, *session hijacking* dan DNS *spoofing*

### 3.5 Pengujian

Pengujian keamanan di jaringan lokal dilakukan melalui dua tahap, yaitu *reconnaissance (passive attacks)* dan *active attacks*. Gambar 3.3 berikut adalah diagram alir dari tahap pengujian keamanan pada penelitian ini.



**Gambar 3.3** Tahap Pengujian Jaringan Lokal

#### 3.5.1 Reconnaissance

Tahap ini bertujuan untuk mengumpulkan informasi yang selanjutnya digunakan untuk menentukan jenis dan tujuan serangan terhadap jaringan lokal. Beberapa informasi yang dibutuhkan adalah:

1. Kondisi pengamanan jaringan, yaitu pada *access point* apakah bersifat *open* atau menggunakan pengamanan seperti WEP (*Wired Equivalent Privacy*) maupun WPA (*Wi-Fi Protected Access*)

2. Informasi target serangan, yaitu menentukan target yang akan diserang beserta informasi alamat IP dan alamat MAC
3. Berjalannya protokol ARP, yaitu memeriksa apakah protokol ARP digunakan oleh target sehingga celah tersebut bisa dimanfaatkan untuk melakukan serangan

### 3.5.2 Active Attacks

Tahap ini dilakukan setelah informasi yang dibutuhkan sudah didapat pada tahap *reconnaissance*, serangan dilakukan pada jaringan lokal sesuai dengan informasi tersebut. Pada tahap ini terdiri dari beberapa metode yang memanfaatkan celah protokol ARP seperti *sniffing*, penangkapan *traffic*, *SSL hijacking*, *session hijacking* dan *DNS spoofing*.

#### 3.5.2.1 Pengujian Sniffing

Serangan *sniffing* yang dilakukan pada penelitian ini berupa *ARP poisoning* yang bertujuan untuk mengelabui *client* dan *router* agar semua pengiriman paket ditujukan pada penyerang terlebih dahulu sebelum diteruskan kepada penerima paket yang sebenarnya. Serangan ini menggunakan *tool* Ettercap dan menjadi dasar dari empat metode serangan yang akan diujikan selanjutnya.

#### 3.5.2.2 Pengujian Penangkapan Traffic

Setelah *ARP poisoning* berhasil dilakukan dengan menggunakan *tool* Ettercap, metode serangan pertama dilakukan dengan melakukan penangkapan *traffic* menggunakan *tool* Wireshark. Wireshark pada Penyerang akan menangkap semua *traffic* data antara Target dan Server, informasi yang akan diambil adalah teks yang berisi *username* dan *password* yang berbentuk *cleartext* atau teks yang belum dienkripsi. *Cleartext* biasanya terdapat pada *post* HTTP yang belum aman. Contoh tabel pengujian akan ditunjukkan pada tabel 3.1.

**Tabel 3.1** Tabel Pengujian Pola Serangan Penangkapan *Traffic*

No.	Aplikasi	Alamat IP	Traffic HTTP ditemukan?
1			
...			
10			

### 3.5.2.3 Pengujian SSL Hijacking

Metode serangan kedua dilakukan dengan melakukan SSL *hijacking* menggunakan *tool* SSLStrip. SSLStrip pada Penyerang akan meneruskan koneksi aman antara Target dan Server melalui HTTPS, menggantikan posisi Target dengan Penyerang serta mengembalikan halaman HTTP ke Target. Target hanya akan melakukan *post* dalam bentuk *cleartext* karena koneksi aman (dalam bentuk *secure post*) baru dimulai antara Penyerang dan Server, dari sinilah Penyerang dapat mendapatkan teks yang berisi *username* dan *password*. Contoh tabel pengujian akan ditunjukkan pada tabel 3.2.

**Tabel 3.2** Tabel Pengujian Pola Serangan SSL *Hijacking*

No.	Aplikasi	Alamat IP	Cleartext ditemukan?
1			
...			
10			



### 3.5.2.4 Pengujian Session Hijacking

Metode serangan ketiga dilakukan dengan melakukan *session hijacking* menggunakan *tool* Ferret dan Hamster. Penyerang menggunakan kedua *tools* ini secara berkesinambungan, Ferret digunakan untuk menangkap *session* yang dibuat oleh Target untuk mengakses informasi pada Server, misalnya informasi data akun dalam keadaan *logged in*. Setelah *session* ditangkap, Hamster digunakan untuk menjalankan *service* lokal dalam menampilkan halaman *web* dengan *session* yang sebelumnya sudah ditangkap tersebut, dengan ini Penyerang dapat melakukan *login* dengan akun yang dimiliki target. Contoh tabel pengujian akan ditunjukkan pada tabel 3.3.

**Tabel 3.3** Tabel Pengujian Pola Serangan *Session Hijacking*

No.	Aplikasi	Alamat IP	Berhasil akses akun?
1			
...			
10			

### 3.5.2.5 Pengujian DNS Spoofing

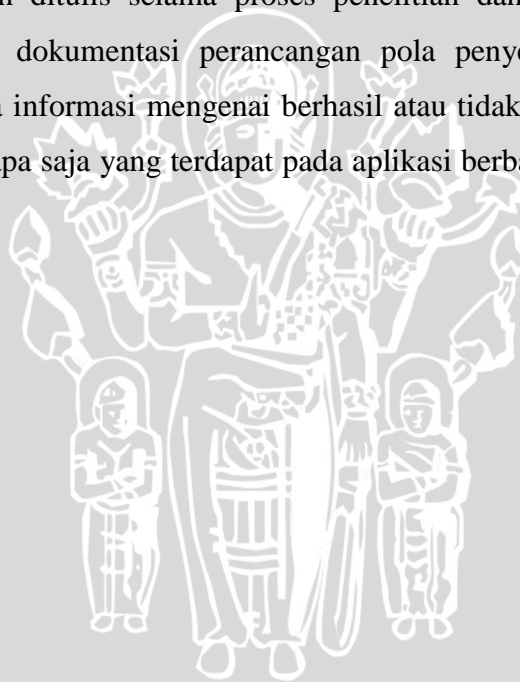
Metode serangan keempat dilakukan dengan melakukan *DNS spoofing* menggunakan *tool* Ettercap yang disertai dengan *plugin* *dns\_spoof* serta SEToolkit. Penyerang memanfaatkan *dns\_spoof* agar dapat memberikan *DNS reply* kepada Target yang melakukan *DNS request*, dengan berisikan informasi alamat IP yang dimiliki Penyerang sebagai alamat logis sebuah *website*. Dengan hal tersebut, Target akan menuju Penyerang ketika mengetikkan sebuah alamat *website*, disamping itu penyerang akan menggunakan SEToolkit untuk melakukan *clone* terhadap sebuah *website* serta menampilkan halaman hasil *clone* kedalam sebuah *service* lokal. Contoh tabel pengujian akan ditunjukkan pada tabel 3.4.

**Tabel 3.4** Tabel Pengujian Pola Serangan DNS *Spoofing*

No.	Aplikasi	Alamat IP	Cleartext ditemukan?
1			
...			
10			

### 3.6 Penulisan Laporan

Laporan penelitian ditulis selama proses penelitian dan pengerjaan tugas akhir. Laporan berisi dokumentasi perancangan pola penyerangan dan hasil penelitian yang berupa informasi mengenai berhasil atau tidaknya serangan yang dilakukan serta celah apa saja yang terdapat pada aplikasi berbasis *web* yang diuji serang.



## BAB IV

### IMPLEMENTASI DAN PENGUJIAN

Pada bab ini akan dijelaskan langkah-langkah yang akan dilakukan dalam melakukan implementasi dan melakukan uji serangan dengan menggunakan metode *sniffing* dengan *ARP poisoning*. Adapun jenis serangan yang dilakukan dibagi kedalam empat serangan sesuai dengan pengamanan pada aplikasi yang diuji serang.

Pada tahap implementasi, langkah-langkah yang akan dilakukan penulis antara lain instalasi dan konfigurasi yang mengacu pada sistem operasi yang digunakan. Pada tahap pengujian, penulis akan melakukan uji serang pada aplikasi yang diterapkan pada beberapa lingkungan jaringan komputer.

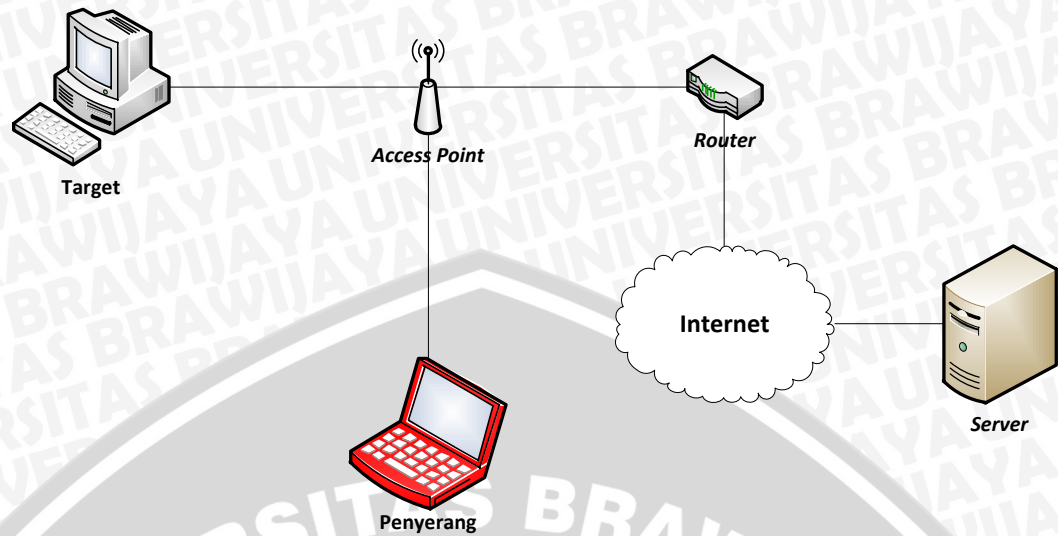
#### 4.1 Implementasi

Implementasi dari uji serang yang akan dilakukan terdiri dari tiga bagian, yaitu implementasi skenario jaringan, *reconnaissance* dan *active attacks*.

##### 4.1.1 Implementasi Skenario Jaringan

Uji serang akan dilakukan pada sebuah jaringan lokal yang didalamnya terdapat dua buah perangkat, yaitu perangkat yang digunakan oleh target dan perangkat yang digunakan oleh penyerang. Jaringan lokal terhubung dengan *server* sebagai tujuan dari target serta sebagai penyedia layanan atau aplikasi berbasis *web* yang akan diuji serang. Gambar 4.1 akan menunjukkan bagaimana skenario jaringan dibuat.





**Gambar 4.1** Skenario Jaringan

Pada perangkat yang digunakan oleh penyerang terpasang sistem operasi berbasis Linux yaitu Kali Linux 1.0.5 i386 dan terhubung pada satu jaringan lokal dengan target. Sistem operasi ini bersifat *free* dan *open source*, jadi penyerang dapat menggunakan, memanfaatkan *tools* yang ada didalamnya, serta melakukan modifikasi pada sistem operasi tersebut secara bebas.

Pada perangkat penyerang terdapat beberapa *tools* yang disiapkan untuk melakukan uji serang, diantaranya adalah *tool* untuk melakukan *ARP poisoning* yaitu Ettercap serta *tools* untuk melakukan uji serang yaitu Wireshark, SSLStrip, Ferret, Hamster dan SEToolkit.

Pada perangkat yang digunakan oleh target terpasang sistem operasi berbasis Windows yaitu Windows 7 64 bit dan terhubung pada satu jaringan lokal dengan penyerang.

Pada perangkat target terdapat browser yaitu Google Chrome dan Mozilla Firefox yang digunakan untuk melakukan akses ke aplikasi berbasis *web* yang ada pada *server* tujuan.

#### 4.1.2 Implementasi Reconnaissance

Tahap *reconnaissance* dilakukan untuk mengumpulkan informasi yang sudah ditentukan sebelumnya pada metodologi penelitian. Tabel 4.1 adalah beberapa informasi yang dibutuhkan serta *tool* yang digunakan untuk mendapatkan informasi tersebut:

**Tabel 4.1** Kebutuhan Informasi

No.	Jenis Informasi	Tool Yang Digunakan
1	Kondisi pengamanan jaringan	Airmon-ng Airodump-ng
2	Informasi target serangan	Ettercap
3	Berjalannya protokol ARP	ARPing

Airmon-ng adalah *tool* yang digunakan untuk mengaktifkan mode *monitor* pada *wireless network card* dan Airodump-ng adalah *tool* yang digunakan untuk melakukan *monitor* pada jaringan *wireless*. Informasi alamat IP dan MAC dari target dan *router* didapatkan dari pemindai jaringan yang dimiliki Ettercap serta informasi bahwa adanya protokol ARP yang berjalan didapatkan dari penggunaan ARPing.

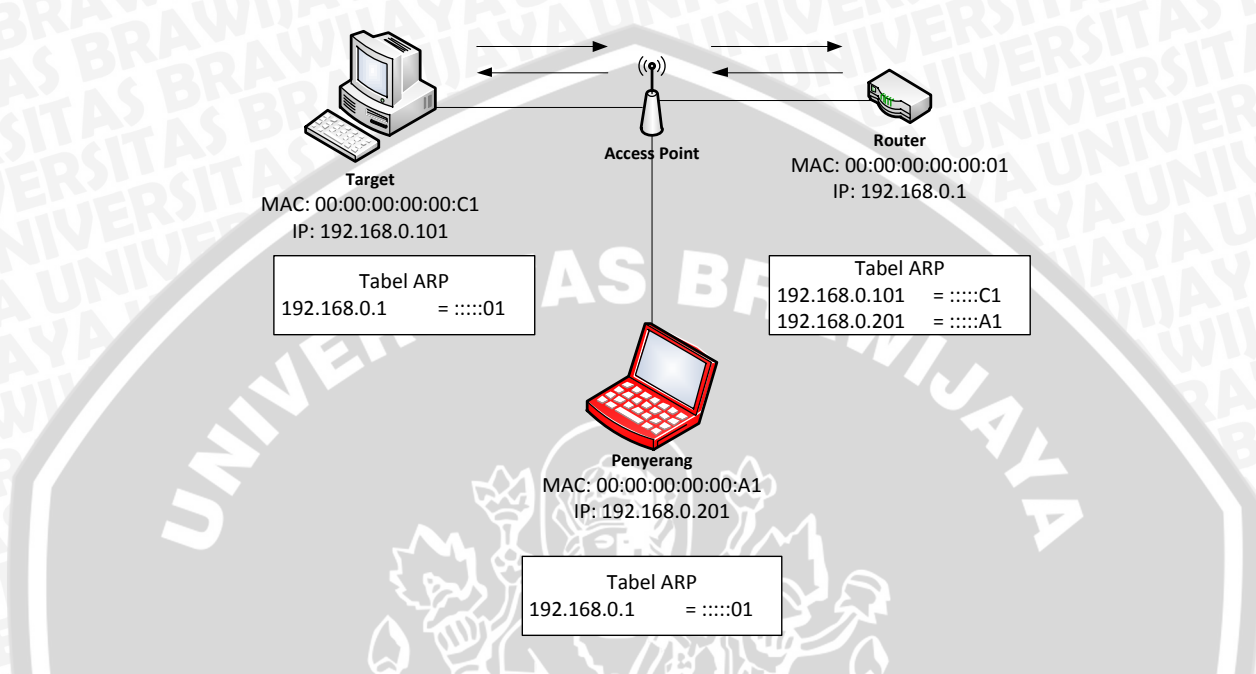
#### 4.1.3 Implementasi Active Attacks

Tahap *active attacks* dilakukan setelah informasi sudah didapatkan dari tahap *reconnaissance*, pada tahap ini penyerang akan melakukan serangan kepada target dan *router* yang sudah ditentukan sebelumnya. Metode serangan pada tahap ini berupa *sniffing*, penangkapan *traffic*, *SSL hijacking*, *session hijacking* dan *DNS spoofing*.

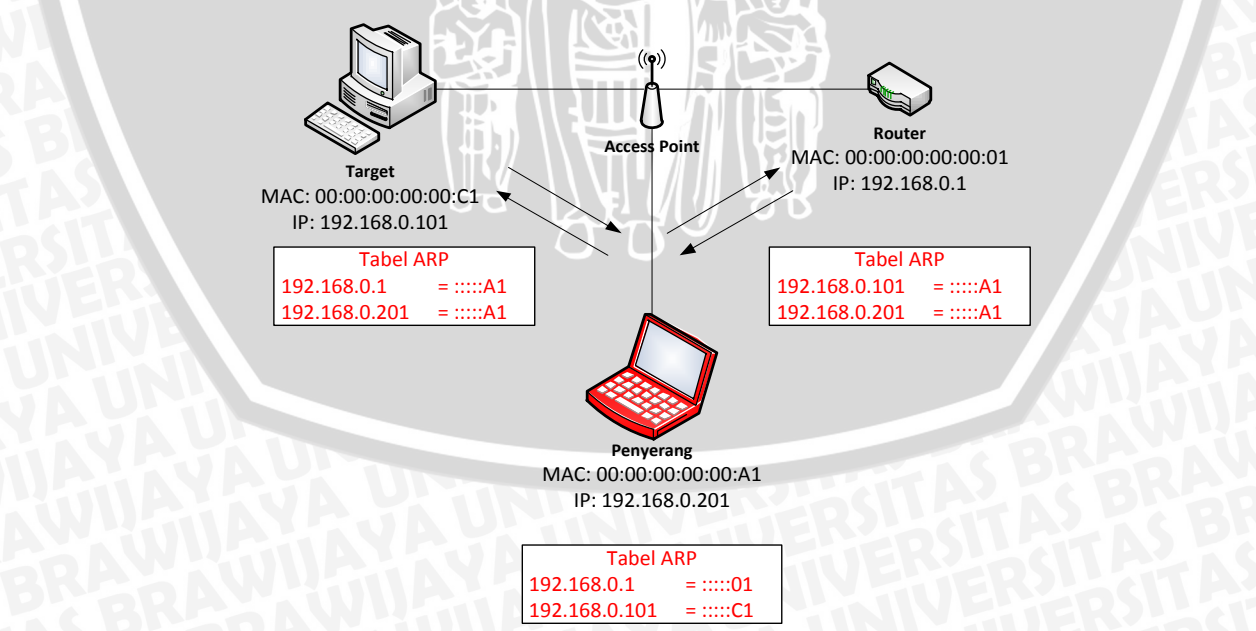
##### 4.1.3.1 Sniffing

*Sniffing* dilakukan oleh penyerang dengan menggunakan metode *ARP poisoning*, dimana penyerang berusaha untuk merubah tabel ARP yang dimiliki oleh target dan *router* sehingga semua *traffic* antara target dan *router* akan diarahkan melalui penyerang.

