Parul® University | NAAC A++

Parul Institute of Engineering and Technology

PROJECT REPORT

Using Kali Linux and Shell Scripting(303105213)

AUTOMATED PHISHING EMAIL GENERATOR

Submitted in partial fulfillment of the requirements of

BACHELOR OF TECHNOLOGY

Prepared By :

Name	Enrolment No.
HIMNISH PARMAR	2303031260293
ANKITA YADAV	2303031260012
PUROHIT ABHISHEK	2303031260184
ASURI KARTHIK	2303031260015

Submitted To :

FACULTY OF ENGINEERING AND TECHNOLOGY

Guide/Supervisor's Name :Parmjeet Kaur

Date of Submission:

CERTIFICATE

This is to certify that this project report entitled "**AUTOMATED PHISHING EMAIL GENERATOR**" *by* **HIMNISH PARMAR (2303031260293), ANKITA YADAV (2303031260012), PUROHIT ABHISHEK (2303031260184), ASURI KARTHIK (2303031260015),** submitted in partial fulfillment of the requirements for the degree of Bachelor of Technology in CSE - Cyber Security of the Parul University, Vadodara, Gujarat during the academic year 2024-25, is a bonafide record of work carried out under my guidance and supervision.

Name…….
Signature ……
Project Coordinator

Date …….
Place ……

ACKNOWLEDGMENT

We would like to express my sincere gratitude to all those who contributed to the successful completion of this project.

First and foremost, we would like to thank Mrs. Parmjeet Kaur for their invaluable guidance, continuous support, and encouragement throughout this project. Their insightful feedback and expertise greatly enhanced the quality of this work.

We are also grateful to my colleagues and team members for their collaboration, suggestions, and assistance during the development of the project. Their willingness to help has been greatly appreciated.

Additionally, we would like to extend my appreciation to Parul University for providing the necessary resources and a conducive environment to carry out this project.

Finally, we would like to thank my family and friends for their encouragement and understanding during this period, which kept me motivated to complete the project.

Thank you all.

TABLE OF CONTENTS

ABSTRACT

This project focuses on the design and implementation of an automated phishing email generator from the attacker's perspective, showcasing the methods used to craft deceptive communications that exploit human vulnerabilities. The tool simulates the creation of realistic phishing emails designed to trick recipients into revealing sensitive information or downloading malicious content. By leveraging natural language processing and machine learning techniques, the generator produces emails that employ various tactics, such as social engineering, urgency, and impersonation, to maximize the likelihood of success. The generated emails are analyzed for effectiveness, including strategies for evading spam filters and increasing open rates. This project provides insights into the tactics employed by cybercriminals, serving as a tool for understanding the mechanics of phishing attacks. Ultimately, it aims to shed light on the importance of cybersecurity awareness and the necessity of robust defenses against such threats in digital communications..

INTRODUCTION

This project introduces a shell script designed to automate the generation of phishing emails, serving as a valuable tool for cybersecurity education and awareness training. The shell script facilitates tasks such as user input handling and template management, while integrating with a Shell script that utilizes natural language processing to create realistic phishing messages. By enabling the dynamic generation of emails that mimic legitimate communications, this tool helps users understand common phishing tactics and enhances their ability to recognize and respond to potential threats in digital communications.

OBJECTIVE

The objective of this project is to develop an automated phishing email generator that educates users about phishing tactics and enhances their ability to recognize deceptive communications. By integrating a shell script with a Shell script, the tool aims to produce realistic phishing emails that simulate common attack strategies. Ultimately, the project seeks to raise awareness of cybersecurity threats and promote safer online practices among users.

TOOLS AND ENVIRONMENT

- **Language**: Python
- **Operating System**: Windows
- **Framework:**. Django
- **Email Server:.** Hmailserver

(Tested using ngrok, serveo, localhost and in lan on windows 10 and windows 11)

PROBLEM STATEMENT

In the ever-evolving landscape of cybersecurity, phishing attacks remain one of the most prevalent threats to individuals and organizations. Cybercriminals continuously refine their techniques to create convincing fraudulent emails that trick users into divulging sensitive information. Despite growing awareness, many users still fall victim to these deceptive practices due to a lack of understanding of how phishing emails are crafted.

