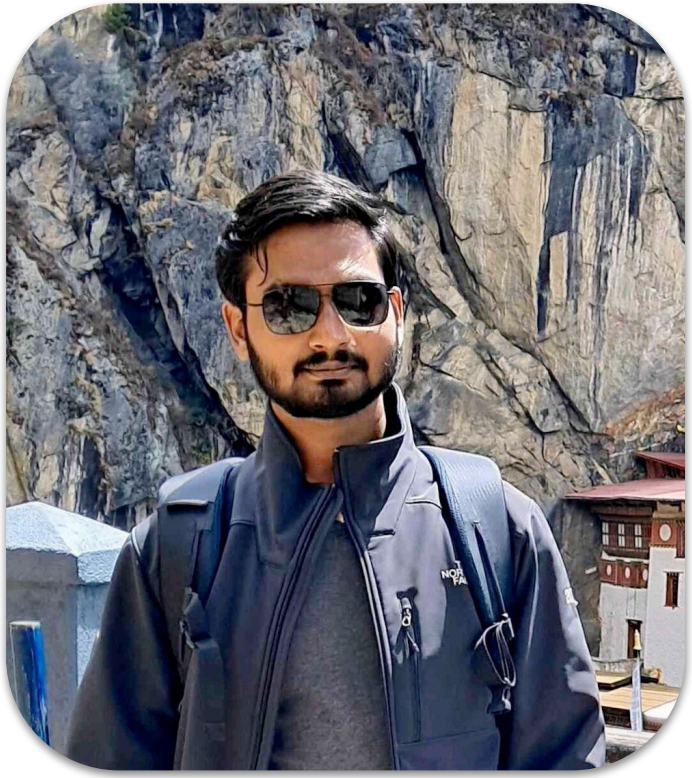
Well, it's just an AWS Account ID

Is it a secret? Really?

About Me



Chandrapal Badshah

Helping Lean Teams Secure AWS @ **Cloud Security Club**

5+ Years of Experience

Cloud & Cloud Native Security

Blogs at badshah.io

- @bnchandrapal

Compromise targetdomain.com
whose entire infra is on AWS

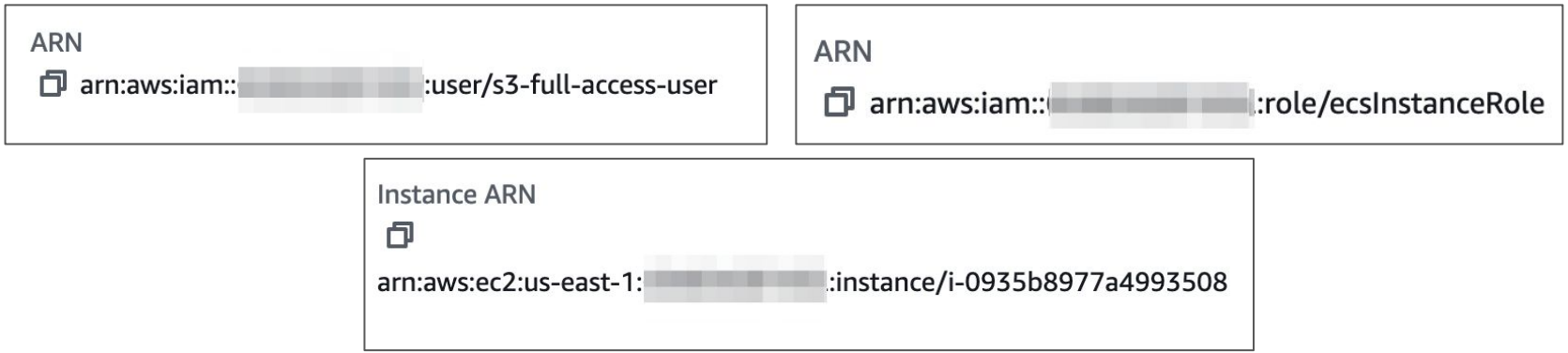
Recon

- Subdomain finder
- Subdomain bruteforcer
- Certificate transparency logs (crt.sh)
- Online Platforms - GitHub, Pastebin
- Scrape Websites, Internet Archive, etc
- S3 Buckets - GreyHat Warfare & osint.sh/buckets/

One recon technique might get you more leverage!