ARP poisoning dilakukan dengan menggunakan *tool* Ettercap yang sudah terdapat didalam sistem operasi Kali Linux. Gambar 4.2 akan menunjukkan bagaimana tabel ARP berjalan secara normal dan Gambar 4.3 akan menunjukkan bagaimana ARP poisoning dilakukan serta bagaimana tabel ARP dirubah.



Gambar 4.2 Tabel ARP Normal



Gambar 4.3 Tabel ARP Setelah Proses Poisoning



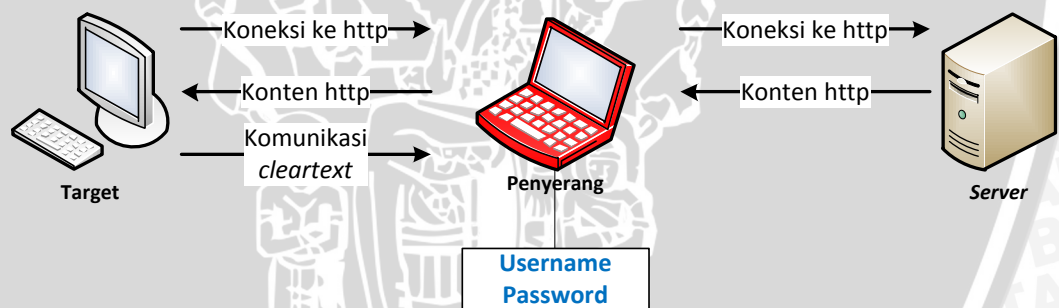


Setelah tabel ARP yang dimiliki oleh target dan router sudah berhasil dirubah (*sniffing* berhasil), langkah selanjutnya adalah melakukan uji serang pada target yang mengakses aplikasi berbasis *web*.

#### 4.1.3.2 Penangkapan Traffic

Serangan ini dilakukan dengan menggunakan *tool* Wireshark yang berfungsi untuk menangkap semua *traffic* pada sebuah interface yang digunakan perangkat untuk terhubung pada sebuah jaringan. *Traffic* yang ditangkap dapat berupa *traffic* masuk maupun *traffic* keluar, namun pada pengujian ini, data yang dicari adalah data *cleartext* yang berisi informasi *username* dan *password*.

Serangan ini tidak dapat menangkap *traffic* yang aman seperti SSL (HTTPS), sehingga hanya dapat dilakukan pada saat target mengakses aplikasi berbasis *web* yang tidak menerapkan protokol keamanan. Pada gambar 4.4 akan ditunjukkan bagaimana implementasi penangkapan *traffic* dilakukan.

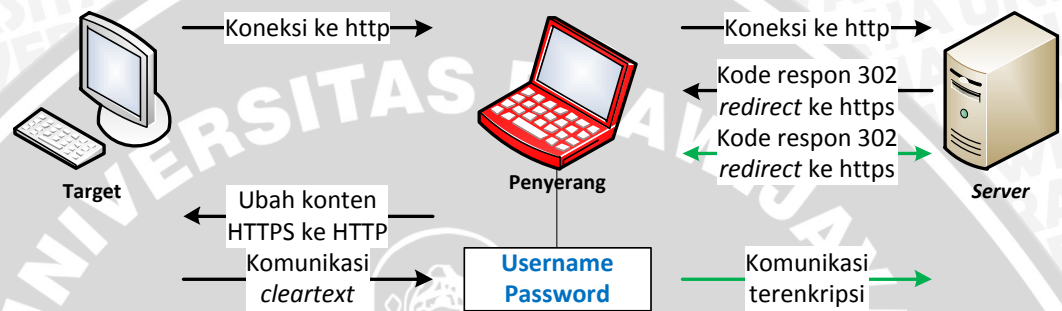


Gambar 4.4 Implementasi Penangkapan Traffic

#### 4.1.3.3 SSL Hijacking

Serangan ini dilakukan dengan menggunakan *tool* SSLStrip yang berfungsi untuk meneruskan koneksi aman dari target menuju ke *server*. SSLStrip pada penyerang akan berkomunikasi dengan *server* melalui jalur aman (HTTPS) dan mengembalikan konten yang didapat kepada target melalui jalur tidak aman (HTTP).

Target akan mendapatkan konten melalui jalur tidak aman dan akan mengirimkan sebuah *cleartext* yang berisi *username* dan *password*, dikarenakan koneksi aman baru dimulai antar penyerang dan *server*. Serangan ini dapat dilakukan pada aplikasi berbasis *web* yang menerapkan protokol keamanan seperti SSL (HTTPS) dikarenakan fungsinya yang mencegah target mengakses *server* melalui koneksi aman. Pada gambar 4.5 akan ditunjukkan bagaimana implementasi SSL *hijacking* dilakukan.



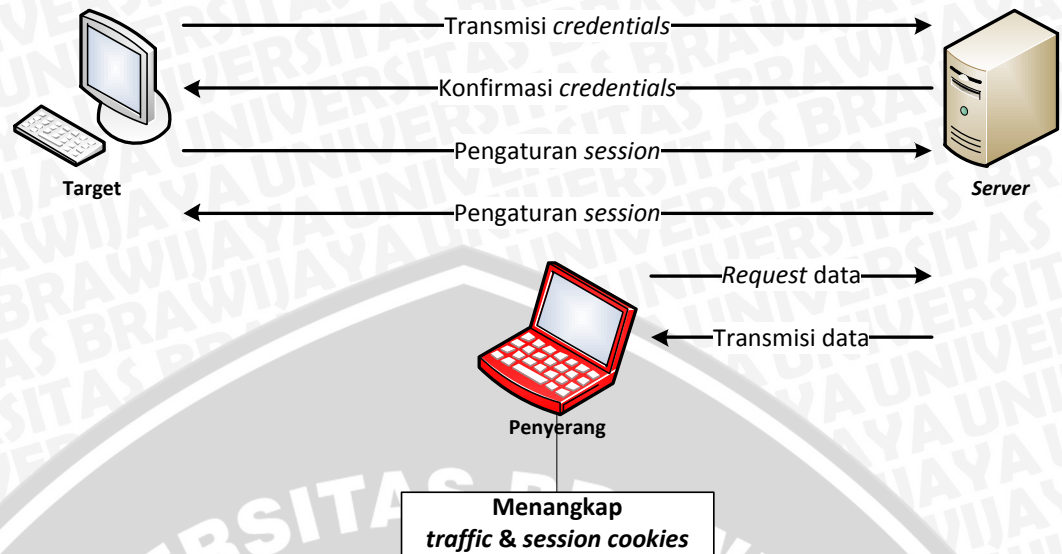
Gambar 4.5 Implementasi SSL Hijacking

#### 4.1.3.4 Session Hijacking

Serangan ini dilakukan dengan menggunakan kombinasi *tool* Ferret dan Hamster yang bekerja secara berkesinambungan. Ferret berguna untuk menangkap *session* yang dibuat antar target dan *server* dengan jalan otentikasi *login*, selanjutnya *session* yang ditangkap akan dimasukkan kedalam sebuah *file* berekstensi *pcap*. Setelah *session* berhasil ditangkap, Hamster akan memproses *session* tersebut dan menampilkan pada *service* lokal di perangkat penyerang. Penyerang dapat mengakses aplikasi berbasis *web* yang ada dengan menggunakan akun target dimana *session* dari target tersebut sudah didapatkan oleh penyerang.

Serangan ini dapat dilakukan pada aplikasi berbasis *web* yang tidak menerapkan protokol keamanan pada halaman otentikasi (*login*), dikarenakan *session* dapat mudah ditangkap jika tidak ada protokol keamanan tersebut. Pada gambar 4.6 akan ditunjukkan bagaimana implelementasi *session hijacking* dilakukan.





Gambar 4.6 Implementasi *Session Hijacking*

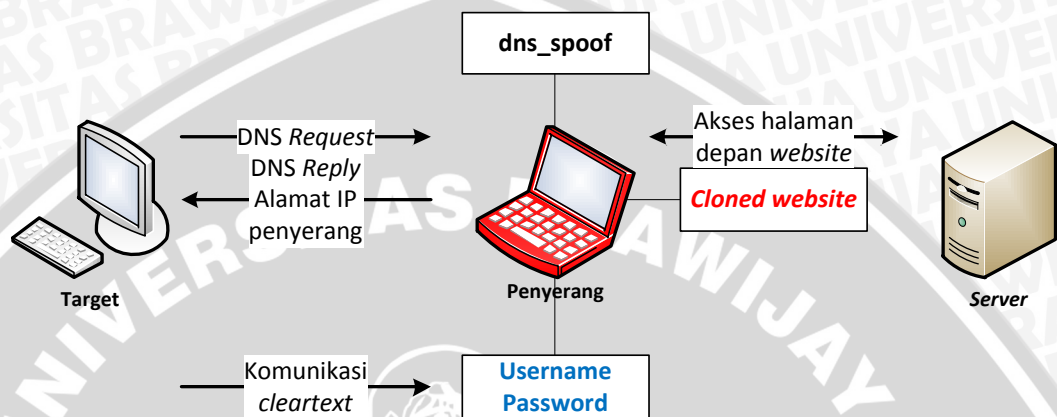
#### 4.1.3.5 DNS Spoofing

Serangan ini dilakukan dengan memanfaatkan *plugin dns\_spoof* yang ada pada *tool* Ettercap, dengan mengaktifkan *plugin* tersebut penyerang akan dapat mengirimkan DNS *reply* kepada target yang sedang melakukan DNS *request*. Penyerang dapat menentukan sebuah DNS memiliki alamat IP penyerang dengan menambahkan informasi tersebut pada *file* etter.dns dalam *folder* Ettercap. Jadi, pada saat target menuliskan sebuah alamat aplikasi berbasis *web*, target tersebut akan menuju ke alamat IP penyerang bukan alamat IP *server*.

Selain *plugin* diatas, penyerang menggunakan *tool* SEToolkit yang berguna untuk melakukan *clone* pada sebuah halaman aplikasi berbasis *web* dan menjalankannya pada *service* lokal penyerang. Saat target menuju alamat IP penyerang dan memasukkan *username* serta *password* pada halaman *clone* yang dibuat sebelumnya, informasi tersebut akan dapat ditangkap oleh penyerang.



Serangan ini dapat dilakukan pada aplikasi berbasis *web* yang tidak menerapkan protokol keamanan maupun yang menerapkan, dikarenakan fungsinya yang hanya melakukan *clone* aplikasi berbasis *web* apapun dan ditampilkan pada saat target mengakses. Pada gambar 4.7 akan ditunjukkan bagaimana implementasi DNS *spoofing* dilakukan.



**Gambar 4.7** Implementasi DNS *Spoofing*

## 4.2 Pengujian

Pengujian dilakukan dengan dua lingkungan yang masing-masing memiliki dua tahap, lingkungan terdiri dari lingkungan jaringan dengan konfigurasi standar (rumah) dan lingkungan jaringan dengan konfigurasi lanjut (kampus). Tahap pengujian pada masing-masing lingkungan yaitu tahap pengujian *sniffing* dan uji serang yang terdiri dari pengujian penangkapan *traffic*, *SSL hijacking*, *session hijacking* serta *DNS spoofing*.

Setelah rencana tahap pengujian sudah ditentukan, maka dapat ditentukan pula aplikasi berbasis *web* milik Universitas Brawijaya yang akan diuji serang dengan masing-masing pola serangan yang sudah dibuat, aplikasi tersebut adalah:

1. nas.ub.ac.id
2. bais.ub.ac.id
3. siam.ub.ac.id
4. gapura.ub.ac.id

5. blog.ub.ac.id
6. siado.ub.ac.id
7. e-complaint.ub.ac.id

#### 4.2.1 Lingkungan Jaringan Konfigurasi Standar

Pengujian pertama dilakukan di lingkungan jaringan dengan konfigurasi standar, seperti di jaringan nirkabel rumah ataupun jaringan nirkabel umum yang tersedia diluar rumah. Jaringan nirkabel rumah atau umum dapat dinyatakan jaringan dengan konfigurasi standar karena biasanya hanya menggunakan *access point* yang disediakan oleh *internet provider* dan sudah terkonfigurasi secara standar oleh teknisi dari provider tersebut. Lingkungan jaringan dengan konfigurasi standar yaitu jaringan rumah dipilih untuk lingkungan pengujian pertama.

##### 4.2.1.1 Reconnaissance

Tahap *reconnaissance* atau pengumpulan data dilakukan sebelum memasuki tahap *active attacks* yang terdiri dari pengujian *sniffing*, penangkapan *traffic*, *SSL hijacking*, *session hijacking* dan *DNS spoofing*. Pada tahap ini terdapat tiga informasi yang dibutuhkan yaitu:

1. Kondisi pengamanan jaringan

Kondisi pengamanan yang diterapkan pada jaringan yang akan dimasuki oleh penyerang dapat diketahui dengan menggunakan *tool* Airmon-ng dan Airodump-ng. Pada gambar 4.8 akan ditunjukkan bahwa penyerang membuat sebuah *interface* yang digunakan untuk melakukan *monitor* jaringan nirkabel yang ada di sekitar serta pada gambar 4.9 akan ditunjukkan adanya sebuah jaringan nirkabel dengan SSID (*Service Set Identifier*) "TANIA" yang bersifat *open* dan dapat dimasuki oleh penyerang serta memiliki alamat MAC CC:1A:FA:9B:3D:B0.

```

root@kali: ~
File Edit View Search Terminal Help
root@kali:~# airmon-ng start wlan0

Found 2 processes that could cause trouble.
If airodump-ng, aireplay-ng or airtun-ng stops working after
a short period of time, you may want to kill (some of) them!
-e
PID      Name
2492    NetworkManager
2789    wpa_supplicant

Interface      Chipset      Driver
wlan0          Broadcom    b43 - [phy0]
              (monitor mode enabled on mon0)

```

Gambar 4.8 Tampilan Airmon-ng

```

root@kali: ~
File Edit View Search Terminal Help
CH 2 ][ Elapsed: 32 s ][ 2014-07-03 17:44

BSSID          PWR  Beacons    #Data, #/s  CH  MB  ENC  CIPHER AUTH ESSID
CC:1A:FA:9B:3D:B0 -47   70         212  0  1  54e  OPN           TANIA

BSSID          STATION    PWR   Rate    Lost    Frames  Probe
(not associated) EC:89:F5:75:E7:D5 -52   0 - 1    0       24
CC:1A:FA:9B:3D:B0 9C:2A:70:11:69:D1 -23   0 - 1    8      232  TANIA

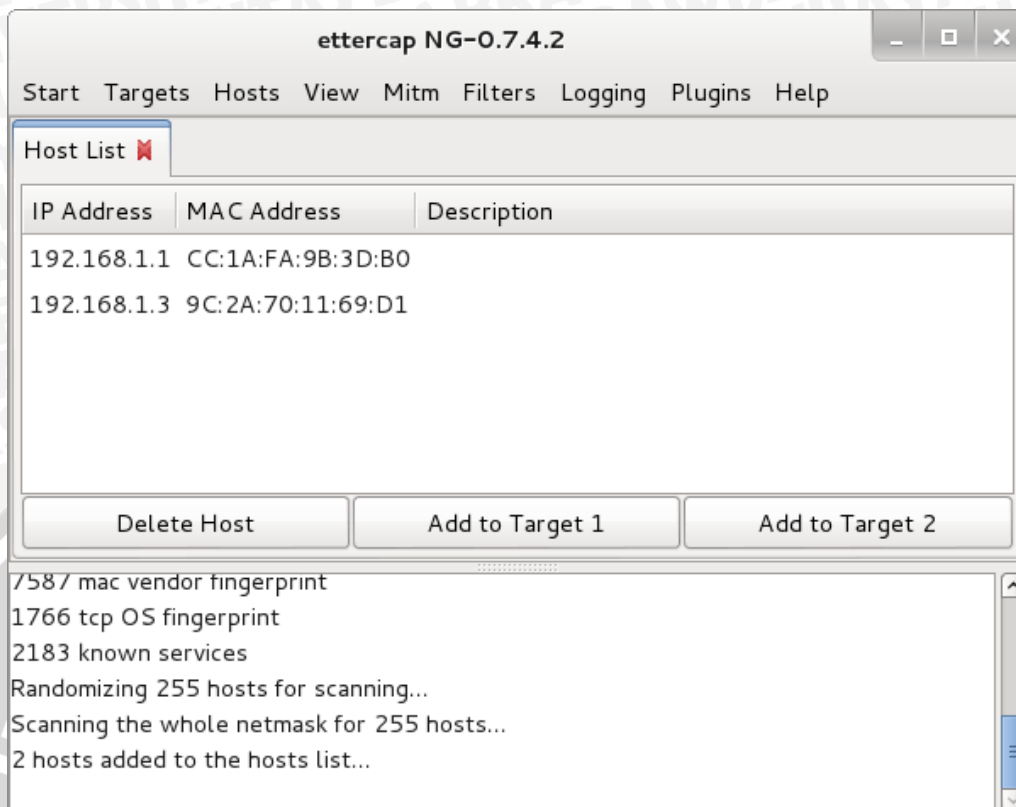
```

Gambar 4.9 Tampilan Airodump-ng

## 2. Informasi target serangan

Informasi target yang akan diserang didapatkan dari *host scanning* yang terdapat pada *tool* Ettercap. Pada gambar 4.10 akan ditunjukkan jendela dari *tool* Ettercap yang menunjukkan informasi alamat IP dan MAC dari *router* dan sebuah *client* yang selanjutnya akan dipilih menjadi target.

