Project Report

Web Scraping and Forensic Analysis of Phishing Websites: Identifying Patterns and Extracting Digital Evidence

---

Executive Summary

The Phishing Forensics project aims to analyze phishing websites to extract patterns, geographical hotspots, and high-risk behaviors, providing actionable insights to forensic investigators. Through a combination of web scraping, data analysis, and visualization techniques, the project delivers a detailed toolkit for understanding phishing attacks, empowering cybersecurity professionals with insights to mitigate risks.

Goal

The goal of this project was to analyze phishing websites comprehensively, extract actionable insights, and provide forensic investigators with a robust framework for identifying phishing patterns. Specific goals included:

- Identifying high-risk phishing websites.
- Analyzing metadata such as geographical origins, suspicious links, and form fields.
- Providing visual representations of phishing behaviors and trends.
- Creating a reusable forensic repository and interactive dashboards for effective reporting.

Scope

Inclusions

1. **Phishing Data Collection**:
    - Phishing website URLs were collected using the PhishTank API.
    - Invalid or inaccessible URLs were filtered out, and valid URLs were stored in valid_phishing_urls.csv.
2. **Automated Web Scraping**:
    - Metadata, suspicious links, external links, and form fields were extracted from validated phishing websites.
    - APIs like ipinfo.io were used to fetch geolocation details.
3. **Data Cleaning and Risk Scoring**:

- The dataset was cleaned and processed to handle anomalies.
- A risk scoring system classified websites into Low, Medium, and High risk levels.

4. **Analysis and Visualization**:
    - Key insights were extracted, and visualizations were created using libraries like matplotlib, seaborn, and Plotly.
5. **Forensic Repository**:
    - An SQLite database was implemented to store and query forensic evidence.
6. **Dashboard Development**:
    - An interactive dashboard was developed using Dash and Plotly to showcase results.

Exclusions

- Advanced phishing techniques like cloaking or encrypted attacks were not analyzed.
- Real-time phishing detection was not implemented.

Methodology

The methodology followed a systematic process involving the following steps:

1. **Phishing Data Collection**:
    - Collected phishing website URLs using the PhishTank API, stored in phishing_urls.csv.
    - Validated URLs and saved accessible ones in valid_phishing_urls.csv.
2. **Web Scraping**:
    - Used Python libraries like BeautifulSoup, requests and IP geolocation APIs to scrape 1000 phishing websites.
    - Extracted the following metadata:
        - **IP Address**, **Geolocation** (country, region, city, latitude, longitude).
        - **Website Metadata** (title, domain, status code, forms, external links, redirection chains).
        - **Visual Elements** (hidden elements, buttons, inline styles).
        - **Third-Party Resources** (external scripts and stylesheets).
    - Saved scraped data to scraped_data_1000_more_advanced.csv.

