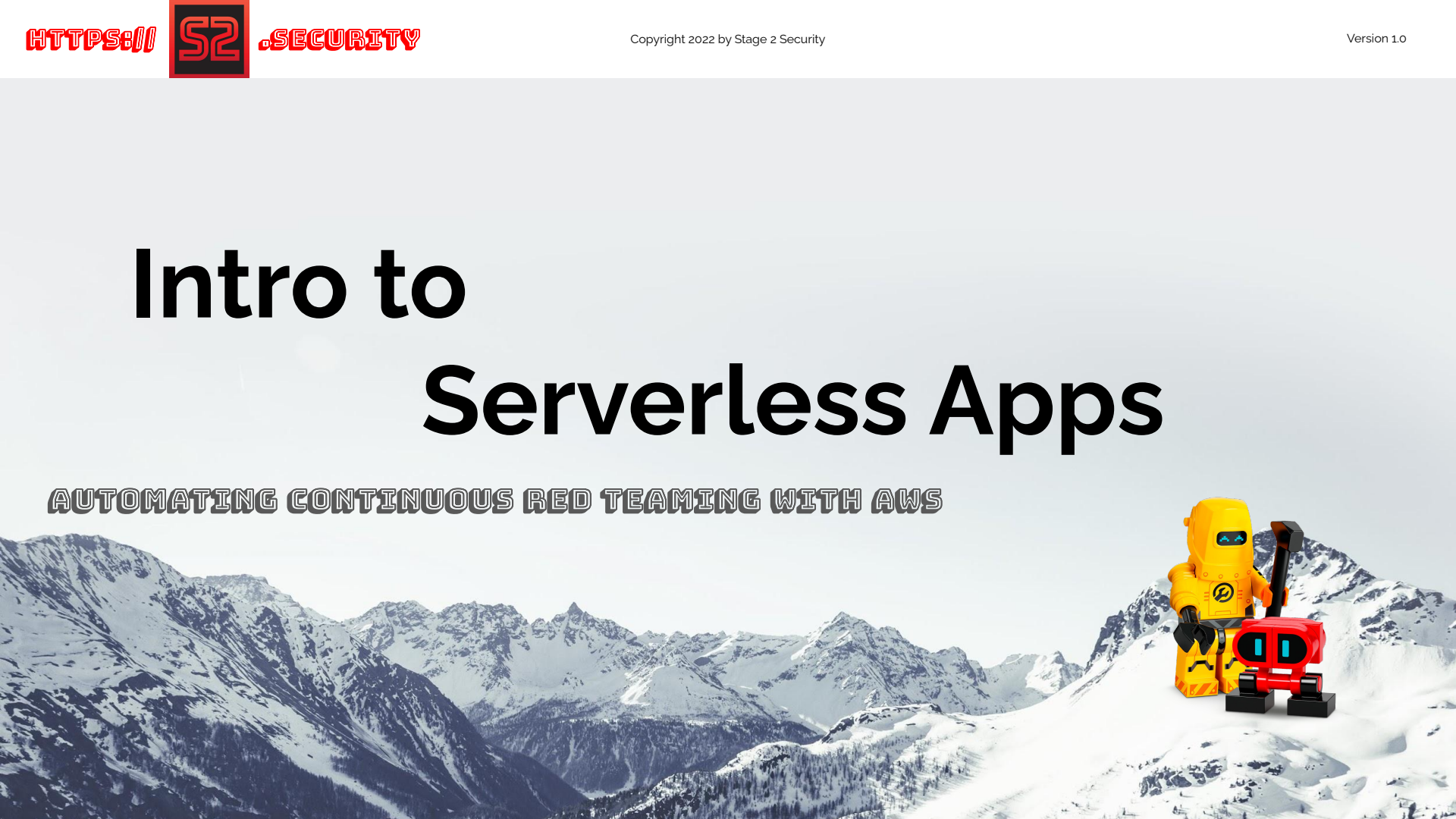
S2 STAGE 2 SECURITY

Outplay Your Adversary!

Bryce Kunz // @TweekFawkes

Intro to Serverless Apps

AUTOMATING CONTINUOUS RED TEAMING WITH AWS



TODO: Agenda

Bryce Kunz
@TweekFawkes

- Who Am I?
- Go From Zero to Cloud Admin!
- SSO Tokens & Browser Cookies
- Graph Database Technologies
- Overview: Bridging to On-Prem
- Attacking the Bridges!
- One More Thing!

Bryce Kunz; @TweekFawkes

Defense
DHS SOC

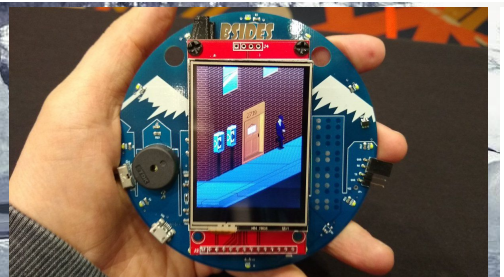
Offense
NSA

Red Team
Adobe
Digital Exp. (DX)

S2 STAGE 2 SECURITY

Services

- Hack (Pentest)
- Hunt (Splunk ES)
- Train (Cloud Sec.)



BSIDES SALT LAKE CITY

Hardware Badge by

@professor_plum

Friday Dec. 16th 2022!

