Homograph Attack Detection Using Python

Objective

This document outlines a Python-based method to detect phishing links using homoglyph characters that mimic legitimate domain names. The approach relies on the `homoglyphs` library to flag such suspicious links.

Background

Homoglyphs are characters that look similar but originate from different Unicode scripts. These characters are commonly exploited in domain spoofing attacks called IDN Homograph Attacks.

Example:

- Legitimate: `google.com`
- Spoofed: `google.com` (uses Cyrillic "o")

The spoofed version appears identical in browsers but directs users to a potentially harmful website.

Python Script

The following Python script extracts links from text and flags those that use homoglyph characters:

```
import re

import sys


try:

    import homoglyphs as hg

except ImportError:

    print("Please install the homoglyphs package: pip install homoglyphs")

    sys.exit(1)


def extract_links_from_text(text):
```

```
    url_pattern = re.compile(r'(https?://[^\s<>"]+|www\.[^\s<>"]+)', re.IGNORECASE)
    return [link.rstrip('.,);]>"\'') for link in url_pattern.findall(text)]

def is_link_suspicious(link, homoglyphs_obj):
    ascii_versions = homoglyphs_obj.to_ascii(link)
    return bool(ascii_versions and link not in ascii_versions)

def check_links_in_text(text):
    homoglyphs_obj = hg.Homoglyphs(languages={'en', 'ru', 'el'})
    links = extract_links_from_text(text)
    suspicious_links = [link for link in links if is_link_suspicious(link, homoglyphs_obj)]

    print(f"All Links ({len(links)}):")
    for link in links:
        print(f"  {link} {'<-- SUSPICIOUS' if link in suspicious_links else ''}")

    print("\nSuspicious Links:")
    if suspicious_links:
        for link in suspicious_links:
            print(f"  {link}")
    else:
        print("  None detected.")

if __name__ == "__main__":
    sample_text = """
    https://google.com
```

```
https://google.com

example.com

www.paypa1.com

"""

check_links_in_text(sample_text)
```

How It Works

1. Extracts URLs using regular expressions.
2. Compares each link with its ASCII counterpart using the homoglyphs library.
3. Flags suspicious links that differ in Unicode but appear visually similar.
