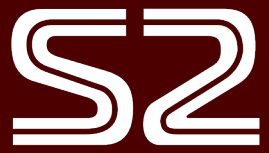


Relentlessly Secure.

Easy Serverless Apps for Automating Red Teaming on AWS Training

Content Location

Github.com/
Stage2Sec/
CaptureTheCloud/
SaintCon2021.zip

perhaps04MAKES!!sense57

Agenda

Event Agenda

3pm thru 5pm:

- Easy Serverless Apps for Automating Red Teaming Training

Training Format & Agenda

Slides (5 min), Demo of Lab (5 min), Lab Time (10-20 min), Repeat!

- Q&A Throughout / Anytime

Easy Red Team Serverless Application Model (SAM) Training

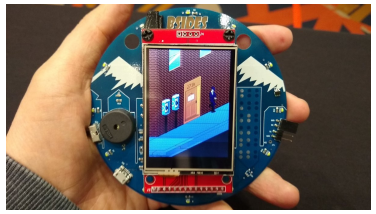
- Lab: C9 & SAM 101 (w/ Lab Environment Access)
- Lab: HTTP GET Parameters
- Lab: Local Debug & Testing
- Lab: Open Port Check
- Lab: Subdomain Discovery

Introduction

About Me

Bryce Kunz

@TweekFawkes



Defense
DHS SOC

Offense
NSA

Red Team
Adobe DX

Cloud Security Challenges

Borderless networks with continuously evolving workloads and data .

Explosion of cloud and container workloads with security tools in use that were not designed to work together

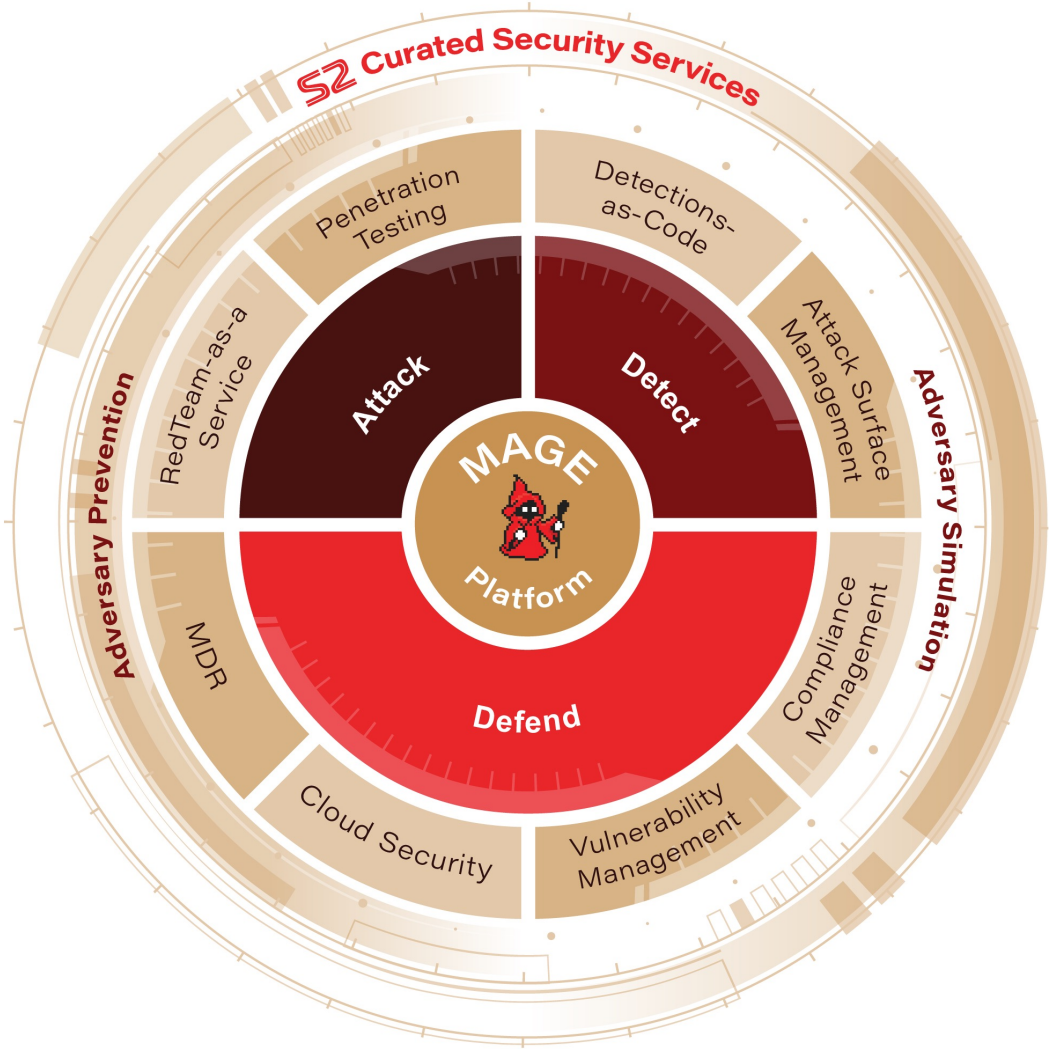
Manual, inconsistent processes often relying on legacy attack surface discovery and identification techniques