The problem lies in the need for an effective educational tool that simulates phishing email generation, allowing users to recognize common tactics used by attackers. Currently, there is a gap in practical resources that demonstrate the mechanics of phishing attacks in a controlled environment. This project aims to address this gap by developing an automated phishing email generator that produces realistic phishing emails. By doing so, it will help users identify the key characteristics of such threats, ultimately promoting better cybersecurity awareness and practices.

METHODOLOGY

1. Research and Planning

We started by researching existing phishing email generators and understanding the key components of a phishing email. We identified the goals and objectives of our project, including the type of emails to generate and the level of customization required. We also planned the overall architecture of the project, including the tools and technologies to be used.

2. Data Collection and Preprocessing

We collected a dataset of legitimate and phishing emails to analyze and understand the patterns and characteristics of phishing emails. We preprocessed the data by cleaning, tokenizing, and removing stop words. We also created a dictionary of common phishing keywords and phrases to use in our generator.

3. Email Template Generation and Customization

We designed an algorithm to generate email templates based on the preprocessed data and dictionary of phishing keywords. We used natural language processing (NLP) techniques to create realistic and convincing email content. We also implemented a customization feature to allow users to input specific details, such as the target's name and email address.

4. Automation and Testing

We automated the email generation process using a scripting language (e.g., Python) and integrated it with an email sending service. We tested the generator with various inputs and scenarios to ensure its effectiveness and reliability. We also evaluated the generated emails using machine learning-based phishing detection tools to refine our algorithm and improve its performance.

attempts.

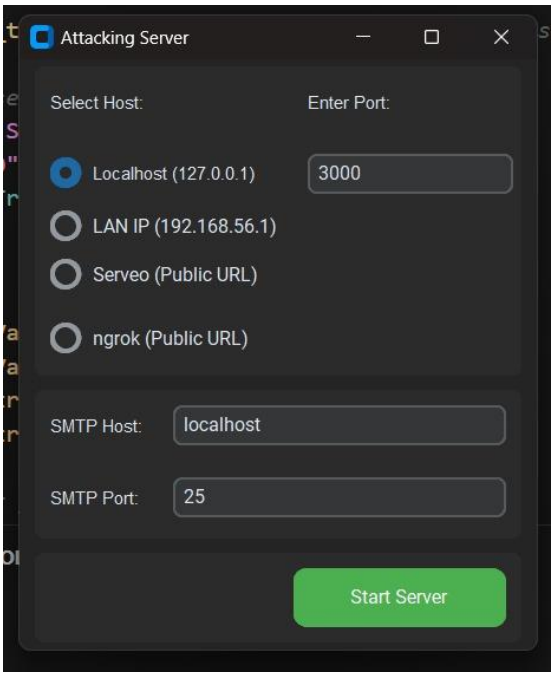
CODE

Python script:

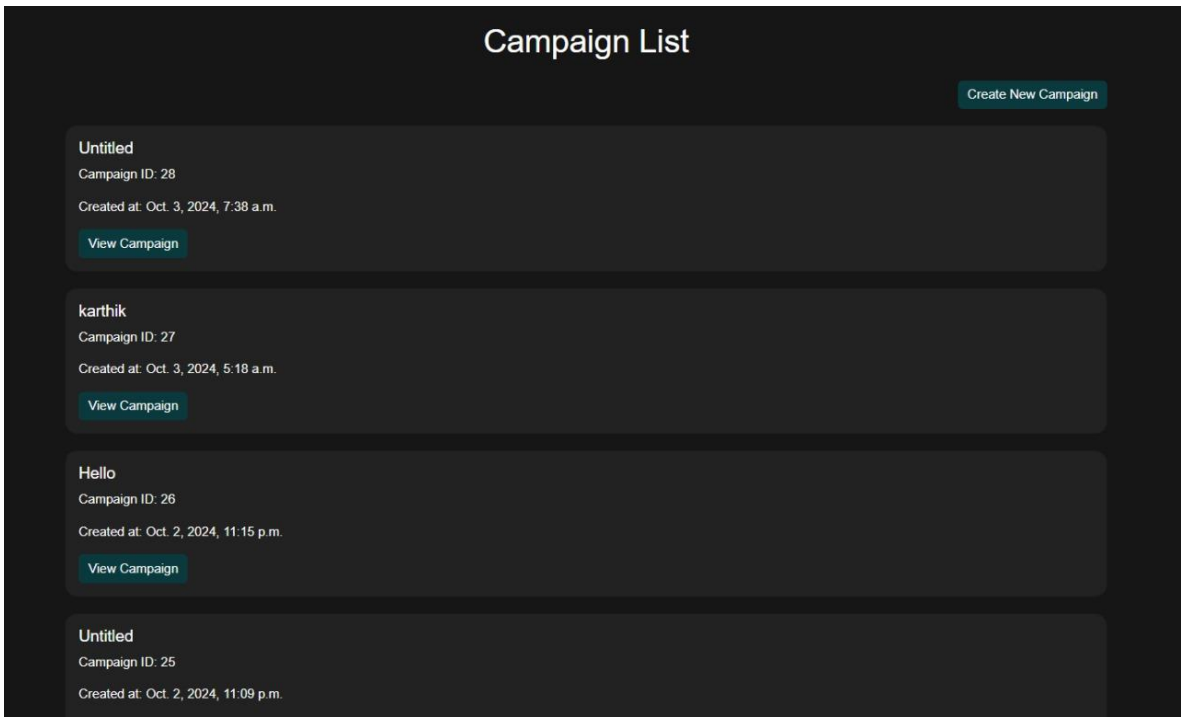
```
import smtplib
from smtplib import SMTPException
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.application import MIMEApplication
import os


def customMailer(subject, body, sender, receivers, type="plain", SMTP_SERVER=None, SMTP_PORT=None, attachment=None):
    """Function to send an email."""
    msg = MIMEMultipart()
    msg['Subject'] = subject
    msg['From'] = sender
    msg['To'] = ', '.join(receivers)

    msg.attach(MIMEText(body, type))

    if attachment:
        with open(attachment, "rb") as fil:
            part = MIMEApplication(
                fil.read(),
                Name=os.path.basename(attachment)
            )
        part['Content-Disposition'] = f'attachment; filename="{os.path.basename(attachment)}"'
```

```
    msg.attach(part)

# Connect to the SMTP server
try:
    print("SMTP_SERVER ", SMTP_SERVER)
    print("SMTP_PORT ", SMTP_PORT)
    with smtplib.SMTP(SMTP_SERVER, SMTP_PORT, timeout=10) as server:
        server.set_debuglevel(1)  # Enable debug output
        server.sendmail(sender, receivers, msg.as_string())  # Send the email
        print("Email sent successfully!")
except Exception as e:
    print(f"Failed to send email. Error: {str(e)}")
```