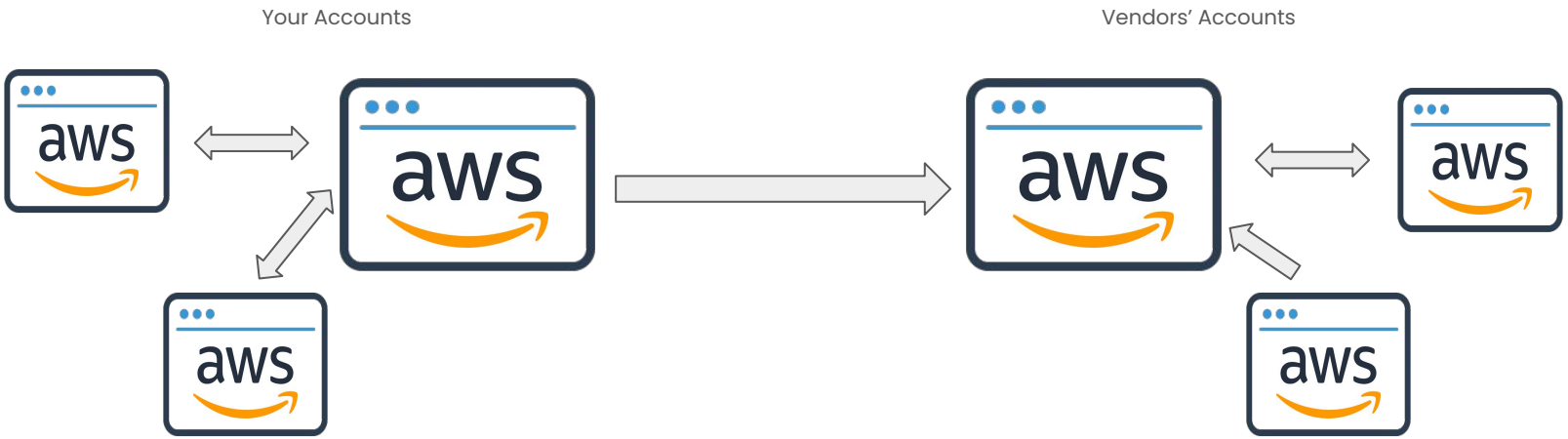
If you don’t know, now you know

- Each AWS account has an account ID (12 digit number)
- Every resource is associated with an AWS account
- Majority of resources get their ARN with account ID in it
- arn:aws:<service>:<region>:<account>:<resourceType>/<resourceName>



Why are Account IDs important?

- To share resources with other accounts
- To create “trust relationships” with other accounts (especially vendors)



Enumerating Account IDs of Companies

Access Keys

- Access Keys disclose AWS Account ID

```
aws sts get-caller-identity
```

- Even if a key is revoked, you can still find it's Account ID

```
aws sts get-access-key-info --access-key-id <KEY>
```

- *Cognito Identity credentials* will also disclose Account ID

DEMO

Find the account ID from leaked creds

```
AKIAZQ3DPS3MCUE5DYLX
xb6cz2KGxZMJT0mfN8sVRKSdW6jy/qxwai4MAjPy
```

S3 Buckets

- Public buckets can fetch you account IDs

  s3-account-search arn:aws:iam::123456789012:role/s3_read s3://my-bucket

- https://github.com/WeAreCloudar/s3-account-search
- Private buckets can also fetch account IDs*
- https://tracebit.com/blog/how-to-find-the-aws-account-id-of-any-s3-bucket

* Looks like AWS has rolled out protection to limit the amount of wildcard chars https://x.com/thiezn_/status/1808477407659069442

