

# Simulating Cyber Attacks and Designing Malware using Python

Shubham Inder<sup>a,1</sup>, Akash Ankit<sup>b,1</sup>, Anuj Sharma<sup>c,2</sup>, Rahul Johari<sup>1d,1</sup>,  
Deo Prakash Vidyarthi<sup>e,3</sup>

<sup>1</sup>SWINGER: Security, Wireless, IoT Network Group of Engineering and Research, University School of Information, Communication and Technology, Guru Gobind Singh Indraprastha University, Dwarka, 110078, New Delhi, India.

<sup>2</sup>Royal Melbourne Institute of Technology (RMIT University), Melbourne, 3000, Victoria, Australia.

<sup>3</sup>Parallel and Distributed System Lab, School of Computer and Systems Sciences, JNU, New Delhi, 110067, New Delhi, India.

Received: date / Accepted: date

**Abstract** In this Internet era, we have become heavily dependent on the World Wide Web and it has become a basic amenity for humans. A normal user can be trapped easily in this vicious web if not careful enough. It is a field that requires users to be on their toes every time they try to connect to this enormous interconnection of networks. Reports and Surveys have said that in the coming years, Cybercrime could become the biggest threat. In this proposed work, the authors have deployed applications and designed malware for the purpose of simulating various kinds of cyber attacks so that they could study their effects & affects and ultimately devise solutions to resolve the vulnerabilities in a system. These applications included networking concepts & protocols and a software based keylogger which was developed with no malicious intent, but to analyse the architectural view or design of a malware and to study the stages of a malware attack like entry, distribution, exploitation, infection and execution. Efforts have been made to demonstrate spoofing attacks with an idea of identifying vulnerabilities and security loopholes in a particular system that can be exploited by an attacker if not resolved within time.

**Keywords** Cyber Security · Address Resolution Protocol(ARP) · Vulnerabilities · Malware Design · Networking · Network Scanner · ARP Spoofer · Packet Sniffer · KeyLogger · Networking

## 1 Introduction

Cyber Security is the art of protecting our networks and devices from unauthorised access. It can also be termed as the protection of valuable assets in the form of data. The reason cyber security exists in the first place is the rise of a digital world, where everything is interconnected like a web. This interconnection has resulted in increased demand to protect data, the increased demand to steal data, the presence of cyber armies, increased financial frauds, etc. Cyber Security comes into the picture whenever there is a financial fraud or an attack by the cyber army. Cyber security requires a strategy to prevent these kinds of attacks and that is why the field has been divided into categories like Network Security, Application Security, Operational Security, Information Security, Disaster Recovery. Cyber attacks are given different names according to their purposes. Sometimes these attacks are done mainly for financial gain or to create war hysteria while sometimes they are carried out with a politically motivated intent. To all those problems, ethical hacking comes as a solution. Ethical hacking is a method used to tap into a system with proper authorisation. The main motive of this technique is to find vulnerabilities in the system and think of a solution to fix those vulnerabilities before an attacker exploits them. It gives an edge to the security analysts by providing them with the essential data which they would not have gotten otherwise.

Main requirements of ethical hacking: Stay legitimate, get the approval before getting access and working for the security evaluation, determine the limit for the evaluation so that the ethical hacker's work stays legitimate and well within the association's limits, de-

<sup>a</sup>e-mail: indershubham@gmail.com

<sup>b</sup>e-mail: akash6727e@gmail.com

<sup>c</sup>e-mail: anujsharma030701@gmail.com

<sup>d</sup>e-mail: rahul@ipu.ac.in

<sup>e</sup>e-mail: dpvidyarthi08@gmail.com

<sup>1</sup>corresponding author

termine the vulnerabilities, inform the association of all vulnerabilities found during the evaluation, provide solutions to resolve the vulnerabilities. The cyber ecosystem has a set of specific Free and Open Source Software (FOSS) which are very helpful in this field as they secure systems and networks with ease. The authors of this paper suggest that advancements, cycles and practices intended to secure PCs from hacking and unapproved access is crucial and so they have extensively talked about “Man In The Middle” attacks in this paper. It was observed in the study that one of the main reasons why cyber crimes take place is because of the elusive strategies and dangerous nature of cybercriminals. Based on these strategies, this paper follows a minimalistic approach for an exhaustive and systematic attack to exploit the network vulnerabilities of the victim. The research begins with a python program to change the MAC Address for our device and then scan the network using an Address Resolution Protocol (ARP) scan, the host sends an ARP request with the IP address of the target to identify all the active network assets. Identification of the network is followed by redirection of the flow of packet i.e small bundles of huge information being transferred every second. A man in middle approach is followed here known as ARP spoofing. These packets are assessed in the next program called packet sniffer where important information is extracted from various distinct packages like username, password, name, log in etc. The next program fulfils the main objective of malware analysis known as file interceptor in which authors exploited the vulnerability of HTTP protocol. In the program, they demonstrate how can someone intercept downloads that the target is downloading and replace the original download file of the targeted victim with a backdoor, malware, keylogger or any other file. The final attack will be a keylogger program installed on the victim computer using “File Interceptor” that will intercept all the keystrokes of the user and mail them to the attacker that will expose the victim’s usernames, passwords and other personal information. These attacks are strategically planned to exploit the target’s security and vulnerability to make the necessary changes for a better and sustainable network protocol.

In summing up the features, this paper discusses various spoofing techniques that are used by attackers to exploit the system. Efforts were made to illustrate and simulate the concepts like Address Resolution Protocol, Domain Name System, Open Systems Interconnection Model in an uncomplicated yet explanatory way. Successful simulations of ARP spoofing, File interception, DNS Spoofing Packet Sniffing were carried out

in a testing lab created on Kali-Linux. In this research, the authors demonstrated a successful software version of keylogger using Python, able of gathering keystrokes and sending back to an email ID. The work also discusses Keystroke logging, characteristics of Keyloggers and aims at educating the user about its characteristics. To reach the root of the problem, the people need to bring changes to the existing elements so that a new highly secure method can be proposed. The study has outlined practices that can be used to manipulate information which would lead to some awareness in the citizens about the threats. Like most cyber attacks, the best way to keep our data safe is regular scanning for any odd connections or traffic and most importantly, self-awareness and basic technical education.

## 2 Problem Statement

Ethical Hacking is about foreseeing things before attackers think of them. These days with the rising technology, a massive data breach can take place very easily if a server gets compromised due to these attacks. This includes intellectual property, sensitive data, protected health information (PHI), personal information, personally identifiable information (PII), and governmental and industry information systems. Without a cybersecurity program, organizations would not be able to defend themselves against data breach campaigns, which will make it an easy target for cybercriminals. Thus, it very important for the people to be aware of these attacks so that they can protect themselves on a personal level as well as on a mass level.

## 3 Objective and Motivation

The advancement in technology and accessibility to the World Wide Web has made our lives so much easier. It has become more accessible, secure and affordable than it was a few years back. The Internet is often called ‘the network of networks’ because it is the result of the interconnection of different networks, a system so enormous and complicated that it could create chaos if shut for just a few minutes. This superhighway of information has opened so many doors of opportunities to mankind whether it is STEM or politics it is being used everywhere regardless of demographics, but all those merits come at a cost. Such a huge network entails vulnerabilities and risks which are then assessed by attackers to exploit them and make a profit out of it, not always translates to financial gain. Cyberspace has not only helped normal users, but it has helped attackers who

have the technical knowledge to exploit the vulnerabilities for financial gain, cyber extortion, phishing or to create war hysteria while sometimes these exploitations or attacks are carried out with a politically motivated intent. Cyber technology has given a new face to fear and that is Cyber Terrorism and it has become an attractive option for modern zealots and extremists. Cyber terrorism is the amalgamation of cyberspace and terrorism. The term translates to unauthorised and violent attacks against networks, databases, servers of a particular organisation or country. These attacks are mainly done with the intent of spreading communal and political propaganda.

The paper follows a minimalistic approach for an exhaustive and systematic study on how network vulnerabilities are exploited by attackers and how these techniques can be used to analyse a person's machine ethically. The research tries to provide a perspective on how things work behind the scene and how an attacker could proceed further with the brute force attack. The whole paper has been presented in such a way that it clearly shows how the attacks are strategically planned to exploit the target's security and vulnerability if the user failed to pay heed to the various aspects of their machine. This paper makes an effort in educating users about the cyber security field and how they can weaponize their machines by understanding concepts like Networking and Protocols against the forces of evil.

#### 4 Literature Survey

**Dewar**[1] explains that the objective of network safety is to engage in exercises in the web freed from the risk of physical or electronic hurt (p. 18). How countries see the conglomeration of connections inside securitisation parts in digital protection issues and the attribution issue makes their network security technique and procedure different from each other. Dewar uses three broad terms to explain the norms of network protection, which are Active Cyber Défense (ACD) that revolves around removing danger and black hat hackers from both inside and outside the organization, Fortified Cyber Defense (FCD) that shapes a defensive environment, and Resilient Cyber Defense (RCD) that bases on ensuring system congruity. Furthermore, the author shows that the ACD is embraced by the US and the gathered Domain, while Germany uses the FCD, and the EU and Japan take on the RCD.

