Phish-Blitz: A Web Resources Collection Tool.

An aid in Phishing website Detection.

Abstract:

To work on projects like Phishing website Detection, the collection of web resources (HTML, CSS, JS, Images) of the URLs is mandatory. In this project, we have come up with a tool that downloads all the required web resources along with their screenshots. This can also be used to view the website completely offline. Not only do we help in downloading the web resources, but this tool also helps to extract some important URL-based and HTML-based features from these downloaded web pages. Using this tool, we can fetch live phishing URLs from Phish Tank. We can also fetch Legitimate URLs based on Page Rank. Not just that, we can also give a list of URLs of our choice and get the web resources downloaded for those URLs too. Using this tool, we have also collected many Phishing Web Resources from various anti-phishing sites like Phish Tank, Phish Stats, Phish Storm, and Open Phish. We have created a web interface for this tool, that makes your experience with this tool, smooth and easy.

Running the application:

Firstly, we have to clone the GitHub repository from the link below:

https://github.com/Duddu-Hriday/Phish-Blitz

Then, we need to fulfill the following requirements:

- Linux (Ubuntu) Operating System
- Chrome Driver
- Python3
- Python3 Libraries:

beautifulsoup4	Pillow
requests	opencv-python
urllib3	scikit-image
tldextract	pandas
chardet	Flask
selenium	
webdriver_manager	

- WGET

Now, we can start the web application by running the app.py file

```
python3 app.py
```

Introduction:

Although there are several tools available to download the web resources, the links (src, href) still point to the direct links of the website. So, when we try to browse these downloaded web pages offline, the resources like images, JS, and CSS are not activated causing the website to look odd. So, after downloading webpages, if we can redirect the links (src, href) to the locally available web resources, then we will be able to browse the webpage locally along with the same look of the website when it was online. So, our first aim was to get the basic HTML source code of the website. The tool we chose to download HTML source code is wget.

About WGET:

GNU Wget is a command-line utility for downloading files from the web. With Wget, we can download files using HTTP, HTTPS, and FTP protocols. Wget provides several options allowing us to download multiple files, resume downloads, limit the bandwidth, recursive downloads, download in the background, mirror a website, and much more.

The command that we use to fetch the HTML source code of the URL is as follows:

```
wget --mirror --convert-links --adjust-extension --page-requisites --no-parent --U user_agent URL
```

Here, user_agent is an array containing user agents and the URL will be the actual URL.

The user agents we used for this tool are as follows:

- Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.3
- Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:89.0) Gecko/20100101 Firefox/89.0
- Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.3
- Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.114 Safari/537.36

For each URL, a user agent will be chosen randomly to avoid bot-related issues. The parameters used in the command are as follows:

- **--mirror**: Creates mirror of the website
- **--convert-links**: Converts links to make them suitable for offline viewing
- **--adjust-extension**: Adds appropriate extensions to downloaded files
- **--page-requisites**: Ensures all necessary files for displaying a webpage are downloaded
- **-- no-parent**: Prevents wget from following links outside of specific site/Directory

Note: WGET not only gets us an index.html file but also fetches us other resources. But we do not use these resources in our tool. Our whole work depends only on index.html.

Is WGET sufficient?

The work is not complete, just by using wget. The wget fetches us the basic HTML source code of the website with the name index.html. Now, our next step is to get all the important resources, which are images, CSS files, and JS files.

We use a scrapping tool called BeautifulSoup to scrape the index.html file. Using this we can fetch the links to all the images, JS files, and CSS files.

Now, we need to download all these resources. So, we use a Python library called requests to fetch all these resources.

Till now, we have a basic HTML file and all the required resources. But, the links (src and href) still point to their original links, meaning, if we browse the HTML file offline, we cannot see the images, CSS Styling, and JS features. Now we aim to redirect the links to point to the local resources that we have downloaded earlier. This work is also possible using Beautiful Soup. We simply fetch the link attributes (href, src) of the images, and link tags with rel attribute as stylesheet and script tags. Then we rewrite these values with the locally downloaded resources path. In this way, for images, CSS, and JS instead of going to the website server and fetching them online, now we use locally downloaded resources, which helps us in making the local webpage static and also useful for offline viewing.