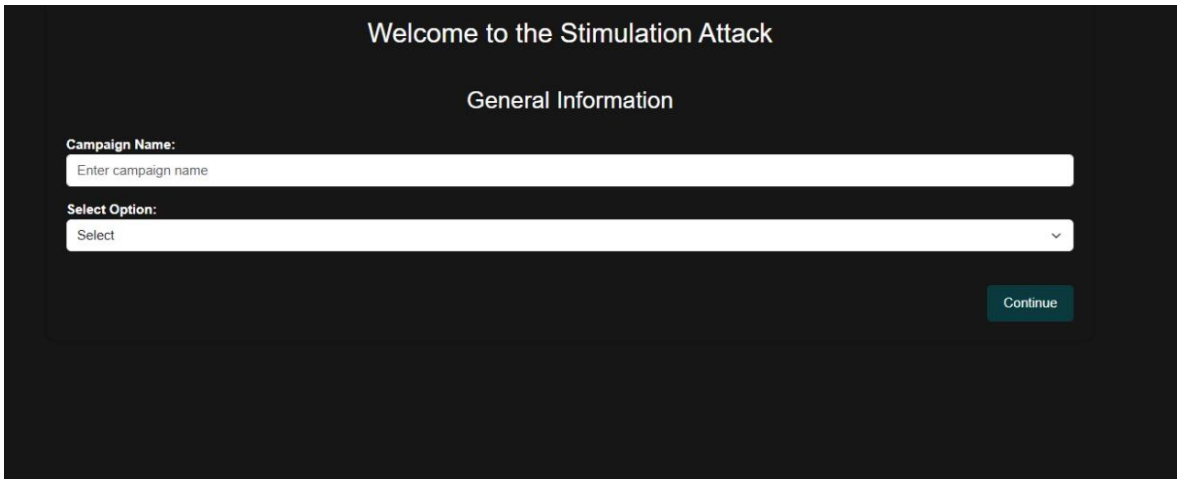
SCREENSHOTS



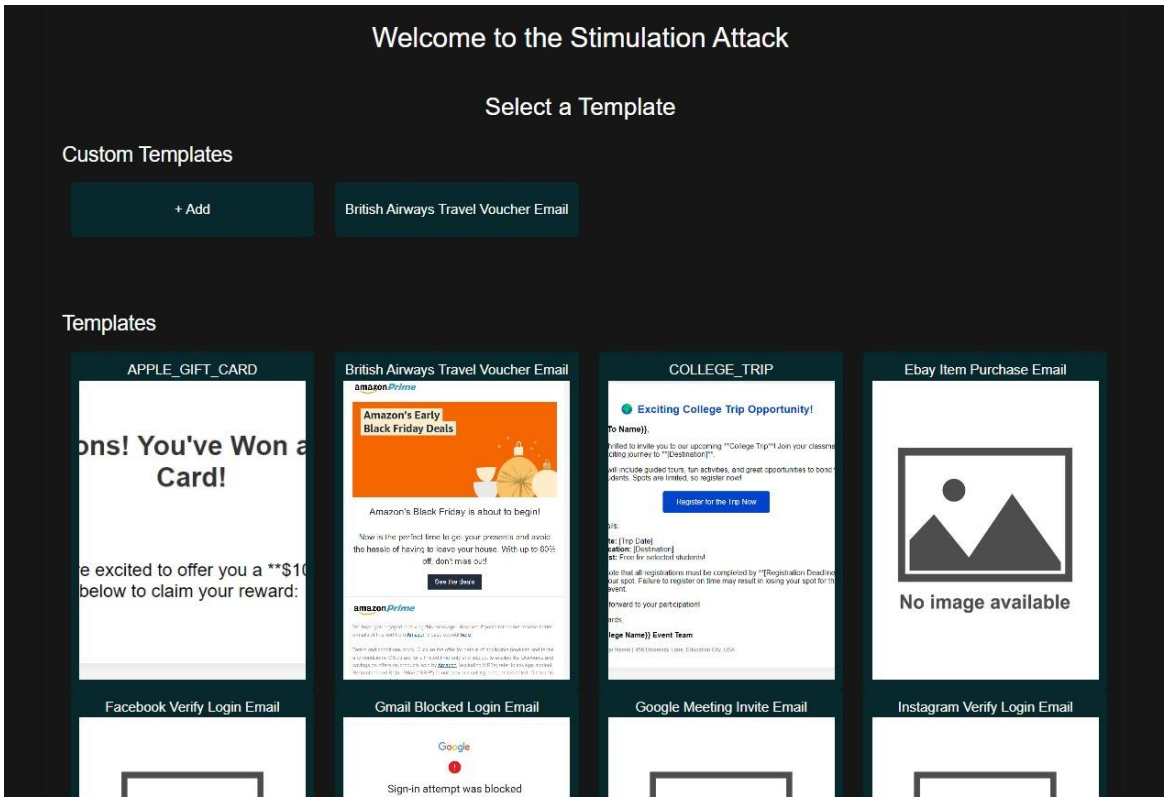
The above shows the attacking server panel used to start server