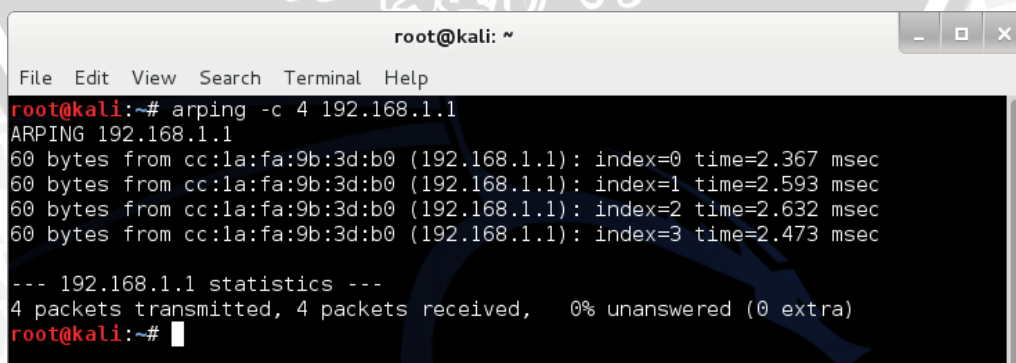




**Gambar 4.10** Tampilan Informasi Host yang Terhubung di Jaringan

3. Berjalannya protokol ARP

Penyerang akan memastikan bahwa protokol ARP pada target berjalan agar *sniffing* (ARP *poisoning*) dapat dilakukan. Pada gambar 4.11 dan 4.12 akan ditunjukkan proses ARPing yang dilakukan pada *router* dengan alamat IP 192.168.1.1 dan target dengan alamat IP 192.168.1.3.



**Gambar 4.11** ARPing pada Router

```

root@kali: ~
File Edit View Search Terminal Tabs Help

root@kali: ~ x root@kali: ~ x root@kali: ~ x root@kali: ~ x
root@kali:~# arping -c 4 192.168.1.3
ARPING 192.168.1.3
42 bytes from 9c:2a:70:11:69:d1 (192.168.1.3): index=0 time=3.284 msec
42 bytes from 9c:2a:70:11:69:d1 (192.168.1.3): index=1 time=3.243 msec
42 bytes from 9c:2a:70:11:69:d1 (192.168.1.3): index=2 time=3.204 msec
42 bytes from 9c:2a:70:11:69:d1 (192.168.1.3): index=3 time=3.223 msec

--- 192.168.1.3 statistics ---
4 packets transmitted, 4 packets received, 0% unanswered (0 extra)
root@kali:~#

```

Gambar 4.12 ARPing pada Target

#### 4.2.1.2 Pengujian Sniffing

Uji *sniffing* dilakukan sebelum penyerang melakukan uji serang pada target, hal ini bertujuan agar penyerang dapat berada di tengah-tengah komunikasi antara target dan *server*. Berikut ini adalah tampilan dari alamat IP yang dimiliki oleh target dan penyerang, dimana target memiliki alamat IP 192.168.1.3 dan penyerang memiliki alamat IP 192.168.1.5 serta router memiliki alamat IP 192.168.1.1.

```

ca: C:\Windows\system32\cmd.exe

Wireless LAN adapter Wireless Network Connection:

Connection-specific DNS Suffix . . :
Link-local IPv6 Address . . . . . : fe80::a955:7a54:3951:757b%13
IPv4 Address. . . . . : 192.168.1.3
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.1.1

Ethernet adapter Local Area Connection:

Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :

Tunnel adapter isatap.{2D735D4C-99A7-4201-8276-0AE371C09A02}:

Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :

Tunnel adapter Teredo Tunneling Pseudo-Interface:

Connection-specific DNS Suffix . . :
IPv6 Address. . . . . : 2001:0:9d38:90d7:30ac:b284:9174:b314
Link-local IPv6 Address . . . . . : fe80::30ac:b284:9174:b314%11
Default Gateway . . . . . :

```

Gambar 4.13 Alamat IP Target

```

root@kali: ~
File Edit View Search Terminal Tabs Help

root@kali: ~ x root@kali: ~ x
UP LOOPBACK RUNNING MTU:65536 Metric:1
RX packets:108 errors:0 dropped:0 overruns:0 frame:0
TX packets:108 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:8400 (8.2 KiB) TX bytes:8400 (8.2 KiB)

mon0
-00
Link encap:UNSPEC HWaddr 00-21-00-52-35-5A-00-00-00-00-00-00-00-00-00-00

UP BROADCAST NOTRAILERS RUNNING PROMISC ALLMULTI MTU:1500 Metric:1
RX packets:1835 errors:0 dropped:1329 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:366592 (358.0 KiB) TX bytes:0 (0.0 B)

wlan0
Link encap:Ethernet HWaddr 00:21:00:52:35:5a
inet addr:192.168.1.5 Bcast:192.168.1.255 Mask:255.255.255.0
inet6 addr: fe80::221:ff:fe52:355a/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:13 errors:0 dropped:0 overruns:0 frame:0
TX packets:17 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:1536 (1.5 KiB) TX bytes:2198 (2.1 KiB)

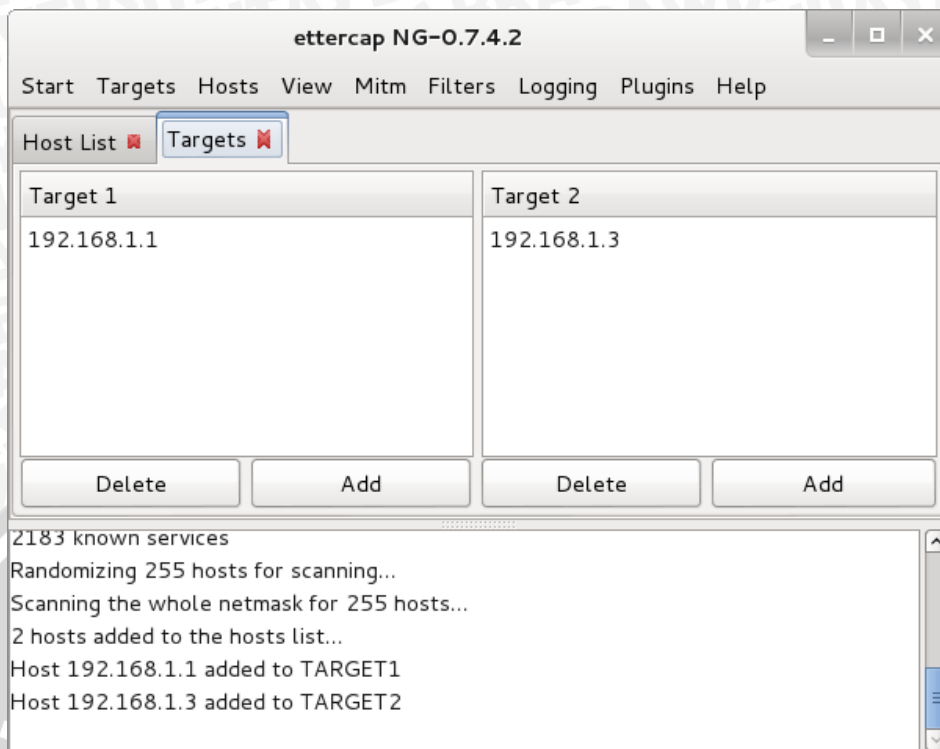
root@kali:~#

```

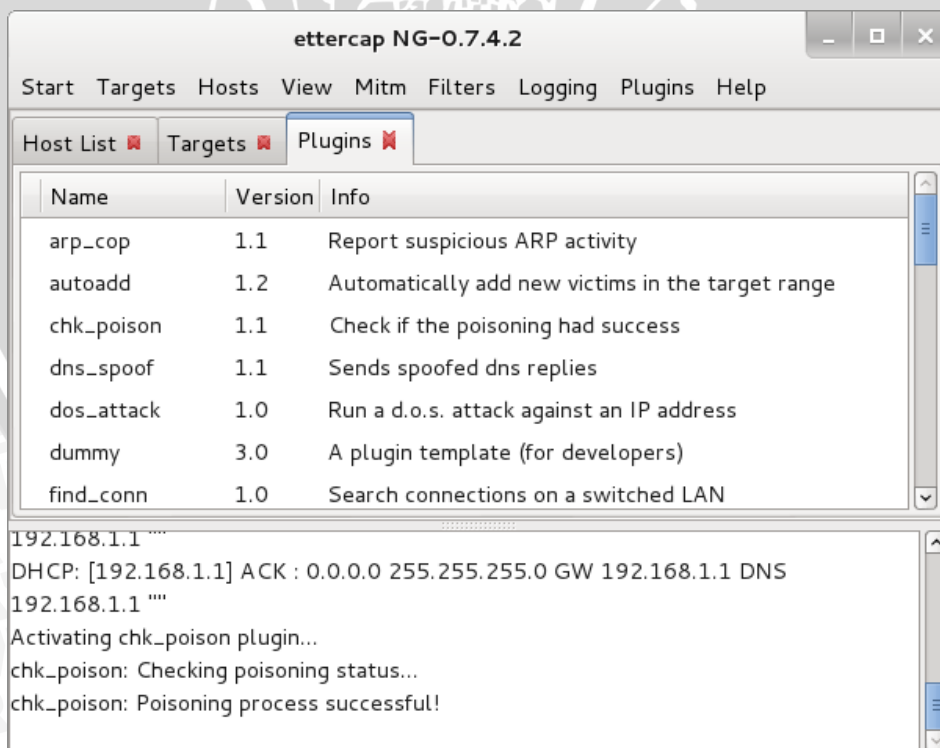
**Gambar 4.14** Alamat IP Penyerang

Penyerang melakukan *sniffing* pada target dengan mendaftarkan alamat IP *router* pada kolom target 1 serta alamat IP target pada kolom target 2. Setelah kedua alamat IP tersebut didaftarkan, penyerang melakukan ARP *poisoning* lalu melakukan cek keberhasilan *sniffing* dengan *plugin* `chk_poison`.



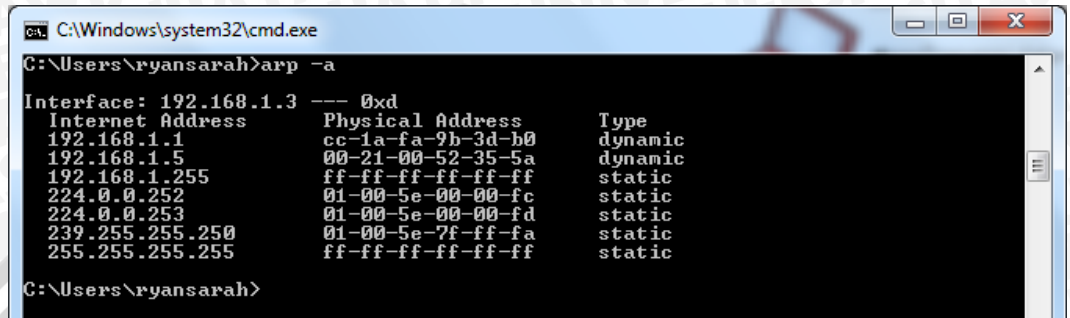


**Gambar 4.15** Penyerang Menentukan Target



**Gambar 4.16** Penyerang Melakukan Cek ARP Poisoning

Keberhasilan dari *sniffing* ditunjukkan pada perubahan tabel ARP (ARP *cache*) yang dimiliki oleh target, gambar 4.17 akan menunjukkan tabel ARP yang normal dan gambar 4.18 akan menunjukkan tabel ARP yang sudah dikenai ARP *poisoning*.



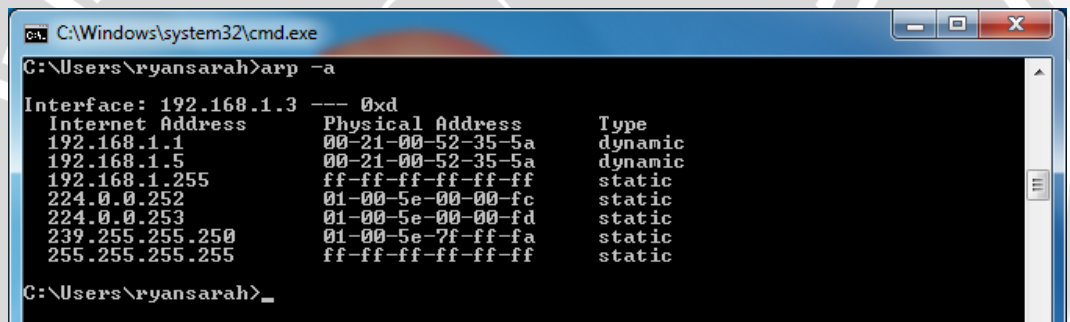
```

C:\Windows\system32\cmd.exe
C:\Users\ryansarah>arp -a

Interface: 192.168.1.3 --- 0xd
Internet Address      Physical Address      Type
192.168.1.1          00-1a-fa-9b-3d-b0    dynamic
192.168.1.5          00-21-00-52-35-5a    dynamic
192.168.1.255        ff-ff-ff-ff-ff-ff    static
224.0.0.252          01-00-5e-00-00-fc    static
224.0.0.253          01-00-5e-00-00-fd    static
239.255.255.250      01-00-5e-7f-ff-fa    static
255.255.255.255      ff-ff-ff-ff-ff-ff    static

C:\Users\ryansarah>
  
```

Gambar 4.17 Tabel ARP Normal



```

C:\Windows\system32\cmd.exe
C:\Users\ryansarah>arp -a

Interface: 192.168.1.3 --- 0xd
Internet Address      Physical Address      Type
192.168.1.1          00-21-00-52-35-5a    dynamic
192.168.1.5          00-21-00-52-35-5a    dynamic
192.168.1.255        ff-ff-ff-ff-ff-ff    static
224.0.0.252          01-00-5e-00-00-fc    static
224.0.0.253          01-00-5e-00-00-fd    static
239.255.255.250      01-00-5e-7f-ff-fa    static
255.255.255.255      ff-ff-ff-ff-ff-ff    static

C:\Users\ryansarah>_
  
```

Gambar 4.18 Tabel ARP Ter-*poison*

Pada gambar 4.18 terlihat bahwa alamat MAC yang dimiliki *router* berubah menjadi sama dengan alamat MAC yang dimiliki oleh penyerang, dengan demikian maka semua pengiriman paket data dari target maupun menuju target akan melewati *device* yang digunakan oleh penyerang.

#### 4.2.1.3 Pengujian Penangkapan Traffic

Uji penangkapan *traffic* bertujuan untuk mengetahui seberapa besar tingkat keberhasilan ARP *poisoning* yang berjalan di lingkungan jaringan dengan konfigurasi standar. Penyerang dapat melihat *traffic* yang melintas antara *client* dan *server* dengan menggunakan *tool* Wireshark, pengujian ini menangkap *traffic* HTTP yang mengandung informasi penting dari sebuah aplikasi berbasis *web*. Berikut ini adalah daftar aplikasi yang diuji serang menggunakan metode penangkapan *traffic*:

**Tabel 4.2** Hasil Uji Penangkapan *Traffic*

No.	Aplikasi	Alamat IP	Traffic HTTP ditemukan?
1	bais.ub.ac.id	175.45.184.222	Tidak
2	siam.ub.ac.id	175.45.184.86	Tidak
3	gapura.ub.ac.id	175.45.184.74	Tidak
4	blog.ub.ac.id	175.45.184.5	Tidak
5	siado.ub.ac.id	175.45.184.104	Tidak
6	e-complaint.ub.ac.id	175.45.184.75	Ya

Aplikasi nas.ub.ac.id merupakan aplikasi berbasis *web* yang khusus digunakan didalam jaringan kampus UB sebagai otentikasi mahasiswa, jadi aplikasi tersebut tidak dapat diakses diluar jaringan kampus UB dan tidak diujikan di lingkungan jaringan dengan konfigurasi standar. *Traffic* HTTP yang berhasil ditangkap adalah pada saat target melakukan akses *login* melalui aplikasi e-complaint.ub.ac.id, gambar 4.19 akan menunjukkan tampilan informasi yang berhasil ditangkap oleh penyerang.



89624	1381.39187	192.168.1.3	175.45.184.75	HTTP	911	[TCP Retransmission]	GET /assets/css/~boot	
89625	1381.39227	192.168.1.3	175.45.184.75	HTTP	911	[TCP Retransmission]	GET /assets/css/~boot	
89897	1386.91302	192.168.1.3	175.45.184.75	HTTP	1039	POST /login.html	HTTP/1.1 (application/x-	
89898	1386.91306	192.168.1.3	175.45.184.75	HTTP	1039	[TCP Retransmission]	POST /login.html	HTTP
89899	1386.91336	192.168.1.3	175.45.184.75	HTTP	1039	[TCP Retransmission]	POST /login.html	HTTP
89940	1389.03847	192.168.1.3	175.45.184.75	HTTP	1281	GET /index.php/borang/form_civitas.html	HT	
89941	1389.03850	192.168.1.3	175.45.184.75	HTTP	1281	[TCP Retransmission]	GET /index.php/borang	
89942	1389.03873	192.168.1.3	175.45.184.75	HTTP	1281	[TCP Retransmission]	GET /index.php/borang	
89994	1389.38580	192.168.1.3	175.45.184.75	HTTP	1237	GET /assets/css/~bootstrap.css	HTTP/1.1	
89995	1389.38582	192.168.1.3	175.45.184.75	HTTP	1237	[TCP Retransmission]	GET /assets/css/~boot	
89996	1389.38606	192.168.1.3	175.45.184.75	HTTP	1237	[TCP Retransmission]	GET /assets/css/~boot	
90006	1389.39337	192.168.1.3	175.45.184.75	HTTP	1248	GET /assets/css/~bootstrap-responsive.css	I	
90007	1389.39340	192.168.1.3	175.45.184.75	HTTP	1248	[TCP Retransmission]	GET /assets/css/~boot	
90008	1389.39367	192.168.1.3	175.45.184.75	HTTP	1248	[TCP Retransmission]	GET /assets/css/~boot	
<ul style="list-style-type: none"> <li>▣ Frame 89897: 1039 bytes on wire (8312 bits), 1039 bytes captured (8312 bits) on interface 0</li> <li>▣ Ethernet II, Src: HonHaiPr_11:69:d1 (9c:2a:70:11:69:d1), Dst: GemtekTe_52:35:5a (00:21:00:52:35:5a)</li> <li>▣ Internet Protocol Version 4, Src: 192.168.1.3 (192.168.1.3), Dst: 175.45.184.75 (175.45.184.75)</li> <li>▣ Transmission Control Protocol, Src Port: nifty-hmi (4134), Dst Port: http (80), Seq: 1, Ack: 1, Len: 985</li> <li>▣ Hypertext Transfer Protocol</li> <li>▣ Line-based text data: application/x-www-form-urlencoded LoginName=C...&amp;LoginPass=s...</li> </ul>								

Gambar 4.19 Tampilan *Traffic* e-complaint.ub.ac.id

Traffic HTTP yang berisi informasi *login* tidak dapat ditangkap pada aplikasi selain e-complaint.ub.ac.id, hal ini dikarenakan lima aplikasi lain yang dimaksud sudah menggunakan pengamanan SSL pada halaman *web* yang dimiliki misalnya bais.ub.ac.id. Pada gambar 4.20 akan ditunjukkan bahwa pada aplikasi bais.ub.ac.id hanya ditangkap *traffic* HTTPS dimana penyerang tidak bisa mendapatkan informasi dari *traffic* tersebut karena bersifat *encrypted*.