DEMO

Find account ID for the following bucket

https://cloudsecclub-bucket.s3.amazonaws.com/65ba6793e4b00910b5ba380a.jpg

S3 Presigned URLs

- Security best practice
- Share private bucket objects to users or allow users to upload objects to private buckets
- Always discloses AWS Access Key ID in the `X-Amz-Credential` param

```
https://presignedurldemo.s3.eu-west-2.amazonaws.com/image.png?X-Amz-
Algorithm=AWS4-HMAC-SHA256&X-Amz-
Credential=AKIAJJWZ7B6WCRGMKFGQ%2F20180210%2Feu-
west-2%2Fs3%2Faws4_request&X-Amz-Date=20180210T171315Z&X-Amz-
Expires=1800&X-Amz-
Signature=12b74b0788aa036bc7c3d03b3f20c61f1f91cc9ad8873e3314255dc479a
25351&X-Amz-SignedHeaders=host
```

Source: https://medium.com/@aidan.hallett/securing-aws-s3-uploads-using-presigned-urls-aa821c13ae8d

DEMO

Do you know the AWS Account ID of GitHub.com?

GitHub

- Leading provider of AWS Account IDs disclosed in IaC code

GitHub

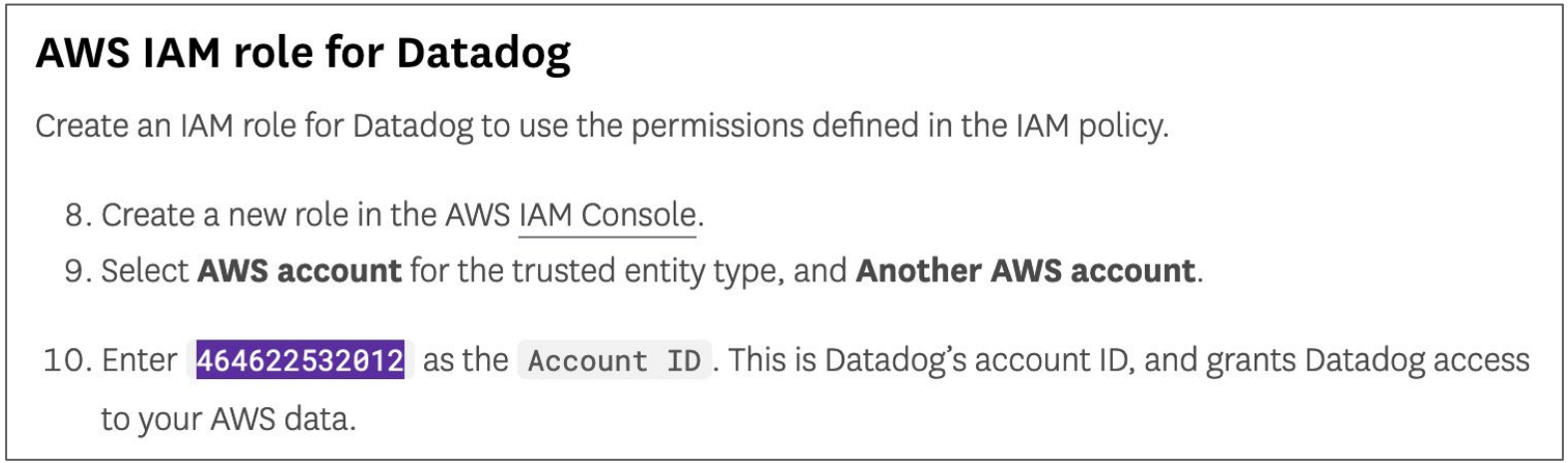
openstreetmap / chef Public

```
[osm-pds]
aws_access_key_id = AKIAZFVRMSDZE2DANIFS
aws_secret_access_key = <%= @credentials["osm-pds"] %>

[osm-pds-upload]
role_arn=arn:aws:iam::630658470130:role/osm-pds-upload-role
source_profile=osm-pds

[osm-osmdbt-state]
aws_access_key_id = AKIASQUXHPE7BNEKJFRQ
aws_secret_access_key = <%= @credentials["osm-osmdbt-state"] %>

[osm-osmdbt-state-upload]
role_arn=arn:aws:iam::173189593406:role/osm-osmdbt-state-upload-role
source_profile=osm-osmdbt-state
```

Source: https://github.com/openstreetmap/chef/blob/master/cookbooks/planet/templates/default/aws-credentials.erb

Documentation

- Product documentations (especially cloud security vendors) expose their Account IDs



AWS IAM role for Datadog

Create an IAM role for Datadog to use the permissions defined in the IAM policy.