> The question may arise: Why do we need to redirect the links? The answer to this can be quite simple. The lifetime of a phishing webpage is much less on average. So, if we do not redirect to the local resources after the phishing web page expires, all these important links will just become invalid. So, to use the dataset for a long time, we need to redirect these links to local resources

How good is this tool?

To check if the downloaded webpage looks similar to the original website, we take screenshots of both of these and compare them. The comparison is done based on a metric called Histogram Correlation. This metric can be calculated by importing a Python library called scikit-learn. We term all the webpages with a Histogram Correlation value greater than 0.8 as completely downloaded webpages and others as partially downloaded webpages.

Directory Structure of the Web Resources

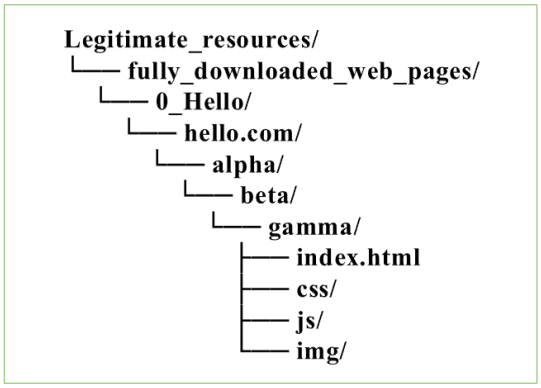
We maintain separate directories for these web pages. The outermost directory will be named legitimate_resources in case of legitimate URLs and your choice URLs. For Phishing URLs, it will be named as phishing_urls. In this outermost directory, we find 2 directories: fully_downloaded_web_pages and partially_downloaded_web_pages. A webpage belongs to partially_downloaded_webpages because of any one of the following reasons:

- File index.html does not exist.
- Offline/Online screenshots cannot be taken
- Histogram Correlation less than 0.8

In all the remaining cases, web resources are stored in fully_downloaded_web_pages.

The inner directory structure in these directories depends on the URL.

For example, the URL is **https://hello.xyz/alpha/beta/gamma**, and let it be a fully downloaded legitimate web page, then the directory structure will be as follows:



About URLs

Now that we have a tool that helps us download web resources, we can go a step further and get users a set of URLs to work on. So, users can fetch URLs by giving the count, and they will be returned with the top page rank URLs. This case is for Legitimate URLs. Now, for Phishing URLs that are live, users can get these URLs by entering the number of pages of Phish Tank, they would like to scrape. Not just these URLs, users can also give the set of URLs of their choice and download resources corresponding to these URLs.

Features

If we want to just extract URL-based features, we can do that without downloading web resources. But, if we want to extract HTML-based features, then firstly we need to download the web resources and then extract HTML-based features. Along with downloaded web resources we also additionally get a CSV file with information on the URL, location of index.html

URL Based Features:

- **domain_is_IP:** Checks if the domain part of the URL is an IP Address
- **symbol_count:** Counts the number of @, -, ~
- **https:** Checks if the URL uses HTTPS
- **domain_len:** Returns the length of the full domain, including subdomain and TLD
- **url_len:** Returns the length of the URL
- **sensitive_word:** returns True, if the following words are present in the URL: "secure", "account", "webscr", "login","signin", "ebayisapi", "banking", "confirm"
- **tld_in_domain:** Checks if any TLD appears in the domain or subdomain part of the URL
- **tld_in_path:** Checks if TLD appears in the path, parameters, query, or fragment of the URL
- **https_in_domain:** Checks for the presence of HTTPS in the domain part of the URL
- **abnormal_url:** Returns True if the domain and hostname of the URL do not match

HTML Based Features:

- **len_html_tag:** Returns the sum of lengths of style, link, form, script tags, and comments.
- **len_html:** Returns the length of HTML content as plain text.
- **hidden:** Returns True, if there are hidden divs, inputs or buttons
- **internal_external_link:** returns the counts of number of internal and external links. (anchor tag hrefs only)
- **empty_link:** Counts empty or placeholder links, such as:
  - href = “”
  - href = “#”
  - href = “#javascript::void(0)”
  - href = “#content”
  - href = “skip”
- **login_form:** Returns True, if the “name” attribute of input tags in a form is one of the following: password, pass, login, signin
- **internal_external_resource:** Counts the number of internal and external resources (link, img, script, noscript tags)
- **redirect:** Checks if HTML content contains the string “redirect”
- **alarm_window:** Checks if any script tag contains ‘alert’ or ‘window.open’ function
- **title_domain:** Checks if the domain appears in the title tag of the HTML content.
- **domain_occurence:** Counts the occurrence of domain in the HTML content.
- **brand_freq_domain:** Counts how many well-known brand names appear in the URL.
- **is_link_valid:** Checks if the response code is 200 or not for all the links in HTML.
- **multiple_https_check:** Checks if “https:” is present more than once in all the links in HTML