Sandy UT

Conference Center at Miller Campus

https://BSidesSLC.org

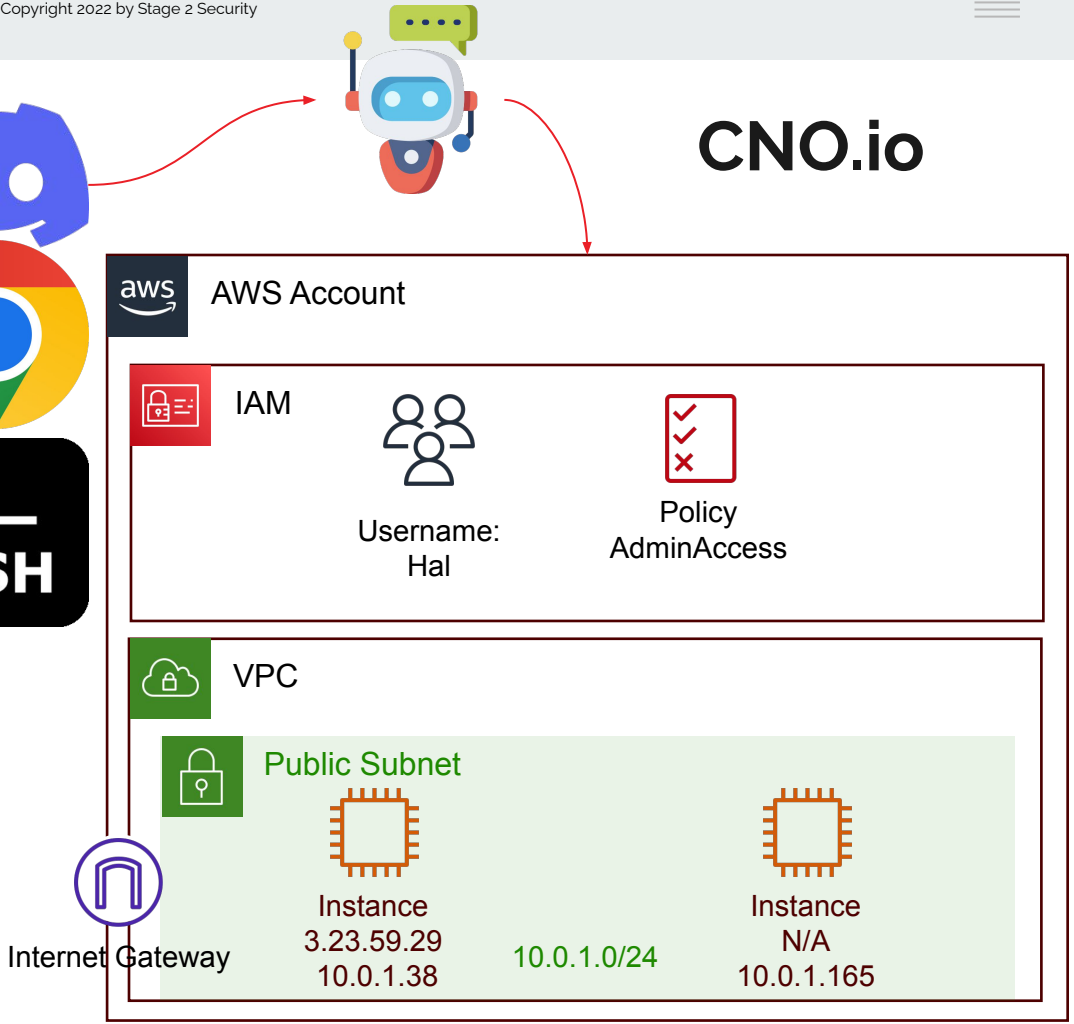
Lab Access

OVERVIEW

Deploy Lab

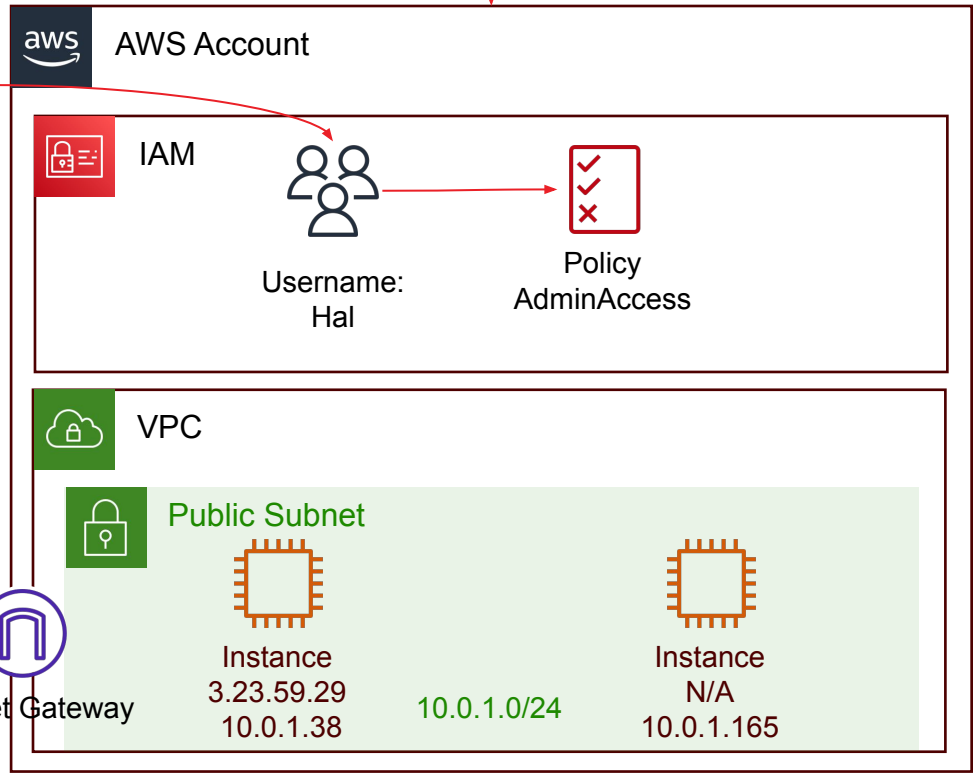
Steps:

- https://cno.io
-
- “SaintCon2022”
- “Intro to Severless”
- “Join Discord”
-
- #itsa20221027
- !itsa_deploy
- !itsa_login



Access AWS

CNO.io



https://console.aws.amazon.com/

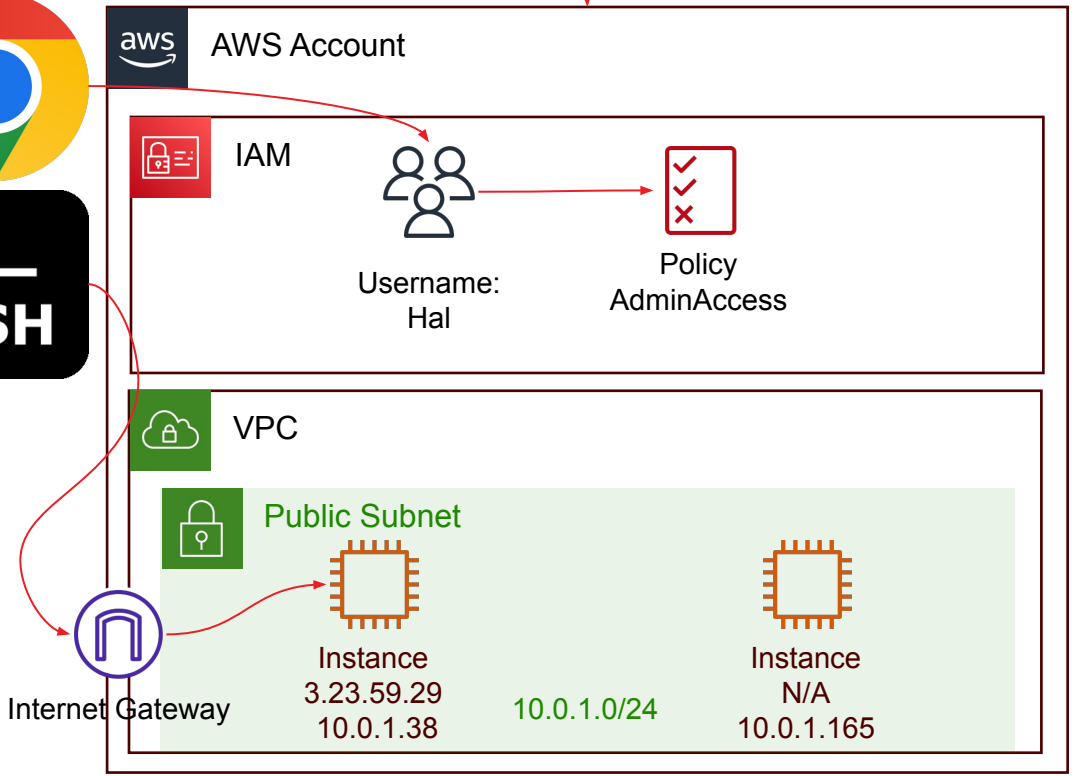
- Account #: (From Bot)
- Username: Hal
- Password: (From Bot)

SSH Access

Steps:

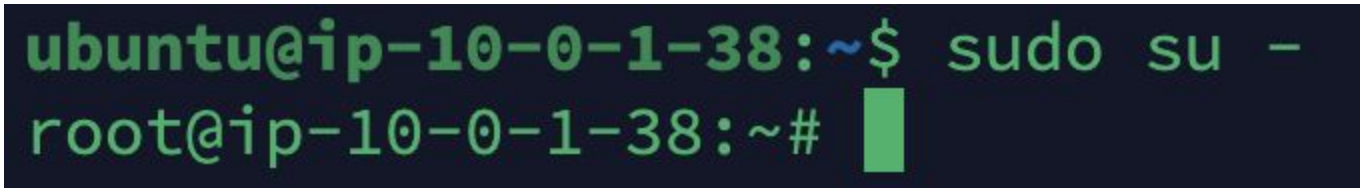
- https://cno.io
-
- “SaintCon2022”
- “Intro to Serverless”
- “SSH Key”
-
- IP: <Find_In_EC2>
- Username: ubuntu
- Password: <SSH_Key>