**Sreenivas** [2] perceived Keylogger by using TAKD estimations that can without a doubt consolidated into routine devices, for instance, switch, door, firewall, IDS,

and so on to further develop its keylogging ID. TAKD computation intertwined quirk-based acknowledgement part and log-based technique to beat the issue of imprint based distinguishing proof.

In 2013, **E. Ladakis**[3] introduced a secretive keylogger with an original methodology by investigating the graphics card as a substitute for facilitating a climate for the keylogger to work. Since cell phones have now turned into an indispensable piece of our lives, banking administrations are additionally being given on mobile applications. Individuals are dynamically becoming OK with utilizing the application since it makes the monetary interaction laidback.

**Ortoline** [4] executed the Discovery approach to perceive the most notable Keyloggers. Their model relied upon the strategies for a keystroke to the I/O configuration formed by Keyloggers.

**A. Bhardwaj** [5] recommends that the keyloggers should be characterized on two standards: the area of execution and functionalities advertised. Further, the keylogger can be either equipment based or software-based. With new keyloggers being presented, the location of keyloggers is likewise a space that is advancing.

**Y.Albatain** [6] likewise focuses on the location of keyloggers utilizing Graphics Preparing Units. Keyloggers are being tried against the most well-known and viable programming to check whether keyloggers can be distinguished or not.

**S.Moses** [7] in "Contact Interface and Keylogging Malware" explores the capacities of keyloggers to catch keystrokes from a virtual console. Later how other keyloggers react to it are likewise illustrated.

**Espoir K. Kabanga et al** [8] discuss the use of deep learning for malware classification. This paper suggests that Deep Learning Algorithms and Convolutional Neural Networks can be used for malware analysis. a model was proposed by the authors that were using Convolutional Neural Networks (CNN) to visualise malware as grayscale images and was able to categorise images obtained from the model with 98% accuracy.

**Andreas Moser et al** [9] suggest a way that allows the readers to explore paths that can be executed in multiple ways, in simpler terms, multiple execution paths. The authors also explained how these malicious programs get activated subject to certain instances. This approach provides a better understanding of circumstances under which suspicious actions are carried out and provided a clearer view and an extensive study on the topic of Malware Analysis.

**P. Ramesh Babu et al** [10] describes a variety of spoofing attacks and provides a clear view on how to detect and prevent these spoofing attacks. This paper was able to explain spoofing techniques like Internet

Protocol (IP) Spoofing, Address Resolution Protocol (ARP) Spoofing, E-Mail Spoofing, Domain Name System (DNS) Spoofing, Web Spoofing. Authors suggested ways on how these attacks can be pre-empted using methods like Ingress Filtering Method - IFM, Spoofing Prevention Method - SPM, Egress Filtering Method - EFM and encouraged users to monitor all logs to mitigate the potential damage of resources.

**Ryan Spangler[11]** in his research paper explains the various cyber-attacks in the form of packet sniffing is carried out. Paper follows a set path from explaining a basic sniffing attack to explaining types of sniffing attacks. This research states sniffing attacks on 2 major types of network traffic switched and non switched.

It also talks about the ARP cache that stores both layer 2 MAC addresses & the layer 3 IP addresses of the victim on the local network. Research also throws light on several procedures for packet sniffing on Layer 2 switched networks. The attacks carried out were 'ARP Cache Poisoning', 'CAM (Content Addressable Memory) Table Flooding' and 'Switch Port Stealing'. Every sniffing method is provided by a clear description. The research paper displays the process of sniffing attacks being executed on the switched networks and provided methods on how it can be prevented.

**Blagoj Nenovski et al [12]** in their paper throws some light on real Real-World ARP Attacks and Packet Sniffing. The paper gives a brief introduction on the working of LAN networks, ARP (Address Resolution Protocol) poisoning, MAC addresses and MitM (Man in the middle attacks). The research follows a real-world packet sniffing attack and goes on to explain how can these attacks be mitigated. The attack goes on to poison ARP of a network and captures the packets using Wireshark on hosts like Windows and android machines for capturing login forms. The paper goes on to explain some of the practices to prevent ARP attacks for example setting up a static entry to the ARP cache with the Gateway's IP and MAC address. This journal explains the vulnerability of users on the local network to DoS or MitM attacks despite having all of the latest technologies, protocols, modern hardware and software systems.

## 5 Modifying MAC (Media Access Control) Address

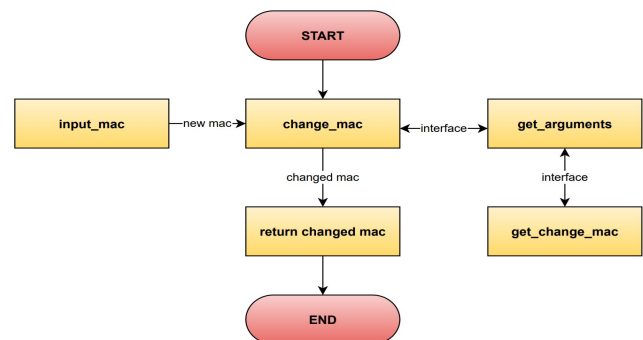
The first step in this research was a program to change the MAC (Media Access Control) address of the device that was used for attacking purposes. An algorithm was designed and implemented using Python.

**MAC ADDRESS :** MAC address (Media Access Control) is a hardware identification number that is used within the network to identify devices and transfer data within devices.

**IP ADDRESS :** An Internet Protocol address is a numerical label that is used in the internet to identify computers and devices

### 5.1 Algorithm for modifying MAC address

- 1) Creating a new project in PyCharm
- 2) Shebang line - `#!/usr/bin/env python`
- 3) Import subprocess module
- 4) Import optparse module
- 5) Import regex module
- 6) Put arguments using get\_argument function
- 7) **get\_arguments:**
- 8) create parser to return value - `from optparse.OptionParser()`
- 9) pass argument for interface
- 10) pass argument for mac address
- 11) use if-else for not mentioning any of the two arguments
- 12) Change mac address using change\_mac function
- 13) **change\_mac:**
- 14) use call function to disable the interface
- 15) modify the mac address to a new one
- 16) enable the interface
- 17) Use regex to find the old mac address by using `get_current_mac` function
- 18) **get\_current\_mac:**
- 19) store variable to identify mac address in REGEX format
- 20) use if-else to return the value if found
- 21) Call the functions
- 22) END



**Fig. 1** The above diagram illustrates the data flow in the MAC changer algorithm using various functions.

## 5.2 Result

Figure 2 shows a script running in the Kali Linux terminal to input the new mac address and if the user does not put the interface, the system will show the message as shown in figure 2.

Script: `python mac_changer.py -m 00:44:55:88:66:77`

```
root@kali: ~/PycharmProjects/mac_changer 126x35
root@kali:~/PycharmProjects/mac_changer# python mac_changer.py -m 00:44:55:88:66:77
Usage: mac_changer.py [options]

mac_changer.py: error: [-]Please specify an interface,use --help for more info.
root@kali:~/PycharmProjects/mac_changer#
```

**Fig. 2** Data flow diagram for MAC changer

Figure 3 shows the case when the user does not put the MAC address, the system will show the following message.

Script: `python mac_changer.py -i eth0`

```
root@kali: ~/PycharmProjects/mac_changer 126x35
root@kali:~/PycharmProjects/mac_changer# python mac_changer.py -i eth0
Usage: mac_changer.py [options]

mac_changer.py: error: [-]Please specify a mac address,use --help for more info.
root@kali:~/PycharmProjects/mac_changer#
```

**Fig. 3** Running the script without MAC

Figure 4 shows the case when the user inputs both the interface and the MAC address, the program will be executed successfully, thus changing the MAC address to 00:33:44:22:88

Script: `python mac_changer.py -i eth0 -m 00:33:44:22:88`

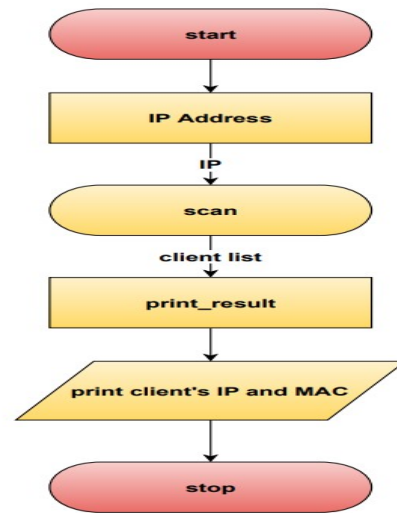
```
root@kali: ~/PycharmProjects/mac_changer 126x35
root@kali:~/PycharmProjects/mac_changer# python mac_changer.py -i eth0 -m 00:33:44:22:88
current MAC=08:00:27:83:18:31
[+]Changing MAC Address for eth0 to 00:33:44:22:88:88
[+] MAC Address was successfully changed to 00:33:44:22:88:88
root@kali:~/PycharmProjects/mac_changer#
```

**Fig. 4** Running the script with both the parameters

## 6 Network Scanner

Network scanning helps in distinguishing the active hosts on a network and connects them to their respective IP addresses. Network scanners send a packet to each IP address and wait for their response to tell the status of these devices (hosts) if it gets back a reply. The hosts are referred to as active if the response is received, while

others are declared inactive. Using an Address Resolution Protocol (ARP) scan, the host sends an ARP request with the IP address of the target. The ARP-scan network discovery tool is made to map the MAC addresses to their IP addresses. The ARP-scan tool uses the Address Resolution Protocol (ARP) to recognize the active networks that may not be easily identified by the network scanning devices.