- **form_empty_action:** Checks if the action attribute of the form tag is empty or 'about:blank'
- **is_mail:** Checks the presence of the mail function or mailto attribute.
- **num_of_redirects:** Returns the count of the number of times the website is redirected
- **status_bar_customization:** Checks if HTML has the onmouseover attribute and has windows.status in it.

Pre-Collected Dataset:

Using this tool, we have collected the web resources of phishing websites from anti-phishing sites like Phish Tank, Open Phish, Phish Storm, and Phish Stats on different dates. We have uploaded these web resources to GitHub which can be accessed using this tool. This may help researchers who are in urgent need of the dataset of phishing websites and want the output of this tool. Not just Phishing web resources, we have also collected a huge dataset of web resources of top Legitimate websites which can also be accessed in the same manner.

Conclusion:

This tool is a very useful aid to researchers who are working on Phishing Website Detection. We have tried to include all the best HTML and URL-based features, that would make your work a bit easier. This tool is a very useful tool to preserve the resources of the websites whose lifetime may not be long but is needed for the research. WGET is a wonderful tool, using which we tried to build a much-advanced tool.

Screenshots:

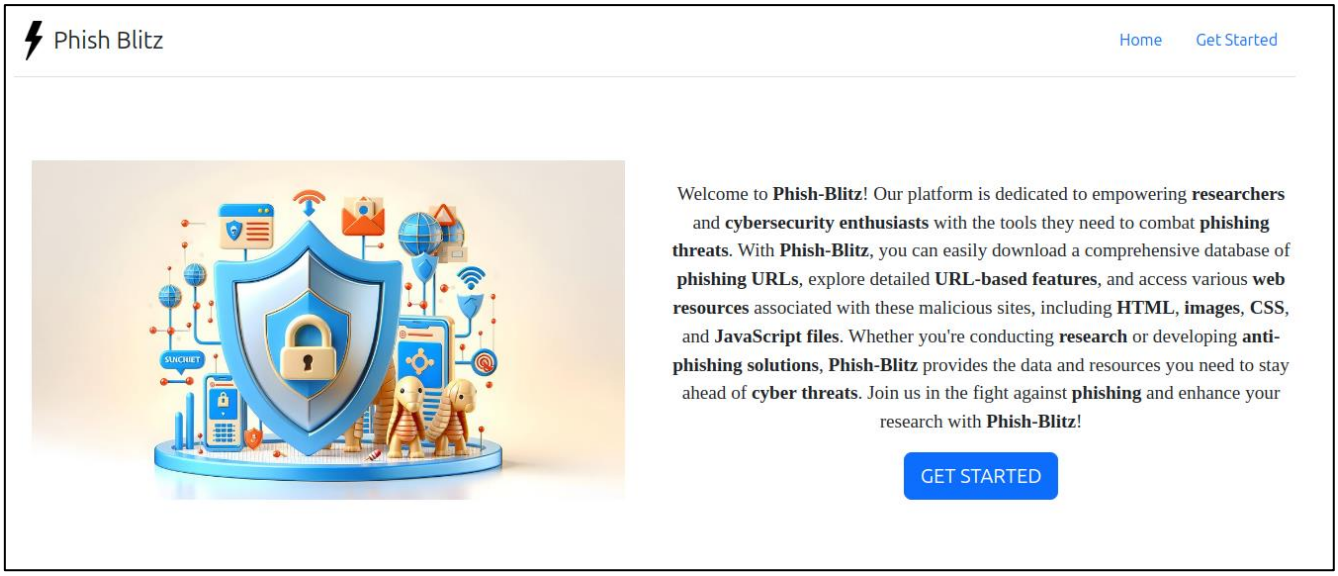


Fig 1: Main Page of Phish Blitz

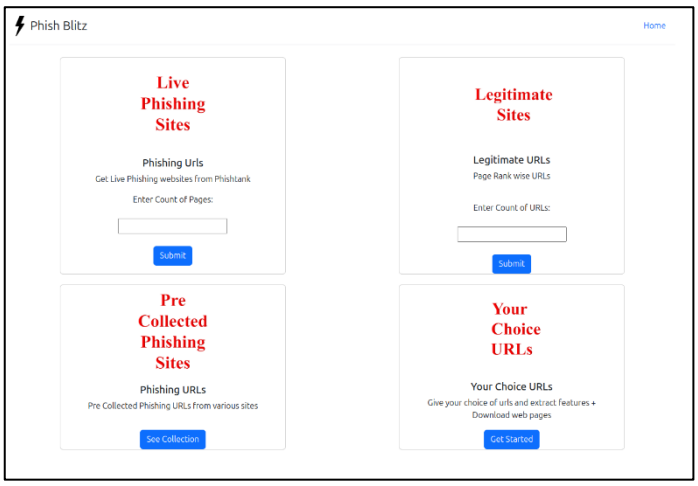


Fig 2: Resources of Phish Blitz

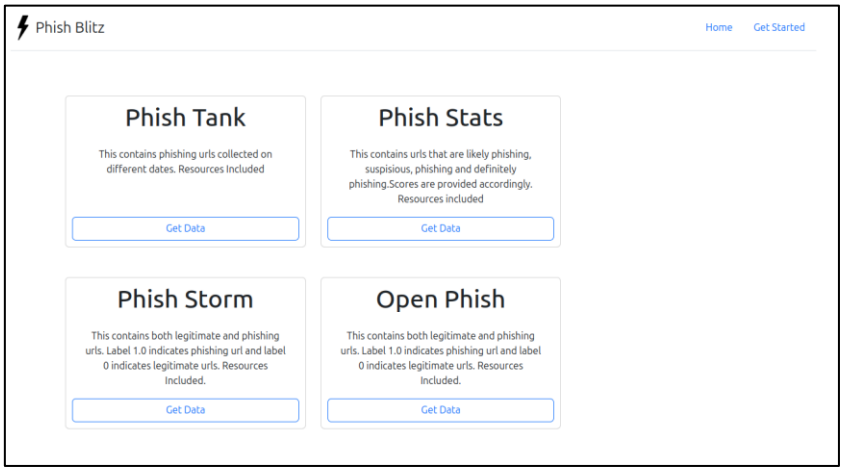
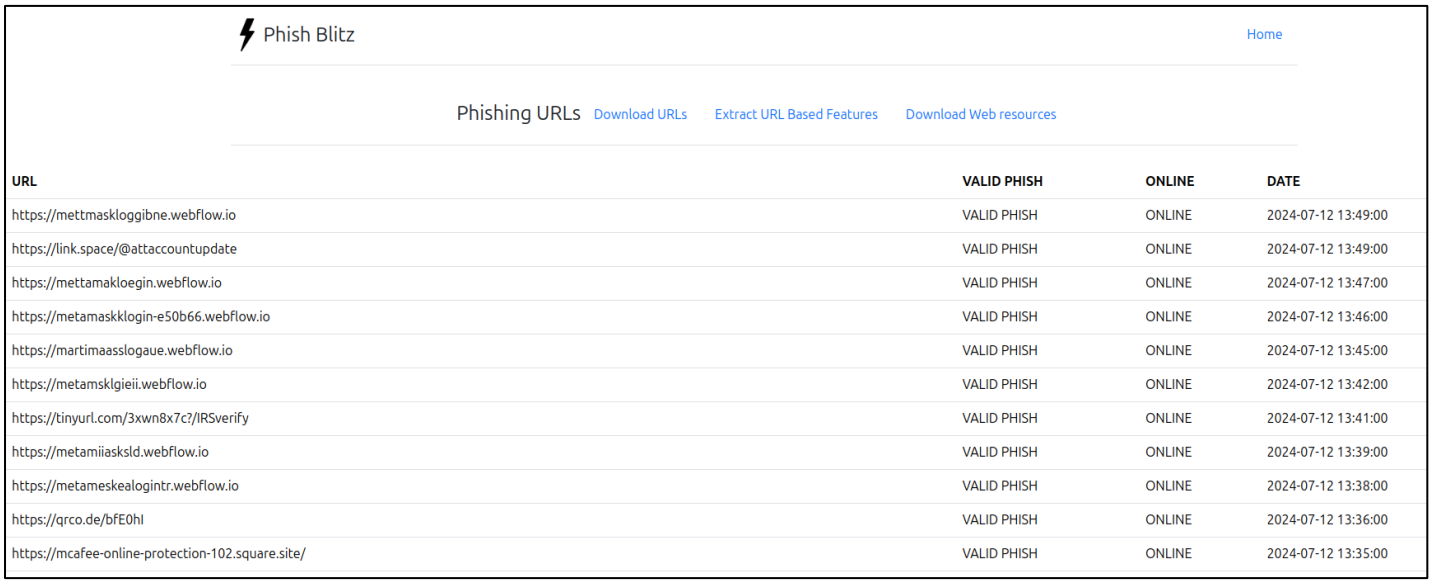