CNO.io

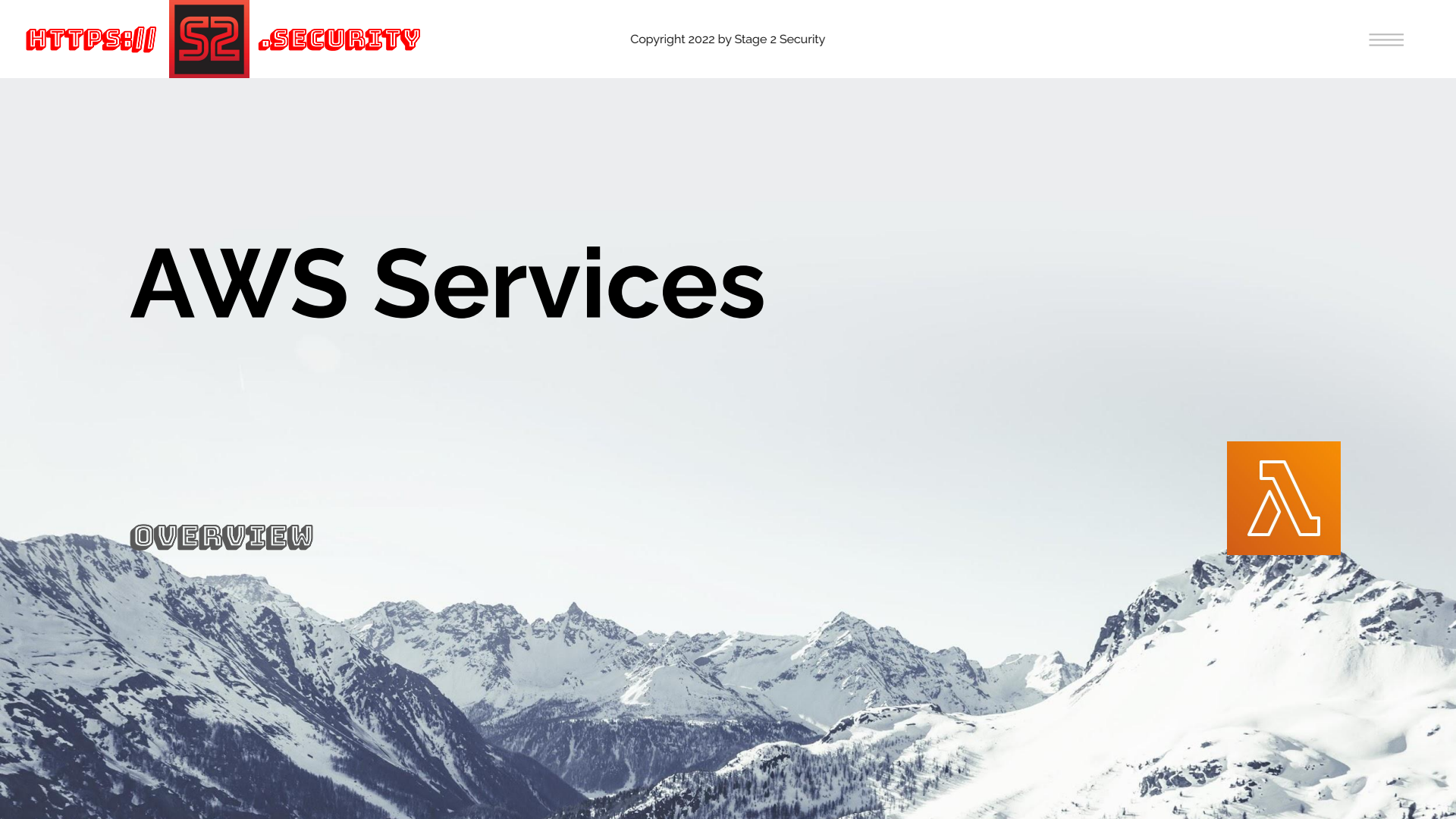


sudo su -

For Root Access



AWS Services



OVERVIEW

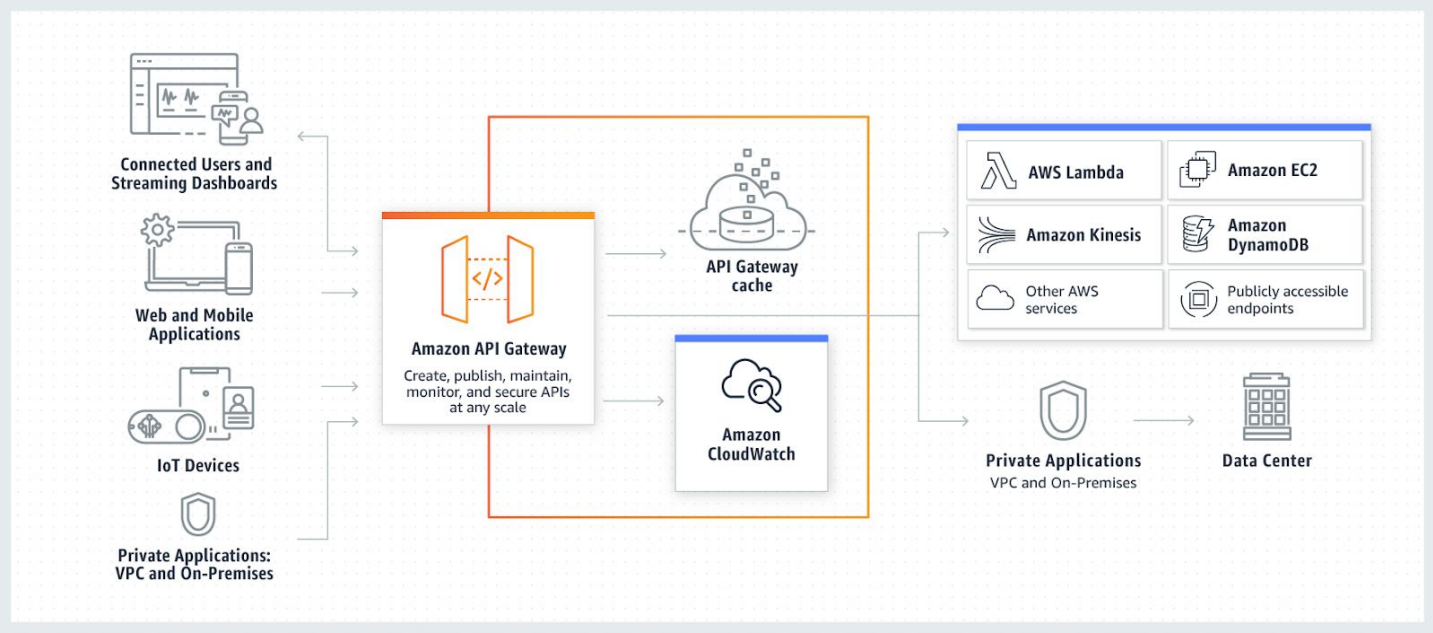
Cloud9

Cloud9 IDE is an Online IDE

It supports multiple programming languages, including:

- C, C++
- PHP
- Ruby
- Perl
- Python
- JavaScript with Node.js
- Go

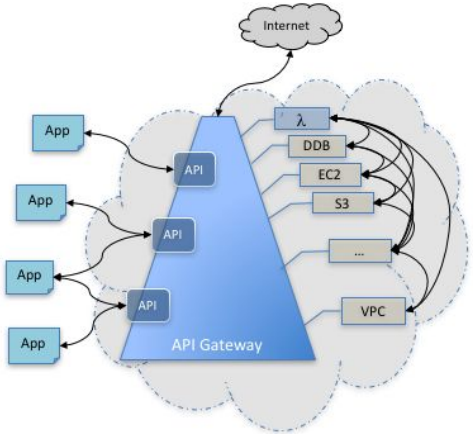
API Gateway



API Proxy (HTTP or REST) as a managed service

https://aws.amazon.com/api-gateway/

API Gateway Overview



API Gateway is a managed service that creates APIs at scale

- API that acts as a “front door” for applications to access data, business logic, or functionality from your back-end services,
    - Such as … code running on AWS Lambda, or any web application.

API Gateway can be considered a backplane in the cloud to connect AWS services and other public or private websites.

Provides consistent RESTful application programming interfaces (APIs) for web applications to access AWS services.

API Gateway Types

HTTP APIs are designed for low-latency, cost-effective integrations with AWS services

e.g. Lambda, and HTTP endpoints

REST APIs currently offer more features, and full control over API requests and responses.

The following tables summarize core features that are available in HTTP APIs and REST APIs.

Authorizers	HTTP API	REST API
AWS Lambda	✓	✓
IAM	✓	✓
Amazon Cognito	✓ *	✓
Native OpenID Connect / OAuth 2.0	✓	

Security	HTTP API	REST API
Client certificates		✓
AWS WAF		✓
Resource policies		✓

https://docs.aws.amazon.com/apigateway/latest/developerguide/http-api-vs-rest.html

Lambda

AWS Scripts as a managed service

https://aws.amazon.com/lambda/

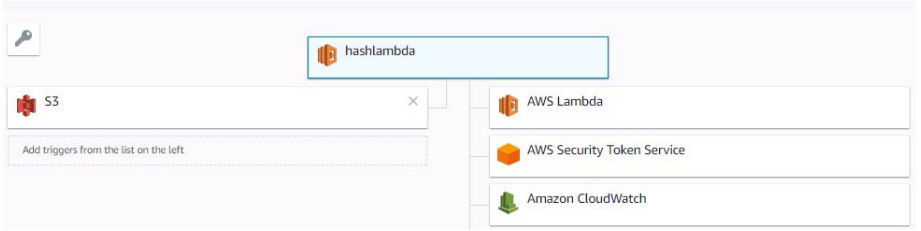
Lambda Overview

AWS Lambda is a serverless compute service

- Runs your code in response to events and automatically manages the underlying compute resources for you.
- Lambda can automatically run code in response to multiple events, such as:
  - HTTP requests via Amazon API Gateway,
  - PutObject calls to Amazon S3 Buckets,
  - Table updates in Amazon DynamoDB, and
  - State transitions in AWS Step Functions.
- You can use AWS Lambda to extend other AWS services with custom logic