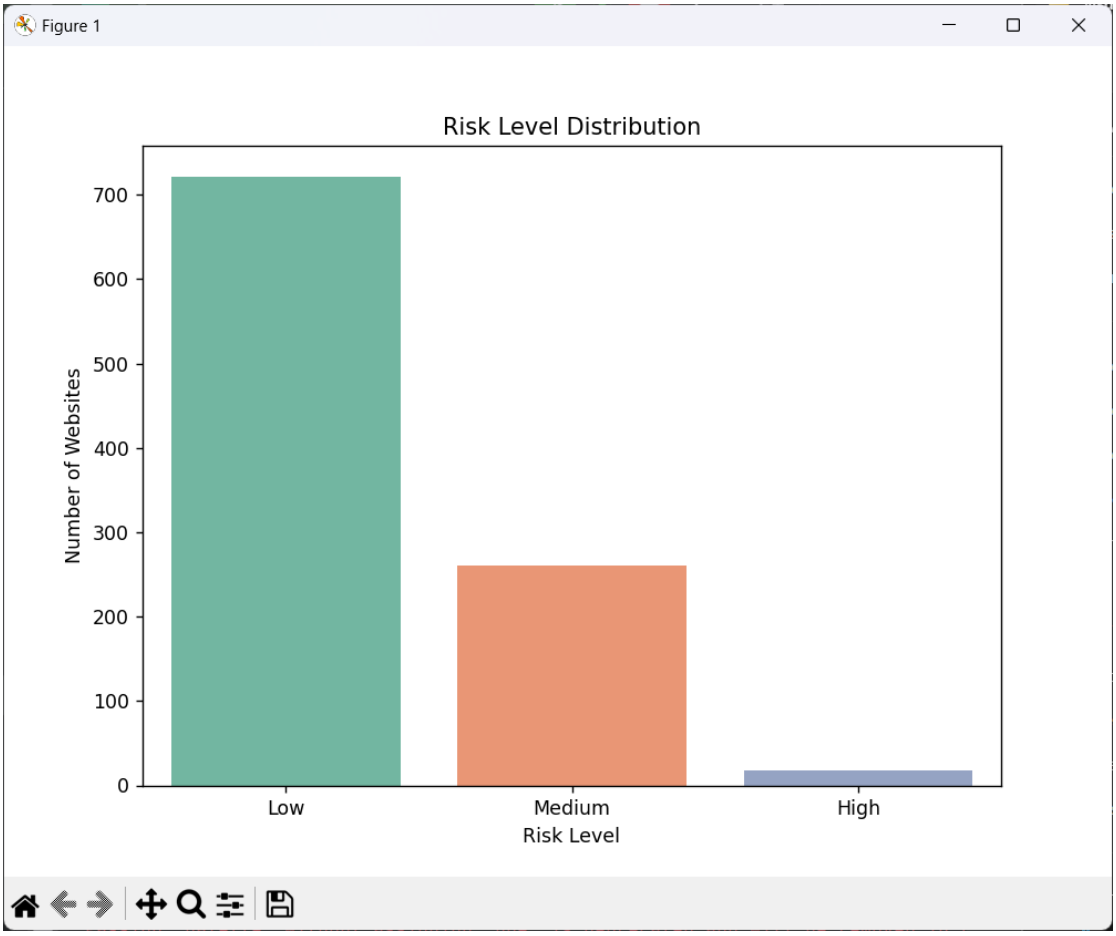
3. **Data Processing**:
    - Processed raw data to handle missing values and anomalies.
    - Data cleaning handled missing values, anomalies, and outliers, with results saved in Cleaned_data.csv.
    - Developed a **Risk Scoring System**:
        - Scores were assigned based on suspicious links, sensitive fields, and metadata anomalies.
        - Websites were classified into Low, Medium, and High-risk levels.
    - Cleaned and finalized dataset saved as Cleaned_data.csv.
4. **Analysis and Visualization**:
    - Insights were derived using Python tools like pandas and Counter.
    - Visualizations created with matplotlib, seaborn, and Plotly.
    - Conducted extensive analysis using pattern_analyzer.py:
        - Geographical analysis of phishing trends.
        - Identification of high-risk fields and suspicious links.
        - Clustering of phishing websites based on metadata.
    - Created visualizations such as bar charts, pie charts, and word clouds.
5. **Repository Setup**:
    - Implemented an SQLite database to store evidence for forensic queries.
    - Created tables for metadata, geolocation, and risk levels.
6. **Dashboard Development**:
    - Developed an interactive dashboard using Dash and Plotly to present results and insights effectively.