8. Create a new role in the AWS IAM Console.
9. Select **AWS account** for the trusted entity type, and **Another AWS account**.
10. Enter `464622532012` as the `Account ID`. This is Datadog's account ID, and grants Datadog access to your AWS data.

Source: https://docs.datadoghq.com/integrations/guide/aws-manual-setup/?tab=roledelegation

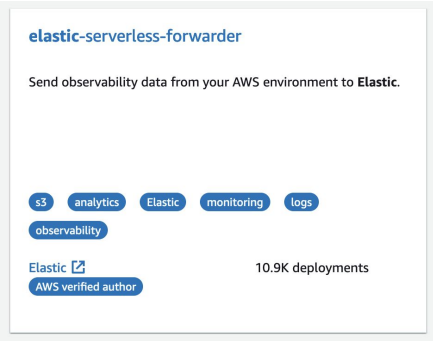
Documentation

Collection of known AWS accounts scraped from documentation:
https://github.com/fwdcloudsec/known_aws_accounts

```yaml
- name: 'Cloudhealth'
  source: 'https://github.com/mozilla/security/blob/master/operations/cloudformation-templates/cloudhealth_iam_role.json'
  accounts: ['454464851268']
- name: 'SegmentIO'
  source: ['https://segment.com/docs/destinations/amazon-s3/', 'https://segment.com/docs/destinations/amazon-kinesis/']
  accounts: ['107630771604', '595280932656']
- name: 'StackDriver'
  source: ['https://web.archive.org/web/20150423044518/https://support.stackdriver.com/customer/portal/articles/1491790-setting-up-stackdri
  accounts: ['314658760392']
- name: 'Zencoder'
  source: 'https://support.brightcove.com/using-zencoder-s3'
  accounts: ['395540211253']
- name: 'Datadog'
  source: ['https://docs.datadoghq.com/integrations/guide/aws-manual-setup/','https://docs.datadoghq.com/integrations/amazon_ec2/']
  accounts: ['464622532012', '865078226113']
- name: 'Cloudability'
  source: ['https://github.com/edrans/tf-aws-iam-cloudability', 'https://developers.cloudability.com/docs/vendor-credentials-end-point']
  accounts: ['165736516723']
```

Other places to find Account IDs or Access Keys

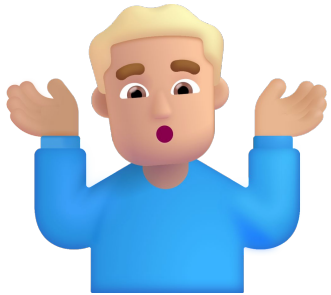
- Source Code
  - GitLab & Bitbucket
  - DockerHub & Public ECR Images
- Error output
  - GitHub, GitLab & Bitbucket Issues
  - Stackoverflow
  - Online forums
- Serverless Application Repositories
- Mobile apps
- SQS Queue URLs




```
ElasticServerlessForwarderEventMacro:
  Type: AWS::Serverless::Application
  Properties:
    Location:
      ApplicationId: arn:aws:serverlessrepo:eu-central-1:267093732750:applications/
      SemanticVersion: 1.14.0
  Metadata:
    SamResourceId: ElasticServerlessForwarderEventMacro
ElasticServerlessForwarderApplication:
  Type: AWS::Serverless::Application
  Properties:
    Location:
      ApplicationId: arn:aws:serverlessrepo:eu-central-1:267093732750:applications/
      SemanticVersion: 1.14.0
```

Where are we now?

Looked at so many places → AWS Account ID →



Leveraging Account IDs of Companies

Bruteforce for IAM Entities

- You can find IAM principals (users and roles) from account ID
- Can help with phishing
- Script: https://github.com/dagrz/aws_pwn/blob/master/reconnaissance/validate_iam_principals.py
- Wordlist: https://github.com/righteousgambit/quiet-riot/blob/main/wordlists/service-linked-roles.txt

```
python3 validate_iam_principals.py -i service-linked-roles.txt -a <accountID>
```

Enumerate Security Services

- AWS can’t access your resources by default
- AWS Security Services create Service Linked IAM Roles to access your resources
- External Cloud Security vendors require you to create IAM Roles
- **Existence of an IAM Role** implies that the service **may or may not be enabled** right now. **Non-existence of the IAM Role** implies the service is **definitely not enabled**.