Lambda Runtimes & Environment

- Supports many native runtimes
  - Python2.7 & 3, Java, Ruby, Go, C# +

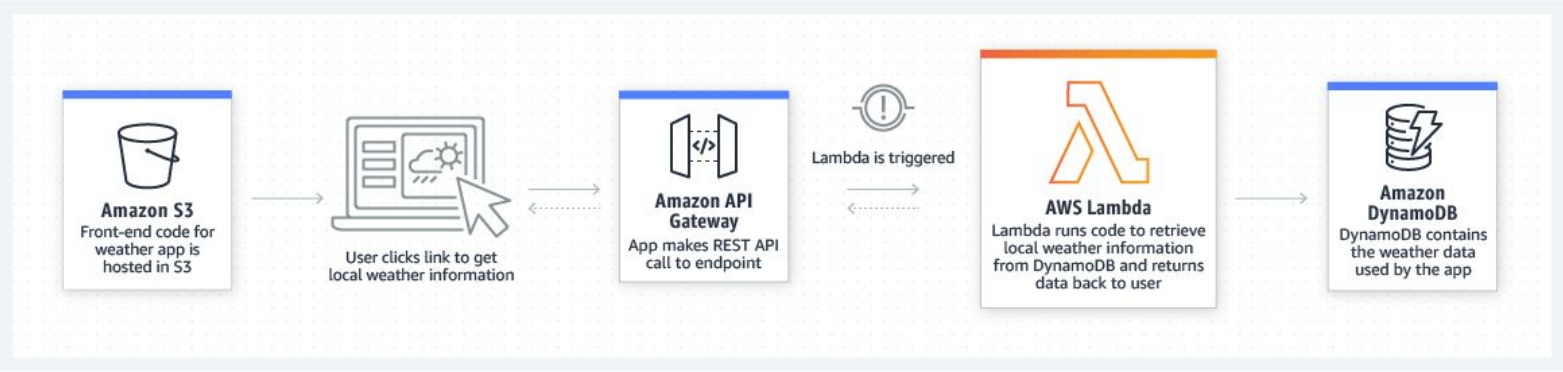


- AWS SDKs built in (e.g. boto3)
- Role Credentials placed in Environment Variables
- Max 15 Minute Runtime

lambda_function

```python
import boto3
import hashlib

def lambda_handler(event, context):
    bucketname = event['Records'][0]['s3']['bucket']['name']
    keyname = event['Records'][0]['s3']['object']['key']

    s3 = boto3.client("s3")
    obj = s3.get_object(Bucket=bucketname, Key=keyname)
    obj_body = obj['Body'].read()
    s = hashlib.sha256()
    s.update(obj_body)
    h = s.hexdigest()

    dynamodb = boto3.resource('dynamodb')
    table = dynamodb.Table('hash-myhashdb')
    response = table.put_item(
       Item={
            'objectid': keyname,
            'hash': h,
            'hashtype': "sha256"
        }
    )
```

Lambda Common Web App Architecture



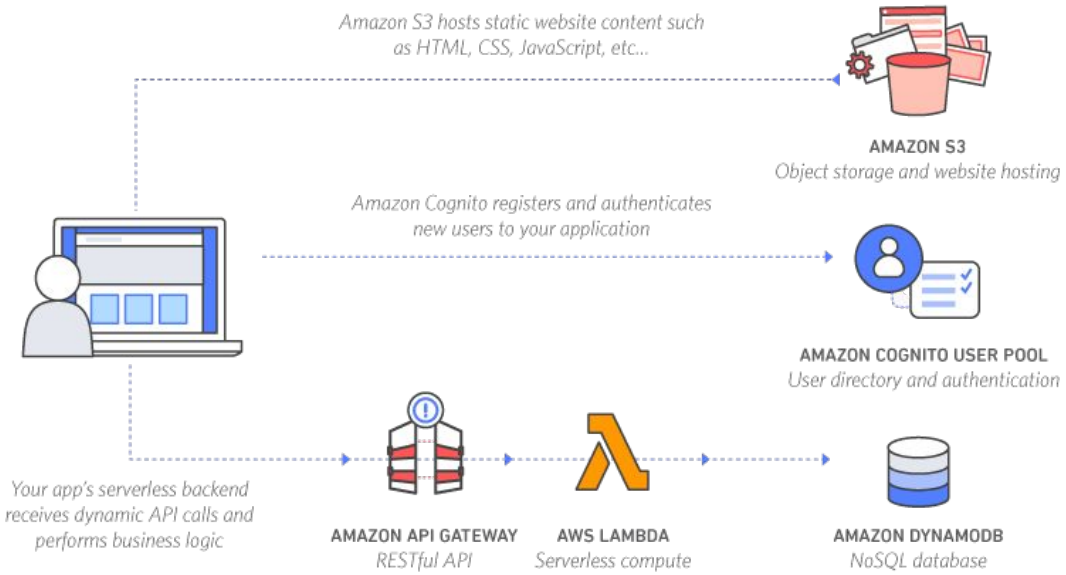
https://aws.amazon.com/lambda/

CloudFormation

Amazon Services Setup

https://aws.amazon.com/cloudformation/

AWS Serverless Application Model (SAM)

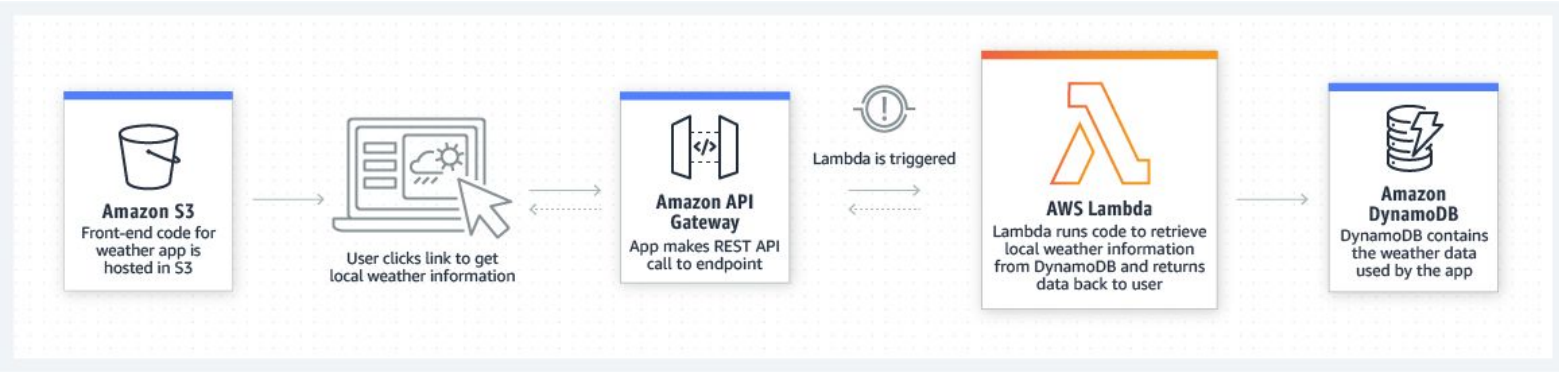


https://aws.amazon.com/serverless/build-a-web-app/

AWS Serverless Application Model (SAM)

Infrastructure as Code

Based on CloudFormation, but much simpler to implement common Serverless Application Models...



https://aws.amazon.com/serverless/sam/

AWS Serverless Application Model (SAM)

The Serverless Application Model is commonly referred to as "**SAM**" in AWS documentation

SAM is NOT an AWS Service, it's more similar to Zappa or Terraform

New Resource Types with SAM:

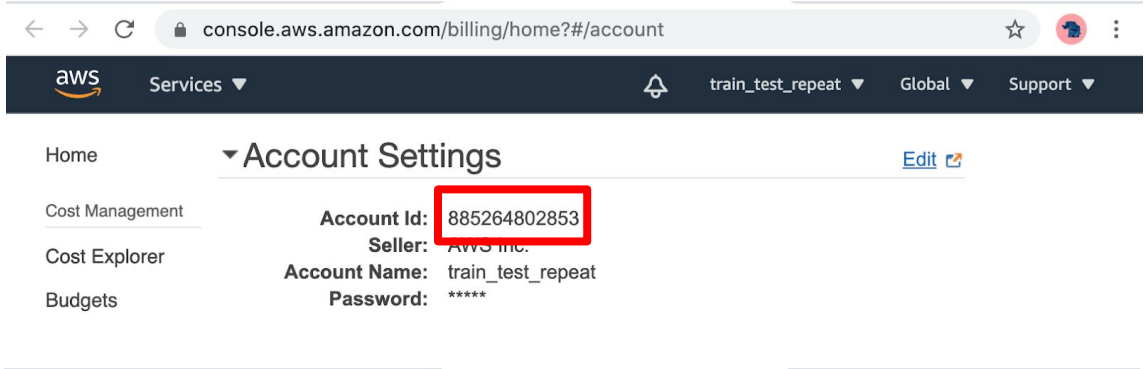
- AWS::Serverless::Function -> AWS Lambda Functions
- AWS::Serverless::Api -> AWS API Gateway APIs
- AWS::Serverless::SimpleTable -> AWS Dynamo DB Tables

https://aws.amazon.com/serverless/sam/

Use Case # 1
Discover Account IDs

Discovering semi-sensitive information (e.g. AWS Account IDs) via analyzing responses from various Cloud services.