Global talent shortage

Alert and Vulnerability fatigue
Constant threats and public exploits (ransomware)

Difficulty rising above the daily firefighting to track KPIs and drive improvement

Attack. Detect. Defend. Repeat



In cybersecurity there are only two teams, and it's not red and blue. It's the good guys and the bad guys.

At S2, we break down the barriers between red team and blue team so threats don't break down your defenses.

We simulate the advanced threats your enterprise faces and automate detection and response, all in one Adversary Simulation/Detection and Response platform MAGE.

With continuous red-team, we find the latest vulnerabilities not just the threats bad actors know you know about. And with as-a-service offerings, it's expertise that protects your enterprise and your budget.

Red Teaming as a Service

MAGE

S2

OSINT

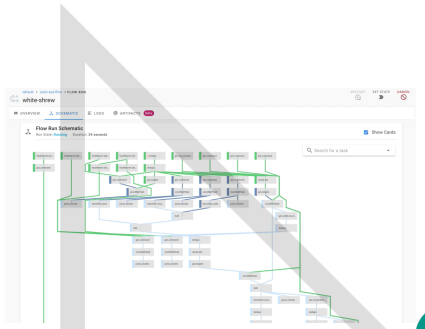
- OSINT++, Secrets in Repos
- Subdomain Takeovers
- Breached Creds, Dark CTI

External Analysis - EASM

- Service & Port Discovery
- Web App Enumeration, IoT
- Weak Credential Checks

Cloud Analysis - CSPM

- Public Service Discovery
- Access Misconfigurations
- Best Practices (e.g. CIS)