**Fig. 5** Data flow diagram for Network scanner

Figure 6 illustrates the algorithm in the form of a DFD of a network scanner program that scans and distinguishes all the active hosts on a network and connects them to their respective IP address.

### 6.1 Algorithm for detecting all active hosts on a network

- 1) Creating a new project in PyCharm
- 2) Shebang line - `#!/usr/bin/env python`
- 3) Import scapy module
- 4) Extracting MAC address using `scan(ip)` function.
- 5) **scan(ip):**
- 6) create an ARP request
- 7) create broadcast for ethernet frame to find available IP's
- 8) send and receive function to give list of answered packets
- 9) for loop to iterate elements in answered list
- 10) create dictionary consisting the elements
- 11) Execute information using function `print result`
- 12) Call function with an argument `"10.0.2.1/24` to broadcast over all the channels
- 13) END

## 6.2 Result

Running a script in the Kali Linux terminal to get all the active hosts.

Script: `python3 network_scanner.py`

```
root@kali:~/PycharmProjects/network_scanner# python3 7_net_scan_FINAL.py
IP IS 10.0.2.1/24
IP          MAC
10.0.2.1    52:54:00:12:35:00
10.0.2.2    52:54:00:12:35:00
10.0.2.3    08:00:27:98:8d:91
10.0.2.15   08:00:27:4e:33:7d
```

**Fig. 6** Terminal window

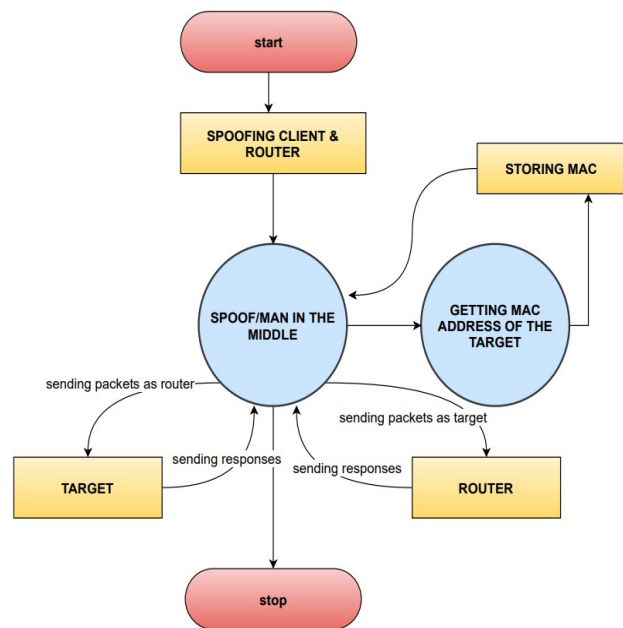
Figure 7 displays the working of the program where we receive the list of all the active hosts on a network along with their MAC and IP Address.

The advantages of using the network scanner are:

1. It permits the clients to find the IPv4 network connected devices.
2. It can identify and map the IP addresses of the devices to their MAC addresses in no time.
3. It can easily distinguish between replicated IP addresses.
4. It will find and dis-engage with devices showing an unusual activity.
5. It recognizes the devices by NIC vendor.

## 7 ARP Spoofer

ARP (Address Resolution Protocol) is a network protocol. It is used to discover devices connected to a network with the help of their IP (Internet Protocol) addresses. It allows the linking of IP (Internet Protocol) addresses to MAC (Media Access Control) addresses. It is used when a machine wants to establish a connection with some other device on a local network. For example, to establish a connection with the router, ARP is used by most of the devices to make a connection with the router and then finally to the internet. An ARP request is sent by the host and contains the IP address of the target machine. All the devices on the same network will see this ARP request, but only the machine having the same IP address that was being sent by the host will respond to this ARP request with the ARP reply containing its MAC address.



**Fig. 7** ARP data flow diagram

Figure 7 explains the working of the ARP Spoofer program where A middle man (MitM) pretends to be the router, re-routes the traffic and receives all the connection requests and packets from the victim's machine and send responses without the machine realizing that it is not the router.

ARP spoofing allows the attacker to redirect the flow of packets instead of flowing through a legitimate machine it would flow through a different machine, any requests sent or any responses received by the target computer. This allows the hacker to modify, read or drop information flowing through the hacker machine. It is often termed as a Man in the Middle (MitM) attack because it allows attackers to tap into systems and make them the man in the middle in a communication channel between two or more network devices. In order to simulate this attack, authors made a ARP spoofing program in Python and tested it in a virtual network setup on a remote computer using the attacking computer which was a Kali Linux system.

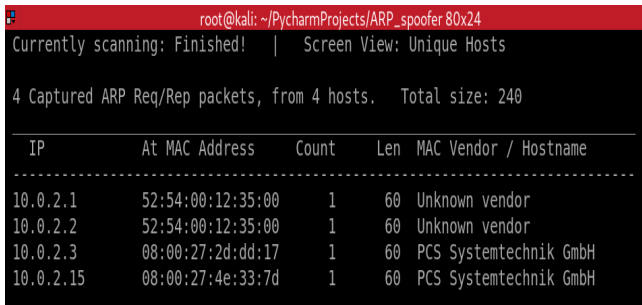
### The attack works as follows:

1. The whole network was scanned to determine the IP addresses. netdiscover -r 10.0.2.1/24 was used to scan the IP addresses.

**Router - 10.0.2.1**

**Attacking Machine - 10.0.2.17**

**Victim machine's IP address - 10.0.2.15**

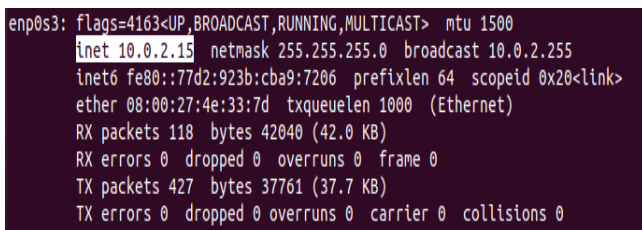


IP	At MAC Address	Count	Len	MAC Vendor / Hostname
10.0.2.1	52:54:00:12:35:00	1	60	Unknown vendor
10.0.2.2	52:54:00:12:35:00	1	60	Unknown vendor
10.0.2.3	08:00:27:2d:dd:17	1	60	PCS Systemtechnik GmbH
10.0.2.15	08:00:27:4e:33:7d	1	60	PCS Systemtechnik GmbH

**Fig. 8** Scanned networks

Figure 8 shows the IP and MAC addresses of all the devices that were connected to the same network as the attacking machine. Scanning was done by using the inbuilt module known as netdiscover.

Figure 9 displays the IP address of the victim machine before initiating the ARP spoofing attack. The IP address of the Ubuntu system is shown in figure 9 as inet 10.0.2.15



```

enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
    inet6 fe80::77d2:923b:cba9:7206 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:4e:33:7d txqueuelen 1000 (Ethernet)
    RX packets 118 bytes 42040 (42.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 427 bytes 37761 (37.7 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
  
```

**Fig. 9** IP address of the victim machine (Ubuntu)

2. Forged responses were sent using the python program. A Python-based interactive packet manipulation program known as Scapy was used in this program which was used for network scanning, sending & receiving packets, dropping a payload and other network-related tasks and after implementing this algorithm, ARP entries were updated for the devices and connection was established with the attacker machine.

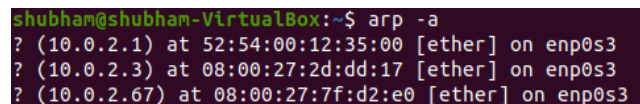
### 7.1 Algorithm for sending ARP responses

- 1) Creating a new project in PyCharm
- 2) Shebang line - `#!/usr/bin/env python`
- 3) Importing time & scapy module
- 4) Extracting MAC address using scan(ip) function. It takes only one argument - ip & returns a list containing MAC address
- 5) **scan(ip):**
- 6) creating an ARP request
- 7) creating broadcast to find out who has the respective IP
- 8) Spoofing function with 2 arguments (target\_ip & spoof\_ip)
- 9) **spoof(target\_ip,spoof\_ip):**
- 10) calling scan & passing argument target\_ip
- 11) creating packet using scapy module with destination as target\_ip, -
- 12) source as spoof\_ip & hardware destination as hwdst
- 13) sending packet
- 14) Creating a counter variable and assigning 0
- 15) **While true:**
- 16) calling spoof("10.0.2.15", "10.0.2.1") - Spoofing Client
- 17) calling spoof(("10.0.2.1", "10.0.2.15") - Spoofing Router
- 18) increment the counter
- 19) print "packet sent"
- 20) halt the program for 2 seconds
- 21)END

### 7.2 Result

Running a script in the Kali Linux terminal to allow the flow of packets through the attacker machine.