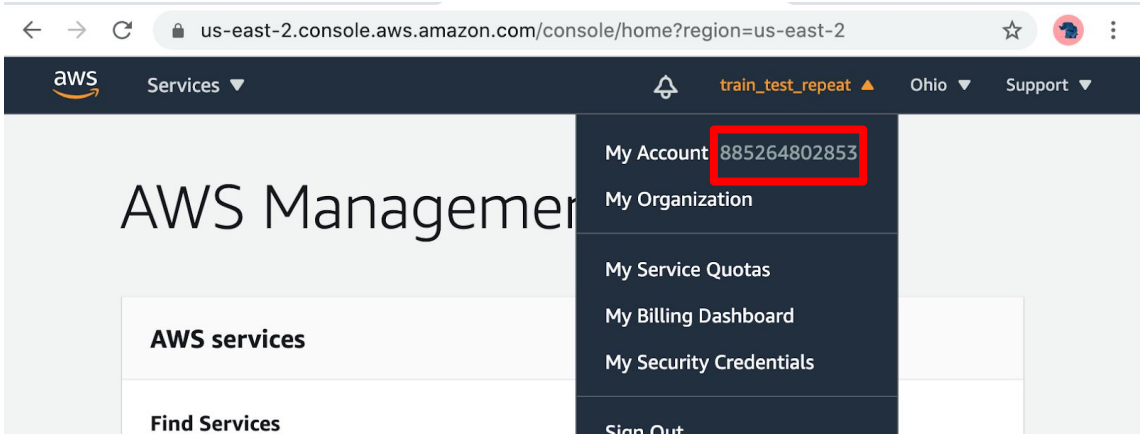
AWS Account IDs



Every AWS account is given a unique ID

Account IDs are 12 digit numbers

e.g. 885264802853

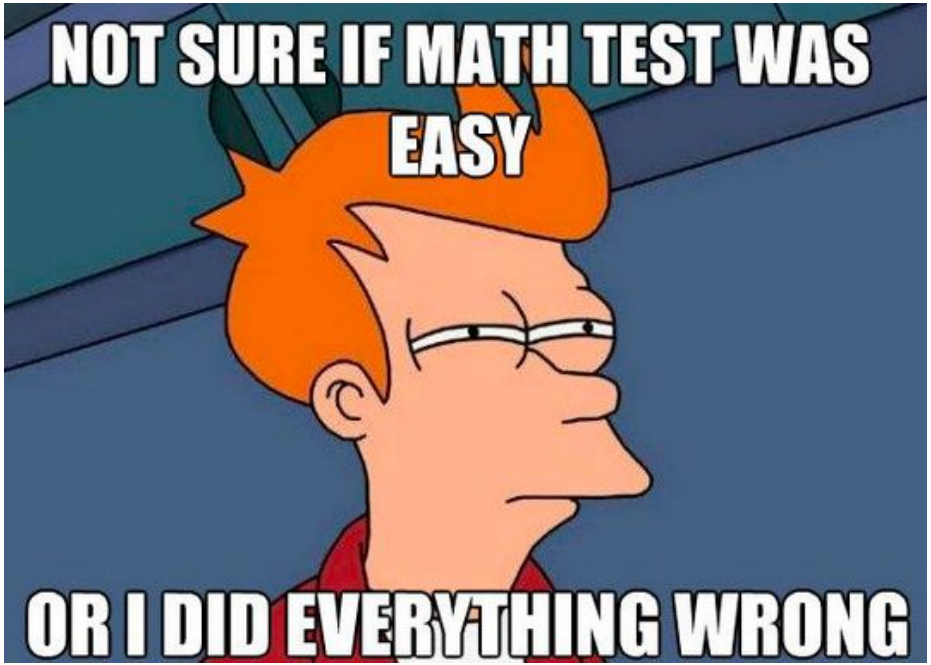


Semi-Sensitive?

Usually kept private...

But historically, considered to be equivalent to a username disclosure

AWS Account IDs - The Math



Account IDs are 12 digit numbers

- e.g. 885264802853

10 possible digits (0,1,2,3,4,5,6,7,8,9)

- 12 digits long

10^12 means...

- 1 Trillion! Possible Account IDs

AWS Account IDs - Should They be Protected?

Many vendors share their Account IDs:

https://github.com/duo-labs/cloudmapper/blob/main/vendor_accounts.yaml

To enable clients to use their services

Should They be Protected?

Tech industry is saying…

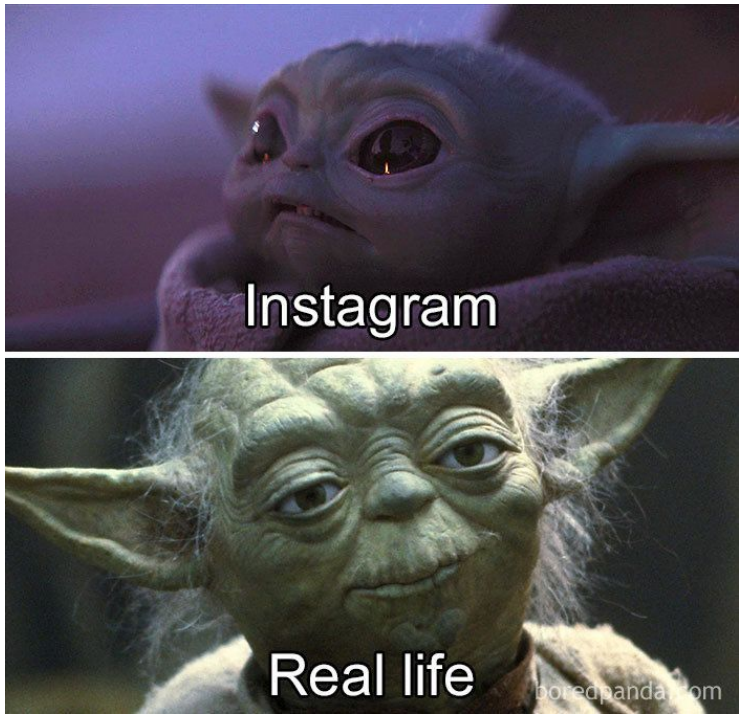
NO 😎

Including, but not limited to:

- Rackspace
- TrendMicro
- Redlock
- Sumo Logic
- Bridgecrew
- Onelogin
- Rapid7
- Threat Stack
- Lucidchart
- Palo Alto Networks
- Tenable

etc...

Known Attacks via Knowing Account IDs



AWS error messages disclose whether a role exists or not, w/ a given Account ID

Role Names can disclose:

- AWS Services being used
- Software & Technologies being used
- Names of IAM users (social engineering)
- 3rd party integrations being used (Okta, Datadog, Cloudsploit, etc.)

Once roles are enumerated, one can try to assume any open roles and pilfer the role credentials.

Same attack vector applies for IAM Usernames

https://rhinosecuritylabs.com/aws/assume-worst-aws-assume-role-enumeration/
https://rhinosecuritylabs.com/aws/aws-iam-user-enumeration/
https://github.com/dagrz/aws_pwn/blob/master/reconnaissance/validate_iam_principals.py

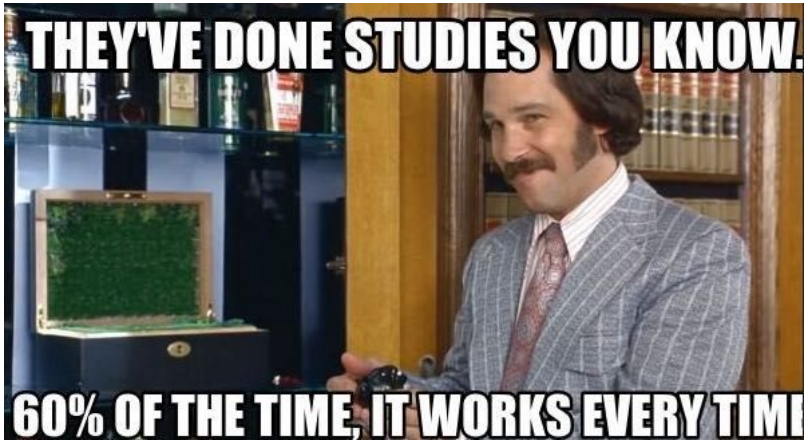
Other Attacks via Knowing Account IDs

This is the way!

Areas for Future Research:

- Discovering overly privileged topics and/or queues with guessable names

Known Ways to Discover Account IDs



Including, but not limited to:

- Compromise a resource (e.g. Lambda, etc.)
- Public resources (e.g. Snapshots, AMI, etc.)
- Source Code Review (e.g. GitHub)
- Error Messages from Services
- Screenshots and/or Documentation
- Forums and/or Discussion Boards
- etc.

Known Ways to Discover Account IDs

```
https://[account].signin.aws.amazon.com/
```

```
1. HTTP/1.1 404 Not Found
2. HTTP/1.1 302 Found
```

By default, login URL for the web console, includes the Account ID as a subdomain.