FastAPI

PREFECT

Amazon EKS

AWS

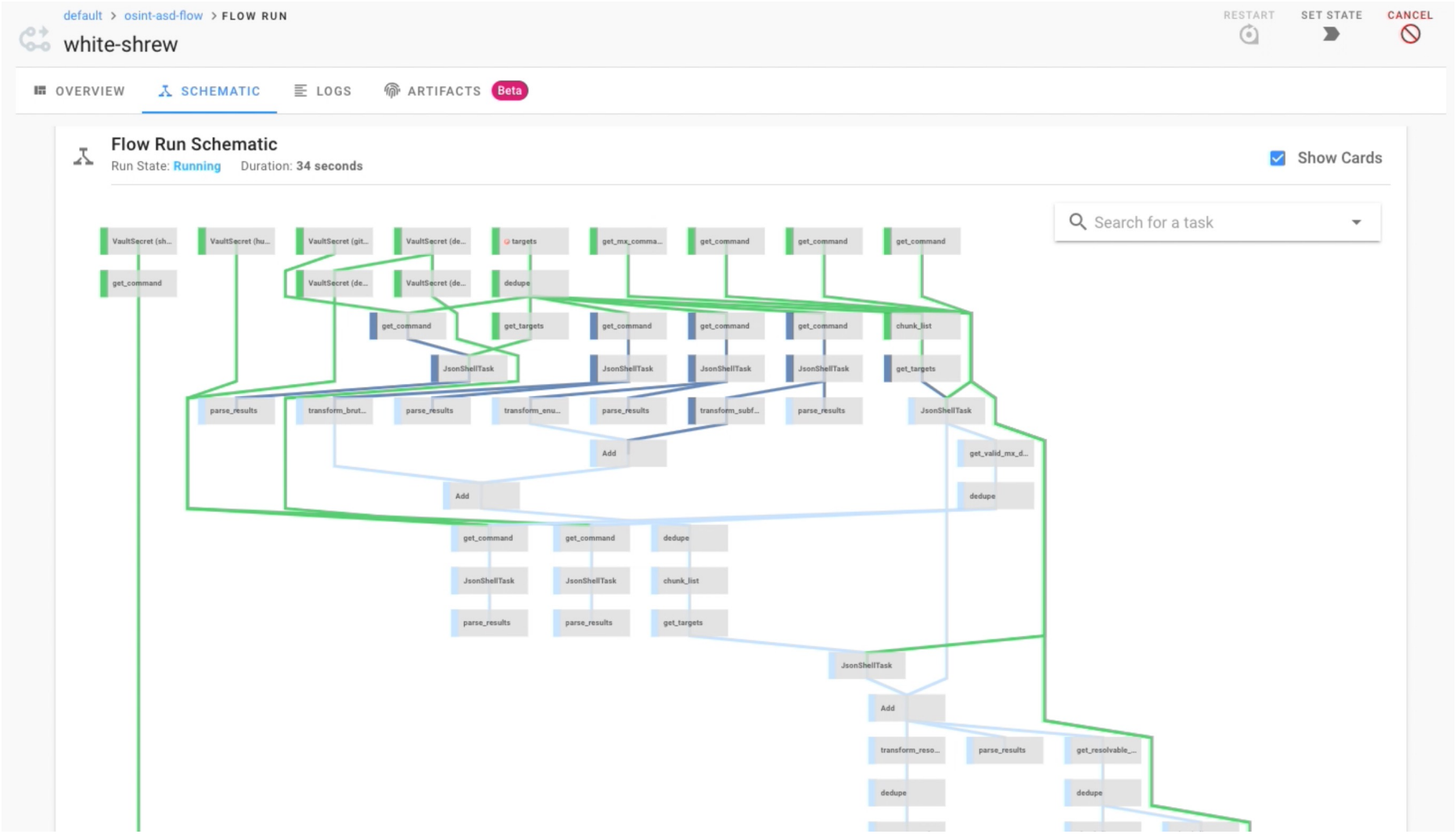
Targeting & Analysis

Client Experience

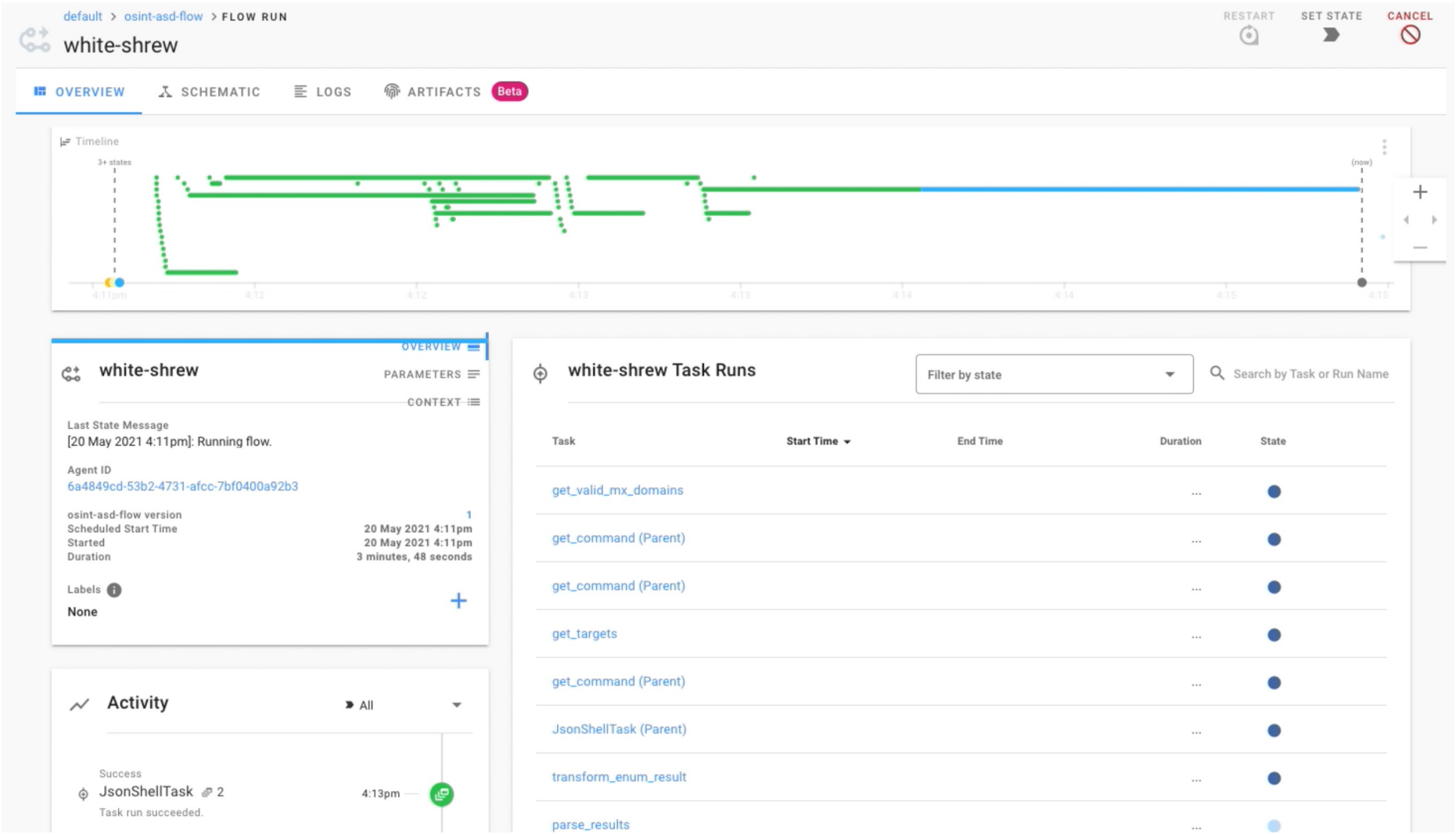