Before attack



```

shubham@shubham-VirtualBox:~$ arp -a
? (10.0.2.1) at 52:54:00:12:35:00 [ether] on enp0s3
? (10.0.2.3) at 08:00:27:2d:dd:17 [ether] on enp0s3
? (10.0.2.67) at 08:00:27:7f:d2:e0 [ether] on enp0s3
  
```

**Fig. 10** Ubuntu terminal window (Victim machine)

Figure 10 displays the IP and MAC address of all the devices connected to the same network and Figure 11 shows the ARP spoofer script running on the attacker machine.



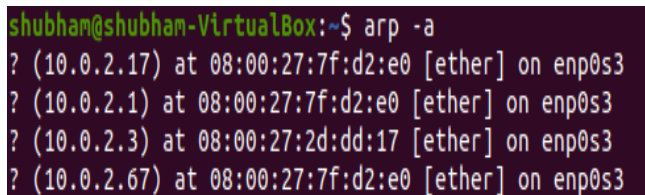
Running program



```
root@kali:~/PycharmProjects/ARP_spoofers# python3 2_arp.py
[+] Packets sent 10
```

**Fig. 11** Kali Linux terminal window (Attacker machine)

After attack



```
shubham@shubham-VirtualBox:~$ arp -a
? (10.0.2.17) at 08:00:27:7f:d2:e0 [ether] on enp0s3
? (10.0.2.1) at 08:00:27:7f:d2:e0 [ether] on enp0s3
? (10.0.2.3) at 08:00:27:2d:dd:17 [ether] on enp0s3
? (10.0.2.67) at 08:00:27:7f:d2:e0 [ether] on enp0s3
```

**Fig. 12** Now router's IP address (10.0.2.1) had been linked with Kali Linux machine's (Attacker Machine) MAC address (08:00:27:7f:d2:e0)

Figure 12 illustrates how router's IP address (10.0.2.1) had been linked with Kali Linux machine's (Attacker Machine) MAC address (08:00:27:7f:d2:e0). Authors were able to run the ARP spoofing attack successfully, but they needed something to collect all the packets and analyze them properly. In order to run this attack efficiently they went ahead with packet sniffing to capture and analyze each packet flowing through the victim machine.

## 8 Packet Sniffer

In networking, a packet is a small segment of a larger message and data is transferred in the form of these packets.

It is divided into 3 main sections –

1. Header – Contains sender's and receiver's IP address, protocol and packet number
2. Payload – It is the data
3. Trailer – Also known as footer, contains additional information

ARP spoofing attack was allowing the redirection of the flow of packets in the network and placing the attacker as the man in the middle.

The attack allowed the flow of information or packets through the main attacking machine, a Kali Linux system. Packet sniffer came into the picture because authors needed something to help us read all that information flowing through their machine.

Packet Sniffing is the process of monitoring every packet that is passing through a network. In networking, packets are one of the essential elements because they represent data in a segmented way. If someone wants to know what these fragments of data represent they need to build a program that could capture these packets in order to give information about browsing data, download & upload requests, URL's, login credentials, etc. In a regular network, data flows from source to destination without any interceptions along the way. When a packet is transmitted over a network, it passes through several nodes in a network. All the devices examine the packet individually to see if the packet is meant for them and if the packet is meant for some other node, nodes that are not being addressed just ignore the packet. But in packet sniffing, some nodes are programmed to not behave like this and designed for the collection of packets even if those packets are not meant for them.

First, the Scapy module was used to capture data that was being sent to or from one of the interfaces that were eth0 on their machine, and then authors ran this packet sniffer program against another machine on the same NAT network. Scapy library has a sniffer function and supports HTTP packets by default. It can capture the data using face parameter and uses prn parameter to call a function on each packet.



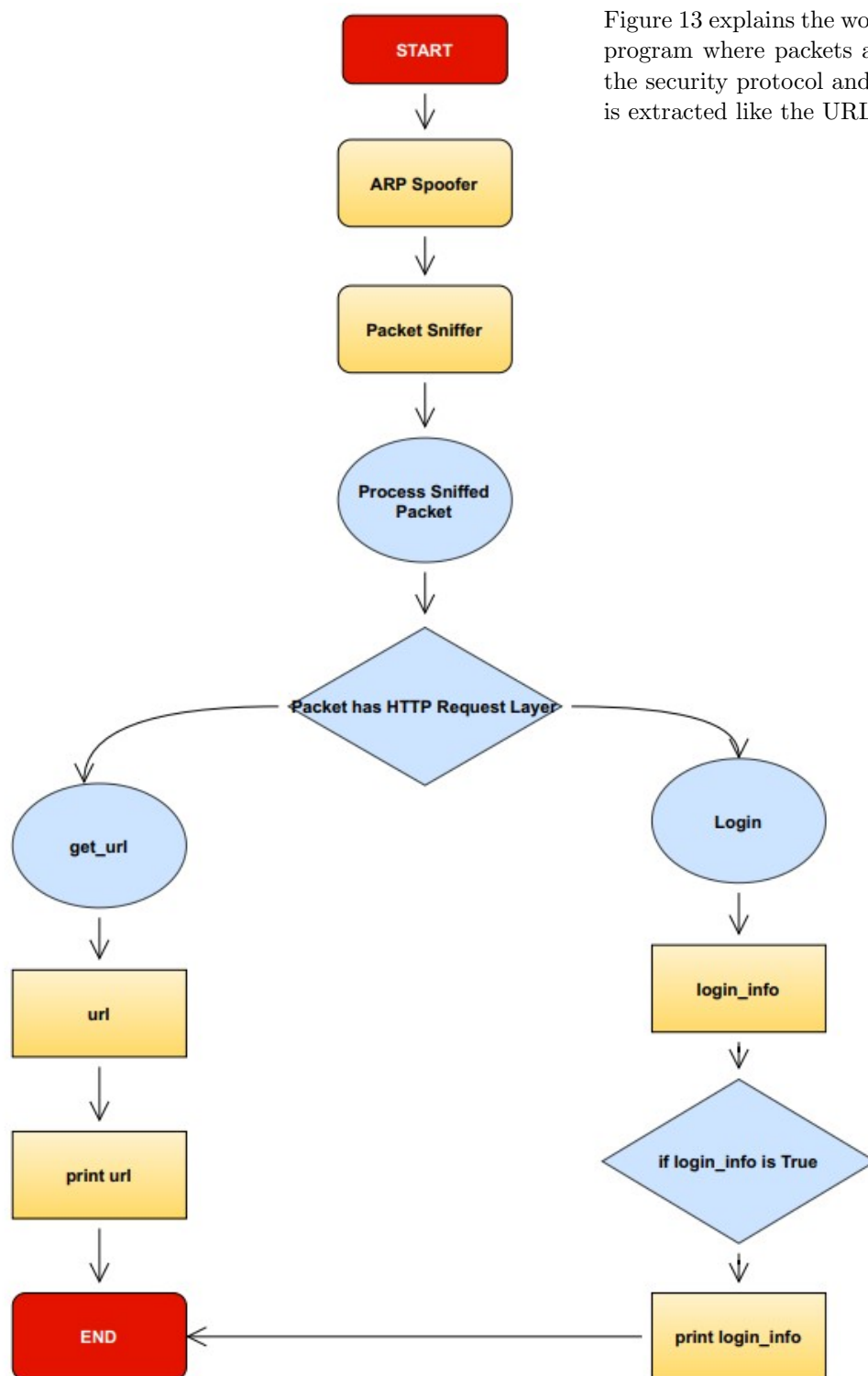


Figure 13 explains the working of a packet sniffer attack program where packets are captured and analysed for the security protocol and if vulnerable the information is extracted like the URL visited and the login info.

Fig. 13 Data flow diagram for Packet Sniffer

## 8.1 Algorithm for Packet Sniffer

- 1) Creating a new project in PyCharm
- 2) Shebang line - `#!/usr/bin/env python`
- 3) Import the scapy module
- 4) From scapy module import the http library for filtering http packets
- 5) Define a sniff function with interface (eth0) as the parameter for sniffing :
- 6) use scapy.sniff function with iface argument for data capturing, and a call back function on every captured on each packet.
- 8) Define a function to capture victim's IP address and returning the URL visited :
- 9) return the information extracted by concatenating packets' host and path
- 10) Define a login function with the interface as the parameter :
- 11) check if the packets captured contains a TCP layer, if yes then:
- 12) store the packets with raw TCP/IP layer and load field of all packets
- 13) make a list of important information to sniff from the packets like "username", "password", "login" etc.
- 15) now run a loop and find the required information on the list from every packets captured
- 17) now return the list of information gathered from the packets
- 18) Define function with packet as the parameter to take a packet that was sniffed
- 19) if packets has a http layer
- 20) extract the url from the packet using url function defined above
- 21) print the extracted url
- 22) extract the login information from the packets
- 23) store the information in a variable
- 24) if variable contains the required information:
- 25) print the login information of the packets stored in the variable
- 26) Passing the interface (eth0 in this case) as the interface is connected to the network
- 27) END

A packet sniffer was then used to analyse this collected data and convert it into a human-readable format so it can be further examined by the intercepting entity. Now authors wanted to run this program against another computer on the same network.