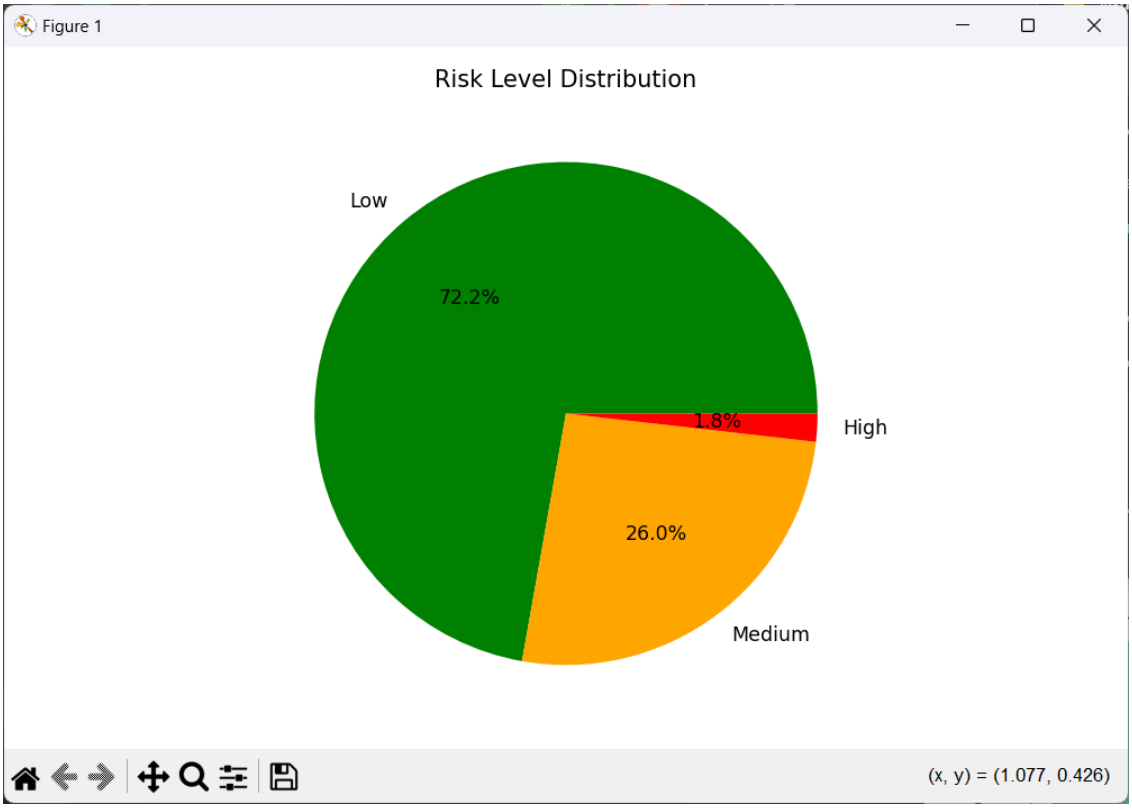
Results

The following insights were derived from the analysis of 1,000 phishing websites:

1. **1. Risk Level Distribution**
    - A majority of websites (72.2%) were classified as **Low Risk**, followed by **Medium Risk** (26.0%) and **High Risk** (1.8%).

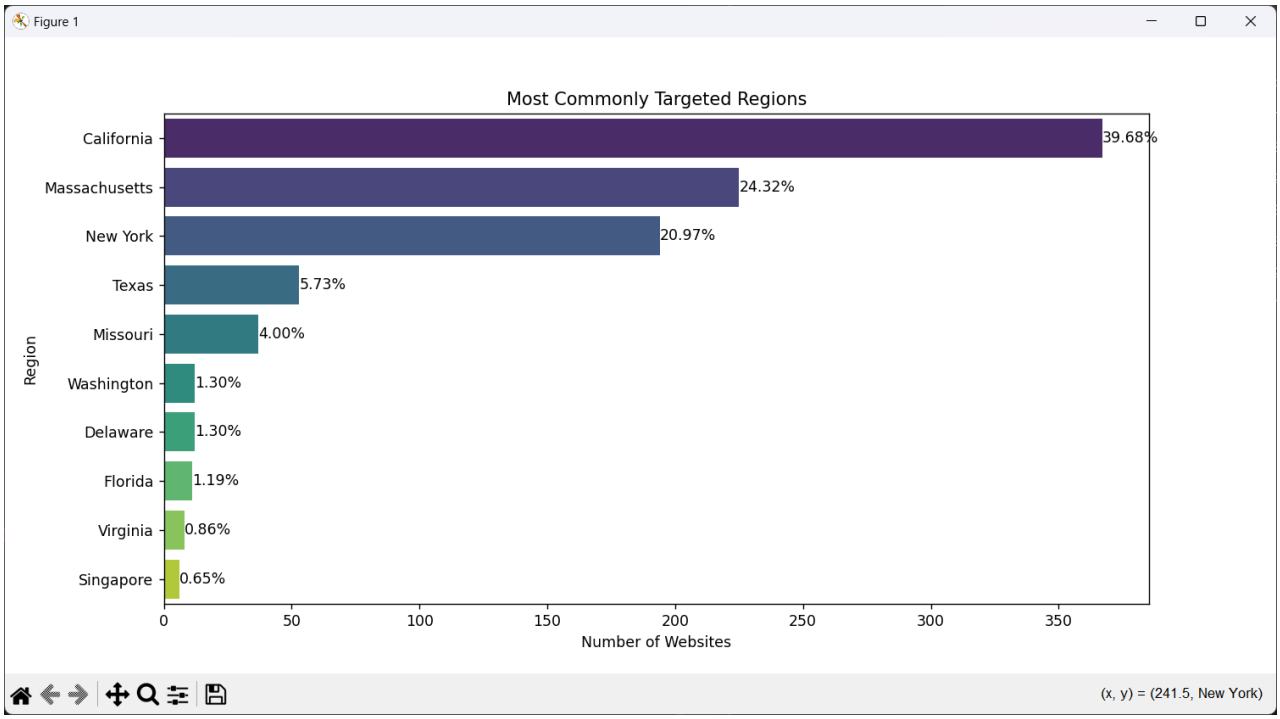
- Bar and pie charts illustrate the distribution:





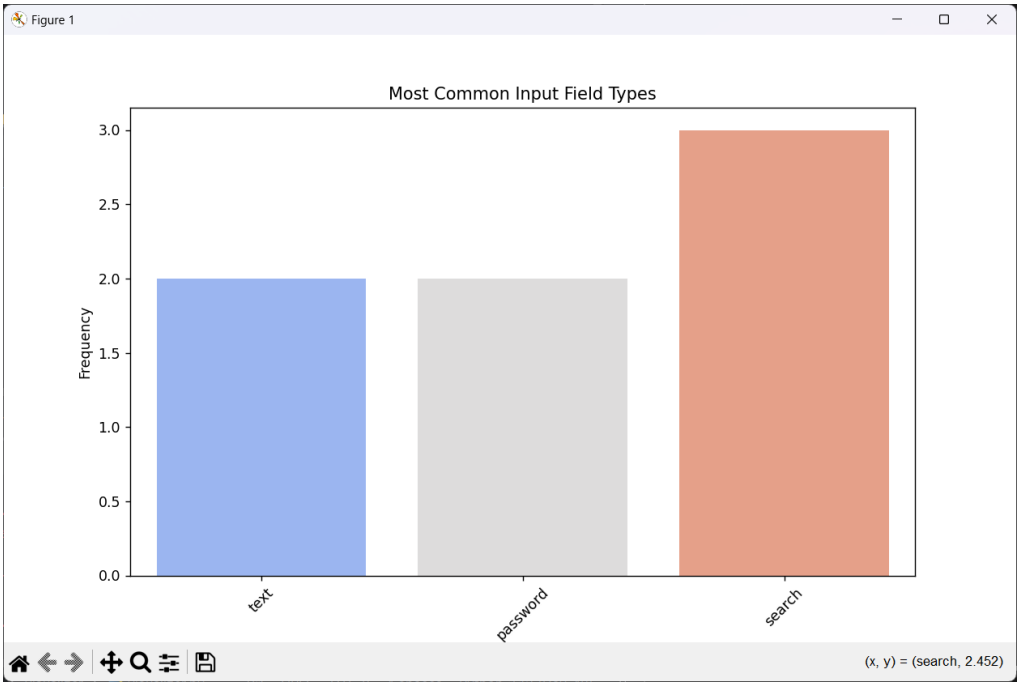
2. Most Targeted Regions

- California (39.68%) emerged as the most targeted region, followed by Massachusetts (24.32%) and New York (20.97%).
- The bar chart highlights the frequency of targeted regions:



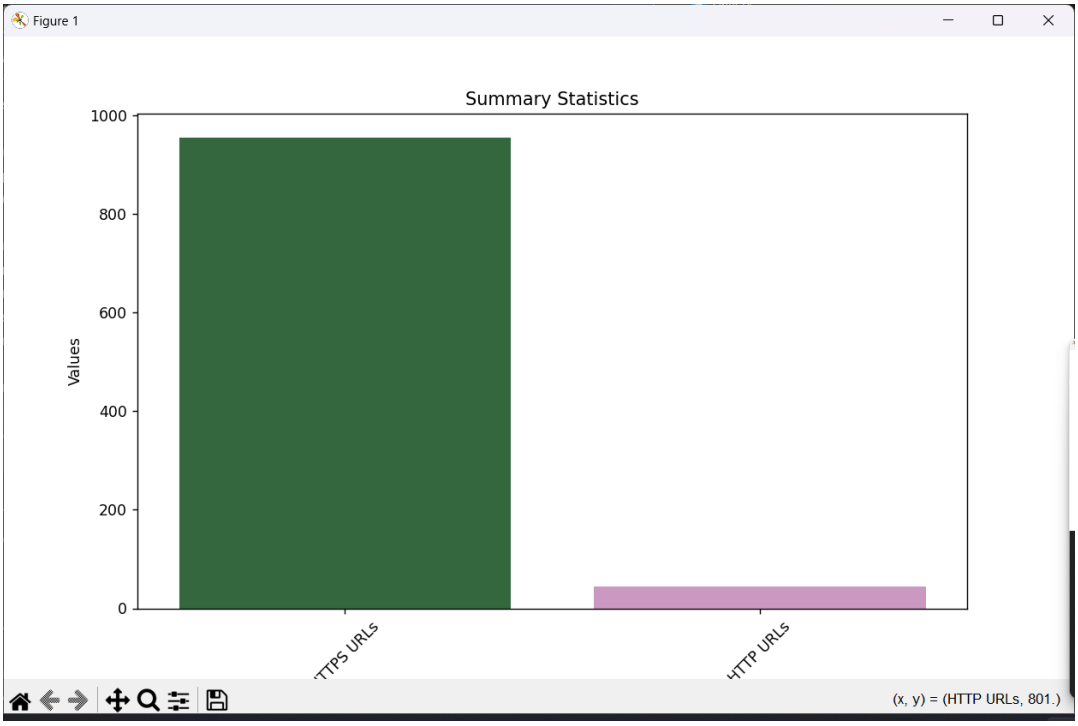
3. Most Common Input Field Types

- Phishing websites frequently included form fields for **text**, **password**, and **search** inputs.
- The bar chart showcases the distribution of input field types:



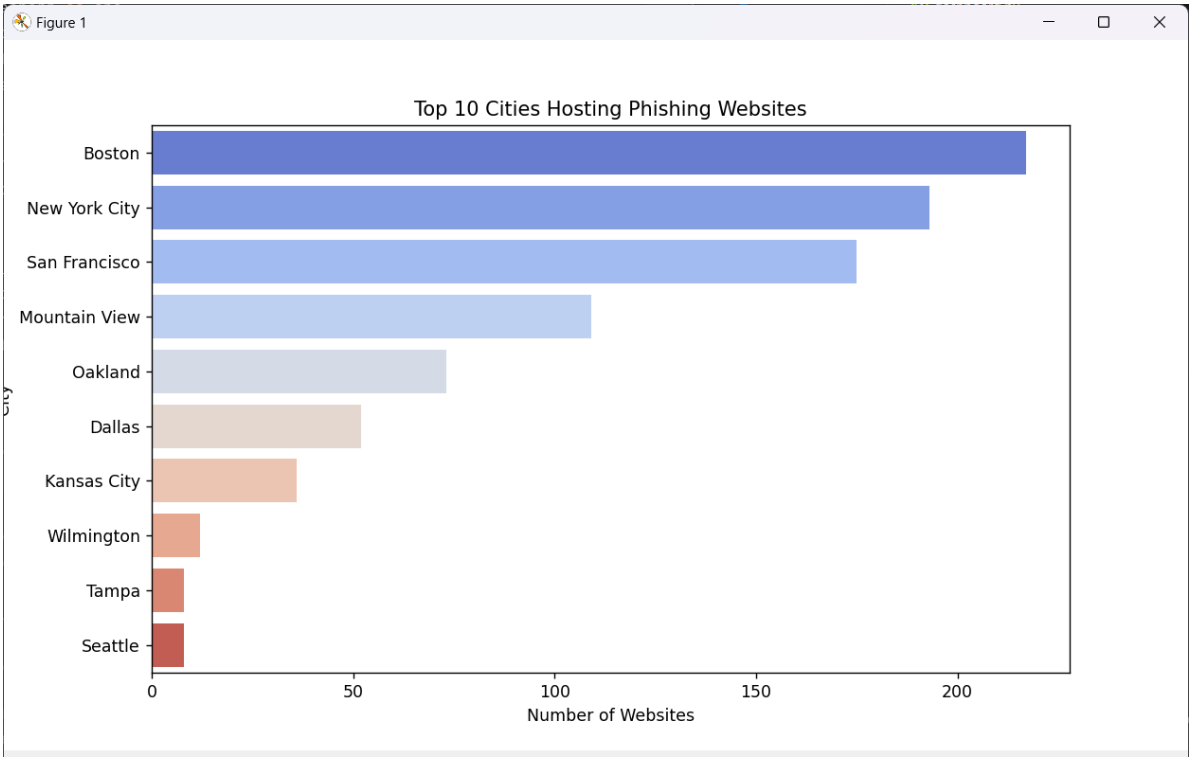
4. HTTPS vs. HTTP URLs

- 80.1% of websites used **HTTPS**, while 19.9% relied on **HTTP**, highlighting the deceptive use of secure protocols.