Demo

What services are being used in target’s accounts?

Target Account ID: 654654281432

Enumerate Public Snapshots

- Many public database snapshots can be queried using Account ID
  - EBS Snapshots (https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Snapshots:visibility=public;sort=snapshotId)
  - RDS Snapshots (https://us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#snapshots-list:tab=public)
  - DocumentDB Snapshots (https://us-east-1.console.aws.amazon.com/docdb/home?region=us-east-1#snapshots)
  - Neptune Snapshots (https://us-east-1.console.aws.amazon.com/neptune/home?region=us-east-1#snapshots:type=public)

DEMO

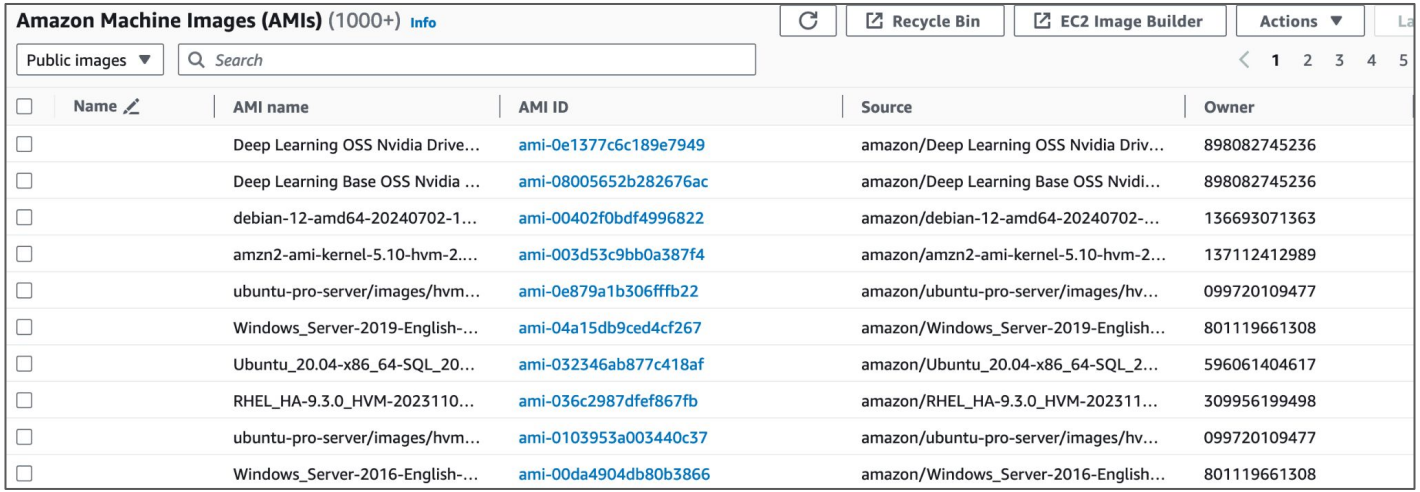
Find public snapshots in target’s accounts?

Target Account ID: 654654281432

Public AMIs

Public AMIs can be queried using Account IDs

https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Images:visibility=public-images