1154	140.687004	192.168.1.3	175.45.184.222	TCP	54	jt400-ssl > https [ACK] Seq=1 Ack=1 win=65800	
1155	140.687027	192.168.1.3	175.45.184.222	TCP	54	[TCP Dup ACK 1154#1] jt400-ssl > https [ACK]	
1156	140.687247	192.168.1.3	175.45.184.222	TCP	54	[TCP Dup ACK 1154#2] jt400-ssl > https [ACK]	
1157	140.687580	192.168.1.3	175.45.184.222	TLSv1	234	Client Hello	
1158	140.687601	192.168.1.3	175.45.184.222	TLSv1	234	[TCP Retransmission] Client Hello	
1159	140.687732	192.168.1.3	175.45.184.222	TLSv1	234	[TCP Retransmission] Client Hello	
1160	140.688075	192.168.1.3	175.45.184.222	TCP	66	[TCP Dup ACK 1159#1] jt400-ssl > https [ACK]	
1161	140.688096	192.168.1.3	175.45.184.222	TCP	66	[TCP Dup ACK 1159#2] jt400-ssl > https [ACK]	
1162	140.688298	192.168.1.3	175.45.184.222	TCP	66	[TCP Dup ACK 1159#3] jt400-ssl > https [ACK]	
1166	140.691116	192.168.1.3	175.45.184.222	TCP	66	[TCP Dup ACK 1159#4] jt400-ssl > https [ACK]	
1167	140.691139	192.168.1.3	175.45.184.222	TCP	66	[TCP Dup ACK 1159#5] jt400-ssl > https [ACK]	
1168	140.691395	192.168.1.3	175.45.184.222	TCP	66	[TCP Dup ACK 1159#6] jt400-ssl > https [ACK]	
1169	140.691734	192.168.1.3	175.45.184.222	TCP	66	[TCP Dup ACK 1159#7] jt400-ssl > https [ACK]	
<ul style="list-style-type: none"> <li>▣ Frame 1157: 234 bytes on wire (1872 bits), 234 bytes captured (1872 bits) on interface 0</li> <li>▣ Ethernet II, Src: HonHaiPr_11:69:d1 (9c:2a:70:11:69:d1), Dst: GemtekTe_52:35:5a (00:21:00:52:35:5a)</li> <li>▣ Internet Protocol Version 4, Src: 192.168.1.3 (192.168.1.3), Dst: 175.45.184.222 (175.45.184.222)</li> <li>▣ Transmission Control Protocol, Src Port: jt400-ssl (3471), Dst Port: https (443), Seq: 1, Ack: 1, Len: 180</li> <li>▣ Secure Sockets Layer</li> <li>▣ TLSv1 Record Layer: Handshake Protocol: Client Hello</li> </ul>							

Gambar 4.20 Tampilan *traffic* bais.ub.ac.id

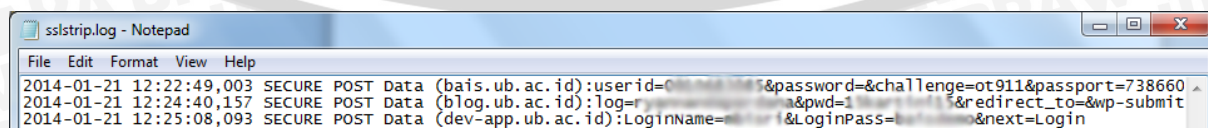
#### 4.2.1.4 Pengujian SSL Hijacking

Pengujian ini adalah tindak lanjut dari kegagalan penyerang dalam menangkap *traffic* pada uji penangkapan *traffic* sebelumnya. Uji SSL *hijacking* dilakukan pada aplikasi yang menggunakan SSL (HTTPS) sebagai standar keamanan pada halaman *web* aplikasi tersebut. Kelebihan dari uji serang ini adalah penyerang dapat melihat *username* dan *password* pada *log file* yang dibuat oleh *tool* SSLStrip setelah *tool* tersebut telah selesai dijalankan. Berikut ini adalah daftar aplikasi yang berhasil diuji serang menggunakan metode SSL *hijacking*:

**Tabel 4.3** Hasil Uji SSL *Hijacking*

No.	Aplikasi	Alamat IP	Cleartext ditemukan?
1	bais.ub.ac.id	175.45.184.222	Ya
2	siam.ub.ac.id	175.45.184.86	Tidak
3	gapura.ub.ac.id	175.45.184.74	Tidak
4	blog.ub.ac.id	175.45.184.5	Ya
5	siado.ub.ac.id	175.45.184.104	Tidak
6	e-complaint.ub.ac.id	175.45.184.75	Ya

Penyerang tidak dapat menangkap *traffic* HTTP pada aplikasi `bais.ub.ac.id` dan `blog.ub.ac.id` di pengujian penangkapan *traffic* sebelumnya, namun dengan uji serang ini penyerang dapat mengambil informasi penting berupa *username* dan *password* saat target mengakses kedua aplikasi tersebut. Gambar 4.21 akan menunjukkan tampilan *log file* yang didapatkan oleh penyerang dari *tool* SSLStrip yang dijalankan.



**Gambar 4.21** Tampilan *log* dari SSLStrip



#### 4.2.1.5 Pengujian Session Hijacking

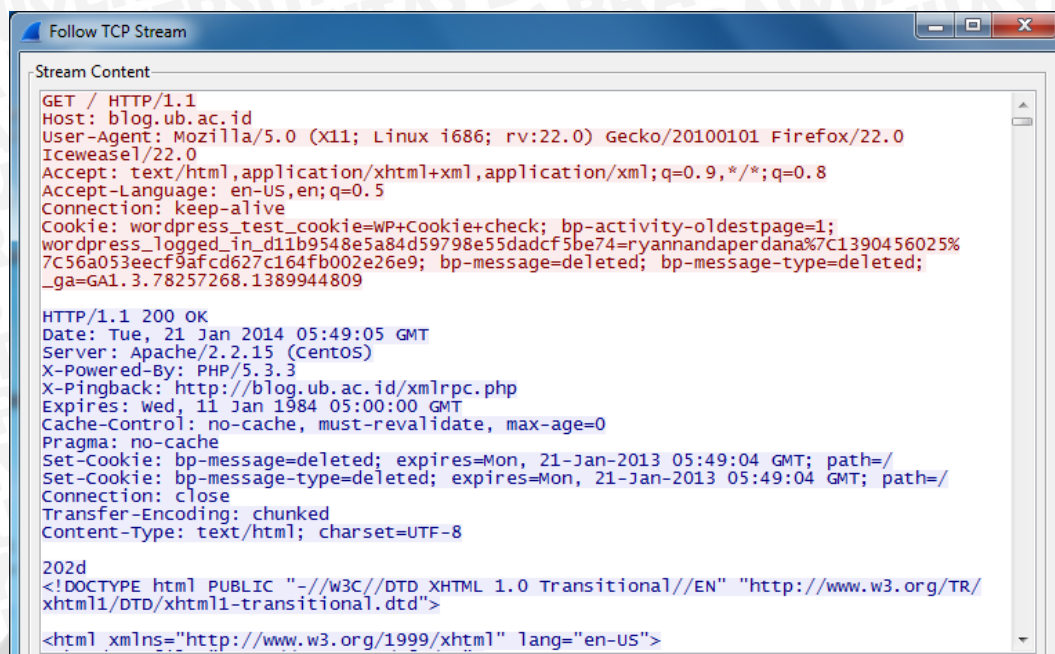
Pengujian ini adalah tindak lanjut kedua dari kegagalan penyerang dalam menangkap *traffic* pada uji penangkapan *traffic* sebelumnya. Uji *session hijacking* dilakukan dengan menangkap *session cookies* dari akun yang sedang membangun *session* dengan *server* melalui otentikasi *login* menggunakan *tool* Ferret. Kelebihan dari uji serang ini adalah penyerang dapat melakukan akses pada suatu aplikasi dengan akun yang dimiliki oleh target menggunakan *tool* Hamster. Berikut ini adalah daftar aplikasi yang berhasil diuji serang menggunakan metode *session hijacking*:

**Tabel 4.4** Hasil Uji *Session Hijacking*

No.	Aplikasi	Alamat IP	Berhasil akses akun?
1	bais.ub.ac.id	175.45.184.222	Ya
2	siam.ub.ac.id	175.45.184.86	Tidak
3	gapura.ub.ac.id	175.45.184.74	Tidak
4	blog.ub.ac.id	175.45.184.5	Ya
5	siado.ub.ac.id	175.45.184.104	Tidak
6	e-complaint.ub.ac.id	175.45.184.75	Ya

Penyerang tidak dapat menangkap *traffic* HTTP pada aplikasi [bais.ub.ac.id](http://bais.ub.ac.id) dan [blog.ub.ac.id](http://blog.ub.ac.id) di pengujian penangkapan *traffic* sebelumnya, namun dengan uji serang ini penyerang dapat mengambil informasi penting berupa *session cookies* saat target mengakses kedua aplikasi tersebut. Gambar 4.22 akan menunjukkan adanya *session cookies* yang ditangkap oleh penyerang lalu *session cookies* yang didapatkan akan digunakan oleh penyerang untuk mengakses akun milik target yang akan ditunjukkan pada Gambar 4.23.





```

Follow TCP Stream
Stream Content
GET / HTTP/1.1
Host: blog.ub.ac.id
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:22.0) Gecko/20100101 Firefox/22.0
Iceweasel/22.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Connection: keep-alive
Cookie: wordpress_test_cookie=WP+Cookie+check; bp-activity-oldestpage=1;
wordpress_logged_in_d11b9548e5a84d59798e55dadcf5be74=ryannandaperdana%7C1390456025%
7C56a053eecf9afcd627c164fb002e26e9; bp-message=deleted; bp-message-type=deleted;
_ga=GA1.3.78257268.1389944809

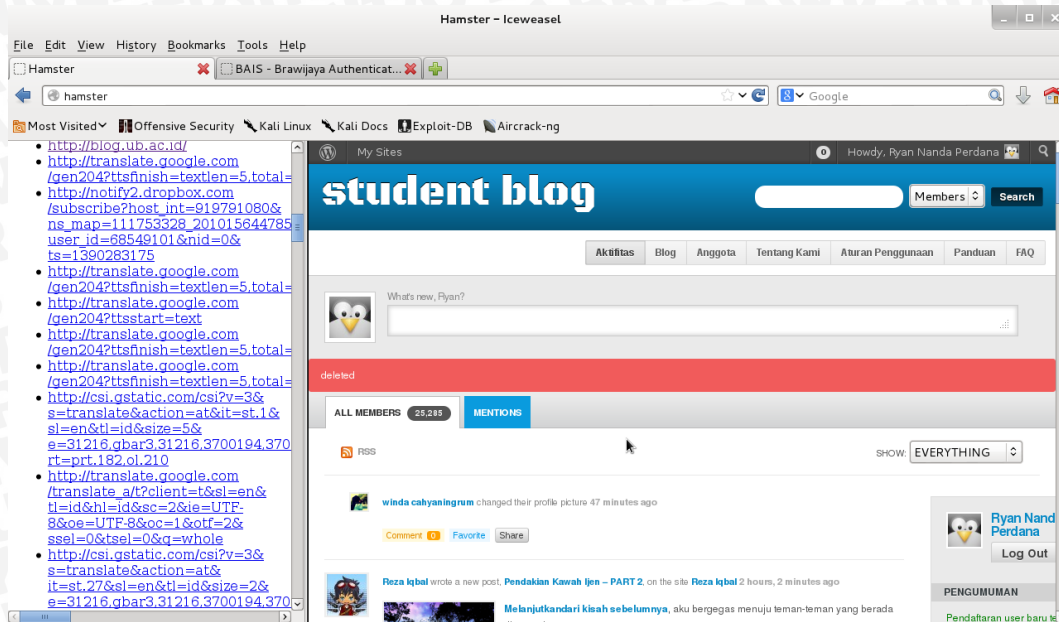
HTTP/1.1 200 OK
Date: Tue, 21 Jan 2014 05:49:05 GMT
Server: Apache/2.2.15 (CentOS)
X-Powered-By: PHP/5.3.3
X-Pingback: http://blog.ub.ac.id/xmlrpc.php
Expires: wed, 11 Jan 1984 05:00:00 GMT
Cache-Control: no-cache, must-revalidate, max-age=0
Pragma: no-cache
Set-Cookie: bp-message=deleted; expires=Mon, 21-Jan-2013 05:49:04 GMT; path=/
Set-Cookie: bp-message-type=deleted; expires=Mon, 21-Jan-2013 05:49:04 GMT; path=/
Connection: close
Transfer-Encoding: chunked
Content-Type: text/html; charset=UTF-8

202d
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/
xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en-US">

```

**Gambar 4.22** Tampilan TCP *stream* bais.ub.ac.id

Gambar 4.22 adalah tampilan TCP *stream* dari komunikasi antara target dengan *server* aplikasi bais.ub.ac.id, terlihat adanya *session cookies* yaitu “wordpress\_test\_cookie=WP+Cookie+check; bp-activity-oldestpage=1; wordpress\_logged\_in\_d11b9548e5a84d59798e55dadcf5be74=ryannandaperdana %7C1390456025%7C56a053eecf9afcd627c164fb002e26e9; bp-message=deleted; bp-message-type=deleted; \_ga=GA1.3.78257268.1389944809” dan dibawahnya terdapat konten halaman *web* dari aplikasi blog.ub.ac.id yang didapatkan oleh penyerang. Gambar 4.23 adalah tampilan halaman *web* dari aplikasi bais.ub.ac.id dalam keadaan *logged in* menggunakan akun milik target.



**Gambar 4.23** Tampilan Penyerang Mengakses Halaman dengan Akun Target

#### 4.2.1.6 Pengujian DNS Spoofing

Pengujian ini adalah tindak lanjut ketiga dari kegagalan penyerang dalam menangkap *traffic* pada uji penangkapan *traffic* sebelumnya. Uji *DNS spoofing* dilakukan dengan memberikan informasi alamat DNS palsu kepada target sehingga target akan menuju ke alamat IP penyerang ketika mengakses alamat suatu aplikasi berbasis *web*. Penyerang dapat melihat *username* dan *password* pada *log* yang ditampilkan oleh *tool* SEToolkit pada saat target melakukan login pada halaman *clone* yang disiapkan oleh penyerang. Berikut ini adalah daftar aplikasi yang berhasil diuji serang menggunakan metode *DNS spoofing*:

Tabel 4.5 Hasil Uji DNS Spoofing

No.	Aplikasi	Alamat IP	Cleartext ditemukan?
1	bais.ub.ac.id	175.45.184.222	Ya
2	siam.ub.ac.id	175.45.184.86	Ya
3	gapura.ub.ac.id	175.45.184.74	Ya
4	blog.ub.ac.id	175.45.184.5	Ya
5	siado.ub.ac.id	175.45.184.104	Ya
6	e-complaint.ub.ac.id	175.45.184.75	Tidak

Informasi berupa *username* dan *password* dapat ditangkap pada sebagian besar aplikasi yang dituju oleh target dikarenakan DNS *cache* pada target memang sudah berisikan informasi alamat palsu dari penyerang. Target akan selalu menuju ke halaman *clone* yang sudah disiapkan oleh penyerang, namun hal ini tidak berlaku untuk aplikasi e-complaint.ub.ac.id dikarenakan kolom *login* yang dimilikinya tidak berada pada halaman utama. Gambar 4.24 akan menunjukkan tampilan ketika informasi *username* dan *password* didapatkan oleh penyerang ketika target mengakses aplikasi siam.ub.ac.id palsu.

```
The best way to use this attack is if username and password form
fields are available. Regardless, this captures all POSTs on a website.
[*] The Social-Engineer Toolkit Credential Harvester Attack
[*] Credential Harvester is running on port 80
[*] Information will be displayed to you as it arrives below:
192.168.88.199 - - [21/Jan/2014 16:43:09] "GET / HTTP/1.1" 200 -
192.168.88.198 - - [21/Jan/2014 16:43:22] "GET / HTTP/1.1" 200 -
[*] WE GOT A HIT! Printing the output:
POSSIBLE USERNAME FIELD FOUND: username=
POSSIBLE PASSWORD FIELD FOUND: password=
POSSIBLE USERNAME FIELD FOUND: login=+LOGIN+
[*] WHEN YOU'RE FINISHED, HIT CONTROL-C TO GENERATE A REPORT.

192.168.88.198 - - [21/Jan/2014 16:44:50] "GET / HTTP/1.1" 200
```

Gambar 4.24 Tampilan SEToolkit dengan Informasi Login



#### 4.2.2 Lingkungan Jaringan Konfigurasi Lanjut

Pengujian kedua dilakukan di jaringan dengan konfigurasi lanjut, seperti di jaringan nirkabel kampus ataupun jaringan nirkabel yang ada di perkantoran. Jaringan nirkabel kampus atau kantor dapat dinyatakan jaringan dengan konfigurasi standar karena biasanya menggunakan perangkat jaringan yang lengkap (seperti *router*, *switch* dan *access point*) dan dikonfigurasi sesuai dengan kebutuhan dari kampus atau kantor tersebut.