Subdomain has different HTTP response codes based on if login page exists

NOTE: URL can be changed by the owner of the AWS, and is done so somewhat frequently

https://github.com/dagrz/aws_pwn/blob/master/miscellanea/Kiwicon%202016%20-%20Hacking%20AWS%20End%20to%20End.pdf

Known Ways to Discover Account IDs

VirusTotal Results:

```
http://www.virustotal.com/vtapi/domain/report?
 domain=signin.aws.amazon.com
```

```
259353407677.signin.aws.amazon.com
431429821356.signin.aws.amazon.com
vodafone-uk.signin.aws.amazon.com
```

https://github.com/dagrz/aws_pwn/blob/master/miscellanea/Kiwicon%202016%20-%20Hacking%20AWS%20End%20to%20End.pdf

Known Ways to Discover Account IDs

VirusTotal has updated the API now to v2, which requires a slightly different syntax to query

```python
import requests
import time
import json

sApiKey = '...REDACTED...'
sUserAgent = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.36'
sDomainName = 'signin.aws.amazon.com'

sUrl = 'https://www.virustotal.com/vtapi/v2/domain/report?apikey=' + sApiKey + '&domain=' + sDomainName

try:
    headers = {
        "User-Agent": sUserAgent
    }
    r = requests.get(sUrl, allow_redirects=False, headers=headers) # verify=False, timeout=5

    sTimeStamp = str(time.strftime('%Y%m%d%H%M%S'))

    f = open('api_' + sTimeStamp + '.text', 'w')
    f.write(r.text)
    f.close()

    with open('api_' + sTimeStamp + '.json', 'w') as f:
        json.dump(r.json(), f)

    dResponse = r.json()
    lSubdomains = dResponse['subdomains']
    with open('api_' + sTimeStamp + '.list', 'w') as f:
        for item in lSubdomains:
            f.write("%s\n" % item)

except requests.RequestException as e:
    print(e)
```

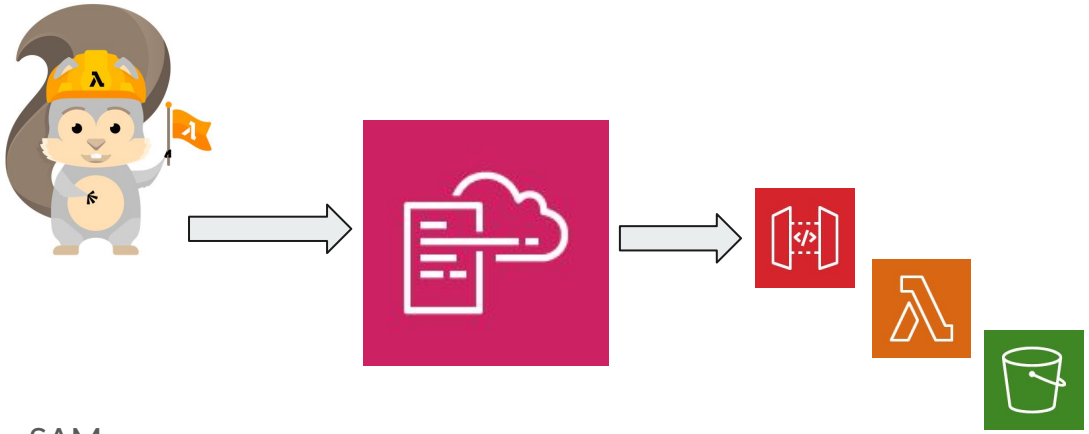
https://github.com/Stage2Sec/CaptureTheCloud/blob/master/find_aws_account_ids_virustotal.py

Other Ways to Discover Account IDs

Other OSINT Sources:

- Passive Total <- This Works Better for Us than VirusTotal
- Amass <- Always Awesome
- etc.

Leverage SAM to Create...



SAM:

- template.yaml
- app.py
- etc.

SAM Template

SAM Template:

- API Gateway
  - GET /hello
- Lambda
  - Python 3.7
- Attach Policy to Enable:
  - S3 Bucket Access
- Set Timeout to Max:
  - 15 Minutes (900 Seconds)

app.py | template.yaml

```yaml
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Description: >
  AWS

  Sample SAM Template for AWS

# More info about Globals: https://github.com/awslabs/serverless-application-model/blob/master/docs/globals.rst
Globals:
  Function:
    Timeout: 900

Resources:
  HelloWorldFunction:
    Type: AWS::Serverless::Function # More info about Function Resource: https://github.com/awslabs/serverless-applicatio
    Properties:
      CodeUri: hello_world/
      Handler: app.lambda_handler
      Runtime: python3.7
      Policies:
        - AmazonS3FullAccess
      Events:
        HelloWorld:
          Type: Api # More info about API Event Source: https://github.com/awslabs/serverless-application-model/blob/mast
          Properties:
            Path: /hello
            Method: get

Outputs:
  # ServerlessRestApi is an implicit API created out of Events key under Serverless::Function
  # Find out more about other implicit resources you can reference within SAM
  # https://github.com/awslabs/serverless-application-model/blob/master/docs/internals/generated_resources.rst#api
  HelloWorldApi:
    Description: "API Gateway endpoint URL for Prod stage for Hello World function"
    Value: !Sub "https://${ServerlessRestApi}.execute-api.${AWS::Region}.amazonaws.com/Prod/hello/"
```

Lambda Application

Python App Source:

- Generate an ID
- Secrets Library

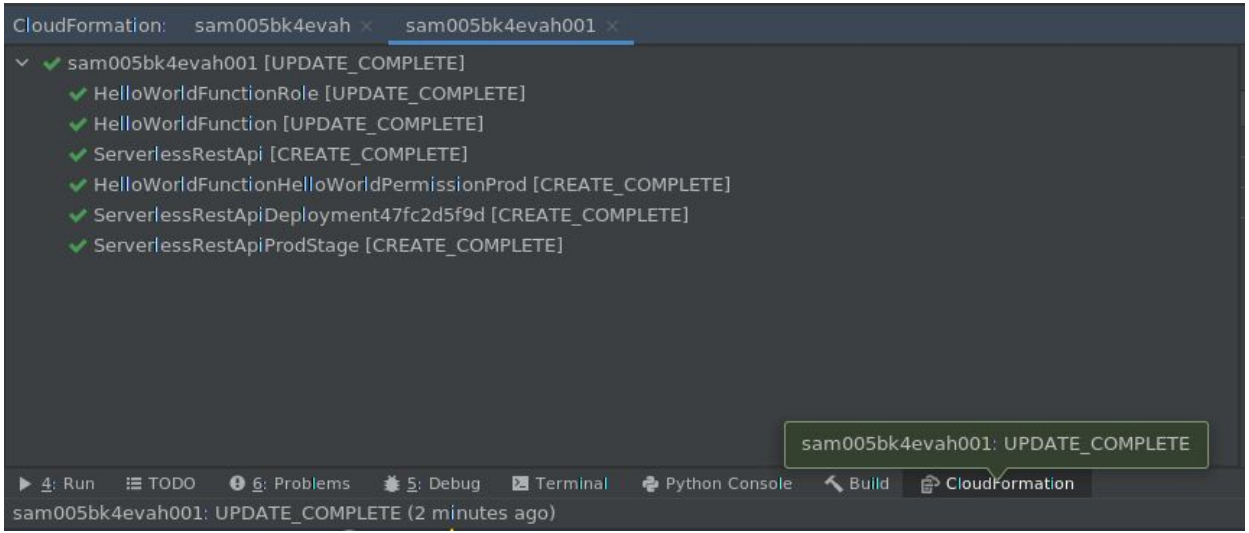
- Try w/ Web Console
- Requests Library

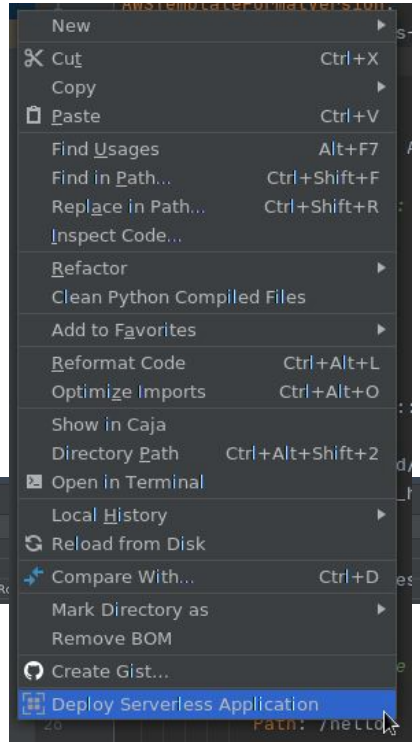
- Analyze Response
- Write Result to S3

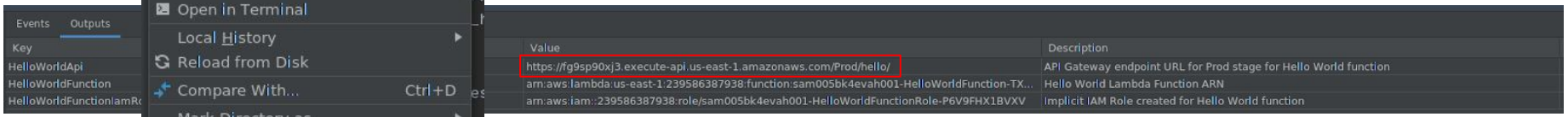
app.py

```python
import secrets
import json
import requests
import time
import boto3

def lambda_handler(event, context):
    sReturn = ""

    iCount = 0
    iLimit = 33

    iFourOhFourCount = 0
    lFourOhFour = []

    iThreeOhTwoCount = 0
    lThreeOhTwo = []

    iOtherCount = 0
    lOther = []

    while iCount < iLimit:
        sRandomNumber = str(secrets.randbelow(1000000000000)).zfill(12)
        sUrl = 'https://' + sRandomNumber + '.signin.aws.amazon.com'
        iCount = iCount + 1

        try:
            headers = {
                "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.36"
            }

            requestsResponse = requests.get(sUrl, allow_redirects=False, headers=headers, verify=False, timeout=5)
            sResponseStatusCode = str(requestsResponse.status_code)
            sLine = sResponseStatusCode + "-" + sUrl
```

