Neo-reGeorg/README-en.md at master · L-codes/Neo-reGeorg

By L-codes

Archived: 2026-04-05 21:49:05 UTC

简体中文 | English

Neo-reGeorg is a project designed to actively restructure reGeorg with the aim of:

- Improve usability and avoid feature detection
- Improve tunnel connection security
- Improve the confidentiality of transmission content
- Solve the existing problems of reGeorg and fix some small bugs

> This tool is limited to safety research and teaching, and the user assumes all legal and related responsibilities caused by the use of this tool! The author does not bear any legal and related responsibilities!

Version

5.3.0 - Change Log

python dependencies

```
python -m pip install requests

# Optional
python -m pip install requests[socks] # SOCKS5 proxy support
python -m pip install curl-cffi       # Switch to the curl-cffi library for improved performance and
python -m pip install requests_ntlm   # NTLM authentication support
```

Features

- The transmission content is encrypted by deformed base64 and disguised as base64 encoding
- Use BLV (Byte-LengthOffset-Value) data format to transmit data
- Direct request response can be customized (such as a disguised 404 page)
- HTTP Headers can be customized
- Support request template
- Custom HTTP response code
- Multiple URL random requests
- Server-side DNS resolution
- Compatible with python2 / python3

- High compatibility of the server environment, such as the server is unstable, the server is only deployed on some machines under load balancing and other special circumstances
- (php only) Refer to pivotnacci to create multiple TCP connections for a single session, to deal with some load balancing scenarios
- aspx/ashx/jsp/jspx no longer depends on Session, and can run normally in harsh environments such as no cookies
- (non-php & non-nodejs) supports intranet forwarding to deal with load balancing environment
- Support process to start the server to deal with more scenarios

Basic Usage

- **Step 1.** Set the password to generate tunnel server.(aspx|ashx|jsp|jspx|php) and upload it to the web server.

```
$ python neoreg.py generate -k password

    [+] Create neoreg server files:
       => neoreg_servers/tunnel.jsp
       => neoreg_servers/tunnel.jspx
       => neoreg_servers/tunnel.ashx
       => neoreg_servers/tunnel.aspx
       => neoreg_servers/tunnel.php
       => neoreg_servers/tunnel.go
```

- **Step 2.** Use `neoreg.py` to connect to the web server and create a socks5 proxy locally.

```
$ python3 neoreg.py -k password -u http://xx/tunnel.php
+-------------------------------------------------------------------------+
  Log Level set to [DEBUG]
  Starting socks server [127.0.0.1:1080]
  Tunnel at:
    http://xx/tunnel.php
+-------------------------------------------------------------------------+
```

Advanced Usage

1. Support the generated server, by default directly requesting and responding to the specified page content (such as a disguised 404 page)

```
$ python neoreg.py generate -k <you_password> --file 404.html
$ python neoreg.py -k <you_password> -u <server_url> --skip
```

2. For example, the server WEB needs to set the proxy to access

```
$ python neoreg.py -k <you_password> -u <server_url> --proxy socks5://10.1.1.1:8080
```

3. To set `Authorization`, there are also custom `Header` or `Cookie` content.

```
$ python neoreg.py -k <you_password> -u <server_url> -H 'Authorization: cm9vdDppcyB0d2VsdmU=' --cook
```

4. Need to disperse requests, upload to multiple paths, such as memory-webshell

```
$ python neoreg.py -k <you_password> -u <url_1> -u <url_2> -u <url_3> ...
```

5. Turn on http forwarding to cope with load balancing

```
$ python neoreg.py -k <you_password> -u <url> -r <redirect_url>
```

6. Use the port forwarding function, do not start the socks5 service (127.0.0.1:1080 -> ip:port)

```
$ python neoreg.py -k <you_password> -u <url> -t <ip:port>
```

7. Set the request content template (you need to specify it when generating)

```
# The request content will be replaced with NEOREGBODY
$ python3 neoreg.py -k password -T 'img=&save=ok'
$ python3 neoreg.py -k password -T 'img=&save=ok' -u http://127.0.0.

# NOTE Allows template content to be written to a file -T file
```