Lingkungan pengujian dengan konfigurasi lanjut yaitu jaringan kampus PTIIK UB dipilih untuk lingkungan pengujian kedua, beberapa informasi tentang konfigurasi lanjut pada jaringan kampus PTIIK UB adalah sebagai berikut:

1. Tiap *router* dipasang mekanisme untuk menanggulangi *ARP spoofing* berupa fitur verifikasi alamat IP dan MAC yaitu DAI (*Dynamic ARP Inspection*)
2. Tiap *router* dipasang *static* DNS yaitu DNS lokal UB
3. Pembatasan pada port tertentu menggunakan ACL (*Access List*)
4. Pembatasan waktu *uptime* pada wilayah dan jam tertentu

##### 4.2.2.1 Reconnaissance

Tahap *reconnaissance* atau pengumpulan data dilakukan sebelum memasuki tahap *active attacks* yang terdiri dari pengujian *sniffing*, penangkapan *traffic*, *SSL hijacking*, *session hijacking* dan *DNS spoofing*. Pada tahap ini terdapat tiga informasi yang dibutuhkan yaitu:

1. Kondisi pengamanan jaringan

Pemeriksaan kondisi keamanan jaringan dilakukan seperti pada lingkungan jaringan dengan konfigurasi standar. Pada gambar 4.25 akan ditunjukkan bahwa penyerang membuat sebuah *interface* yang digunakan untuk melakukan *monitor* jaringan nirkabel yang ada di sekitar serta pada gambar 4.26 akan ditunjukkan adanya sebuah jaringan nirkabel dengan SSID “PTIIK” yang bersifat *open* dan dapat dimasuki oleh penyerang dan memiliki alamat MAC D4:8C:B5:07:0F:91.

```

root@kali: ~
File Edit View Search Terminal Help
root@kali:~# airon-ng start wlan0

Found 2 processes that could cause trouble.
If airodump-ng, aireplay-ng or airtun-ng stops working after
a short period of time, you may want to kill (some of) them!
-e
PID      Name
2506     NetworkManager
2805     wpa_supplicant

Interface      Chipset      Driver
wlan0          Broadcom    b43 - [phy0]
              (monitor mode enabled on mon0)

```

Gambar 4.25 Tampilan Airmon-ng

```

root@kali: ~
File Edit View Search Terminal Tabs Help

root@kali: ~
CH 12 ][ Elapsed: 48 s ][ 2014-07-09 00:45

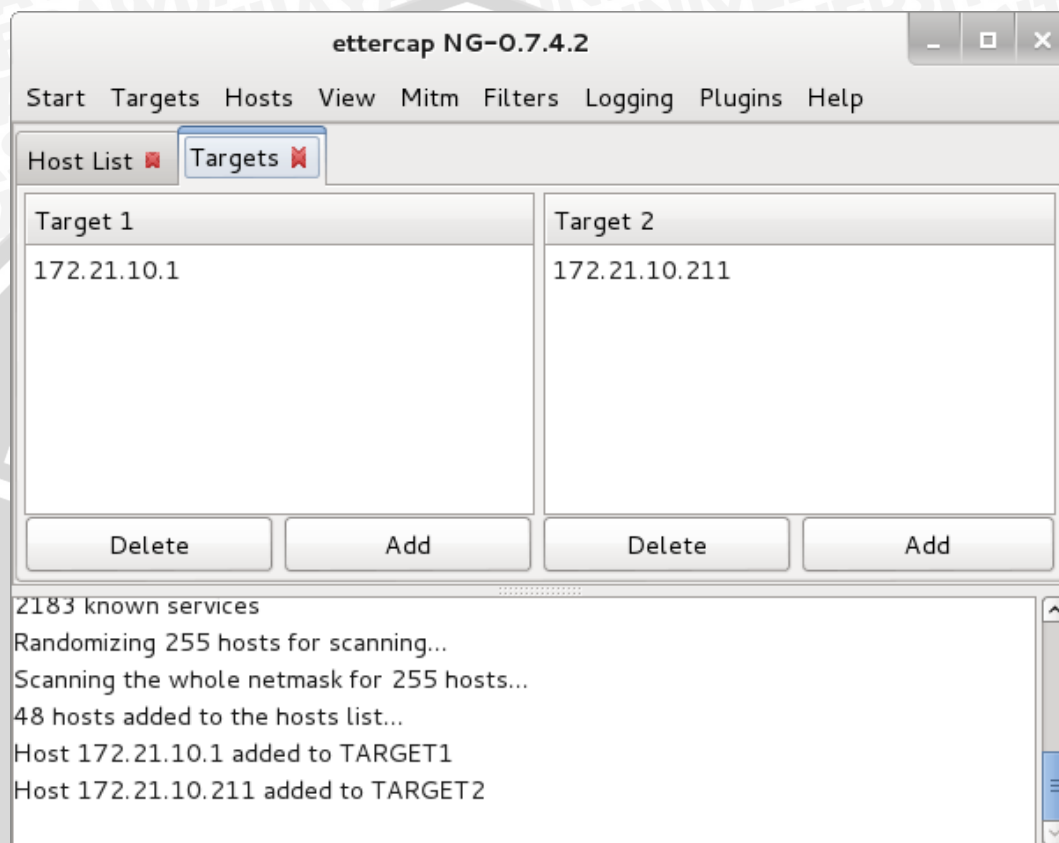
BSSID          PWR  Beacons    #Data, #/s  CH  MB  ENC  CIPHER AUTH  ESSID
C2:9F:DB:67:12:18 -33    79      411   9  2  54e.  OPN          PTIIK
C6:9F:DB:67:12:18 -26    63         0   0  2  54e.  WPA2 CCMP   PSK   <leng
C2:9F:DB:67:12:4D -34   113      140   0  7  54e.  OPN          PTIIK
C2:9F:DB:E7:82:E8 -42    49      360   7  1  54e.  OPN          PTIIK
C2:9F:DB:2B:B5:94 -38   132         0  11  54e.  OPN          PTIIK
C6:9F:DB:2B:B5:94 -44    69         0  11  54e.  WPA2 CCMP   PSK   <leng
C2:9F:DB:67:15:71 -57    66      32   0  6  54e.  OPN          PTIIK
56:DE:2B:43:34:E5 -52    25         0   3  54e.  WPA2 CCMP   PSK   Conne
C2:9F:DB:67:15:B4 -52    42         1   6  54e.  OPN          PTIIK
82:CF:41:24:DF:72 -58    89         0   7  54e.  WPA2 CCMP   PSK   www.b
10:FE:ED:F9:E0:F6 -58     8         0   3  54 .  OPN          UB_VE
00:02:6F:54:82:A5 -65    38         1  12  54 .  OPN          <leng
C2:9F:DB:67:14:23 -64    89         6   8  54e.  OPN          PTIIK
C2:9F:DB:67:12:AF -70     2        30   0  4  54e.  OPN          PTIIK
C2:9F:DB:67:15:A0 -69    33         2  11  54e.  OPN          PTIIK
C2:9F:DB:67:15:B6 -72    75         0  10  54e.  OPN          PTIIK
C2:9F:DB:67:12:D3 -77     3        11   0  5  54e.  OPN          PTIIK
2A:A4:3C:99:8A:30 -79     3         0   6  54e.  OPN          Dekan

```

Gambar 4.26 Tampilan Airodump-ng

## 2. Informasi target serangan

Informasi target yang akan diserang didapatkan dari *host scanning* yang terdapat pada *tool* Ettercap. Pada gambar 4.27 akan ditunjukkan jendela dari *tool* Ettercap yang menunjukkan informasi alamat IP dan MAC dari *router* dan sebuah *client* yang selanjutnya akan dipilih menjadi target.



**Gambar 4.27** Tampilan Informasi Host yang Terhubung di Jaringan

## 3. Berjalannya protokol ARP

Penyerang akan memastikan bahwa protokol ARP pada target berjalan agar *sniffing* (*ARP poisoning*) dapat dilakukan. Pada gambar 4.28 dan 4.29 akan ditunjukkan proses ARPing yang dilakukan pada *router* dengan alamat IP 172.21.10.1 dan target dengan alamat IP 172.21.10.211.



```

root@kali: ~
File Edit View Search Terminal Tabs Help
root@kali: ~ x root@kali: ~ x root@kali: ~ x root@kali: ~ x
root@kali:~# arping -c 4 172.21.10.1
ARPING 172.21.10.1
60 bytes from d4:8c:b5:07:0f:91 (172.21.10.1): index=0 time=5.261 msec
60 bytes from d4:8c:b5:07:0f:91 (172.21.10.1): index=1 time=10.582 msec
60 bytes from d4:8c:b5:07:0f:91 (172.21.10.1): index=2 time=20.859 msec
60 bytes from d4:8c:b5:07:0f:91 (172.21.10.1): index=3 time=10.685 msec

--- 172.21.10.1 statistics ---
4 packets transmitted, 4 packets received, 0% unanswered (0 extra)
root@kali:~#

```

**Gambar 4.28** ARPing pada Router

```

root@kali: ~
File Edit View Search Terminal Tabs Help
root@kali: ~ x root@kali: ~ x root@kali: ~ x root@kali: ~ x
root@kali:~# arping -c 4 192.168.1.3
ARPING 192.168.1.3
42 bytes from 9c:2a:70:11:69:d1 (192.168.1.3): index=0 time=3.284 msec
42 bytes from 9c:2a:70:11:69:d1 (192.168.1.3): index=1 time=3.243 msec
42 bytes from 9c:2a:70:11:69:d1 (192.168.1.3): index=2 time=3.204 msec
42 bytes from 9c:2a:70:11:69:d1 (192.168.1.3): index=3 time=3.223 msec

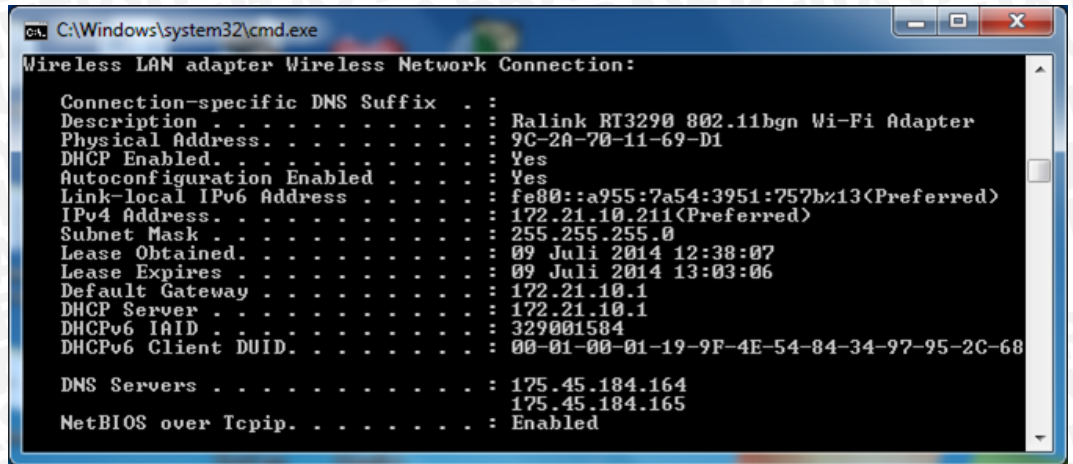
--- 192.168.1.3 statistics ---
4 packets transmitted, 4 packets received, 0% unanswered (0 extra)
root@kali:~#

```

**Gambar 4.29** ARPing pada Target

#### 4.2.2.2 Pengujian Sniffing

Uji *sniffing* pada lingkungan jaringan ini dilakukan sama seperti pengujian pada lingkungan jaringan sebelumnya dengan alamat IP yang berbeda, alamat IP target adalah 172.21.14.14 dan alamat IP penyerang adalah 172.21.14.13 serta alamat IP *router* adalah 172.21.14.1. Berikut ini adalah tampilan alamat IP dari masing-masing elemen:



```

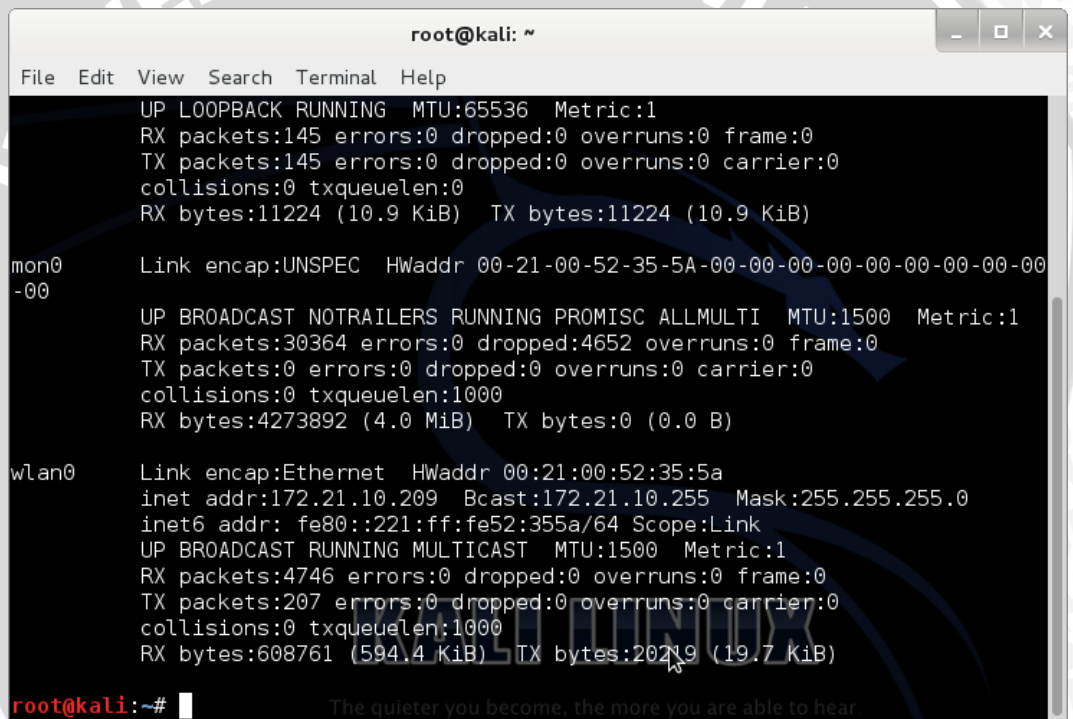
C:\Windows\system32\cmd.exe
Wireless LAN adapter Wireless Network Connection:

Connection-specific DNS Suffix . . . : 
Description . . . . . : Ralink RT3290 802.11bgn Wi-Fi Adapter
Physical Address. . . . . : 9C-2A-70-11-69-D1
DHCP Enabled. . . . . : Yes
Autoconfiguration Enabled . . . . . : Yes
Link-local IPv6 Address . . . . . : fe80::a955:7a54:3951:757b%13(Preferred)
IPv4 Address. . . . . : 172.21.10.211(Preferred)
Subnet Mask . . . . . : 255.255.255.0
Lease Obtained. . . . . : 09 Juli 2014 12:38:07
Lease Expires . . . . . : 09 Juli 2014 13:03:06
Default Gateway . . . . . : 172.21.10.1
DHCP Server . . . . . : 172.21.10.1
DHCPv6 IAID . . . . . : 329001584
DHCPv6 Client DUID. . . . . : 00-01-00-01-19-9F-4E-54-84-34-97-95-2C-68

DNS Servers . . . . . : 175.45.184.164
                       175.45.184.165
NetBIOS over Tcpiip. . . . . : Enabled

```

Gambar 4.30 Alamat IP Target



```

root@kali: ~
File Edit View Search Terminal Help

UP LOOPBACK RUNNING MTU:65536 Metric:1
RX packets:145 errors:0 dropped:0 overruns:0 frame:0
TX packets:145 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:11224 (10.9 KiB) TX bytes:11224 (10.9 KiB)

mon0
-00
Link encap:UNSPEC HWaddr 00-21-00-52-35-5A-00-00-00-00-00-00-00-00-00-00

UP BROADCAST NOTRAILERS RUNNING PROMISC ALLMULTI MTU:1500 Metric:1
RX packets:30364 errors:0 dropped:4652 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:4273892 (4.0 MiB) TX bytes:0 (0.0 B)

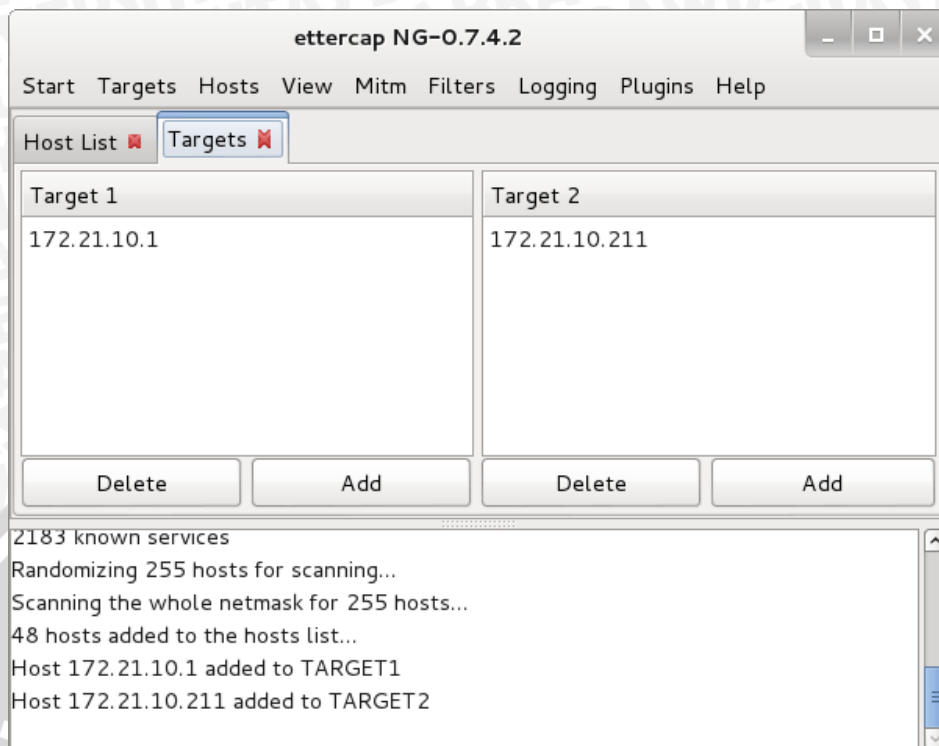
wlan0
Link encap:Ethernet HWaddr 00:21:00:52:35:5a
inet addr:172.21.10.209 Bcast:172.21.10.255 Mask:255.255.255.0
inet6 addr: fe80::221:ff:fe52:355a/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:4746 errors:0 dropped:0 overruns:0 frame:0
TX packets:207 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:608761 (594.4 KiB) TX bytes:20219 (19.7 KiB)

root@kali:~#

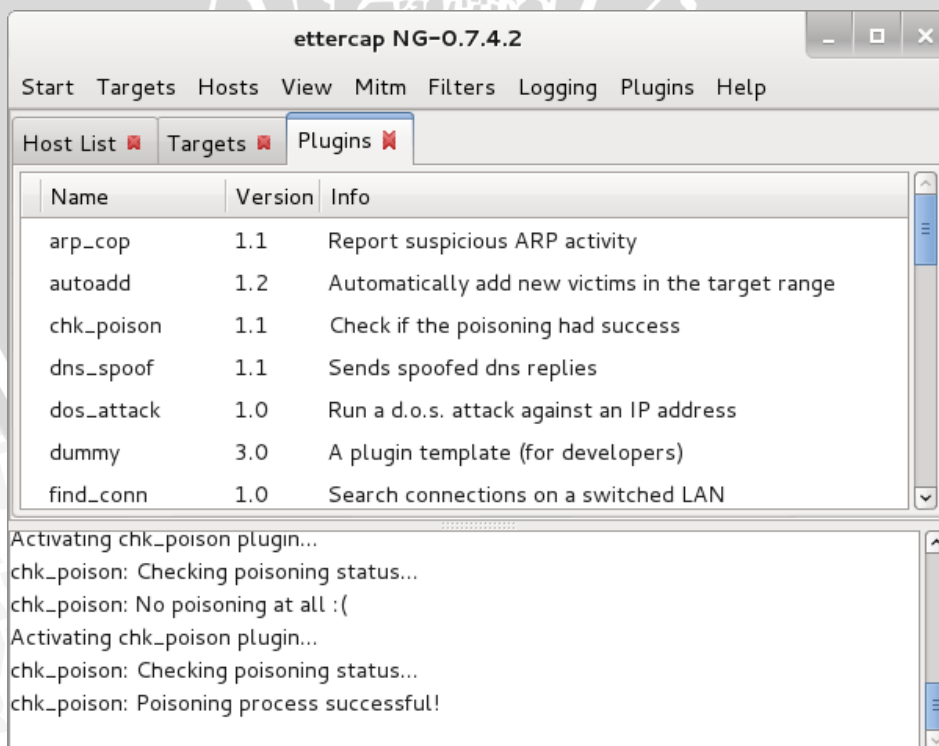
```

Gambar 4.31 Alamat IP Penyerang

Penyerang mendaftarkan alamat IP router dan alamat IP target kedalam daftar target dan melakukan cek keberhasilan ARP *poisoning*, sama seperti pengujian *sniffing* sebelumnya.