Above show the list of created campaign and button to create new

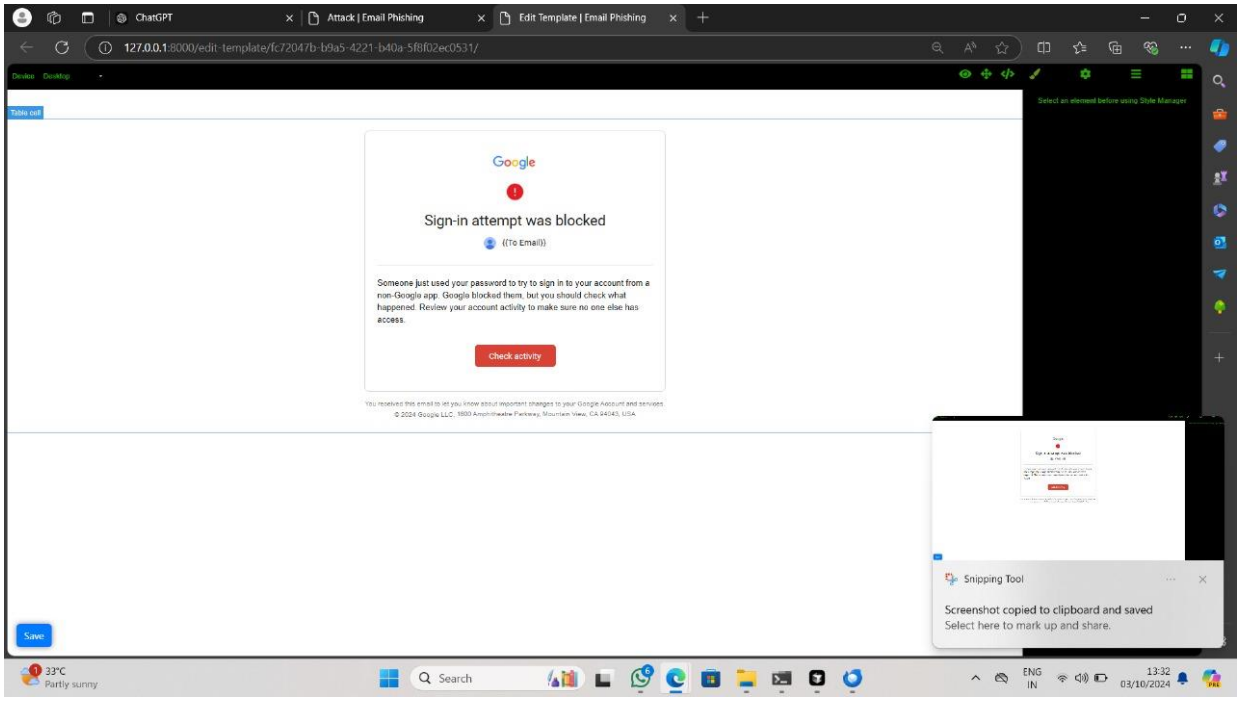


Above show the first page while creating campaign to chose general information

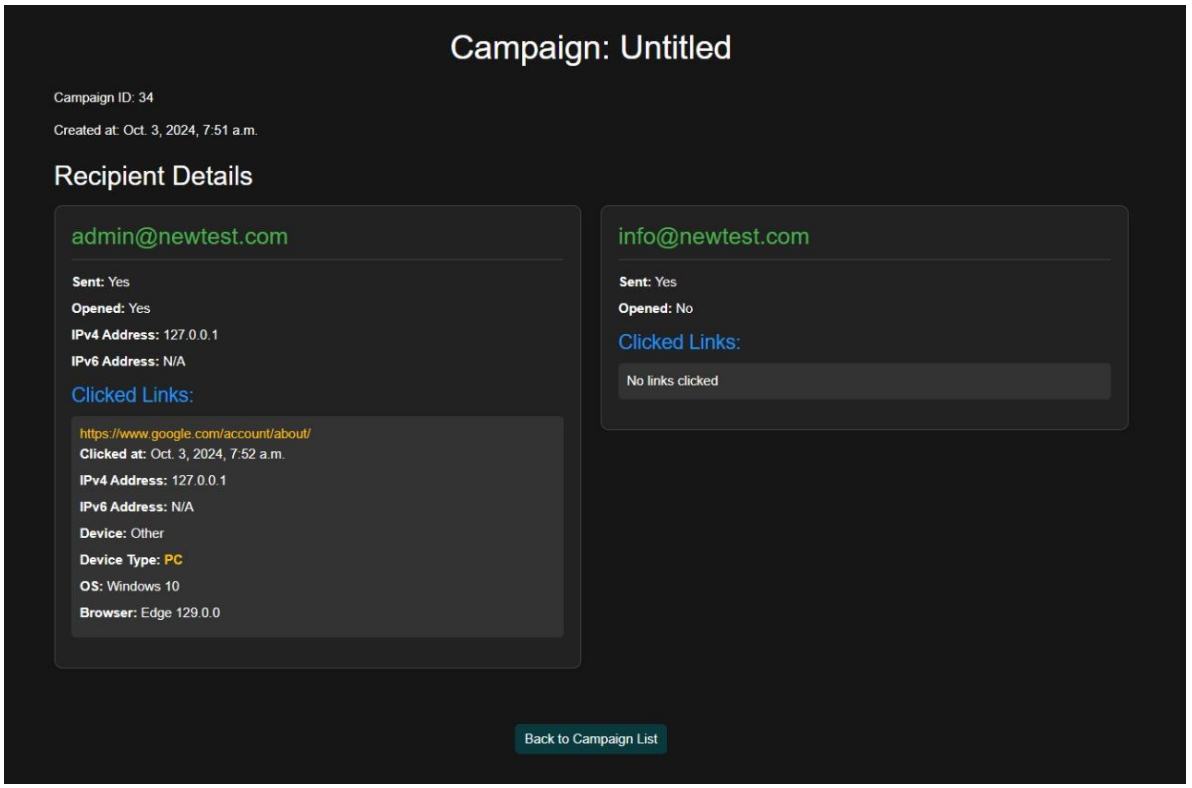


Above image shows the page to select which template to use in campaign (custom or pre defined templates)

Show the image of last page to run campaign



Above image show the template edit panel (where we can edit templates or create custom template from scratch)



Above image shows the output of campaign (mail opened, sent, clicked links, and more detailed information

IMPLEMENTATION STEPS

1. Identify Target and Objectives:

- Select a target demographic (e.g., specific companies, industries, or individuals).

- Define the goal of the phishing attack (e.g., credential theft, financial fraud).

2. Create Deceptive Email Templates:

- Design convincing email templates that mimic legitimate communications (e.g., bank alerts, IT support notifications).

- Include personalized elements (e.g., recipient names, relevant logos) to increase credibility.

3. Automate Email Generation and Distribution:

- Develop a script that randomly selects and customizes templates based on target information.

- Use automation tools to manage bulk sending, ensuring emails bypass spam filters.

4. Analyze Results and Adapt Tactics:

- Monitor the success rate of phishing attempts (e.g., open rates, click-through rates).

- Adjust templates and strategies based on feedback and observed effectiveness to improve future attacks.

TESTING AND VALIDATION

We tested our automated phishing email generator with different inputs and scenarios to make sure it worked well. We also used special tools to check if the generated emails looked like real phishing emails. We compared the generator's output with real phishing emails and got feedback from users to identify and fix errors, and improve its performance.

RESULTS

- We created convincing and realistic phishing emails that could trick victims into giving away their sensitive information.