In order to do so they need to be the man in the middle. They could do so using the program they wrote earlier for ARP spoofing. It redirected the flow of packets and after that they ran the packet sniffer program to capture all the packets including URLs, username and password entered by the user on the victim machine.

## 8.2 Result

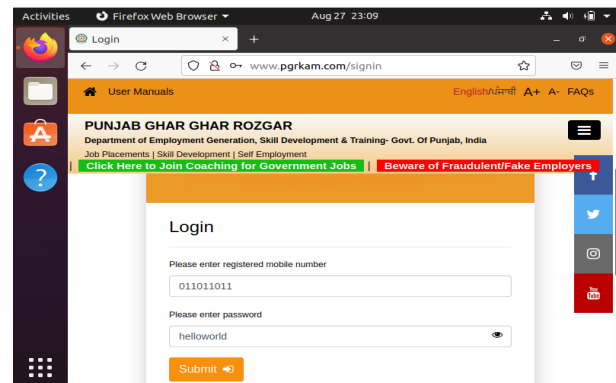


Fig. 14 Logging in on a machine on the same network

Figure 14 displays the working of the packet sniffer program where a URL login is demonstrated and Figure 15 displays the packet information along with login details captured and displayed on the terminal with the help of packet sniffer program. After running the ARP Spoofer and Packet Sniffer authors were getting all the information including URLs, username and password on the terminal entered on another machine on the same network.

Username – 011011011 & Password – helloworld

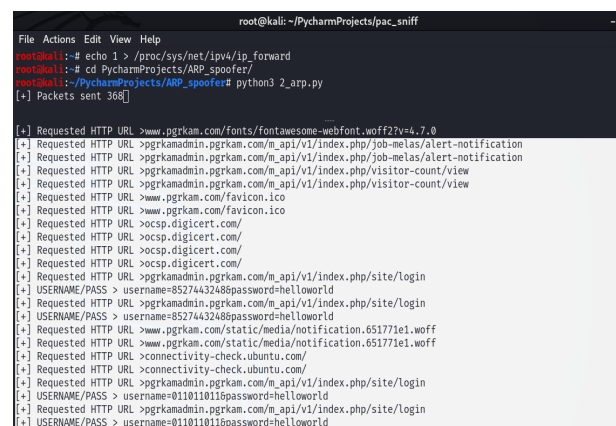


Fig. 15 Receiving packets on terminal window (Kali Linux)

## 9 File Interceptor

Till now authors wrote programs that were allowing them to redirect the flow of packets and allowing them to capture the packets. To achieve their main objective of Malware analysis and to build Malware that they could use against other machines, they had to think of an algorithm that could modify the packets as it flows through the attacker machine giving them an advantage of spoofing DNS requests. In this program, data was modified in the HTTP layer as it is the layer through which data is sent. Through this program, the authors tried to achieve objectives like: Replacing download requests & Modifying requests and responses These objectives meant that they could intercept downloads that the target was downloading. In simpler terms, it was replacing the original download file with a backdoor, malware, keylogger or any other file. Figure 16 on the right, demonstrates the working of the file interceptor program aims to replace the download request of the victims machine and modifying the website requests and responses.

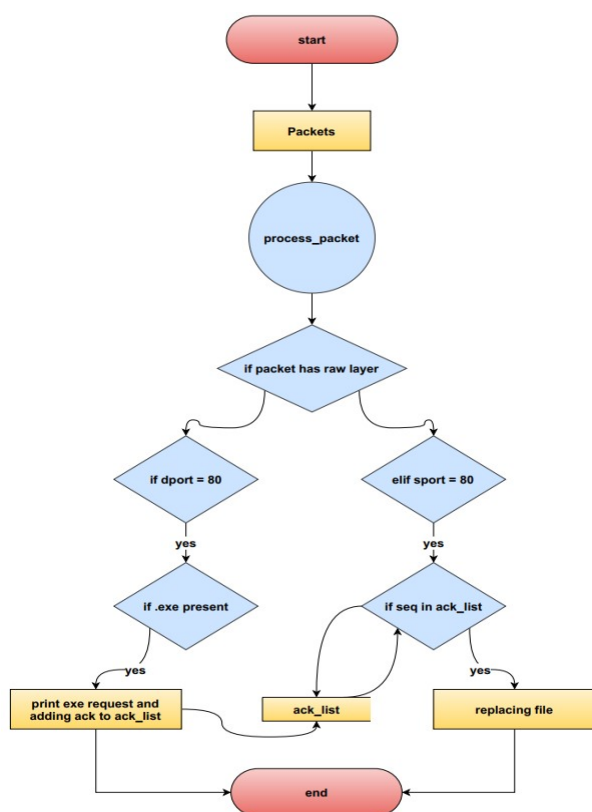


Fig. 16 Receiving packets on terminal window (Kali Linux)

### 9.1 Algorithm for File Interceptor

- 1) Creating a new project in PyCharm
- 2) Shebang line - `#!/usr/bin/env python`
- 3) Importing netfilterqueue & scapy module
- 4) Declaring an empty array or list (ack\_list)
- 5) Defining a function(process\_packet) with only 1 argument - packet
- 6) **process\_packet(packet):**
- 7)     converting packet to scapy packet & getting payload
- 8)     If packet has raw layer
- 9)     If dport == 80
- 10)         If ".exe" is present in load field under raw layer
- 11)             print "exe request"
- 12)             add ack field's data to ack\_list
- 13)     Else If sport == 80
- 14)         If seq field is there in ack\_list
- 15)             removing seq field and print "replacing file"
- 16)             replacing the file by - modifying the load field using 3XX redirection
- 17)             removing len field and - chksum field under IP layer
- 18)             removing chksum field - under TCP layer
- 19)             setting payload
- 20)     packet.accept()
- 21) Creating a variable queue using netfilterqueue module
- 22) Configuring the queue with a bind calling process\_packet as a call back function
- 23) Running queue
- 24) END

## 9.2 Result

```

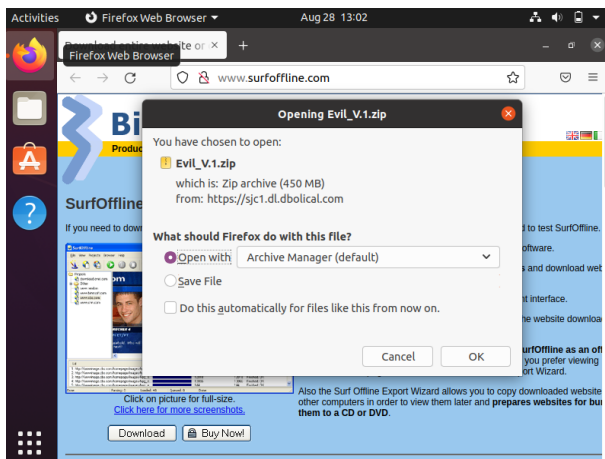
root@kali: ~/PycharmProjects/file_interceptor 99x12
^CTraceback (most recent call last):
  File "3_interceptor.py", line 35, in <module>
    queue.run()
KeyboardInterrupt
root@kali:~/PycharmProjects/file_interceptor# python 3_interceptor.py
/usr/local/lib/python2.7/dist-packages/scapy/config.py:411: CryptographyDeprecationWarning: Python
2 is no longer supported by the Python core team. Support for it is now deprecated in cryptography,
and will be removed in the next release.
  import cryptography
[+] exe request
[+] Replacing file

root@kali:~/PycharmProjects/ARP_spoofing 99x12
root@kali:~/PycharmProjects/ARP_spoofing# python3 2_arp.py
[+] Packets sent 20^CTraceback (most recent call last):
  File ~/root/PycharmProjects/ARP_spoofing/2_arp.py", line 27, in <module>
    time.sleep(2)
KeyboardInterrupt
root@kali:~/PycharmProjects/ARP_spoofing# iptables --flush
root@kali:~/PycharmProjects/ARP_spoofing# iptables -I INPUT -j NFQUEUE --queue-num 0
root@kali:~/PycharmProjects/ARP_spoofing# iptables --flush
root@kali:~/PycharmProjects/ARP_spoofing# iptables -I FORWARD -j NFQUEUE --queue-num 0
root@kali:~/PycharmProjects/ARP_spoofing# python3 2_arp.py
[+] Packets sent 116]

```

**Fig. 17** Terminal window of Kali Linux (Attacker Machine)

Figure 17 displays the terminal commands of the packet sniffer program in working using iptables commands. Instead of getting a file from [www.suroffline.com](http://www.suroffline.com) victim was receiving an evil file named as Evil\_V.1.zip. The authors were able to intercept the download request sent by the victim and were able to replace it with a totally different file. Figure 18 demonstrates the working of the packet sniffer program where the victim's ubuntu machine is redirected to download an infected file when he was on [suroffline](http://suroffline.com).