Interactive Operations

VOODOO

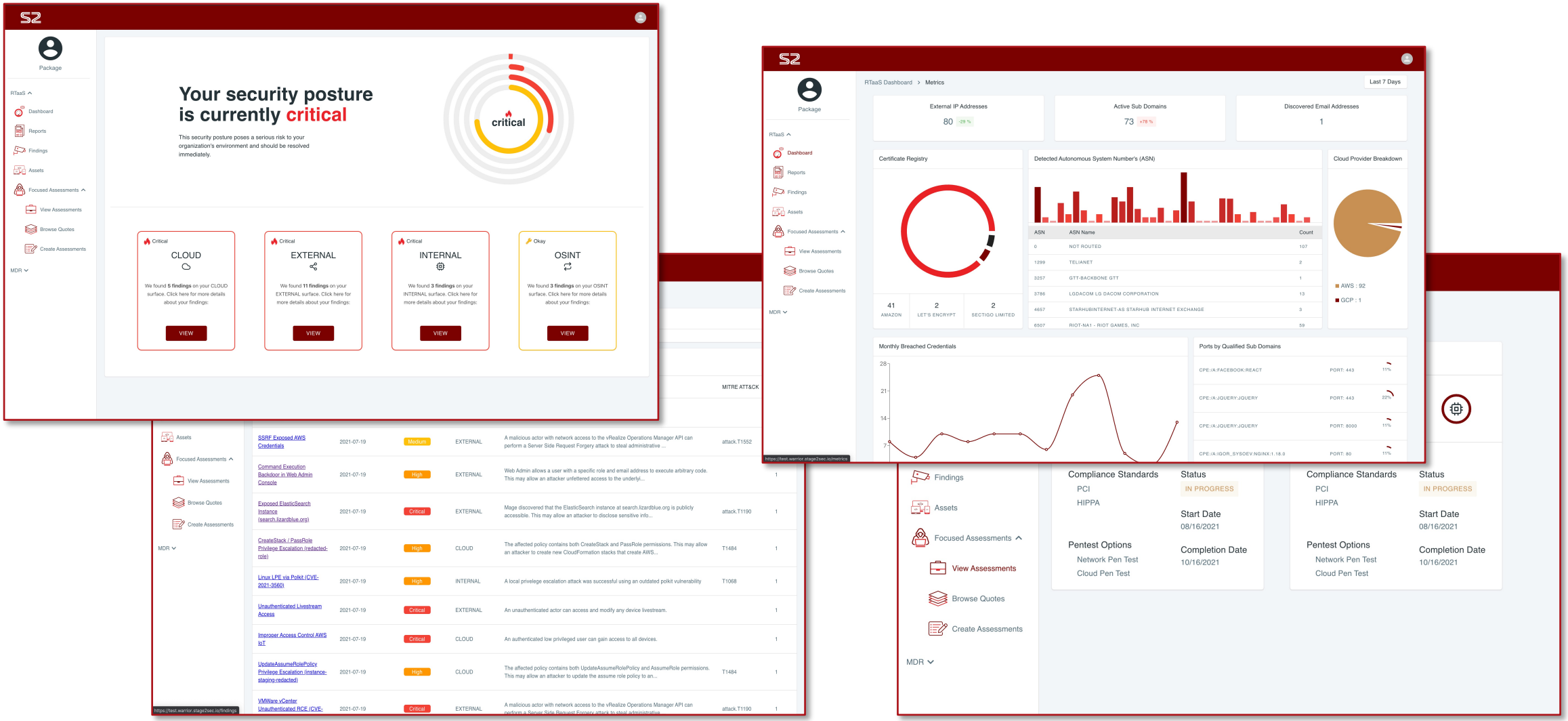
Red Teaming as a Service



Red Teaming as a Service

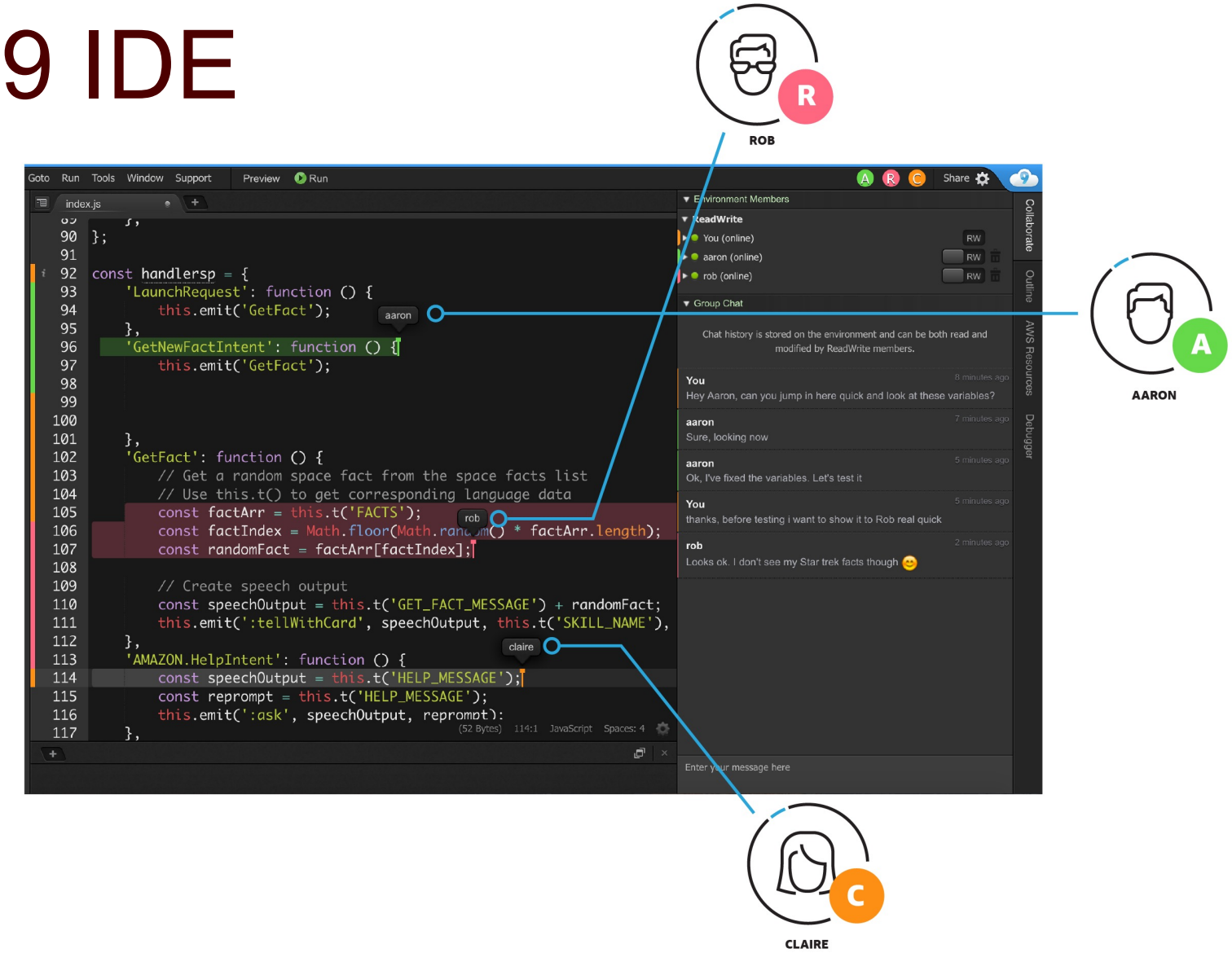


