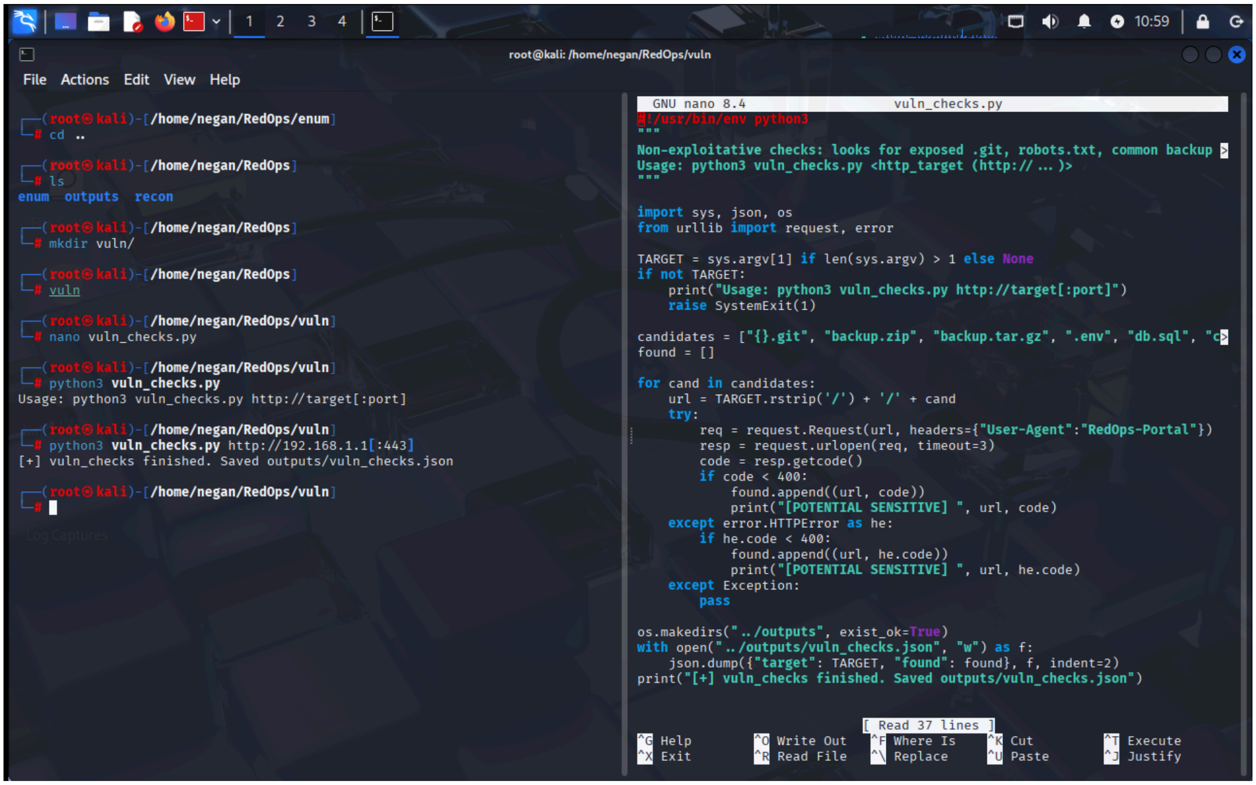
VULN_CHECK

A lightweight misconfiguration checker

Python Script :

```
#!/usr/bin/env python3
"""
Non-exploitative checks: looks for exposed .git, robots.txt, common backup filenames (based
on outputs/http_enum.json).
Usage: python3 vuln_checks.py <http_target (http://…)>
"""

import sys, json, os
from urllib import request, error

TARGET = sys.argv[1] if len(sys.argv) > 1 else None
if not TARGET:
    print("Usage: python3 vuln_checks.py http://target[:port]")
    raise SystemExit(1)

candidates = ["{}.git", "backup.zip", "backup.tar.gz", ".env", "db.sql", "config.php", "robots.txt"]
found = []

for cand in candidates:
    url = TARGET.rstrip('/') + '/' + cand
    try:
        req = request.Request(url, headers={"User-Agent":"RedOps-Portal"})
        resp = request.urlopen(req, timeout=3)
        code = resp.getcode()
        if code < 400:
            found.append((url, code))
            print("[POTENTIAL SENSITIVE] ", url, code)
    except error.HTTPError as he:
        if he.code < 400:
            found.append((url, he.code))
            print("[POTENTIAL SENSITIVE] ", url, he.code)
    except Exception:
        pass

os.makedirs("../outputs", exist_ok=True)
with open("../outputs/vuln_checks.json", "w") as f:
    json.dump({"target": TARGET, "found": found}, f, indent=2)
print("[+] vuln_checks finished. Saved outputs/vuln_checks.json")
```

Working :



```
┌──(root㉿kali)-[/home/negan/RedOps/enum]
└─# cd ..

┌──(root㉿kali)-[/home/negan/RedOps]
└─# ls
enum  outputs  recon

┌──(root㉿kali)-[/home/negan/RedOps]
└─# mkdir vuln/

┌──(root㉿kali)-[/home/negan/RedOps]
└─# vuln

┌──(root㉿kali)-[/home/negan/RedOps/vuln]
└─# nano vuln_checks.py

┌──(root㉿kali)-[/home/negan/RedOps/vuln]
└─# python3 vuln_checks.py
Usage: python3 vuln_checks.py http://target[:port]

┌──(root㉿kali)-[/home/negan/RedOps/vuln]
└─# python3 vuln_checks.py http://192.168.1.1[:443]
[+] vuln_checks finished. Saved outputs/vuln_checks.json

┌──(root㉿kali)-[/home/negan/RedOps/vuln]
└─# 
```

```
  GNU nano 8.4                          vuln_checks.py
#!/usr/bin/env python3
"""
Non-exploitative checks: looks for exposed .git, robots.txt, common backup >
Usage: python3 vuln_checks.py <http_target (http:// ... )>
"""

import sys, json, os
from urllib import request, error

TARGET = sys.argv[1] if len(sys.argv) > 1 else None
if not TARGET:
    print("Usage: python3 vuln_checks.py http://target[:port]")
    raise SystemExit(1)

candidates = ["{}.git", "backup.zip", "backup.tar.gz", ".env", "db.sql", "c>
found = []

for cand in candidates:
    url = TARGET.rstrip('/') + '/' + cand
    try:
        req = request.Request(url, headers={"User-Agent":"RedOps-Portal"})
        resp = request.urlopen(req, timeout=3)
        code = resp.getcode()
        if code < 400:
            found.append((url, code))
            print("[POTENTIAL SENSITIVE] ", url, code)
    except error.HTTPError as he:
        if he.code < 400:
            found.append((url, he.code))
            print("[POTENTIAL SENSITIVE] ", url, he.code)
    except Exception:
        pass

os.makedirs("../outputs", exist_ok=True)
with open("../outputs/vuln_checks.json", "w") as f:
    json.dump({"target": TARGET, "found": found}, f, indent=2)
print("[+] vuln_checks finished. Saved outputs/vuln_checks.json")

                              [ Read 37 lines ]
^G Help       ^O Write Out  ^F Where Is   ^K Cut        ^T Execute
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify
```