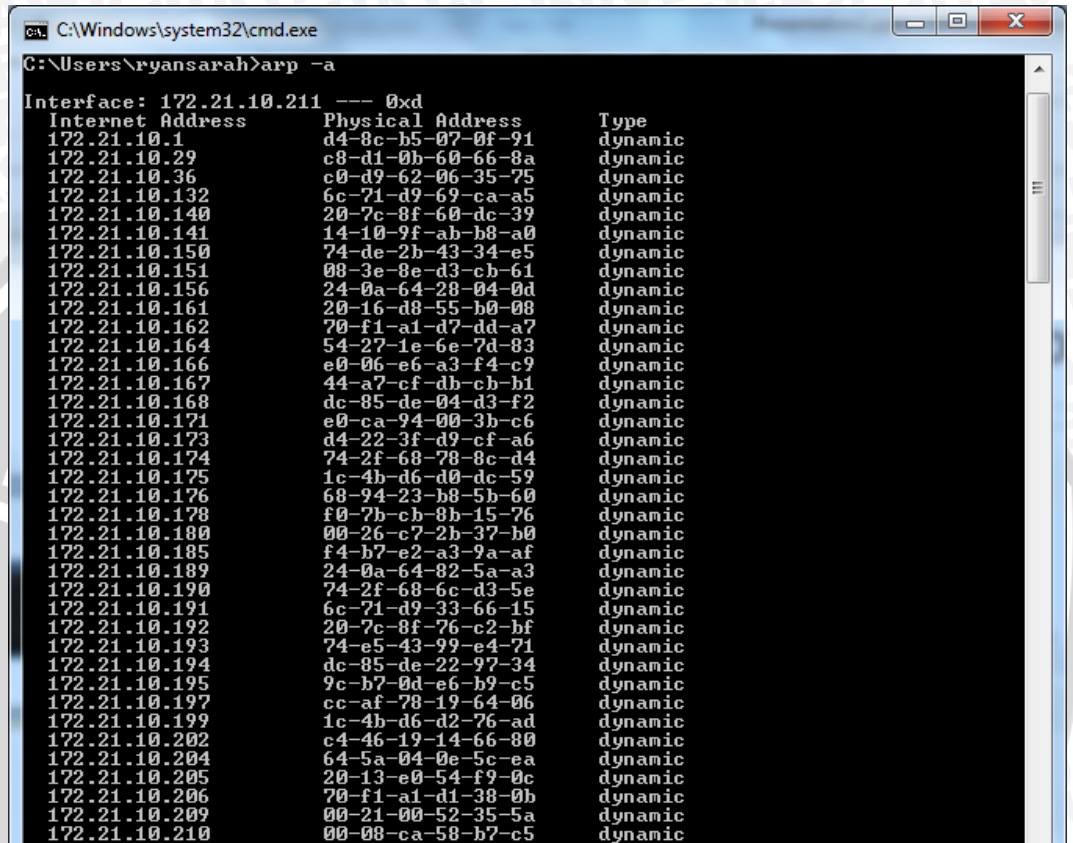
**Gambar 4.32** Penyerang Menentukan Target



**Gambar 4.33** Penyerang Melakukan Cek ARP Poisoning



Keberhasilan dari *sniffing* ditunjukkan pada perubahan tabel ARP (ARP *cache*) yang dimiliki oleh target, gambar 4.34 akan menunjukkan tabel ARP yang normal dan gambar 4.35 akan menunjukkan tabel ARP yang sudah dikenai ARP *poisoning*.



```

C:\Windows\system32\cmd.exe
C:\Users\ryansarah>arp -a

Interface: 172.21.10.211 --- 0xd
Internet Address      Physical Address      Type
172.21.10.1           d4-8c-b5-07-0f-91    dynamic
172.21.10.29          c8-d1-0b-60-66-8a    dynamic
172.21.10.36          c0-d9-62-06-35-75    dynamic
172.21.10.132         6c-71-d9-69-ca-a5    dynamic
172.21.10.140         20-7c-8f-60-dc-39    dynamic
172.21.10.141         14-10-9f-ab-b8-a0    dynamic
172.21.10.150         74-de-2b-43-34-e5    dynamic
172.21.10.151         00-3e-8e-d3-cb-61    dynamic
172.21.10.156         24-0a-64-28-04-0d    dynamic
172.21.10.161         20-16-d8-55-b0-08    dynamic
172.21.10.162         70-f1-a1-d7-dd-a7    dynamic
172.21.10.164         54-27-1e-6e-7d-83    dynamic
172.21.10.166         e0-06-e6-a3-f4-c9    dynamic
172.21.10.167         44-a7-cf-db-cb-b1    dynamic
172.21.10.168         dc-85-de-04-d3-f2    dynamic
172.21.10.171         e0-ca-94-00-3b-c6    dynamic
172.21.10.173         d4-22-3f-d9-cf-a6    dynamic
172.21.10.174         74-2f-68-78-8c-d4    dynamic
172.21.10.175         1c-4b-d6-d0-dc-59    dynamic
172.21.10.176         68-94-23-b8-5b-60    dynamic
172.21.10.178         f0-7b-cb-8b-15-76    dynamic
172.21.10.180         00-26-c7-2b-37-b0    dynamic
172.21.10.185         f4-b7-e2-a3-9a-af    dynamic
172.21.10.189         24-0a-64-82-5a-a3    dynamic
172.21.10.190         74-2f-68-6c-d3-5e    dynamic
172.21.10.191         6c-71-d9-33-66-15    dynamic
172.21.10.192         20-7c-8f-76-c2-bf    dynamic
172.21.10.193         74-e5-43-99-e4-71    dynamic
172.21.10.194         dc-85-de-22-97-34    dynamic
172.21.10.195         9c-b7-0d-e6-b9-c5    dynamic
172.21.10.197         cc-af-78-19-64-06    dynamic
172.21.10.199         1c-4b-d6-d2-76-ad    dynamic
172.21.10.202         c4-46-19-14-66-80    dynamic
172.21.10.204         64-5a-04-0e-5c-ea    dynamic
172.21.10.205         20-13-e0-54-f9-0c    dynamic
172.21.10.206         70-f1-a1-d1-38-0b    dynamic
172.21.10.209         00-21-00-52-35-5a    dynamic
172.21.10.210         00-08-ca-58-b7-c5    dynamic
  
```

Gambar 4.34 Tabel ARP Normal

```

ca. C:\Windows\system32\cmd.exe
C:\Users\ryansarah>arp -a

Interface: 172.21.10.211 --- 0xd
Internet Address      Physical Address      Type
169.254.52.106       3c-77-e6-d6-29-77    dynamic
169.254.92.73        44-6d-57-7c-19-42    dynamic
169.254.121.92       74-de-2b-25-e5-ac    dynamic
172.21.10.1          00-21-00-52-35-5a    dynamic
172.21.10.29         c8-d1-0b-60-66-8a    dynamic
172.21.10.36         c0-d9-62-06-35-75    dynamic
172.21.10.125        44-6d-57-7c-19-42    dynamic
172.21.10.132        6c-71-d9-69-ca-a5    dynamic
172.21.10.140        20-7c-8f-60-dc-39    dynamic
172.21.10.141        14-10-9f-ab-b8-a0    dynamic
172.21.10.150        74-de-2b-43-34-e5    dynamic
172.21.10.151        08-3e-8e-d3-cb-61    dynamic
172.21.10.156        24-0a-64-28-04-0d    dynamic
172.21.10.157        e0-63-e5-09-54-65    dynamic
172.21.10.161        20-16-d8-55-b0-08    dynamic
172.21.10.162        70-f1-a1-d7-dd-a7    dynamic
172.21.10.164        54-27-1e-6e-7d-83    dynamic
172.21.10.166        e0-06-e6-a3-f4-c9    dynamic
172.21.10.167        44-a7-cf-db-cb-b1    dynamic
172.21.10.168        dc-85-de-04-d3-f2    dynamic
172.21.10.171        e0-ca-94-00-3b-c6    dynamic
172.21.10.174        74-2f-68-78-8c-d4    dynamic
172.21.10.176        68-94-23-b8-5b-60    dynamic
172.21.10.180        00-26-c7-2b-37-b0    dynamic
172.21.10.185        f4-b7-e2-a3-9a-af    dynamic
172.21.10.189        24-0a-64-82-5a-a3    dynamic
172.21.10.190        74-2f-68-6c-d3-5e    dynamic
172.21.10.191        6c-71-d9-33-66-15    dynamic
172.21.10.192        20-7c-8f-76-c2-bf    dynamic
172.21.10.193        74-e5-43-99-e4-71    dynamic
172.21.10.194        dc-85-de-22-97-34    dynamic
172.21.10.195        9c-b7-0d-e6-b9-c5    dynamic
172.21.10.197        cc-af-78-19-64-06    dynamic
172.21.10.199        1c-4b-d6-d2-76-ad    dynamic
172.21.10.202        c4-46-19-14-66-80    dynamic
172.21.10.204        64-5a-04-0e-5c-ea    dynamic
172.21.10.205        20-13-e0-54-f9-0c    dynamic
172.21.10.206        70-f1-a1-d1-38-0b    dynamic
172.21.10.209        00-21-00-52-35-5a    dynamic
172.21.10.210        00-08-ca-58-b7-c5    dynamic

```

Gambar 4.35 Tabel ARP Ter-poisson

Pada gambar 4.35 terlihat bahwa alamat MAC yang dimiliki *router* berubah menjadi sama dengan alamat MAC yang dimiliki oleh penyerang, dengan demikian maka semua pengiriman paket data dari target maupun menuju target akan melewati *device* yang digunakan oleh penyerang.

#### 4.2.2.3 Pengujian Penangkapan Traffic

Uji penangkapan *traffic* dilakukan sama seperti pada pengujian sebelumnya, namun pada pengujian kali ini nas.ub.ac.id dapat diakses. Berikut ini adalah daftar aplikasi yang berhasil diuji serang menggunakan metode penangkapan *traffic*:

Tabel 4.6 Hasil Uji Penangkapan *Traffic*

No.	Aplikasi	Alamat IP	Traffic HTTP ditemukan?
1	nas.ub.ac.id	175.45.189.132	Ya
2	bais.ub.ac.id	175.45.184.222	Tidak
3	siam.ub.ac.id	175.45.184.86	Tidak
4	gapura.ub.ac.id	175.45.184.74	Tidak
5	blog.ub.ac.id	175.45.184.5	Tidak
6	siado.ub.ac.id	175.45.184.104	Tidak
7	e-complaint.ub.ac.id	175.45.184.75	Ya

Aplikasi nas.ub.ac.id dapat diakses didalam jaringan kampus UB, oleh karena itu *traffic* HTTP dari aplikasi nas.ub.ac.id dan e-complaint.ub.ac.id berhasil ditangkap oleh penyerang. *Traffic* HTTP yang berhasil ditangkap adalah pada saat target melakukan akses *login* melalui kedua aplikasi tersebut, gambar 4.36 akan menunjukkan tampilan informasi dari aplikasi nas.ub.ac.id yang berhasil ditangkap oleh penyerang.

```

2141 24.7194890 172.21.10.211 175.45.189.132 HTTP 424 [TCP Retransmission] GET /webAuth/images/bg-nas.
2142 24.7199130 172.21.10.211 175.45.189.132 HTTP 424 [TCP Retransmission] GET /webAuth/images/bg-nas.
2158 24.7415670 172.21.10.211 175.45.189.132 HTTP 427 GET /webAuth/images/bg-header.jpg HTTP/1.1
2159 24.7415940 172.21.10.211 175.45.189.132 HTTP 427 [TCP Retransmission] GET /webAuth/images/bg-head
2160 24.7418900 172.21.10.211 175.45.189.132 HTTP 427 [TCP Retransmission] GET /webAuth/images/bg-head
3429 35.0378770 172.21.10.211 175.45.189.132 HTTP 523 POST /webAuth/ HTTP/1.1 (application/x-www-form
3430 35.0378930 172.21.10.211 175.45.189.132 HTTP 523 [TCP Retransmission] POST /webAuth/ HTTP/1.1 (a
3436 35.0384780 172.21.10.211 175.45.189.132 HTTP 523 [TCP Retransmission] POST /webAuth/ HTTP/1.1 (a

Frame 3429: 523 bytes on wire (4184 bits), 523 bytes captured (4184 bits) on interface 0
Ethernet II, Src: HonHaiPr_11:69:d1 (9c:2a:70:11:69:d1), Dst: GemtekTe_52:35:5a (00:21:00:52:35:5a)
Internet Protocol Version 4, Src: 172.21.10.211 (172.21.10.211), Dst: 175.45.189.132 (175.45.189.132)
Transmission Control Protocol, Src Port: cybro-a-bus (8442), Dst Port: http (80), Seq: 1, Ack: 1, Len: 469
Hypertext Transfer Protocol
Line-based text data: application/x-www-form-urlencoded
username= &password= &pwd=aa11223344&secret=true

```

Gambar 4.36 Tampilan *Traffic* nas.ub.ac.id



Traffic HTTP yang berisi informasi *login* tidak dapat ditangkap pada aplikasi selain *nas.ub.ac.id* dan *e-complaint.ub.ac.id*, hal ini dikarenakan lima aplikasi lain yang dimaksud sudah menggunakan pengamanan SSL pada halaman *web* yang dimiliki misalnya *siam.ub.ac.id*. Pada gambar 4.37 akan ditunjukkan bahwa pada aplikasi *siam.ub.ac.id* hanya ditangkap *traffic* HTTPS dimana penyerang tidak bisa mendapatkan informasi dari *traffic* tersebut karena bersifat *encrypted*.

14811	202.652246	172.21.10.211	175.45.184.86	TCP	54 8673	> https [ACK] Seq=836 Ack=1425 Win=16096 Len=0
14812	202.652521	172.21.10.211	175.45.184.86	TCP	54 8673	> https [ACK] Seq=836 Ack=1426 Win=16096 Len=0
14813	202.752822	172.21.10.211	175.45.184.86	TLSv1	113	Change Cipher Spec, Encrypted Handshake Message
14814	202.752857	172.21.10.211	175.45.184.86	TLSv1	113	[TCP Retransmission] Change Cipher Spec, Encrypted
14815	202.753172	172.21.10.211	175.45.184.86	TLSv1	113	[TCP Retransmission] Change Cipher Spec, Encrypted
14816	202.753802	172.21.10.211	175.45.184.86	TLSv1	688	Application Data, Application Data
14817	202.753829	172.21.10.211	175.45.184.86	TLSv1	688	[TCP Retransmission] Application Data, Application
14818	202.753992	172.21.10.211	175.45.184.86	TLSv1	688	[TCP Retransmission] Application Data, Application
14819	202.754383	172.21.10.211	175.45.184.86	TLSv1	113	Change Cipher Spec, Encrypted Handshake Message
14820	202.754407	172.21.10.211	175.45.184.86	TLSv1	113	[TCP Retransmission] Change Cipher Spec, Encrypted
14821	202.754554	172.21.10.211	175.45.184.86	TLSv1	113	[TCP Retransmission] Change Cipher Spec, Encrypted
14822	202.755212	172.21.10.211	175.45.184.86	TLSv1	688	Application Data, Application Data

☐ Frame 14816: 688 bytes on wire (5504 bits), 688 bytes captured (5504 bits) on interface 0  
 ☐ Ethernet II, Src: HonHaiPr\_11:69:d1 (9c:2a:70:11:69:d1), Dst: GemtekTe\_52:35:5a (00:21:00:52:35:5a)  
 ☐ Internet Protocol Version 4, Src: 172.21.10.211 (172.21.10.211), Dst: 175.45.184.86 (175.45.184.86)  
 ☐ Transmission Control Protocol, Src Port: 8674 (8674), Dst Port: https (443), Seq: 250, Ack: 146, Len: 634  
 ☐ Secure Sockets Layer  
 ☐ TLSv1 Record Layer: Application Data Protocol: http  
   Content Type: Application Data (23)  
   Version: TLS 1.0 (0x0301)  
   Length: 32  
   Encrypted Application Data: e213a4cb0c167311c5c380dc6f3b4ee9920702468c08f3d2...  
 ☐ TLSv1 Record Layer: Application Data Protocol: http

Gambar 4.37 Tampilan *Traffic* *siam.ub.ac.id*

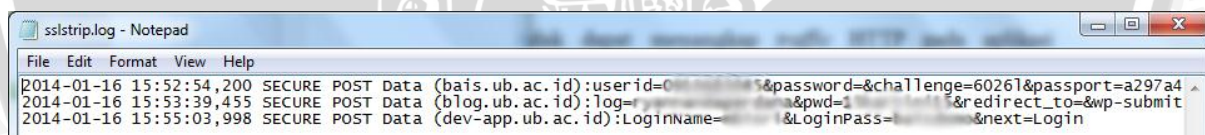
#### 4.2.2.4 Pengujian SSL Hijacking

Uji SSL *hijacking* dilakukan sama seperti pada pengujian sebelumnya, hasil yang didapatkan hanya berbeda pada aplikasi *nas.ub.ac.id* yang bisa diakses dan ditangkap *traffic* berupa *username* dan *password*. Berikut ini adalah daftar aplikasi yang berhasil diuji serang menggunakan metode SSL *hijacking*:

Tabel 4.7 Hasil Uji SSL Hijacking

No.	Aplikasi	Alamat IP	Cleartext ditemukan?
1	nas.ub.ac.id	175.45.189.132	Ya
2	bais.ub.ac.id	175.45.184.222	Ya
3	siam.ub.ac.id	175.45.184.86	Tidak
4	gapura.ub.ac.id	175.45.184.74	Tidak
5	blog.ub.ac.id	175.45.184.5	Ya
6	siado.ub.ac.id	175.45.184.104	Tidak
7	e-complaint.ub.ac.id	175.45.184.75	Ya

Penyerang tidak dapat menangkap *traffic* HTTP pada aplikasi `bais.ub.ac.id` dan `blog.ub.ac.id` di pengujian penangkapan *traffic* sebelumnya, namun dengan uji serang ini penyerang dapat mengambil informasi penting berupa *username* dan *password* saat target mengakses kedua aplikasi tersebut. Gambar 4.38 akan menunjukkan tampilan *log file* yang didapatkan oleh penyerang dari *tool* SSLStrip yang dijalankan.