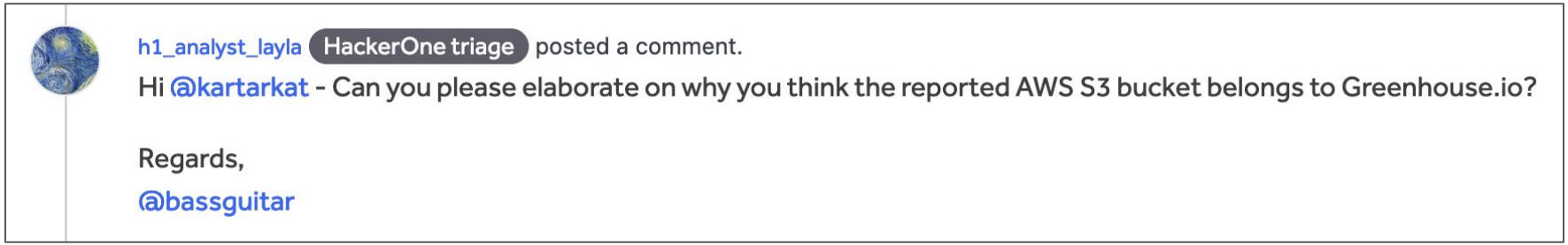


Name	AMI name	AMI ID	Source	Owner
	Deep Learning OSS Nvidia Drive...	ami-0e1377c6c189e7949	amazon/Deep Learning OSS Nvidia Driv...	898082745236
	Deep Learning Base OSS Nvidia ...	ami-08005652b282676ac	amazon/Deep Learning Base OSS Nvidi...	898082745236
	debian-12-amd64-20240702-1...	ami-00402f0bdf4996822	amazon/debian-12-amd64-20240702-...	136693071363
	amzn2-ami-kernel-5.10-hvm-2....	ami-003d53c9bb0a387f4	amazon/amzn2-ami-kernel-5.10-hvm-2...	137112412989
	ubuntu-pro-server/images/hvm...	ami-0e879a1b306fffb22	amazon/ubuntu-pro-server/images/hv...	099720109477
	Windows_Server-2019-English-...	ami-04a15db9ced4cf267	amazon/Windows_Server-2019-English...	801119661308
	Ubuntu_20.04-x86_64-SQL_20...	ami-032346ab877c418af	amazon/Ubuntu_20.04-x86_64-SQL_2...	596061404617
	RHEL_HA-9.3.0_HVM-2023110...	ami-036c2987dfef867fb	amazon/RHEL_HA-9.3.0_HVM-202311...	309956199498
	ubuntu-pro-server/images/hvm...	ami-0103953a003440c37	amazon/ubuntu-pro-server/images/hv...	099720109477
	Windows_Server-2016-English-...	ami-00da4904db80b3866	amazon/Windows_Server-2016-English...	801119661308

Other Interesting Scenarios

Does this ABC resource belongs to XYZ Company?

- If the leaked resource's owner ID == target domain's account ID => confirmed resource leak



Source: https://hackerone.com/reports/819278

Detect CanaryTokens

- If you know an account is used for canary tokens, avoid using it
- Trufflehog has inbuilt support to detect canarytokens.org accounts

