PROJET FINAL : BLACKPYRECONX – FRAMEWORK D'ATTAQUE COMPLET EN PYTHON

Objectif :

Construire un **framework de hacking modulaire** en Python, simulant une opération complète de type Red Team : de la reconnaissance jusqu'à la post-exploitation et à l'exfiltration automatique, avec interface CLI ou Telegram.

Spécifications du projet

1. Arborescence du projet

```
arduino
CopierModifier
BlackPyReconX/
├── main.py
├── config.json
├── modules/
│   ├── osint.py
│   ├── scanner.py
│   ├── exploit_web.py
│   ├── exploit_sys.py
│   ├── persistence.py
│   ├── exfiltration.py
│   ├── evasion.py
│   ├── reporting.py
│   ├── telegram_bot.py
│   ├── crypto_tools.py
│   └── utils.py
├── data/
│   ├── targets.txt
│   ├── dorks.txt
│   ├── passwords.txt
│   └── assets/
│       └── image_stego.png
├── outputs/
│   ├── screenshots/
│   ├── keylogs/
│   ├── exfiltrated.zip
│   └── rapport_final.txt
├── README.md
└── requirements.txt
```

Modules & Fonctions à intégrer

`main.py`

- Interface CLI avec `argparse` pour lancer des modules.
- Exemple :

```
python main.py --target nysafrica.com --osint --scan --web --exploit
--exfil --report
```

`osint.py`

- Utilise les **API suivantes** pour fingerprint la cible :
  - `ipinfo.io`
  - `ip-api.com`
  - `abuseipdb.com`
  - `shodan.io`
- Enregistre tout dans `outputs/osint.txt`.

`scanner.py`

- Scanner de ports TCP/UDP custom.
- Détection OS via TTL (`ping`), reverse DNS.
- Bannière grabbing (22, 80, 443).
- Ping multiple avec % de réussite.
- Affichage ASCII graphique des ports ouverts.
- Enregistre dans `outputs/scan_results.txt`.

`exploit_web.py`

- Test manuel et automatique des vulnérabilités :
  - XSS, SQLi, LFI, Bruteforce de formulaire, Headers, cookies.
  - Scanner de vulnérabilités maison.
- Affiche les résultats et sauvegarde dans `outputs/web_vulns.txt`.

`exploit_sys.py`

- Reverse Shell ou Bind Shell avec `socket`.
- Keylogger (avec sauvegarde dans `keylogs/`).
- Screenshot automatique (`PIL` ou `pyautogui`).
- Caméra snapshot (si activé).
- Peut envoyer automatiquement via Telegram.

`crypto_tools.py`

- Outils de chiffrement :
  - Base64, XOR, Fernet, rot13.
- Outils de stéganographie (`steganogan` ou manuelle avec `PIL`).
- Cache un script Python dans une image et peut l'exécuter ensuite.

exfiltration.py

- Compresse (`zipfile`), chiffre (`Fernet`) les outputs.
- Envoie via :
  - `Telegram API`
  - webhook `httpbin`
  - email (optionnel).

persistence.py

- Persistance basique :
  - Tâche planifiée (Windows)
  - Cron job (Linux)
  - Ajout autorun dans registre.

evasion.py

- Obfuscation de script (base64 encode)
- Split en petits scripts
- Nom de fichier "légitimes"
- Bypass de détection très simple

reporting.py

- Génère `rapport_final.txt` automatiquement :
  - Timestamp début/fin
  - Résultats OSINT
  - Résultats SCAN
  - Vulnérabilités détectées
  - Exploits lancés
  - Chemin d'exfiltration
  - Statut final

telegram_bot.py

- Interface pour commander les modules :
  - `/osint domaine.com`
  - `/scan ip`
  - `/rapport`
- Envoi automatique des résultats compressés et encryptés.
- Notifications push si exfiltration OK.

Livrables obligatoires

- `README.md` détaillant le fonctionnement et l'ordre d'attaque.
- Vidéo de démonstration ou captures écran.
- `rapport_final.txt` bien structuré.
- Archive `.zip` du projet.

Concepts maîtrisés à travers ce projet

Domaine	Compétences couvertes
Cybersec Offensive	Recon, scan, exploit, post-exploit
Python avancé	Sockets, API, fichiers, automation
OSINT	Dorking, Shodan, AbuseIPDB, etc.
Web hacking	XSS, SQLi, LFI, Bruteforce
Réseaux	TCP/UDP, ports, reverse shell
Crypto	Base64, Fernet, Stéganographie
Post-exploitation	Keylogger, persistance, screenshot
Evasion	Obfuscation, encodage, faux noms
Exfiltration	Telegram API, zip, encryption
Interface	CLI + Telegram
Structuration	Projet modulaire, arborescence pro
Rapport	Structuré, horodaté, complet

Fortement recommandé

- 🕸️ Interface web locale (`Flask`) pour exécuter les modules
- 💣 Générateur de payload `.exe` avec `pyinstaller`
- 🦹 Mode agent dormant : activation à date/IP précise
- 🧠 IA : suggestions de cible à partir d'un domaine
- 🌍 Intégration de TOR + Proxychains
- 🪥 Honeypot detection automatique

Barème d'évaluation

Critère	Points
OSINT complet et automatisé	100
Scan réseau fiable	100
Web vulnérabilités détectées	100
Exploits systèmes et Shell	100
Exfiltration automatisée	100
Telegram fonctionnel	100

Critère	Points
Crypto + stégo	100
Rapport généré correctement	100
Persistance & post-exploit	100
Style de code, structure, doc	100
Total	**1000**