Gambar 4.38 Tampilan *log* dari SSLStrip

#### 4.2.2.5 Pengujian Session Hijacking

Uji *session hijacking* dilakukan sama seperti pada pengujian sebelumnya, hasil yang didapatkan juga sama karena aplikasi `nas.ub.ac.id` yang bisa diakses di lingkungan jaringan ini tidak memberikan informasi *session cookies* kepada penyerang. Berikut ini adalah daftar aplikasi yang berhasil diuji serang menggunakan metode *session hijacking*:



Tabel 4.8 Hasil Uji *Session Hijacking*

No.	Aplikasi	Alamat IP	Berhasil akses akun?
1	nas.ub.ac.id	175.45.189.132	Tidak
2	bais.ub.ac.id	175.45.184.222	Ya
3	siam.ub.ac.id	175.45.184.86	Tidak
4	gapura.ub.ac.id	175.45.184.74	Tidak
5	blog.ub.ac.id	175.45.184.5	Ya
6	siado.ub.ac.id	175.45.184.104	Tidak
7	e-complaint.ub.ac.id	175.45.184.75	Ya

#### 4.2.2.6 Pengujian DNS Spoofing

Pengujian DNS *spoofing* pada lingkungan jaringan ini tidak berjalan baik seperti pada pengujian sebelumnya, penyerang tidak dapat memengaruhi DNS *cache* pada target sehingga target akan selalu mengakses *server* asli (DNS lokal UB) walaupun penyerang sudah melakukan DNS *spoofing* pada target. Kegagalan DNS *spoofing* pada pengujian di lingkungan jaringan kampus PTIIK UB dipengaruhi adanya pengaturan *static* DNS pada tiap *router* yang digunakan di area kampus PTIIK UB.

Tabel 4.9 Hasil Uji DNS *Spoofing*

No.	Aplikasi	Alamat IP	Cleartext ditemukan?
1	nas.ub.ac.id	175.45.189.132	Tidak
2	bais.ub.ac.id	175.45.184.222	Tidak
3	siam.ub.ac.id	175.45.184.86	Tidak
4	gapura.ub.ac.id	175.45.184.74	Tidak
5	blog.ub.ac.id	175.45.184.5	Tidak
6	siado.ub.ac.id	175.45.184.104	Tidak
7	e-complaint.ub.ac.id	175.45.184.75	Tidak



## BAB V

### ANALISIS HASIL PENGUJIAN

Berdasarkan hasil pengujian pada bab implementasi dan pengujian yang telah dilakukan pada skenario jaringan yang sudah dirancang, uji serang dapat dilakukan di dua lingkungan jaringan yang berbeda. Terdapat perbedaan dari masing-masing tahap pengujian pada kedua lingkungan jaringan yang dipilih, yaitu lingkungan jaringan konfigurasi standar yang ada di rumah dan lingkungan jaringan konfigurasi lanjut yang ada di kampus PTIIK UB.

Dalam bab ini akan dijelaskan bagaimana analisa dan perbandingan hasil uji serang dari masing-masing lingkungan jaringan yang sudah ditetapkan. Berikut adalah analisa dari uji serang yang terdiri dari uji penangkapan *traffic*, uji SSL *hijacking*, uji *session hijacking* dan uji DNS *spoofing*.

#### 5.1 Analisa Uji Penangkapan Traffic

Uji penangkapan *traffic* dinyatakan berhasil dengan ditemukannya *cleartext* berisi *username* dan *password* yang dikirim dari target ke tujuan. Keberhasilan dari uji *capturing* berdasarkan tidak adanya protokol keamanan SSL (HTTPS) pada halaman yang berisikan kolom *login* serta kolom ganti *password*. Perbedaan hasil uji serang pada dua lingkungan jaringan adalah aplikasi nas.ub.ac.id yang tidak bisa diakses dari luar lingkungan jaringan kampus Universitas Brawijaya. Berikut ini adalah tabel perbandingan keberhasilan dari kedua lingkungan jaringan:

Tabel 5.1 Analisa Uji Penangkapan *Traffic*

Aplikasi	Lingkungan Jaringan	
	Rumah	Kampus UB
nas.ub.ac.id	-	Berhasil
bais.ub.ac.id	Gagal	Gagal
siam.ub.ac.id	Gagal	Gagal
gapura.ub.ac.id	Gagal	Gagal
blog.ub.ac.id	Gagal	Gagal
siado.ub.ac.id	Gagal	Gagal
e-complaint.ub.ac.id	Berhasil	Berhasil

Tingkat keberhasilan pengujian penangkapan *traffic* pada kedua lingkungan jaringan ini ditampilkan pada persentase keberhasilan, pada lingkungan jaringan dengan konfigurasi standar didapatkan tingkat keberhasilan sebesar 17% sedangkan pada lingkungan dengan konfigurasi lanjut didapatkan tingkat keberhasilan sebesar 28%.

## 5.2 Analisa Uji SSL Hijacking

Uji SSL *hijacking* dinyatakan berhasil dengan ditemukannya *cleartext* berisi *username* dan *password* yang dikirim dari target ke tujuan. Kegagalan SSL *hijacking* pada dua aplikasi yaitu *siam.ub.ac.id* dan *gapura.ub.ac.id* adalah terjadinya *redirect loop* yang mengakibatkan *tool* SSLStrip tidak dapat menerjemahkan halaman *web* dari HTTPS ke HTTP. *Redirect loop* diakibatkan oleh permintaan konten aplikasi berbasis *web* secara terus-menerus yang tidak bisa dilayani oleh *server* disebabkan tidak tersedianya konten aplikasi berbasis *web* tersebut pada port HTTP (80).

Perbedaan hasil uji serang pada dua lingkungan jaringan sama seperti pengujian sebelumnya, aplikasi *nas.ub.ac.id* tidak bisa diakses dari luar lingkungan jaringan kampus Universitas Brawijaya. Berikut ini adalah tabel perbandingan keberhasilan dari kedua lingkungan jaringan:



Tabel 5.2 Analisa Uji SSL Hijacking

Aplikasi	Lingkungan Jaringan	
	Rumah	Kampus UB
nas.ub.ac.id	-	Berhasil
bais.ub.ac.id	Berhasil	Berhasil
siam.ub.ac.id	Gagal	Gagal
gapura.ub.ac.id	Gagal	Gagal
blog.ub.ac.id	Berhasil	Berhasil
siado.ub.ac.id	Gagal	Gagal
e-complaint.ub.ac.id	Berhasil	Berhasil

Tingkat keberhasilan pengujian SSL *hijacking* pada kedua lingkungan jaringan ini ditampilkan pada persentase keberhasilan, pengujian ini mempunyai persentase keberhasilan yang lebih besar daripada pengujian sebelumnya. Pada lingkungan jaringan dengan konfigurasi standar didapatkan tingkat keberhasilan sebesar 50% sedangkan pada lingkungan dengan konfigurasi lanjut didapatkan tingkat keberhasilan sebesar 57%.

### 5.3 Analisa Uji Session Hijacking

Uji *session hijacking* dinyatakan berhasil dengan keberhasilan penyerang melakukan akses pada aplikasi berbasis *web* menggunakan akun milik target. Kegagalan *session hijacking* pada tiga aplikasi yaitu *siam.ub.ac.id*, *gapura.ub.ac.id* dan *siado.ac.id* adalah tidak adanya *session cookies* yang didapat oleh penyerang serta mengakibatkan penyerang tidak dapat melakukan akses pada aplikasi tersebut menggunakan akun milik target.

Perbedaan hasil pada dua lingkungan jaringan sama seperti pengujian sebelumnya, aplikasi *nas.ub.ac.id* tidak bisa diakses dari luar lingkungan jaringan kampus Universitas Brawijaya. Berikut ini adalah tabel perbandingan keberhasilan dari kedua lingkungan jaringan:



Tabel 5.3 Analisa Uji *Session Hijacking*

Aplikasi	Lingkungan Jaringan	
	Rumah	Kampus UB
nas.ub.ac.id	-	Gagal
bais.ub.ac.id	Berhasil	Berhasil
siam.ub.ac.id	Gagal	Gagal
gapura.ub.ac.id	Gagal	Gagal
blog.ub.ac.id	Berhasil	Berhasil
siado.ub.ac.id	Gagal	Gagal
e-complaint.ub.ac.id	Berhasil	Berhasil

Tingkat keberhasilan pengujian *session hijacking* pada kedua lingkungan jaringan ini ditampilkan pada persentase keberhasilan. Aplikasi nas.ub.ac.id memang bisa diakses dari dalam lingkungan jaringan kampus Universitas Brawijaya, namun *session cookies* tetap tidak bisa ditangkap oleh penyerang. Pada lingkungan jaringan dengan konfigurasi standar didapatkan tingkat keberhasilan sebesar 50% sedangkan pada lingkungan dengan konfigurasi lanjut didapatkan tingkat keberhasilan sebesar 43%.

#### 5.4 Analisa Uji DNS Spoofing

Uji DNS *spoofing* dinyatakan berhasil dengan ditemukannya *cleartext* berisi *username* dan *password* yang dikirim dari target ke penyerang yang berpura-pura sebagai tujuan (*server*). Perbedaan hasil uji serang pada dua lingkungan jaringan adalah DNS *spoofing* yang tidak bisa berjalan pada lingkungan jaringan kampus Universitas Brawijaya dikarenakan adanya pengaturan pada *router* yang akan selalu memberikan alamat DNS lokal UB kepada siapa saja yang terhubung dengan jaringan kampus PTIIK UB. Berikut ini adalah tabel perbandingan keberhasilan dari kedua lingkungan jaringan:

Tabel 5.4 Analisa Uji DNS *Spoofing*

Aplikasi	Lingkungan Jaringan	
	Rumah	Kampus UB
nas.ub.ac.id	-	Gagal
bais.ub.ac.id	Berhasil	Gagal
siam.ub.ac.id	Berhasil	Gagal
gapura.ub.ac.id	Berhasil	Gagal
blog.ub.ac.id	Berhasil	Gagal
siado.ub.ac.id	Berhasil	Gagal
e-complaint.ub.ac.id	Gagal	Gagal

Tingkat keberhasilan pengujian DNS *spoofing* pada kedua lingkungan jaringan ini ditampilkan pada persentase keberhasilan, pengujian ini mempunyai persentase keberhasilan yang lebih besar daripada pengujian lainnya. Pada lingkungan jaringan dengan konfigurasi standar didapatkan tingkat keberhasilan sebesar 83% sedangkan pada lingkungan dengan konfigurasi lanjut didapatkan tingkat keberhasilan sebesar 0%.

## 5.5 Analisa Penanggulangan Sniffing

Pengujian penangkapan *traffic*, *SSL hijacking*, *session hijacking* dan DNS *spoofing* dapat dilakukan dengan adanya *sniffing* terlebih dahulu oleh penyerang. *Sniffing* dapat ditanggulangi dengan tindakan pencegahan yang dilakukan di sisi target maupun *router* dengan melakukan beberapa konfigurasi. Langkah yang dapat dipilih adalah dengan melakukan pemeriksaan ARP *poisoning*, deteksi ARP *poisoning* serta pencegahan ARP *poisoning*.

### 5.5.1 Pemeriksaan ARP Poisoning

Pemeriksaan ARP Poisoning dilakukan dengan menampilkan tabel ARP pada PC yang dimiliki oleh target, jika tabel ARP tersebut memiliki beberapa alamat IP dengan alamat MAC yang sama maka dapat dinyatakan bahwa target tersebut sudah menjadi korban ARP *spoofing* atau ARP *poisoning*. Berikut adalah gambar tabel ARP normal dan tabel ARP yang sudah dikenai *poisoning*:

```

C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\ryansarah>arp -a

Interface: 192.168.1.4 --- 0xd
Internet Address      Physical Address      Type
192.168.1.1          cc-1a-fa-9b-3d-b0    dynamic
192.168.1.5          00-21-00-52-35-5a    dynamic
192.168.1.255       ff-ff-ff-ff-ff-ff    static
224.0.0.2           01-00-5e-00-00-02    static
224.0.0.22          01-00-5e-00-00-16    static
224.0.0.252         01-00-5e-00-00-fc    static
224.0.0.253         01-00-5e-00-00-fd    static
239.255.255.250     01-00-5e-7f-ff-fa    static
255.255.255.255     ff-ff-ff-ff-ff-ff    static

C:\Users\ryansarah>

```

Gambar 5.1 Tabel ARP Normal

```

C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\ryansarah>arp -a

Interface: 192.168.1.4 --- 0xd
Internet Address      Physical Address      Type
192.168.1.1          00-21-00-52-35-5a    dynamic
192.168.1.5          00-21-00-52-35-5a    dynamic
192.168.1.255       ff-ff-ff-ff-ff-ff    static
224.0.0.2           01-00-5e-00-00-02    static
224.0.0.22          01-00-5e-00-00-16    static
224.0.0.252         01-00-5e-00-00-fc    static
224.0.0.253         01-00-5e-00-00-fd    static
239.255.255.250     01-00-5e-7f-ff-fa    static
255.255.255.255     ff-ff-ff-ff-ff-ff    static

C:\Users\ryansarah>_

```

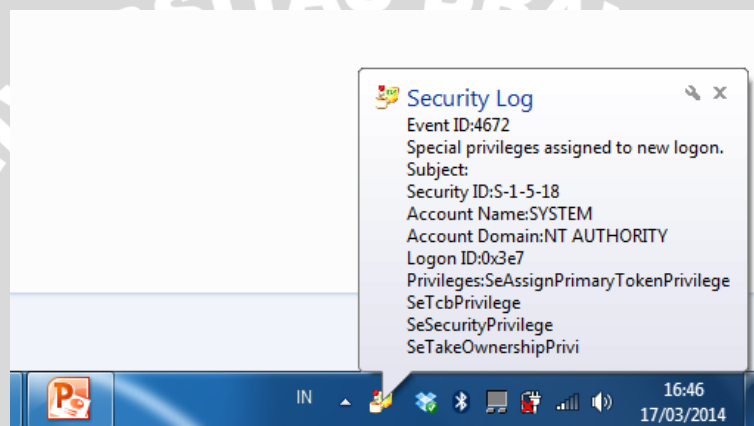
Gambar 5.2 Tabel ARP Ter-poisson

Gambar 5.9 menunjukkan bahwa tabel ARP pada PC target memiliki informasi alamat IP dan MAC yang salah, dapat dilihat pada alamat IP 192.168.1.1 (yang merupakan alamat IP dari *router*) memiliki alamat MAC yang sama dengan alamat IP 192.168.1.5. Dari dua gambar diatas dapat diketahui bahwa PC dengan alamat IP 192.168.1.5 adalah PC yang dimiliki oleh penyerang.