8. Support the creation process to start a new Neoreg server-side, which can deal with harsh special environments

```
$ go run neoreg_servers/tunnel.go 8000
$ python3 neoreg.py -k password -u http://127.0.0.1:8000/anysting
```

9. Supports in-memory proxy format for Node.js. Modify the path in the JS file by adding `const path = '/proxy_path';`, and include the `--async-connect` parameter for connections.

```
$ python3 neoreg.py -k password --async-connect -u <url>
```

- For more information on performance and stability parameters, refer to -h help information

```
# Generate server-side scripts
$ python neoreg.py generate -h
```

```
    usage: neoreg.py [-h] -k KEY [-o DIR] [-f FILE] [-c CODE] [--read-buff Bytes]
                     [--max-read-size KB]

    Generate neoreg webshell

    optional arguments:
      -h, --help            show this help message and exit
      -k KEY, --key KEY     Specify connection key.
      -o DIR, --outdir DIR  Output directory.
      -f FILE, --file FILE  Camouflage html page file
      -c CODE, --httpcode CODE
                            Specify HTTP response code. When using -r, it is
                            recommended to <400 (default: 200)
      -T STR/FILE, --request-template STR/FILE
                            HTTP request template (eg:
                            'img=&save=ok')
      --read-buff Bytes     Remote read buffer (default: 513)
      --max-read-size KB    Remote max read size (default: 512)

# Connection server
$ python neoreg.py -h
    usage: neoreg.py [-h] -u URI [-r URL] [-R] [-t IP:PORT] -k KEY [-l IP]
                     [-p PORT] [-s] [-H LINE] [-c LINE] [-x LINE] [-T STR/FILE]
                     [-a] [--php-skip-cookie] [--go] [--php-connect-timeout S]
                     [--local-dns] [--read-buff KB] [--read-interval MS]
                     [--write-interval MS] [--max-threads N] [--max-retry N]
                     [--cut-left N] [--cut-right N] [--extract EXPR]
                     [--ntlm-auth USER:PASS] [-v]

    Socks server for Neoreg HTTP(s) tunneller (DEBUG MODE: -k debug)

    optional arguments:
      -h, --help            show this help message and exit
      -u URI, --url URI     The url containing the tunnel script
      -r URL, --redirect-url URL
                            Intranet forwarding the designated server (only
                            java/.net)
      -R, --force-redirect  Forced forwarding (only jsp -r)
      -t IP:PORT, --target IP:PORT
                            Network forwarding Target, After setting this
                            parameter, port forwarding will be enabled
      -k KEY, --key KEY     Specify connection key
      -l IP, --listen-on IP
                            The default listening address (default: 127.0.0.1)
      -p PORT, --listen-port PORT
                            The default listening port (default: 1080)
      -s, --skip            Skip usability testing
```

```
    -H LINE, --header LINE
                          Pass custom header LINE to server
    -c LINE, --cookie LINE
                          Custom init cookies
    -x LINE, --proxy LINE
                          Proto://host[:port] Use proxy on given port
    -T STR/FILE, --request-template STR/FILE
                          HTTP request template (eg:
                          'img=&save=ok')
    -a, --async-connect   Asynchronous CONNECT (e.g., in PHP, Node.js)
    --php-skip-cookie     Skip cookie availability check in php
    --go                  Use go connection method
    --php-connect-timeout S
                          PHP connect timeout (default: 0.5)
    --local-dns           Use local resolution DNS
    --read-buff KB        Local read buffer, max data to be sent per POST
                          (default: 7, max: 50)
    --read-interval MS    Read data interval in milliseconds (default: 300)
    --write-interval MS   Write data interval in milliseconds (default: 200)
    --max-threads N       Proxy max threads (default: 400)
    --max-retry N         Max retry requests (default: 10)
    --cut-left N          Truncate the left side of the response body
    --cut-right N         Truncate the right side of the response body
    --extract EXPR        Manually extract BODY content (eg:
                          <html><p>NEOREGBODY</p></html> )
    --ntlm-auth USER:PASS
                          Enable NTLM authentication for web requests (format:
                          DOMAIN\USER:PASSWORD or USER:PASSWORD)
    -v                    Increase verbosity level (use -vv or more for greater
                          effect)
```