**Fig. 18** Intercepting download request at Victim's machine (Ubuntu)

Figure 19 on the right explains how keystrokes are registered in a Keylogger with the help of a Data flow diagram(DFD). On the other hand, Figure 20 explains how the collected data is sent to the attacker by introducing send mail method in the program.

## 10 Malware: Keylogger

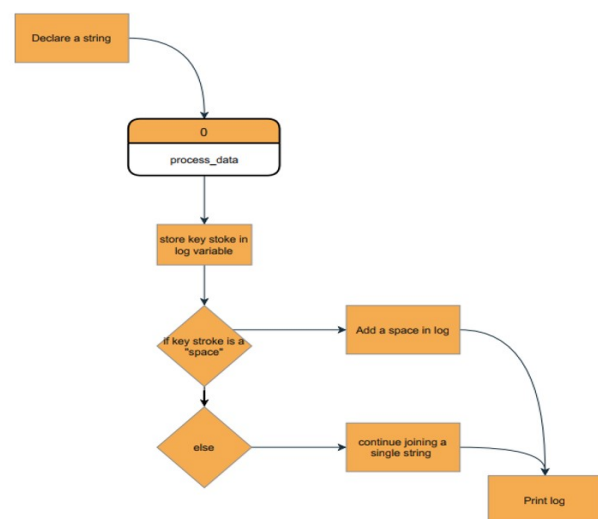
In this section, smaller bits of each program was used like MAC changer program to ensure that we were working anonymously, a Network Scanner to scan the networks, ARP spoofing to simulate Man in the Middle Attack (MitM), Packet sniffer to capture the packets, DNS Spoofing and File Interceptor. All these concepts and some more advanced concepts of Python like Threading & Object-Oriented approach were used to build a Keylogger.

Keylogger is a type of monitoring software used to record every keystroke pressed at the physical keyboard. Thus, the key log is regularly transmitted over the internet to a server, but here the authors have changed the functionality of this keylogger. The data was sent to an email ID after a fixed time interval. The way a keylogger works is by intercepting the keystrokes before they are sent to the respective application. This software is capable of camouflaging as a system application and running in the background.

Mainly there are two types of keyloggers-

1. Local keyloggers - These types of keyloggers will save the data on a local machine.
2. Remote Keyloggers - Remote keylogger will send reports to a server, in our case it is an email ID.

Hardware keyloggers are also there in the form of a plug-in device that needs to be plugged into a port to intercept the signals. In this program pynput library was used, this library allowed the authors to control and monitor input devices like keystrokes and mouse clicks.



**Fig. 19** Keylogger Data flow diagram (Registering keystrokes)



**Fig. 20** Keylogger Data flow diagram

Keylogger is not always malicious, it has various use cases in different fields like:

1. Testing
2. UX improvement
3. Troubleshooting
4. Monitoring
5. Used by Intelligence Agencies

The way a keylogger works is by intercepting the keystrokes before they are sent to the respective application. This software is capable of camouflaging as a system application and running in the background. Threading was used to build the report function because report() function is needed to execute after a certain period of time with the process\_data() function. Threading came in handy here as it allowed the report() function to run in the background without interrupting the function that is handling all the data processing. Threading splits the whole program into two timelines one where all the data processing is going on and other where a timer is triggered every time after a specific

interval of time.

Classes had been used in this final version of the Keylogger program because classes make the code more readable and usable, allow separate implementation, make it easier to maintain.

### 10.1 Algorithm for Keylogger

- 1) Creating a new project in PyCharm
- 2) Shebang line - `#!/usr/bin/env python`
- 3) Importing smtp & pynput library and selecting keyboard specifically
- 4) Importing threading module
- 5) KeyLog class is created
- 6) **class Keylog:**
- 7) constructor with parameters(interval,email,pass):
- 8) declaring attributes
- 9) append function with a parameter(string):
- 10) storing the string in log variable
- 11) process\_data function with a -
- 12) parameter(key) :
- 13) handling exceptions using try & -
- 14) except statements
- 15) If keystroke is space
- 16) current\_key = " "
- 17) Else
- 18) current\_key = str(key)
- 19) report function :
- 20) calling send method with 3 -
- 21) arguments(email, password, log)
- 22) timer class has been used with -
- 23) threading module
- 24) starting timer and report() is called
- 25) after every 120 seconds
- 26) send function (email,password,message):
- 27) starting TLS with SMTP
- 28) passing login info using .login method
- 29) sending email with the data using -
- 30) .sendmail
- 31) quitting server
- 32) start method for starting the program:
- 33) calling process\_data & report
- 34) allowing the execution of the report()
- 35) & process\_data()
- 36) END

10.2 Result

Running the program



Fig. 21 Terminal window (Kali Linux), running keylogger

Figure 21 shows the execution of the Keylogger program and Figure 22 illustrates the case where a user is typing some text on the machine which has been infected –



Fig. 22 Typing some text

Figure 23 confirms that Keylogger execution was successful and shows the receipt of data from the Keylogger program. Receiving data through email -

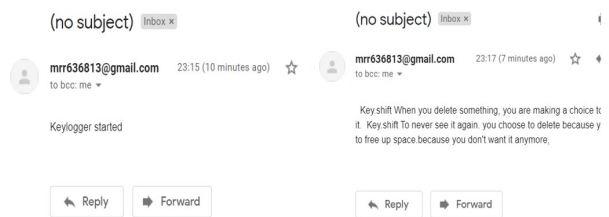


Fig. 23 Typing some text

Figure 24 demonstrates the case where a user is logging in on a website -

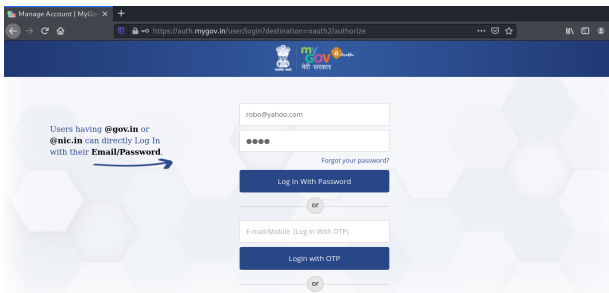


Fig. 24 Logging in on a website

Figure 25 confirms that Keylogger execution was successful and shows the receipt of data from the Keylogger program. Receiving data through email -

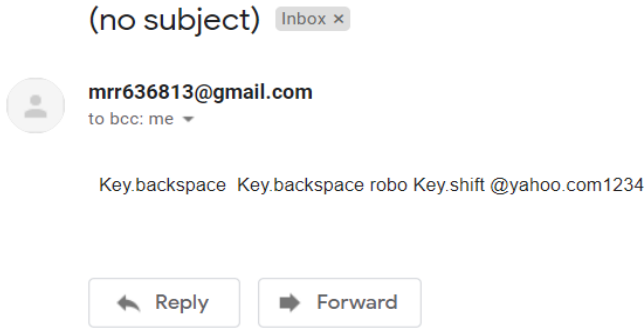


Fig. 25 Receiving login credentials

11 Malware: Packaging

Now taking Keylogging, ARP spoofing, Packet sniffing and File intercepting to one way forward to analyze how in real-world these attacks could violate the public's privacy. Exploiting the vulnerability, we now target other Linux based machines for the ARP poisoning, Packet-sniffing and packaging of a Keylogger. The attack was carried out using the following steps.

11.1 Preparing the attack

1. Packaging (Creating an executable file)

Figure 26 describes how the attack was prepared by creating a package containing an executable file disguised as a surffoffline program wherein reality it was a keylogger that sends every key pressed on the victim's machine to the attacker through email whenever the program was run by the victim.

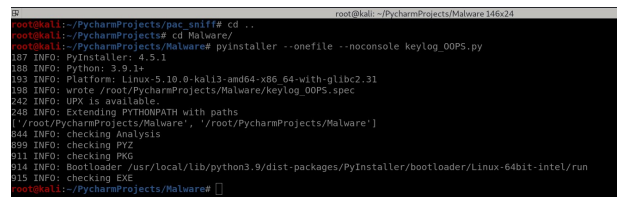
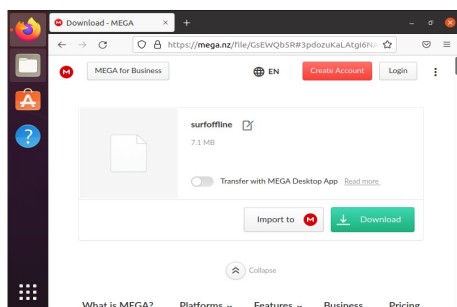


Fig. 26 Terminal window (Kali Linux)



## 2. Uploading the file to a cloud-based server

This file was uploaded to a cloud-based server (MEGA in this case) so that it could be accessed globally by anyone.