PTaaS/RTaaS – Continuous Testing



C9 & SAM

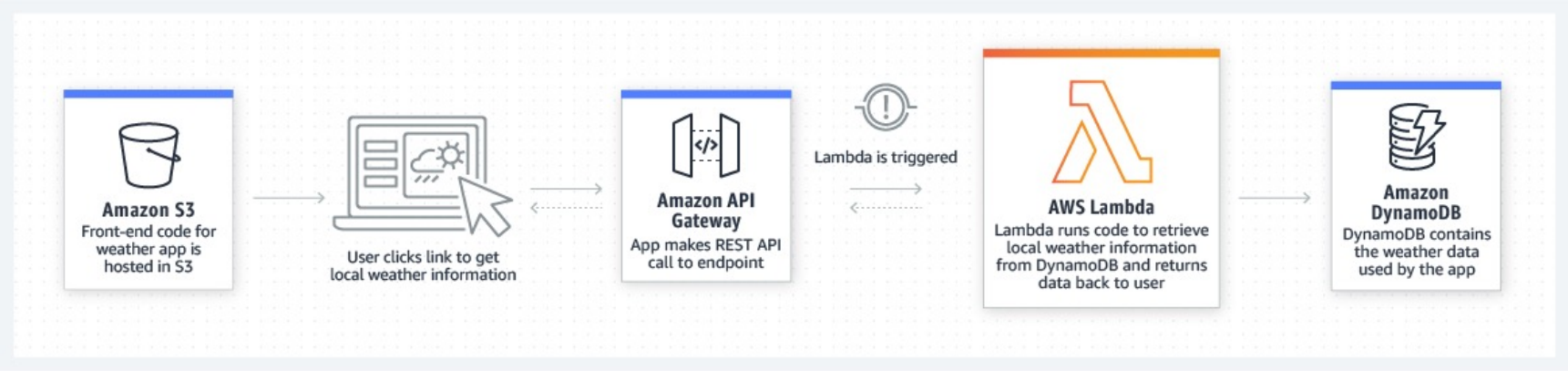
Cloud9 IDE

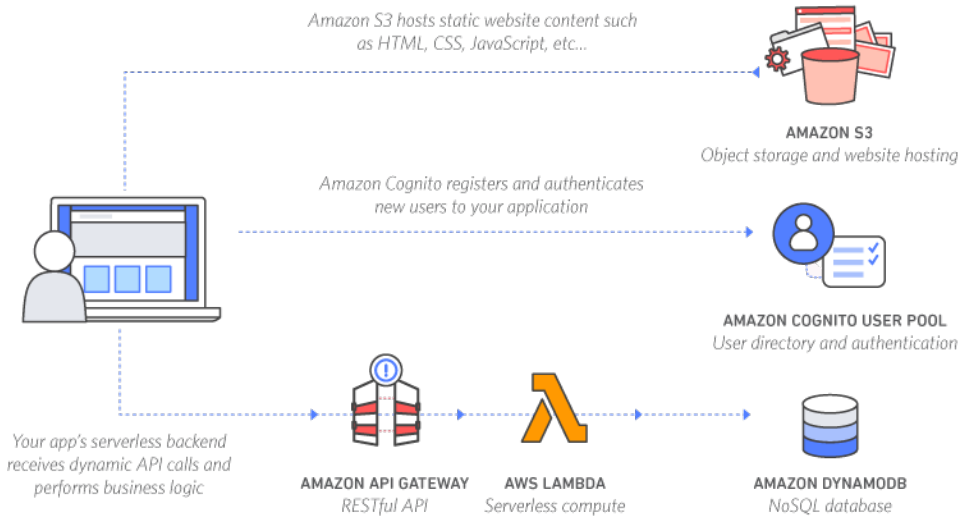