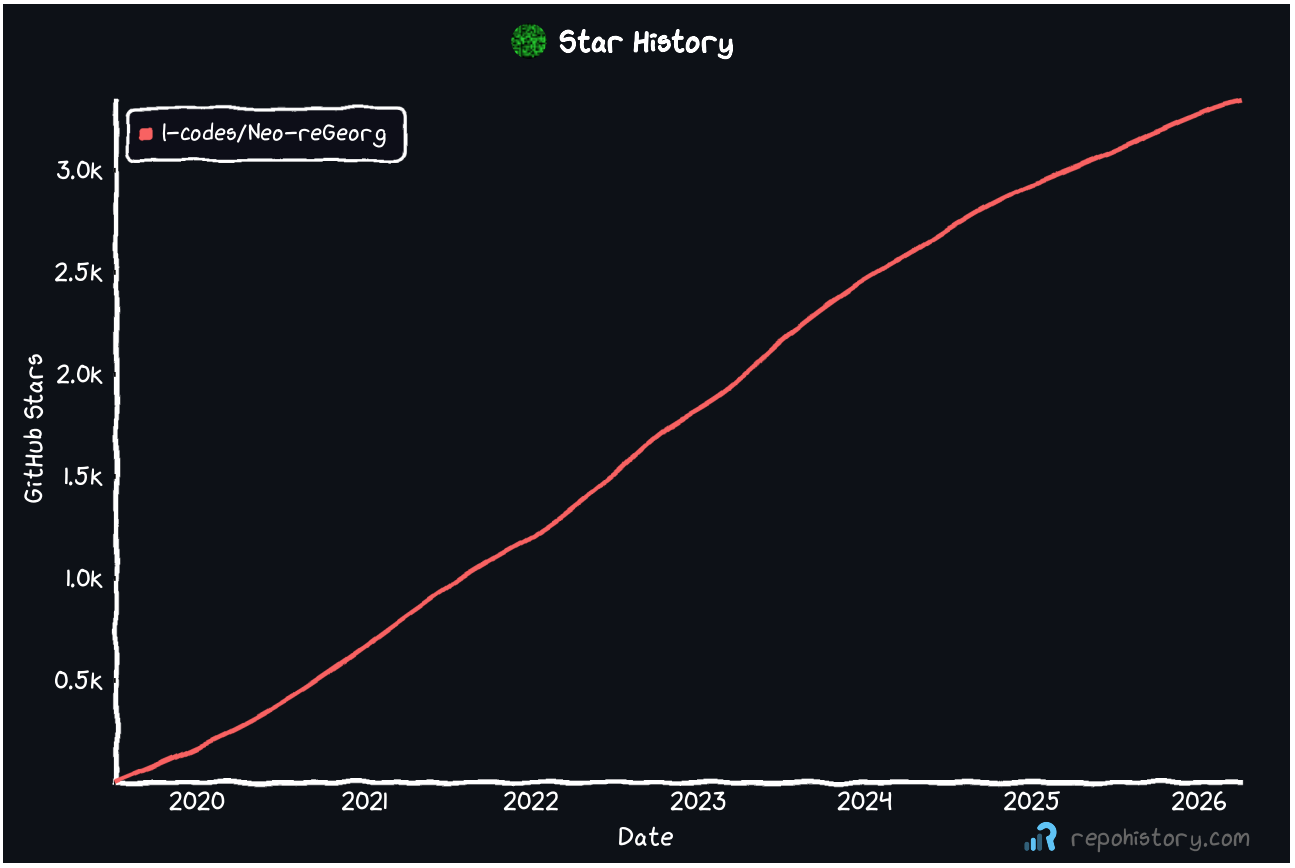
Remind

- When running `neoreg.py` with high concurrency on Mac OSX, a large number of network requests will be lost. You can use `ulimit -n 2560` to modify the "maximum number of open files" of the current shell.

License

GPL 3.0

Star History Chart



Source: https://github.com/L-codes/Neo-reGeorg/blob/master/README-en.md