## 3. ARP Spoofing & Packet Sniffing

The following commands were carried out to start redirecting the packets through the hacker's machine and packet sniffing was started with the help of iptables commands for redirecting all Input and Output packets. Figure 27 shows the execution of ARP spoofer and Packet sniffer to allow the flow of packets.

```
root@kali:~/PycharmProjects/ARP_spoofing# echo 1 > /proc/sys/net/ipv4/ip_forward
root@kali:~/PycharmProjects/ARP_spoofing# iptables --flush
root@kali:~/PycharmProjects/ARP_spoofing# iptables -I INPUT -j NFQUEUE --queue-num 0
root@kali:~/PycharmProjects/ARP_spoofing# iptables -I FORWARD -j NFQUEUE --queue-num 0
root@kali:~/PycharmProjects/ARP_spoofing# python3 2_arp.py
[+] Packets sent 22
```

Fig. 27 Terminal window (Kali Linux)

## 4. Launching the File Interceptor

The packets were redirected to intercept the download request sent by the victim and to replace it with a totally different file. Figure 28 shows the execution of File Interceptor.

```
root@kali:~/PycharmProjects/file_interceptor 80x11
root@kali:~/PycharmProjects/file_interceptor# python2 3 interceptor.py
/usr/local/lib/python2.7/dist-packages/scapy/config.py:411: CryptographyDeprecationWarning: Python 2 is no longer supported by the Python core team. Support for it is now deprecated in cryptography, and will be removed in the next release.
import cryptography
```

Fig. 28 Terminal window (Kali Linux)

## 11.2 On Victim's Machine

### 1. The Website

Figure 29 displays the website which was launched by the victim to download an application. A download request was sent by the victim to the server and then the same request got intercepted by the file interceptor.



Fig. 29 Website (Ubuntu)

### 2. Download request redirected to the uploaded MEGA file

The download request on the website was redirected to the uploaded executable file on the MEGA cloud from where the user unknowingly downloads the file to its system and uses the terminal to install the program.

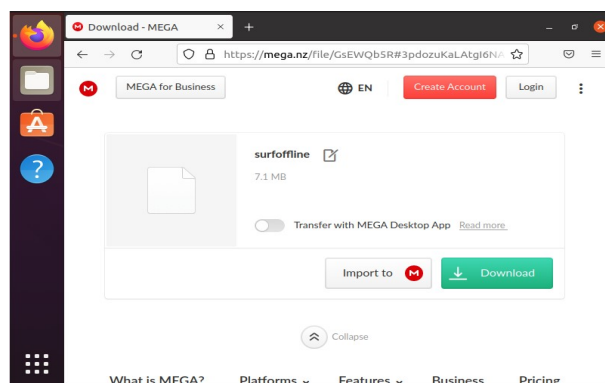
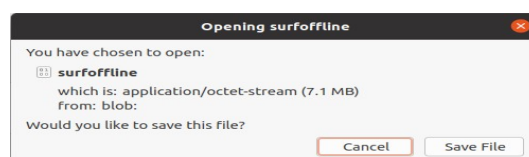
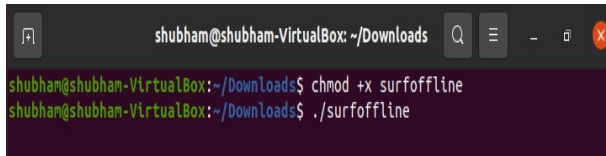


Fig. 30 MEGA cloud





### 3. User running the executable file



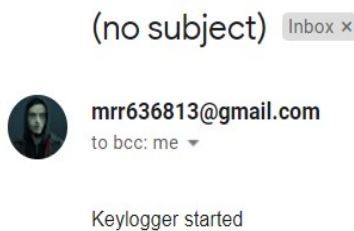
**Fig. 31** Terminal window (Ubuntu)

In Figure 31, creation of an executable file is shown. It is a keylogger disguised as an executable.

## 11.3 On Attacker's Machine

### 1. Acknowledgement via email

The attacker was informed immediately when the infected file was run by the victim and keylogger got activated. Figure 32 shows the receipt of an acknowledgement email.



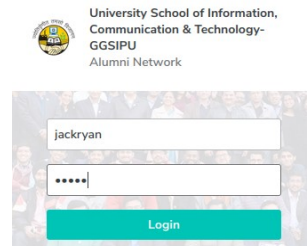
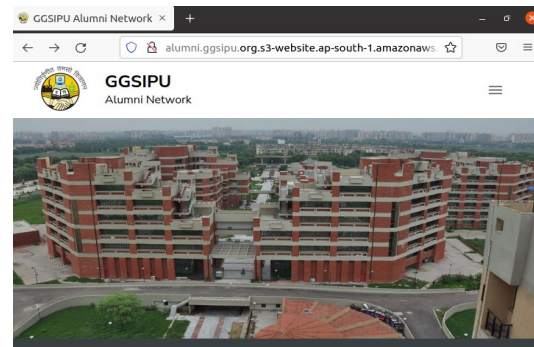
**Fig. 32** Acknowledgement via email

### 2. Launching Packet sniffer

The packet sniffer was started and it redirected information about all the websites browsed by the victim in real time.

### 3. Victim logging in on a website

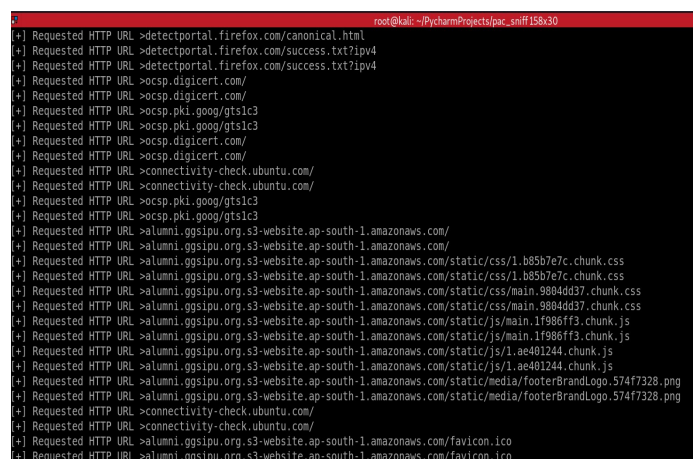
During the user web activity, the victim tried to log in on a website unprotected by the security protocol, (like HTTP instead of HTTPS). All the information about the website including the links were transferred to the attacker's machine along with all the keys pressed by the user throughout the session. Figure 33 describes the user's web activity like URL visited, logging in and sending various requests.



**Fig. 33** Logging In

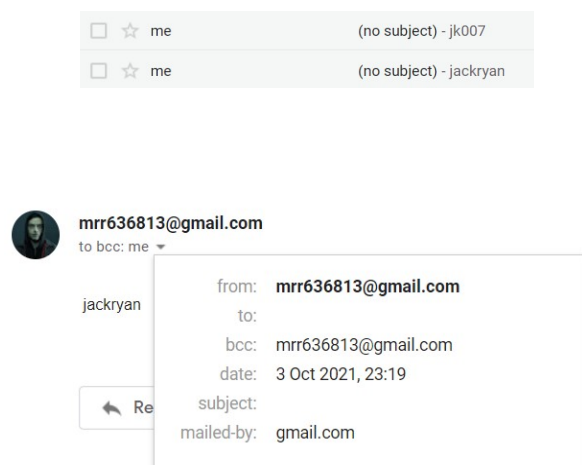
### 4. Receiving data

Unaware of any of the activities going behind, the victim may enter his login credentials on the website which are captured through the keylogger and sent to the attacker to the programmed mail. Now the attacker can go through the victim's packets to search for the website link where they have entered their details with the help of the packet sniffer and analyse the data sent by the keylogger to trace back the keys pressed during the session. After tracing the website and the keys pressed by the victim, an attacker can easily penetrate the victim's account and exploit the personal information.



**Fig. 34** Receiving packets

Figure 34 shows the data received from the packet sniffer like all the URL visited, while Figure 35 displays the information that the user has typed. This information was sent via email from the programmed keylogger automatically after every 30 seconds. In the above case, login information was received.



**Fig. 35** Email received from the Keylogger containing login credentials

## 12 Protection From Attacks

### 12.1 For ARP Spoofing

#### 1. Use Spoofing Identifiers

A few applications help to identify attacks, particularly for ARP spoofing. Consider applications like ARP Watch, NetCut or ARP Screen for protection against spoofing. These tools can assess and affirm genuine information before it reaches an objective machine which can altogether bring down the achievement of spoofing attacks.

#### 2. Validate Clients and Frameworks

On the off chance that gadgets on a network can utilize just IP addresses for access, IP caricaturing can be done for intrusion. The connection between devices should be validated by singular clients or applications, or by utilizing legitimate frameworks.