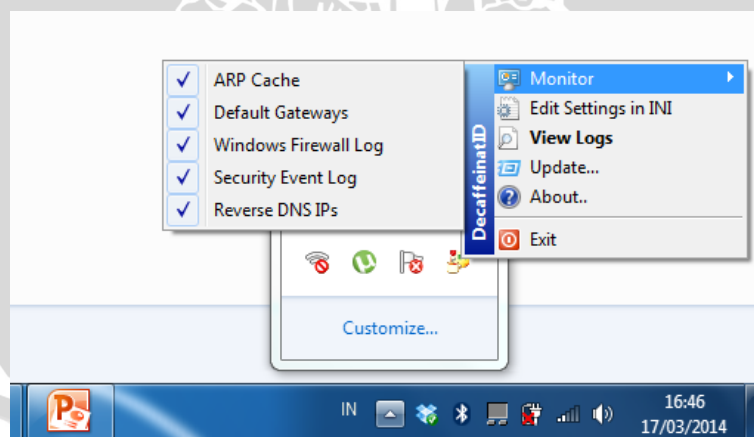


### 5.5.2 Deteksi ARP Poisoning

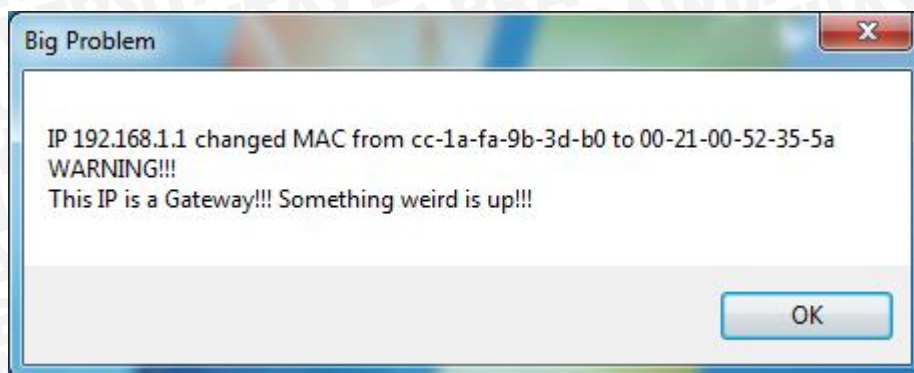
Deteksi ARP Poisoning dapat dilakukan oleh target dengan bantuan *software* tanpa harus memeriksa tabel ARP secara manual, sudah ada beberapa *software* yang disediakan secara gratis seperti DecaffeineatID [ADR-14] dan XARP [CRS-2011] pada sistem operasi windows serta ARPoN pada sistem operasi linux. *Software* DecaffeineatID dan XARP berfungsi untuk melakukan *monitor* pada elemen penting yang menjadi tujuan serangan ARP *poisoning* seperti tabel ARP. Berikut adalah tampilan dari *software* DecaffeineatID dan XARP yang digunakan oleh target:



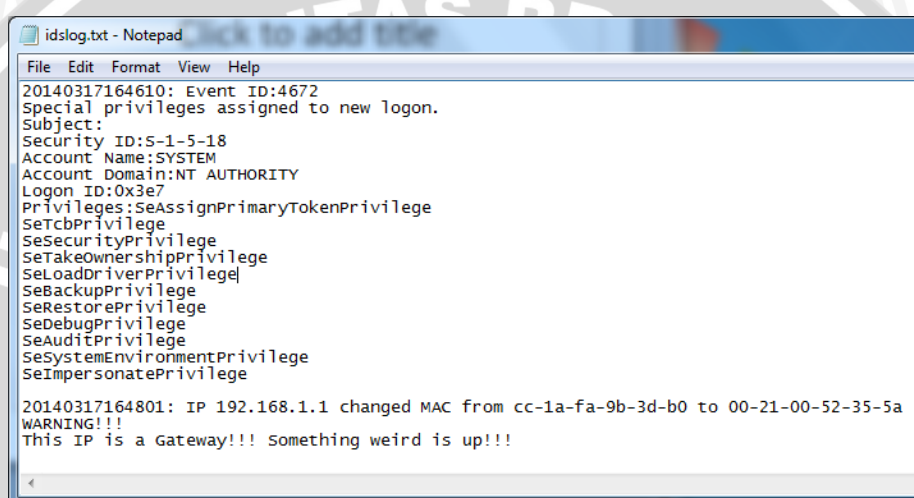
**Gambar 5.3** DecaffeineatID Berjalan di *Taskbar*



**Gambar 5.4** *Monitor* pada DecaffeineatID

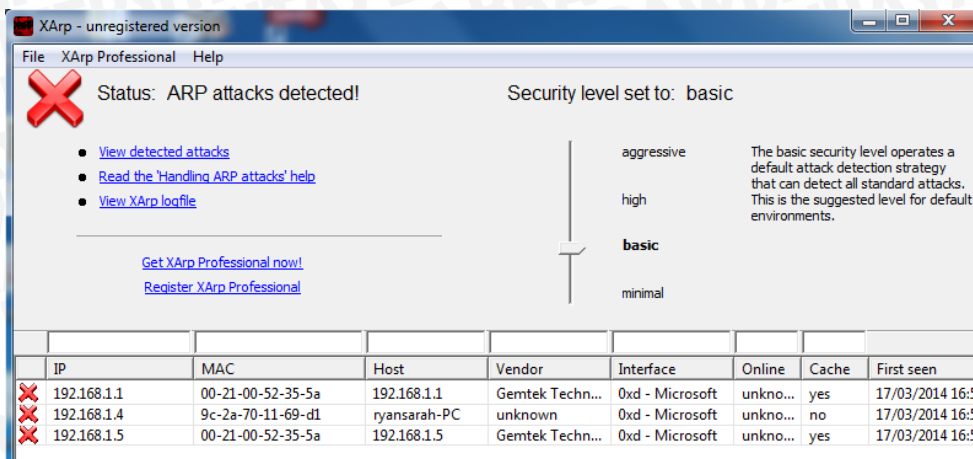


**Gambar 5.5** DecaffeineatID Mendeteksi Adanya Perubahan Tabel ARP

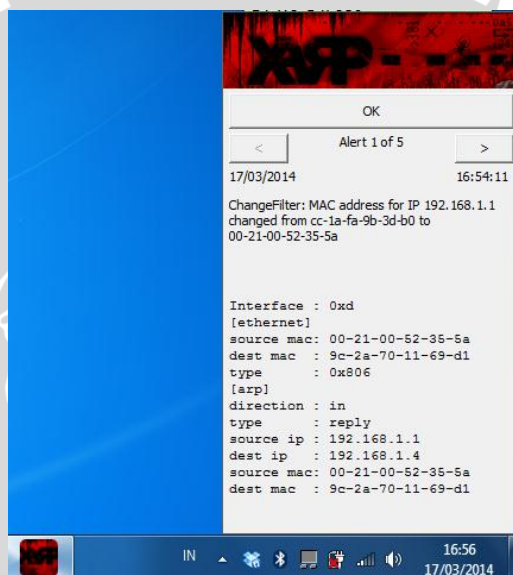


**Gambar 5.6** Log File pada DecaffeineatID

*Software* DecaffeineatID memiliki ukuran file yang sangat ringan namun mempunyai beberapa fitur penting seperti monitor tabel ARP dan *default gateway* serta membuat *log* setiap ada perubahan pada tabel ARP. Gambar 5.12 adalah tampilan jendela yang muncul pada saat terjadi perubahan alamat MAC pada alamat IP *gateway* atau *router* didalam tabel ARP. Jendela yang muncul adalah peringatan pada target bahwa telah terjadi adanya ARP *poisoning*.



Gambar 5.7 Tampilan XARP Dengan Rincian Tabel ARP



Gambar 5.8 XARP Mendeteksi Adanya Perubahan Tabel ARP

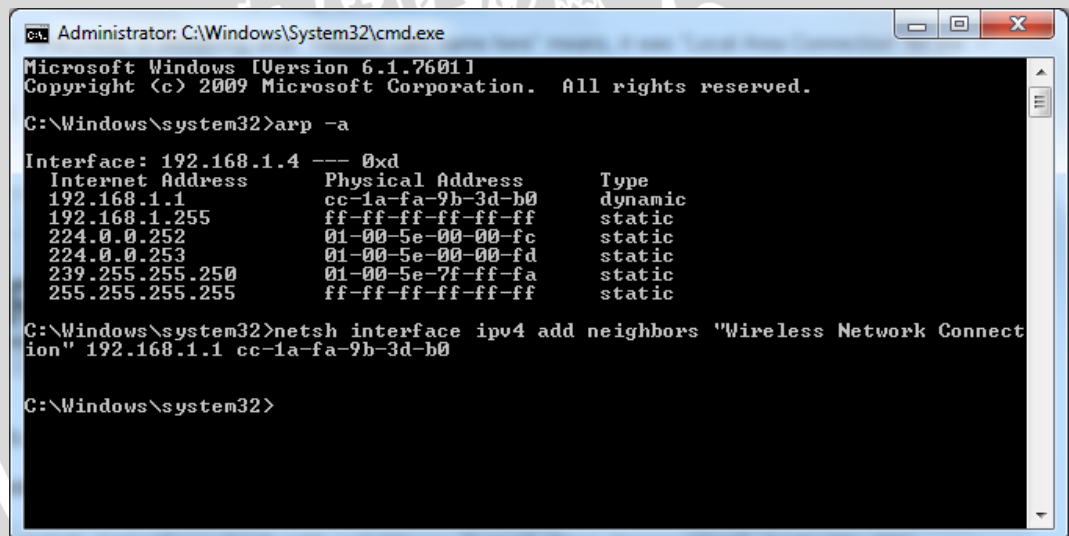
Software XARP memiliki ukuran file yang relatif ringan dan mempunyai fitur dan tampilan yang lebih kompleks daripada DecaffeineatID. Pada software ini terdapat versi *unregistered* dan *professional*, namun pada versi *unregistered* sudah bisa melakukan deteksi ARP poisoning seperti yang ditunjukkan pada gambar 5.14 yang menunjukkan bahwa alamat IP 192.168.1.1 memiliki alamat MAC yang sama dengan alamat IP 192.168.1.5.



Target dapat mengetahui secara realtime jika terjadi serangan ARP *poisoning* dengan adanya notifikasi yang muncul pada taskbar, seperti pada gambar 5.15 yang menunjukkan rincian perubahan alamat MAC pada alamat IP *router*.

### 5.5.3 Pencegahan ARP Poisoning

Pencegahan ARP *Poisoning* dapat dilakukan di sisi target ataupun di sisi *router*, pencegahan di sisi target adalah dengan merubah alamat IP *gateway* pada PC dari *dynamic* menjadi *static*. Perubahan alamat IP *gateway* membuat PC target tidak akan melakukan *update* tabel ARP jika ada ARP *reply* dari penyerang yang berisikan informasi alamat IP dan MAC yang tidak seharusnya (palsu). Saat alamat IP *gateway* dirubah menjadi *static*, penyerang tidak bisa melakukan ARP *poisoning* kepada target sehingga percobaan *sniffing* menjadi gagal.



```

Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32>arp -a

Interface: 192.168.1.4 --- 0xd
Internet Address      Physical Address      Type
192.168.1.1           cc-1a-fa-9b-3d-b0    dynamic
192.168.1.255        ff-ff-ff-ff-ff-ff    static
224.0.0.252          01-00-5e-00-00-fe    static
224.0.0.253          01-00-5e-00-00-fd    static
239.255.255.250     01-00-5e-7f-ff-fa    static
255.255.255.255     ff-ff-ff-ff-ff-ff    static

C:\Windows\system32>netsh interface ipv4 add neighbors "Wireless Network Connect
ion" 192.168.1.1 cc-1a-fa-9b-3d-b0

C:\Windows\system32>

```

Gambar 5.9 Perintah Static IP Gateway pada Windows

```

C:\Windows\system32\cmd.exe
C:\Users\ryansarah>arp -a

Interface: 192.168.1.4 --- 0xd
Internet Address      Physical Address      Type
192.168.1.1          cc-1a-fa-9b-3d-b0    static
192.168.1.255        ff-ff-ff-ff-ff-ff    static
224.0.0.252          01-00-5e-00-00-fc    static
224.0.0.253          01-00-5e-00-00-fd    static
239.255.255.250     01-00-5e-7f-ff-fa    static
255.255.255.255     ff-ff-ff-ff-ff-ff    static

C:\Users\ryansarah>

```

**Gambar 5.10** Alamat IP *Gateway* Menjadi *Static* pada Tabel ARP Target

ettercap NG-0.7.4.2

Start Targets Hosts View Mitm Filters Logging Plugins Help

Host List  Plugins

Name	Version	Info
arp_cop	1.1	Report suspicious ARP activity
autoadd	1.2	Automatically add new victims in the target range
chk_poison	1.1	Check if the poisoning had success
dns_spoof	1.1	Sends spoofed dns replies
dos_attack	1.0	Run a d.o.s. attack against an IP address
dummy	3.0	A plugin template (for developers)
find_conn	1.0	Search connections on a switched LAN

Unified sniffing was stopped.  
Starting Unified sniffing...

Activating chk\_poison plugin...  
chk\_poison: Checking poisoning status...  
chk\_poison: No poisoning at all :(

**Gambar 5.11** ARP *Poisoning* Gagal pada Penyerang

## BAB VI

### PENUTUP

#### 6.1 Kesimpulan

Dari hasil implementasi, pengujian dan analisis uji serang pada jaringan lokal menggunakan metode penangkapan *traffic*, *SSL hijacking*, *session hijacking* dan *DNS spoofing*, dapat disimpulkan bahwa:

1. Pengujian serangan pada jaringan lokal dirancang dengan menggunakan dua tahap pengujian, yaitu tahap *reconnaissance* dan *active attacks*. Tahap *reconnaissance* bertujuan untuk mengumpulkan informasi terkait tiga kondisi yaitu kondisi pengamanan jaringan, informasi target serangan serta berjalannya protokol ARP yang digunakan untuk menentukan jenis dan tujuan serangan terhadap jaringan lokal. Tahap *active attacks* terdiri dari empat metode serangan yang dilakukan setelah informasi dari tahap pertama selesai didapatkan.
2. Tingkat keberhasilan dari empat metode serangan berbeda-beda sesuai dengan fitur dari masing-masing serangan serta kondisi dari masing-masing lingkungan jaringan. Berikut adalah tabel yang berisi persentase dari keempat metode di dua lingkungan jaringan yang berbeda:

**Tabel 6.1** Persentase Tingkat Keberhasilan Pengujian

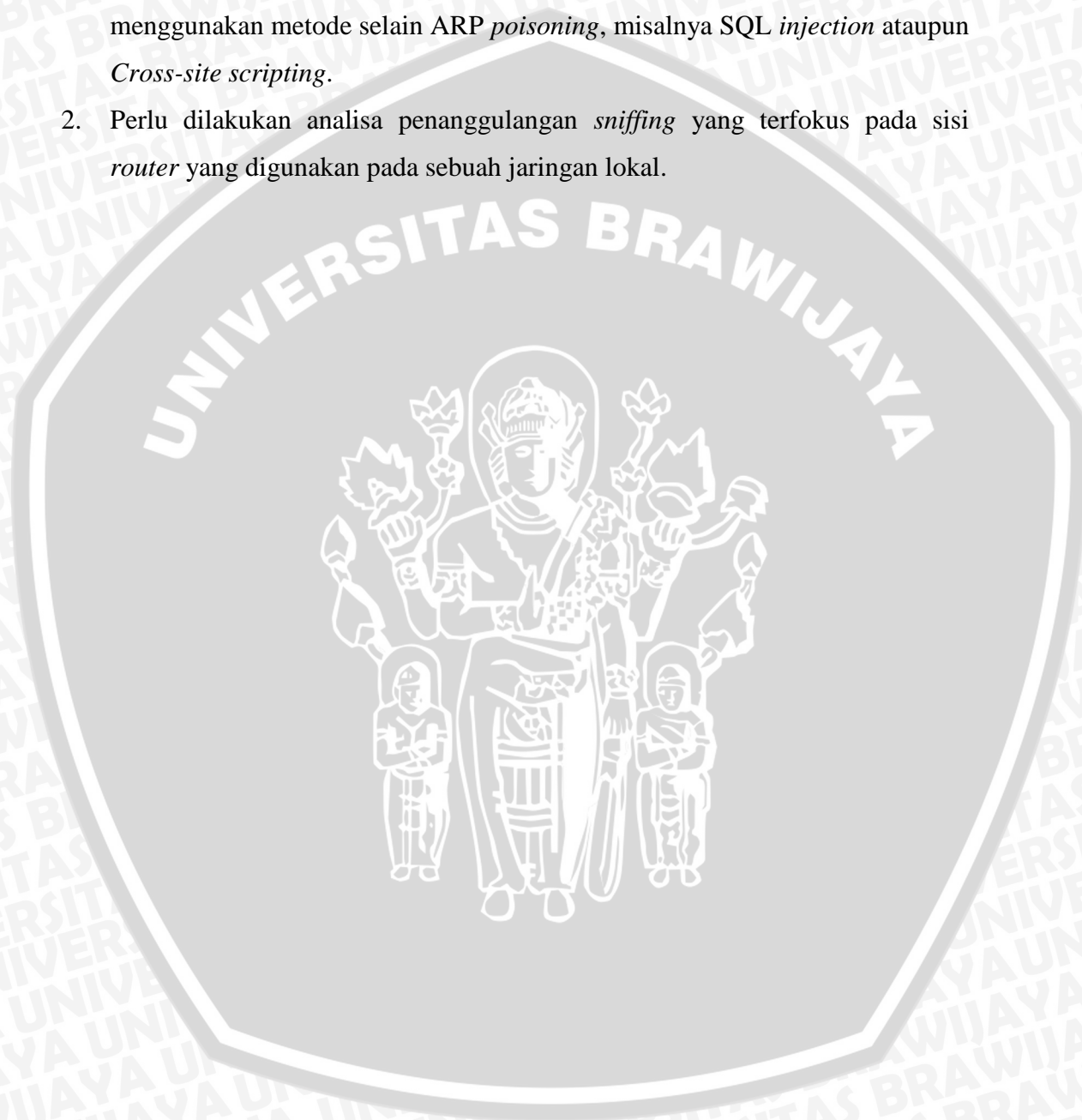
Metode Serangan	Lingkungan Jaringan	
	Konfigurasi Standar	Konfigurasi Lanjut
Penangkapan <i>Traffic</i>	17%	28%
<i>SSL Hijacking</i>	50%	57%
<i>Session Hijacking</i>	50%	43%
<i>DNS Spoofing</i>	83%	0%



## 6.2 Saran

Saran yang dapat disampaikan penulis untuk pengembangan penelitian ini adalah:

1. Perlu dilakukan pengujian serangan pada aplikasi berbasis *web* UB menggunakan metode selain *ARP poisoning*, misalnya *SQL injection* ataupun *Cross-site scripting*.
2. Perlu dilakukan analisa penanggulangan *sniffing* yang terfokus pada sisi *router* yang digunakan pada sebuah jaringan lokal.



## DAFTAR PUSTAKA

- [ABS-13] Ahmad Budi Setiawan. 2013, *Implementasi Tata Kelola Keamanan Informasi Nasional dalam Kerangka e-Government*. Universitas Indonesia.
- [HRY-10] Hary Fernando. 2010, *Studi dan Implementasi Sistem Keamanan Berbasis Web dengan Protokol SSL di Server Students Informatika ITB*. Institut Teknologi Bandung.
- [FQI-09] Fang Qi, Et Al. 2009, *SSL-Enabled Trusted Communication: Spoofing and Protecting the Non-Cautions Users*. Security and Communication Networks, Wiley InterScience.
- [FAM-07] Fata Mukhlis. 2007, *Analisis Keamanan Internet Banking Bank Mandiri*. Institut Teknologi Bandung.
- [NAS-08] Nwabude Arinze Sunday. 2008, *Wireless Local Area Network (WLAN): Security Risk Assessment and Countermeasures*. Blekinge Institute of Technology.
- [VAM-05] Vamshidhar Chillamcharla. 2005, *ARP Spoofing*, diakses pada 02 Januari 2014, <<http://www.data.ks.uni-freiburg.de/download/inetworkSS05/practical/arp/arp-spoofing.pdf>>.
- [SAN-10] Chris Sanders. 2010, *Understanding Man-in-the-Middle Attack*, diakses pada 02 Januari 2014, <[http://www.windowsecurity.com/articles-tutorials/authentication\\_and\\_encryption/Understanding-Man-in-the-Middle-Attacks-ARP-Part1.html](http://www.windowsecurity.com/articles-tutorials/authentication_and_encryption/Understanding-Man-in-the-Middle-Attacks-ARP-Part1.html)>.

[ADR-14] Adrian Crenshaw. 2014, *DecaffeineatID: A Very Simple IDS / Log Watching App / ARPWatch For Windows*, diakses pada 31 Maret 2014,

<<http://www.irongeek.com/i.php?page=security/decaffeinatid-simple-ids-arpwatch-for-windows>>.

[CRS-11] Christoph Mayer. 2011, *XARP – Advanced ARP Spoofing Detection*, diakses 31 Maret 2014,

<<http://www.chrismc.de/development/xarp/>>.