SAM Deploy

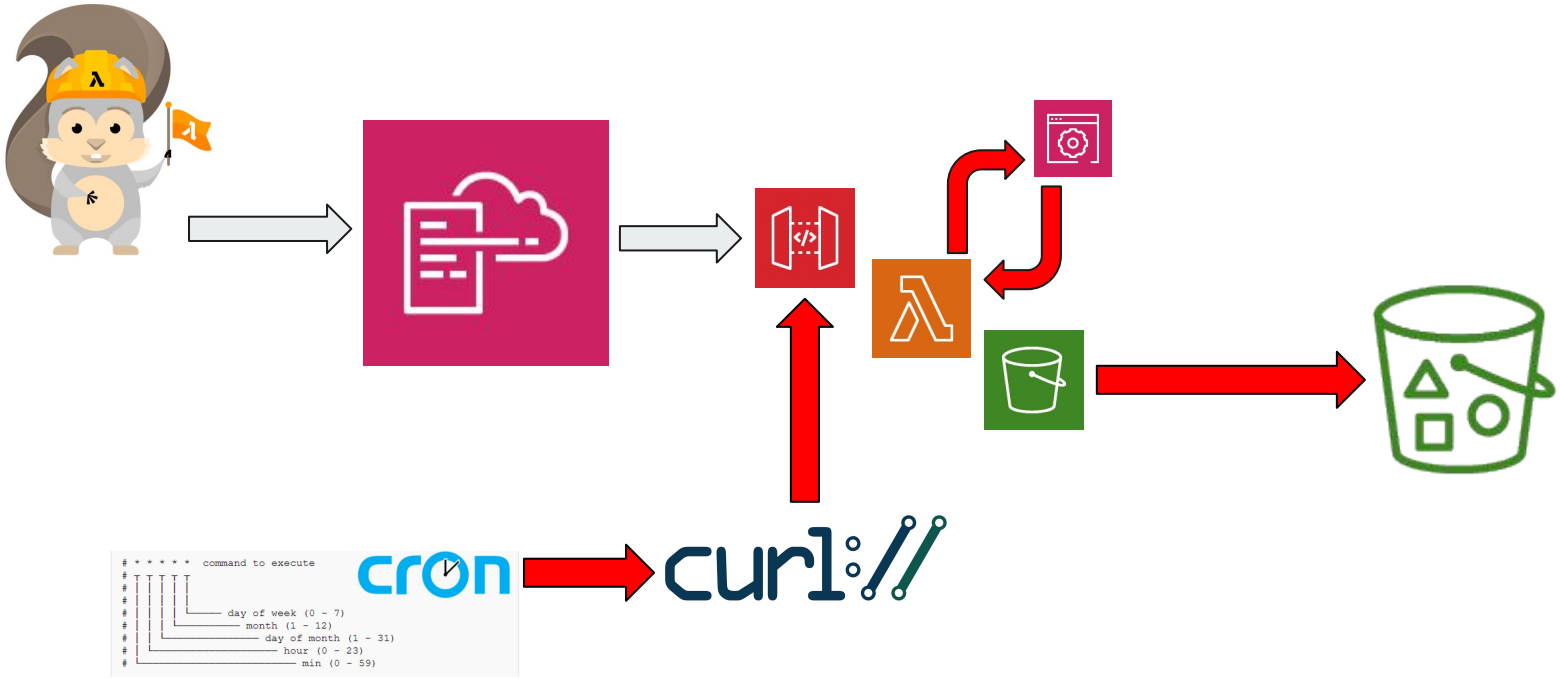






https://aws.amazon.com/blogs/aws/new-aws-toolkits-for-pycharm-intellij-preview-and-visual-studio-code-preview/

Leverage SAM to Create...



Finding Accounts

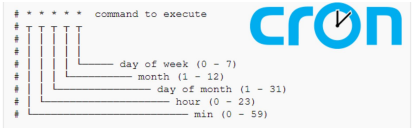
33x

0-3

Second Delay

~40 Account IDs (302s)

Per Hour

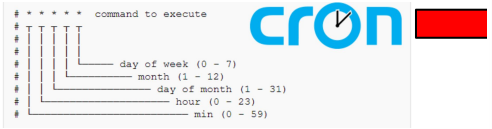


curl://

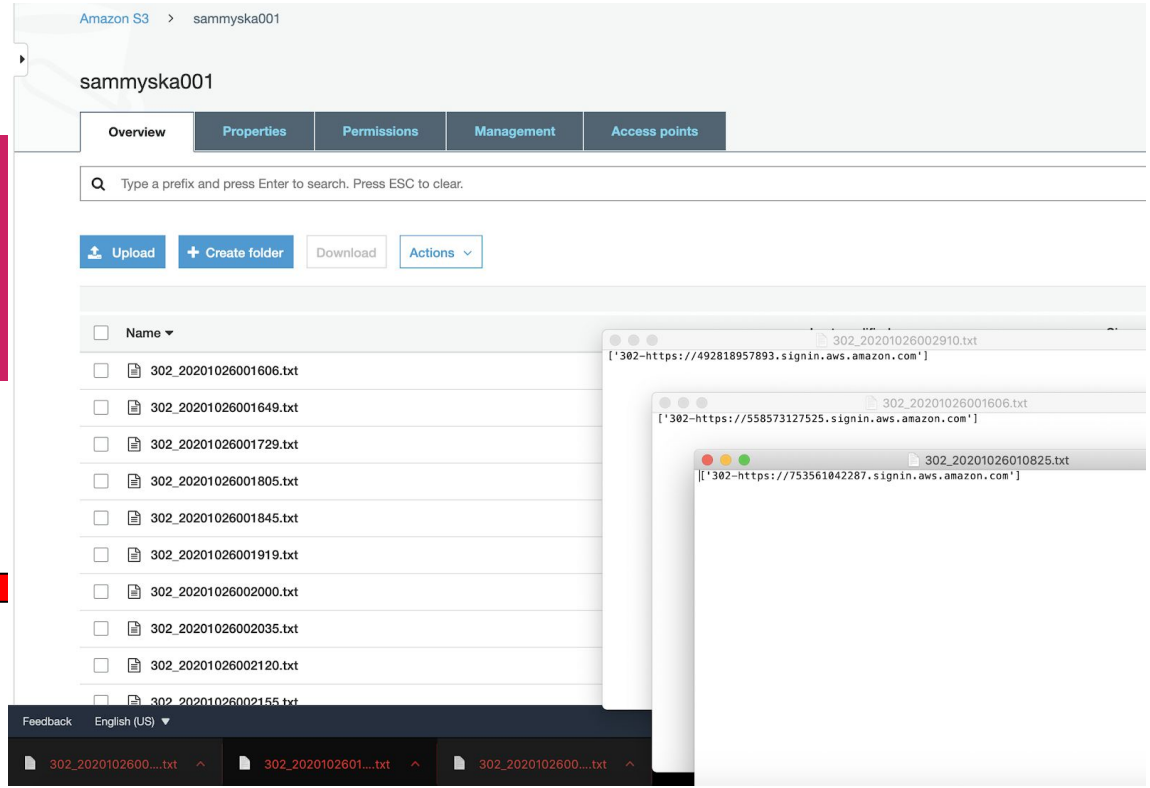
w/ Max 15 Min

At every 3rd minute (*/3 * * * *)