### 3. Use Packet Shifting

Packet shifting dissects the IP bundles and squares those with clashing source data. Since the packets containing malware will come from outside the network, this is a decent method to dispose of the futile IP bundles. Since aggressors have created strategies for dodging basic bundle channels, most packet channel frameworks offer a DPI (Deep Packet Inspection). DPI permits categorizing the rules dependent on both the header and the network bundles, shifting through numerous sorts of IP caricaturing attacks.

### 4. Encode and Verify

Security specialists have fostered few secure ways, including Transport Layer Security (TLS) (used by HTTPS and FTPS), Web System Security (IP-Security), and Secure Shell (SSH) to access the application or gadget while interfacing which encodes the information on the way to decrease the probability of getting attacked.

### 12.2 For Packet Sniffing and DNS Spoofing

#### 1. Protect the messages with a VPN

A viable method to defend from sniffing is to encode all the messages before sharing them with a virtual private organization (VPN). Encryption improves security and makes it hard for programmers to decode the packet information.

#### 2. Network examining

Directors of leading organizations get their data examined and observe their network with the assistance of gadget evaluation. Consequently, this is one of the significant procedures to upgrade the organization's environment and recognize the presence of ransomware. No organization is protected from unapproved interruptions by cybercriminals. Even with the most complex strategies and measures set up, assailants can take advantage and remove information from the organization. Subsequently, the association should have a certified group of moral programmers and managers who can infiltrate the frameworks and carry out these checks intermittently.

#### 3. Keep a distance from unstable networks

We frequently read articles on bank robbery where hackers took the client's banking data to roll out sensitive information.

It can happen if a client connects their gadget to an unstable Wi-Fi connection. Also, assailants utilize such weak networks to introduce packet sniffing to peruse the information communicated over that network. Another way a hacker can do this is by making counterfeit free open Wi-Fi. So, it is recommended to stay away from these free unstable public Wi-Fi connections.

### 12.3 For File Interceptor and Malware

#### 1. Key Encryption

Key encryption disorientates the keys that are entered onto the console to keep the key loggers from identifying the specific movements. They hide the keystrokes as soon as they arrive at an application. This directly blocks the key loggers from decoding the information.

#### 2. Malware Removal Programs

Malware removal programs shield from any malware assortment like key loggers, ransomware, rootkit, and Trojan. It filters the records that enter into a PC, hence identifying and blocking the hack. Also, it routinely checks the PC for malware to keep the hard drive malware-free. It likewise shields the console from direct access. Hence, it does not let any pernicious programmer acquire direct entry to it.

#### 3. No Downloads from Unrecognized Sites

Declining to download from unrecognized websites keeps the key loggers from tainting the PC. These websites are usually filled with malware. Though they are free, yet they could be dangerous for the PC. One may accidentally introduce a key logger camouflaged in a PC application.

#### 4. Regular PC Updates

Updating on a regular basis fixes patches and weaknesses of the PC. Moreover, it resolves the current issues of the PC that programmers can take advantage of. It additionally puts in new provisions on the application making them more proficient.

## 13 Hardware and Software Requirements

In order to achieve the objective, the following hardware and software were used -

NAT (Network Address Translation) network	Virtual Box
Attacking machine - Kali Linux	Kali Linux
Victim machines - Ubuntu and Windows	PyCharm Community
OSI and TCP/IP Modules	Scapy Module

## 14 Conclusion and Future Work

The main objective was to simulate and study various attacks in the cyber security field and the authors were successful in simulating these attacks and programs in the given time. During this period, they got to know a lot about the importance of ethical hacking. The authors got a better knowledge and understanding of how computer systems work. In this paper, they explained the relationship between hacking and programming and how both complement each other. The interaction was made with the Kali Linux terminal and some of the modules such as subprocess, SMTP and functions like regex which allowed the execution of system commands were also taken into account.

The programs were successful in discovering devices connected to a network and were able to display their IP addresses as well as their MAC Addresses. Algorithms were designed that we're able to execute codes that could send information about the connected user profiles directly through email. In addition to that, the authors were also able to design and implement malware on their own. From modelling applications like MAC changer Network Scanner to simulating Man in the Middle Attacks, they designed and simulated all these programs by designing their own testing lab. The authors were successful in building cross-platform programs that work on almost every OS. A deep study was carried out on what really happens behind the scene when carrying out attacks and when our systems are being compromised. All this work gave a clear understanding of how different network layers such as ARP, DNS, HTTP work, how HTTP requests are analysed and modified and how interception modification of network packets is done.

In the future, the authors of this paper will be working more deeply on all these concepts and many other concepts as well which they were not able to study properly due to lack of skills and experience and in some cases unavailability of proper hardware / software. The authors will continue to keep honing their skills in order to make better and real-world oriented projects.

## 15 Compliance of Ethical Standards

**Conflict of interest** The author(s) specify that, to the best of the knowledge and belief, no conflict of interest exist

**Ethical statement** Hereby, we consciously assure that for the manuscript *Simulating Cyber Attacks and Designing Malware using Python* the following is fulfilled:

1. This material is the authors' own original work, which has not been previously published elsewhere.
2. The paper has not been currently considered for publication elsewhere.
3. The paper reflects the authors' own research and analysis in a truthful and complete manner.
4. The paper properly credits the meaningful contributions of co-authors and co-researchers.
5. The results are appropriately placed in the context of prior and existing research.
6. All sources used are properly disclosed (correct citation). Literal copying of text has been indicated by using quotation marks and giving proper reference.
7. All authors have been personally and actively involved in substantial work leading to the paper, and will take public responsibility for its content.
8. The paper does not contain any studies with human participants or animals performed by any of the authors.

## 16 References

1. Dewar, Robert S. "The "trptych of cyber security": A classification of active cyber defence." In 2014 6th International Conference On Cyber Conflict (CyCon 2014), pp. 7-21. IEEE, 2014.
2. Sreenivas, R. Sreeram, and R. Anitha. "Detecting keyloggers based on traffic analysis with periodic behaviour." *Network Security* 2011, no. 7 (2011): 14-19.
3. Ladakis, Evangelos, Lazaros Koromilas, Giorgos Vasiladis, Michalis Polychronakis, and Sotiris Ioannidis. "You can type, but you can't hide: A stealthy GPU-based keylogger." In *Proceedings of the 6th European Workshop on System Security (EuroSec)*. 2013.
4. Ortolani, Stefano, Cristiano Giuffrida, and Bruno Crispo. "Bait your hook: a novel detection technique for keyloggers." In *International workshop on recent advances in intrusion detection*, pp. 198-217. Springer, Berlin, Heidelberg, 2010.
5. Bhardwaj, Akashdeep, and Sam Goundar. "Keyloggers: silent cyber security weapons." *Network Security* 2020, no. 2 (2020): 14-19.
6. Albabtain, Yazeed, and Baijian Yang. "The process of reverse engineering GPU malware and provide protection to GPUs." In *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pp. 1669-1673. IEEE, 2018.
7. Moses, Samuel, Jon Mercado, Allie Larson, and Dale Rowe. "Touch interface and keylogging malware." In *2015 11th international conference on innovations in information technology (IIT)*, pp. 86-91. IEEE, 2015.
8. Espoir K. Kabanga and Chang Hoon Kim. "Malware Images Classification Using Convolutional Neural Network". *Journal of Computer and Communications*, 6, 153-158, 2018.
9. Andreas Moser, Christopher Kruegel, and Engin Kirda. "Exploring Multiple Execution Paths for Malware Analysis".
10. Babu, P. & Bhaskari, Lalitha & CH.Satyanarayana,. (2011). A Comprehensive Analysis of Spoofing. *International Journal of Advanced Computer Sciences and Applications*. 10.14569/IJACSA.2010.010623.
11. Spangler, R. Packet Sniffing on Layer 2 Switched Local Area Networks. December 2003.
12. Blagoj Nenovski and Pece Mitrevski. "Real-World ARP Attacks and Packet Sniffing, Detection and Prevention on Windows and Android Devices". *The 12th International Conference for Informatics and Information Technology (CIIT 2015)*
13. "What Is Packet Sniffing? Definition and Details." Paessler.com. Paessler. 2021. <https://www.paessler.com/it-explained/packet-sniffing>
14. "TryHackMe Introductory Networking." TryHackMe. 2018. <https://tryhackme.com/room/introtonetworking>
15. "Network Scanner: What Is It and How Does It Work?" TekTools. March 12, 2020. <https://www.tek-tools.com/network/network-scanning-tools>
16. "McAfee. 2019. "What Is Malware?" McAfee. November 26, 2019. <https://www.mcafee.com/en-us/antivirus/malware.html>

17. What Is Hacking? What You Need To Know About Hackers , Malwarebytes. 2021.  
<https://www.malwarebytes.com/hacker>
18. “Ethical Hacking, DNS Spoofing - Javatpoint.”  
[www.javatpoint.com](http://www.javatpoint.com) 2011.  
<https://www.javatpoint.com/dns-spoofing>
19. “What is ARP Spoofing, ARP cache poisoning attack explained, Imperva”. May 6, 2020,  
<https://www.imperva.com/learn/application-security/arp-spoofing>