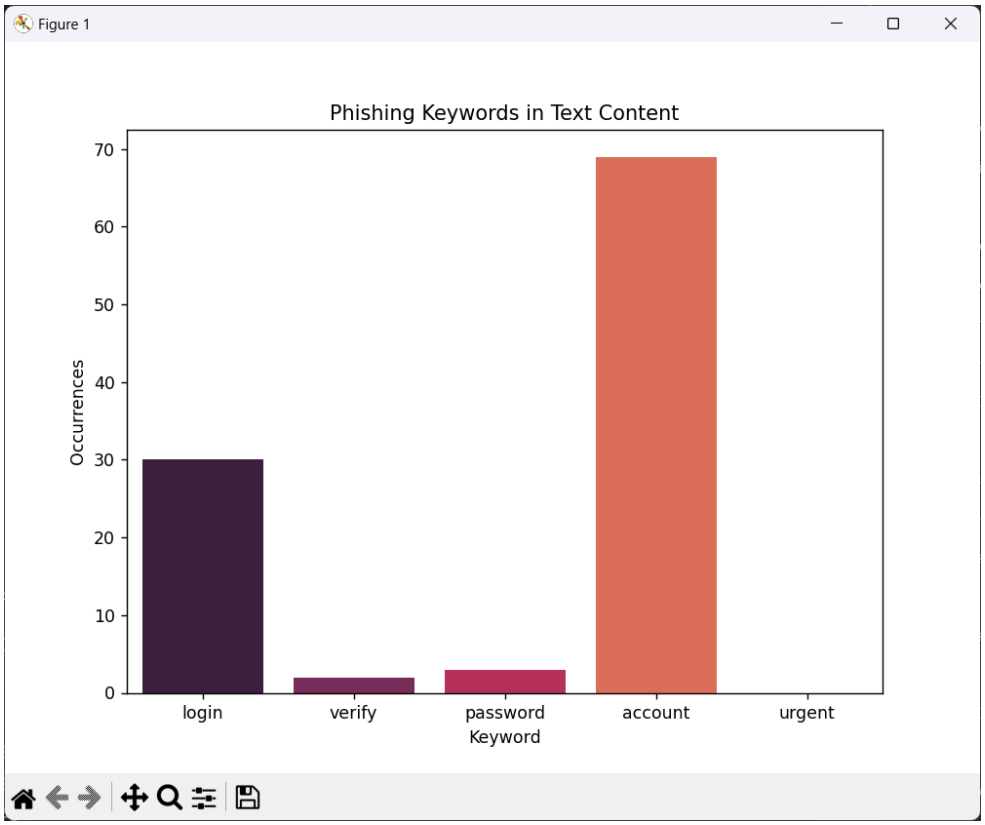
5. Top Cities Hosting Phishing Websites

- Boston, New York City, and San Francisco hosted the most phishing websites, as shown below:



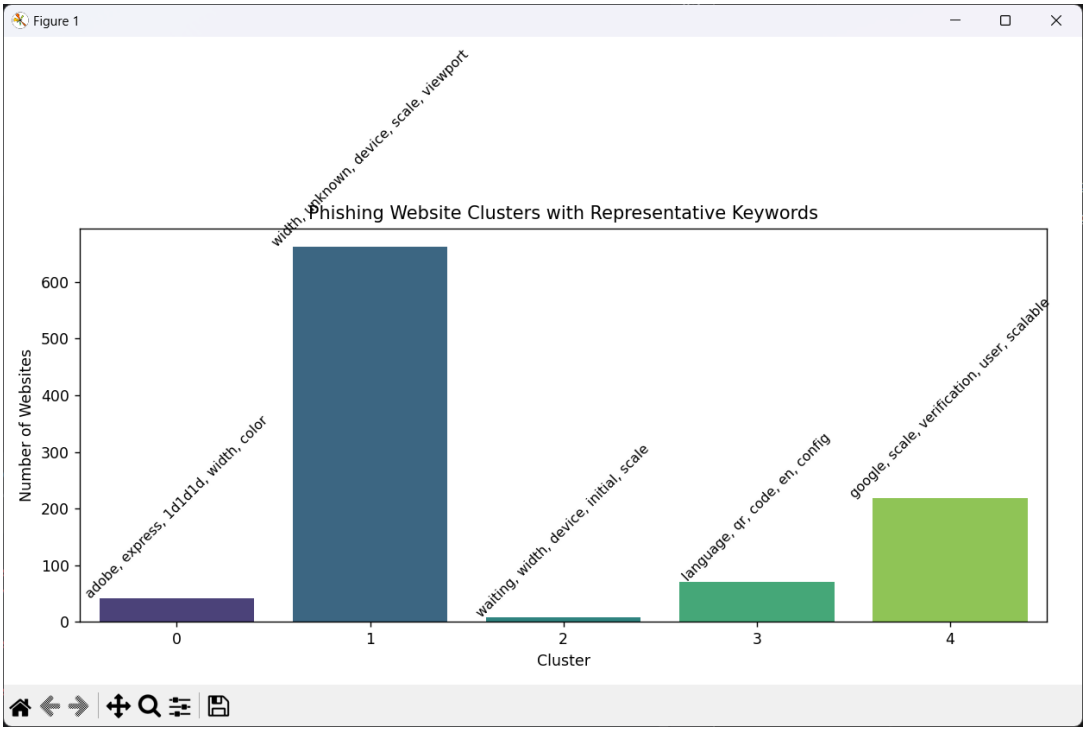
6. Keyword Analysis

- Keywords such as "login," "verify," and "account" were prominently used to deceive users.
- The bar chart and word cloud illustrate keyword frequency:



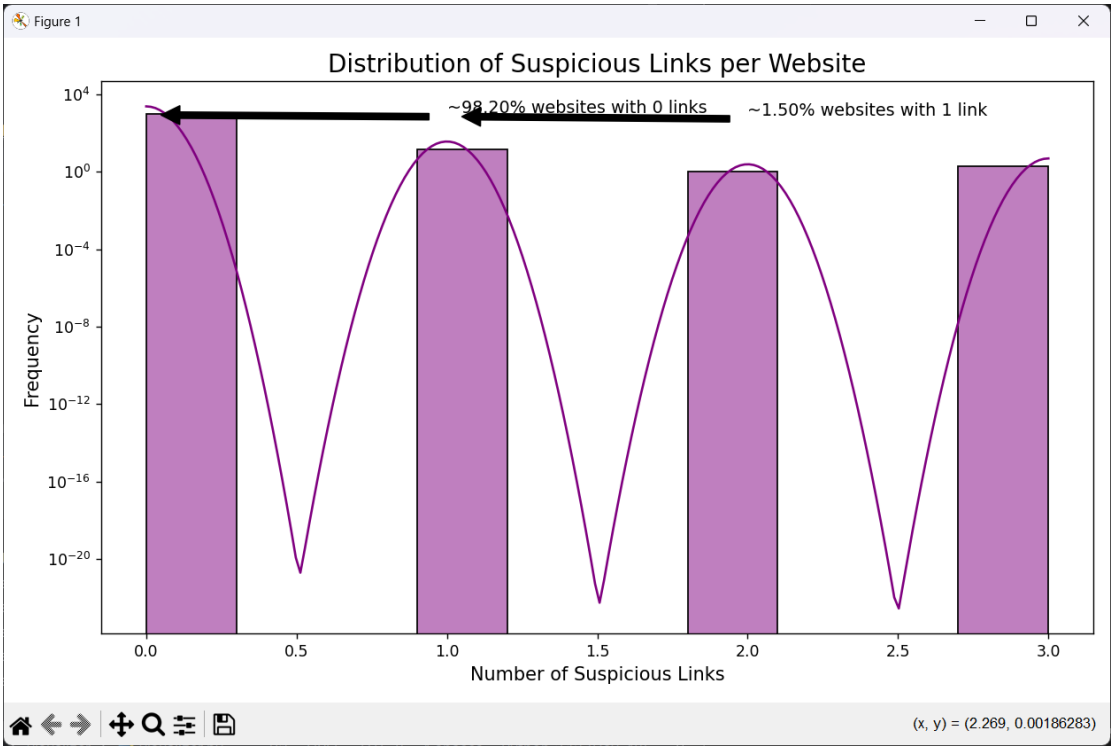
7. Clustering Analysis

- Phishing websites were grouped into clusters based on textual metadata. Each cluster displayed unique patterns, as depicted below:

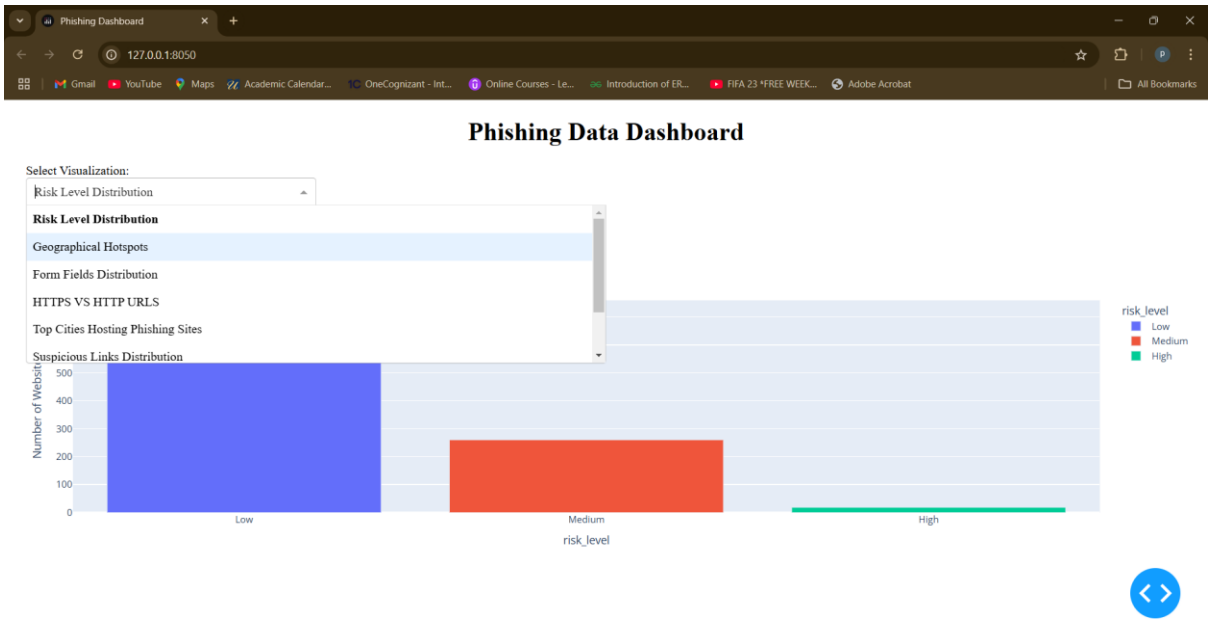


8. Suspicious Links

- While most websites contained no suspicious links, a few exhibited abnormal patterns, as shown in the histogram:



9. **Interactive Dashboard**:
    - Allowed forensic investigators to explore metadata interactively.
    - Included features like cluster analysis and geolocation mapping.



Challenges

1. **Data Collection**:
    - Many URLs were inaccessible due to server restrictions or invalid responses.
    - Resolved by retrying requests and skipping failed URLs.
2. **Data Cleaning**:
    - Missing or malformed metadata required significant preprocessing.
    - Developed custom scripts to handle anomalies.
3. **Complexity in Risk Scoring**:
    - Defining meaningful risk thresholds was challenging.
    - Addressed by iterative refinement and domain expertise.
4. **Visualization Scaling**:
    - Large datasets required optimization for performance.
    - Used efficient libraries like Plotly for interactive visualizations.

Conclusions

- The project successfully met its goal of analyzing phishing websites and providing forensic investigators with actionable insights.
- Visualizations and dashboards enabled effective presentation and exploration of trends.
- The forensic repository ensures that evidence can be queried and reused in the future.
- Future improvements include integrating real-time phishing detection and advanced machine learning models.

Project Deliverables

1. **GitHub Repository**:
    - Link: GitHub Repository for Phishing Forensics (Click on this link)
    - Contains:
        - All scripts (scraper_extractor.py, pattern_analyzer.py, data_cleaning.py, visualizer.py, dashboard.py, Forensic_Report.py, repository.py) in respective directories.
        - Data files (phishing_urls.csv, Cleaned_data.csv, Updated_scraped_data_with_risk_score.csv, scraped_data_1000_more_advanced.csv).
        - SQLite database (phishing_forensics.db, evidence_repository.db).
2. **Comprehensive Report**:
    - Attached as Comprehensive_Forensic_Analysis_Report.pdf.
3. **Installation and Usage Guide**:
    - Provided the README file.

Installation and Usage Guide

System Requirements

- **Python Version**: 3.8+
- **Pycharm IDE or any other IDE with all the required libraries installed**.
- **Dependencies**: pandas, matplotlib, seaborn, dash, sqlite3, fpdf.

Installation Steps

1. Clone the repository from GitHub: [link to repository].
2. Install dependencies: pip install -r requirements.txt.
3. Configure the environment with valid API keys for geolocation.

Usage Instructions

1. **Scraping**:
    - Run API_data.py file from scraper and extractor directory to get data from phishtank API, (not necessary to run this script as data has already been extracted from API and stored in data directory as phishing_urls.csv).
    - Run scraper_extractor.py to extract metadata, present in the same directory i.e. “scraper and extractor” directory.
2. **Analysis**:
    - Run “pattern_analyzer.py” to generate insights, present in “analyzer” directory.
3. **Visualization**:
    - Execute data_cleaning.py, visualizer.py and dashboard.py for interactive exploration, these files are present in visualizer directory along with the images of visualisation.
4. **Repository:**
    - Run “repository.py” to implement an SQLite database.
5. **Report:**
    - Execute Forensic_Report.py to generate a comprehensive report in pdf format.

Troubleshooting

- Ensure all dependencies are installed.
- Verify internet access and API configurations.

Link to Tool

- https://github.com/Prathameshmane1710/Digital_Forensic_project_repository