Fig 3: Pre Collected Phishing Web resources



URL	VALID PHISH	ONLINE	DATE
https://mettmaskloggibne.webflow.io	VALID PHISH	ONLINE	2024-07-12 13:49:00
https://link.space/@attaccountupdate	VALID PHISH	ONLINE	2024-07-12 13:49:00
https://mettamakloegin.webflow.io	VALID PHISH	ONLINE	2024-07-12 13:47:00
https://metamaskklogin-e50b66.webflow.io	VALID PHISH	ONLINE	2024-07-12 13:46:00
https://martimaasslogaue.webflow.io	VALID PHISH	ONLINE	2024-07-12 13:45:00
https://metamsklgieii.webflow.io	VALID PHISH	ONLINE	2024-07-12 13:42:00
https://tinyurl.com/3xwn8x7c?/IRSverify	VALID PHISH	ONLINE	2024-07-12 13:41:00
https://metamiiasksld.webflow.io	VALID PHISH	ONLINE	2024-07-12 13:39:00
https://metameskealogintr.webflow.io	VALID PHISH	ONLINE	2024-07-12 13:38:00
https://qrco.de/bfE0hI	VALID PHISH	ONLINE	2024-07-12 13:36:00
https://mcafee-online-protection-102.square.site/	VALID PHISH	ONLINE	2024-07-12 13:35:00

Fig 4: Live Phishing URLs from Phish Tank

Phishing URL Features Download Features

URL	Domain==IP	Count of @,-,~	HTTPS	Domain Length	URL Length	Number of Dots in Hostname	Presence of Sensitive words	TLD in Domain	HTTPS in Domain	Abnormal URL
https://mettmaskloggibne.webflow.io	False	0	True	27	35	1	False	0	False	True
https://link.space/@attaccountupdate	False	1	True	10	36	0	True	1	False	False
https://mettamakloegin.webflow.io	False	0	True	25	33	1	False	0	False	True
https://metamaskklogin-e50b66.webflow.io	False	1	True	32	40	1	True	0	False	True
https://martimaasslogaue.webflow.io	False	0	True	27	35	1	False	0	False	True
https://metamsklgieii.webflow.io	False	0	True	24	32	1	False	0	False	True
https://tinyurl.com/3xwn8x7c?/IRSverify	False	0	True	11	39	0	False	0	False	False
https://metamiiaskslд.webflow.io	False	0	True	24	32	1	False	0	False	True
https://metameskealogintr.webflow.io	False	0	True	28	36	1	True	0	False	True

Fig 5: URL-Based Features

URL	Internal Links	External Links	Empty Links	Login Form	HTML Length Tag	HTML Length	Alarm Window	Redirection	Hidden Divs	Title Domain	Internal Resources	External Resources	Domain Occurence	Brand Domain	Working Links	Not Working Links	Multiple HTTPs	Form Empty Action	Same Form Action Domain	Is Mail	Number of Redirects	StatusBar Customization
google.com	23	1	0	0	172978	369	1	0	1	1	3	0		1	24	0	True	False	False	False		False
facebook.com	-	-	-	-	-	-	-	-	-	-	-	-		-	-	-	-	-	-	-		-
aajtak.in	318	77	0	0	281331	19994	1	0	0	0	157	23		1	382	11	False	False	None	False		False
whatsapp.com	-	-	-	-	-	-	-	-	-	-	-	-		-	-	-	-	-	-	-		-
cricbuzz.com	250	9	0	0	24509	11430	0	0	1	1	51	0		1	255	5	False	True	True	False		False

Fig 6: HTML-Based Features