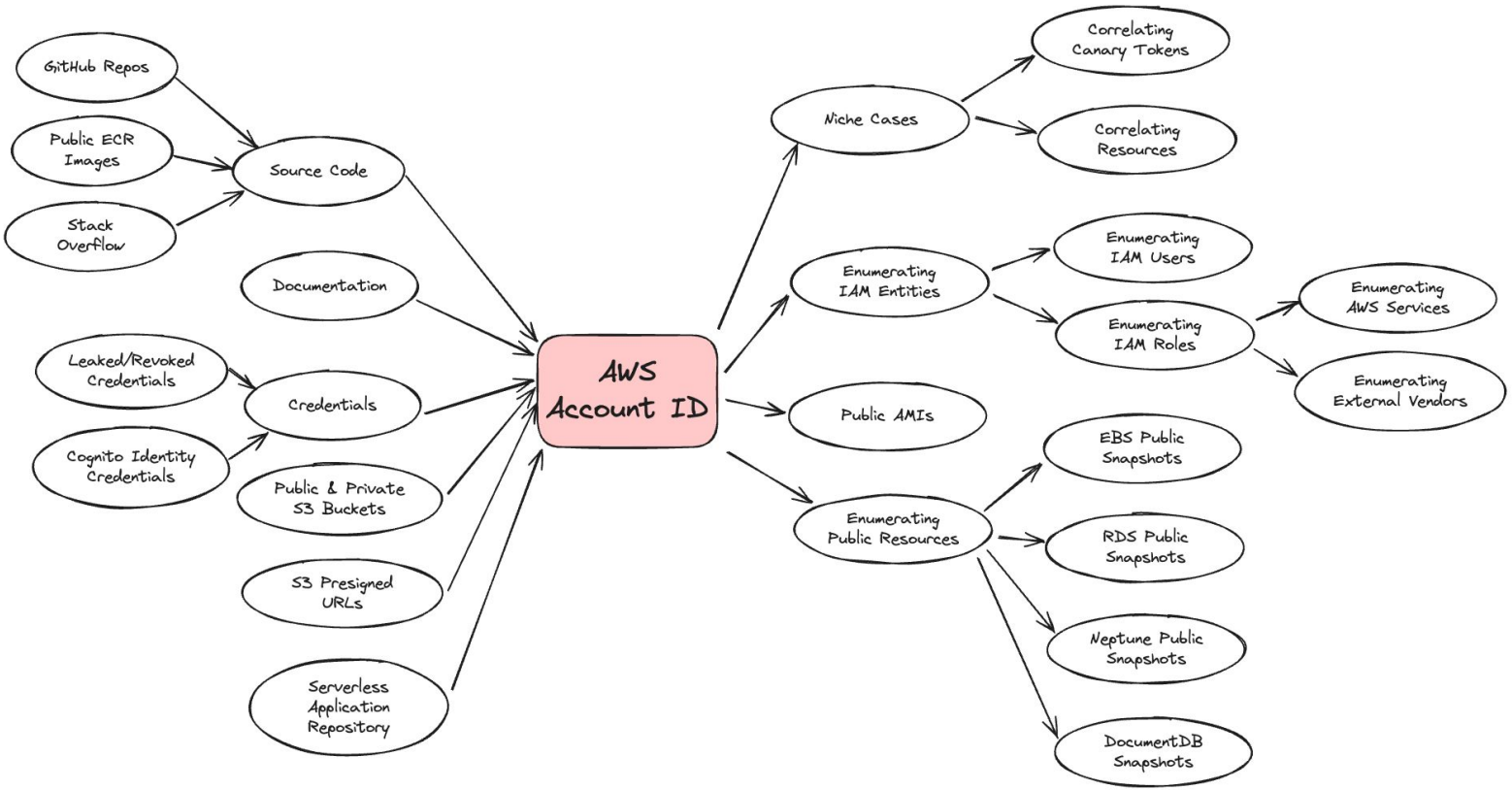
Source: https://trufflesecurity.com/blog/canaries

Where are we now?

Looked at so many places → AWS Account ID → So much more information without compromising anything

Remember, it’s just a ~~*bug*~~ feature

Mindmap


GitHub Repos
Public ECR Images
Stack Overflow
Source Code
Documentation
Leaked/Revoked Credentials
Credentials
Cognito Identity Credentials
Public & Private S3 Buckets
S3 Presigned URLs
Serverless Application Repository
AWS Account ID
Niche Cases
Correlating Canary Tokens
Correlating Resources
Enumerating IAM Entities
Enumerating IAM Users
Enumerating IAM Roles
Enumerating AWS Services
Enumerating External Vendors
Public AMIs
Enumerating Public Resources
EBS Public Snapshots
RDS Public Snapshots
Neptune Public Snapshots
DocumentDB Snapshots

Is AWS Account ID Sensitive?

My Short Answer

The Account ID is useless and not a direct weakness in itself.

It’s sensitivity arises from the fact it can help fetch and/or correlate resources and also gather information that can be used in other attacks.

What I do know is it’s a **POWERFUL** technique in your recon process.

Resources

- https://rhinosecuritylabs.com/aws/assume-worst-aws-assume-role-enumeration/
- https://github.com/fwdcloudsec/known_aws_accounts
- https://medium.com/@TalBeerySec/a-short-note-on-aws-key-id-f88cc4317489
- https://blog.plerion.com/aws-account-ids-are-secrets/
- https://github.com/dagrz/aws_pwn/blob/master/reconnaissance/validate_iam_principals.py
- https://www.youtube.com/watch?v=iMYbne-tD20

Thank You

Any Questions?

Securing AWS: Strategies for Lean Teams