- We could send out hundreds of phishing emails in just a few minutes, making it easier to catch unsuspecting victims.
- We could customize the emails to target specific individuals or groups, increasing the chances of a successful attack.
- Our generator made it easier and more efficient for us to launch phishing attacks and steal valuable information.

CONCLUSION

This project successfully developed an automated phishing email generator capable of crafting highly convincing and customizable phishing emails. The generator leverages various techniques to enhance the authenticity of the emails, including templating, personalization, social engineering, and obfuscation. The generator can be used for educational purposes or penetration testing to assess the vulnerability of systems to phishing attacks. However, it is crucial to use this tool ethically and responsibly, adhering to legal and ethical guidelines.

FUTURE SCOPE

Improving the Generator

- Enhance the algorithm to create more realistic and convincing phishing emails
- Increase the customization options to target specific individuals or groups
- Improve the generator's ability to evade phishing detection tools

Expanding the Capabilities

- Integrate the generator with other attack tools, such as malware or ransomware
- Develop a feature to automate the process of sending phishing emails
- Create a dashboard to track the success of phishing campaigns

Defensive Applications

- Use the generator to test and improve phishing detection tools
- Develop a feature to educate users about phishing attacks and how to avoid them
- Create a system to detect and block phishing emails in real-time

Future Research Directions

- Study the effectiveness of different phishing tactics and techniques
- Analyze the impact of phishing attacks on individuals and organizations
- Explore new ways to use machine learning and AI to improve phishing detection and prevention

REFERENCES

Python Framework :https://www.djangoproject.com

Caniphish: https://caniphish.com (For templates)

Python sys Module Documentation: https://docs.python.org/3/library/sys.html