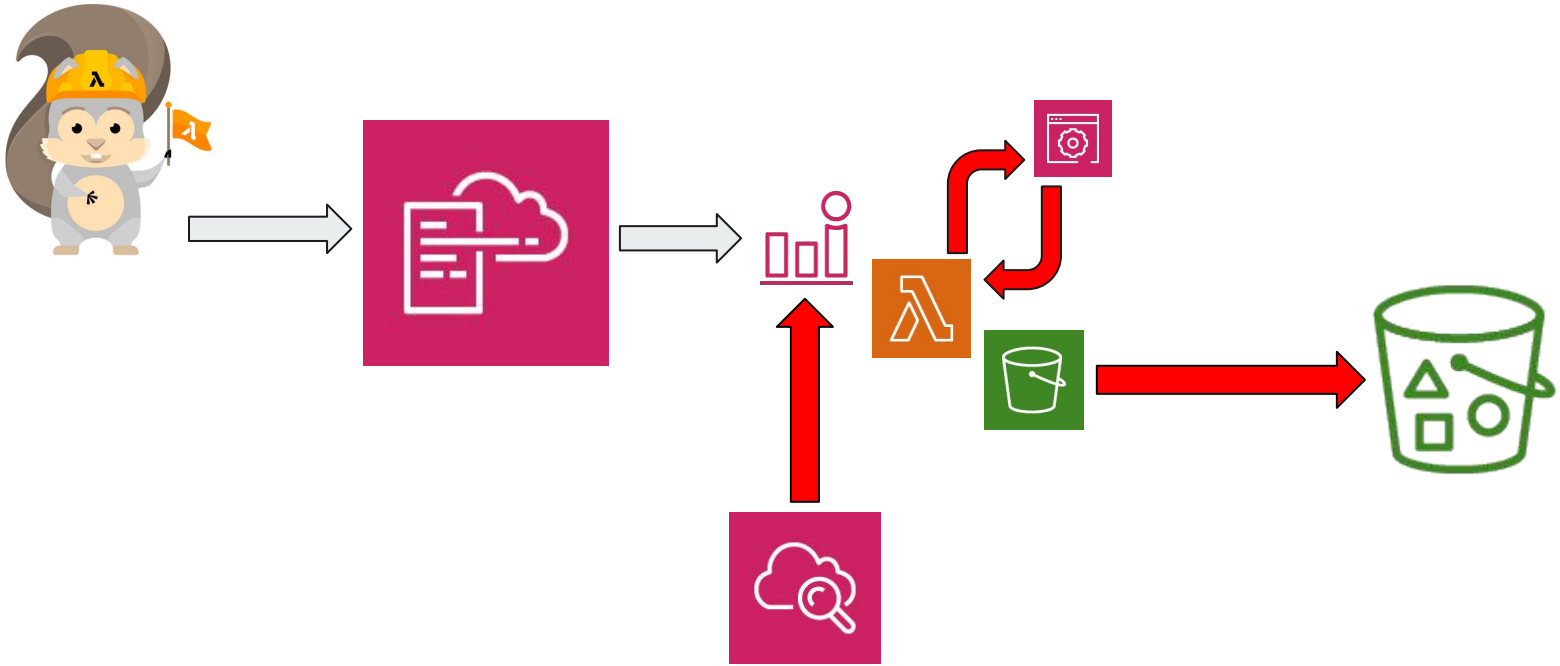
Leverage SAM to Create...



At every 3rd minute (*/3 * * * *)



Another Option: Cron via CloudWatch



https://docs.aws.amazon.com/serverless-application-model/latest/developerguide/sam-property-function-schedule.html

Correlating Account ID w/ Resources

Boto3 Docs 1.16.4 documentation

TABLE OF CONTENTS

Request Syntax

```
response = object.get(
    IfMatch='string',
    IfModifiedSince=datetime(2015, 1, 1),
    IfNoneMatch='string',
    IfUnmodifiedSince=datetime(2015, 1, 1),
    Range='string',
    ResponseCacheControl='string',
    ResponseContentDisposition='string',
    ResponseContentEncoding='string',
    ResponseContentLanguage='string',
    ResponseContentType='string',
    ResponseExpires=datetime(2015, 1, 1),
    VersionId='string',
    SSECustomerAlgorithm='string',
    SSECustomerKey='string',
    RequestPayer='requester',
    PartNumber=123,
    ExpectedBucketOwner='string'
)
```

`get`(***kwargs*)

Retrieves objects from Amazon S3. To use `GET` , you must have `READ` access to the object. If you grant `READ` access to the anonymous user, you can return the object without using an authorization header.

S3 feature to validate correct bucket ownership!

If we know of a publically available object within an S3 bucket

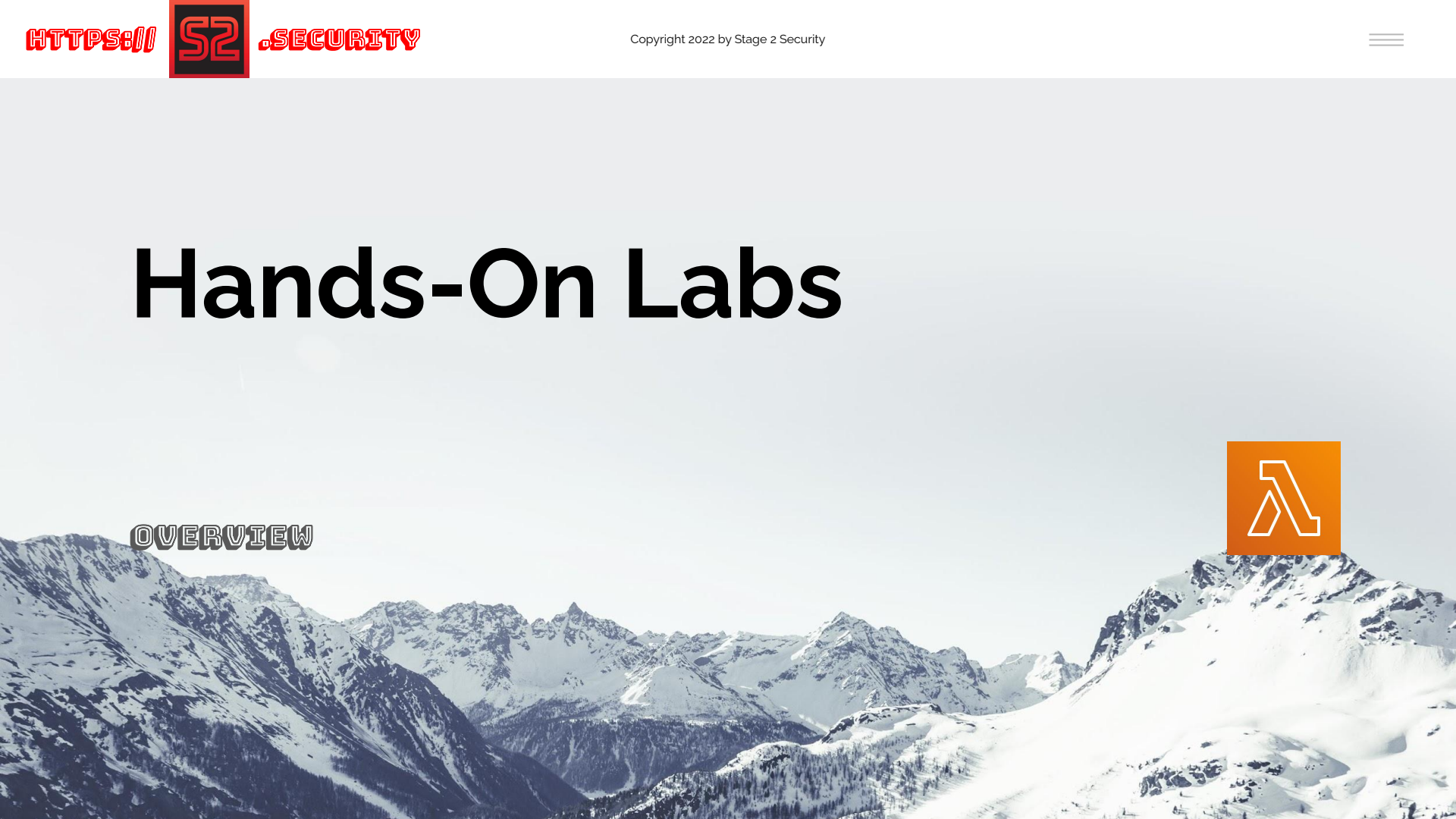
e.g. a static resource (image, javascript, etc.) for a public website

We can check to see if an AWS Account ID is associated with it via this API!

- **ExpectedBucketOwner** (*string*) -- The account id of the expected bucket owner. If the bucket is owned by a different account, the request will fail with an HTTP `403 (Access Denied)` error.

https://aws.amazon.com/about-aws/whats-new/2020/09/amazon-s3-bucket-owner-condition-helps-validate-correct-bucket-ownership/
https://docs.aws.amazon.com/AmazonS3/latest/dev/bucket-owner-condition.html
https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3.html

Hands-On Labs



Hand-On Labs!

Includes:

1. Login_to_the_AWS_Student_Environment
2. SSH_to_Server_for_Labs
3. Cloud9_and_SAM_101
4. HTTP_GET_Parameters
5. Local_Debug_and_Testing
6. Open_Port_Check
7. Subdomain_Discovery

8. (Optionally) Clean_Up

Thank You! Training@Stage2Sec.com

Trainings @ BlackHat & On-Site!

black hat®

STAGE 2 SECURITY

@TweekFawkes

VOODOO.sh

End