AWS Cloud9 is a cloud-based integrated development environment (IDE) that lets you write, run, and debug your code with just a browser.

https://aws.amazon.com/cloud9/

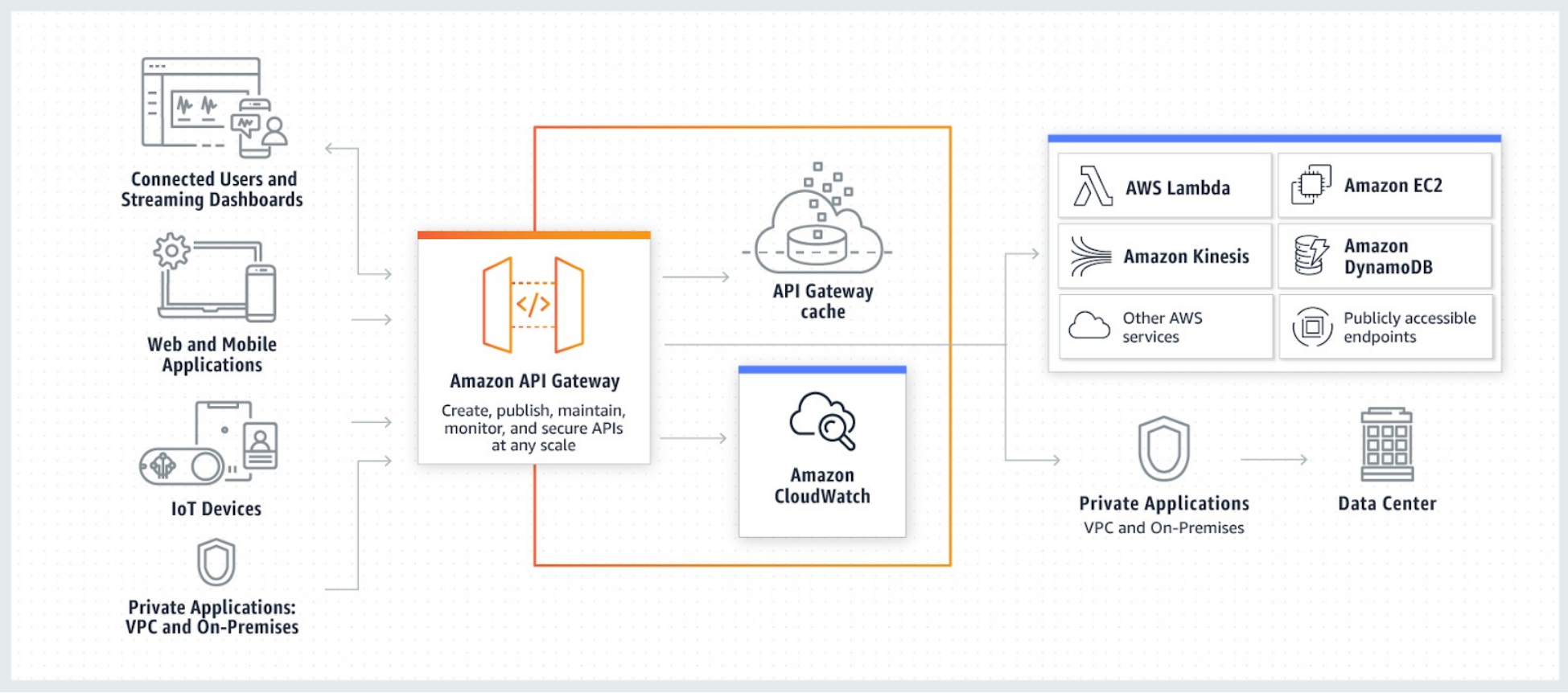
Serverless Application Model (SAM)





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

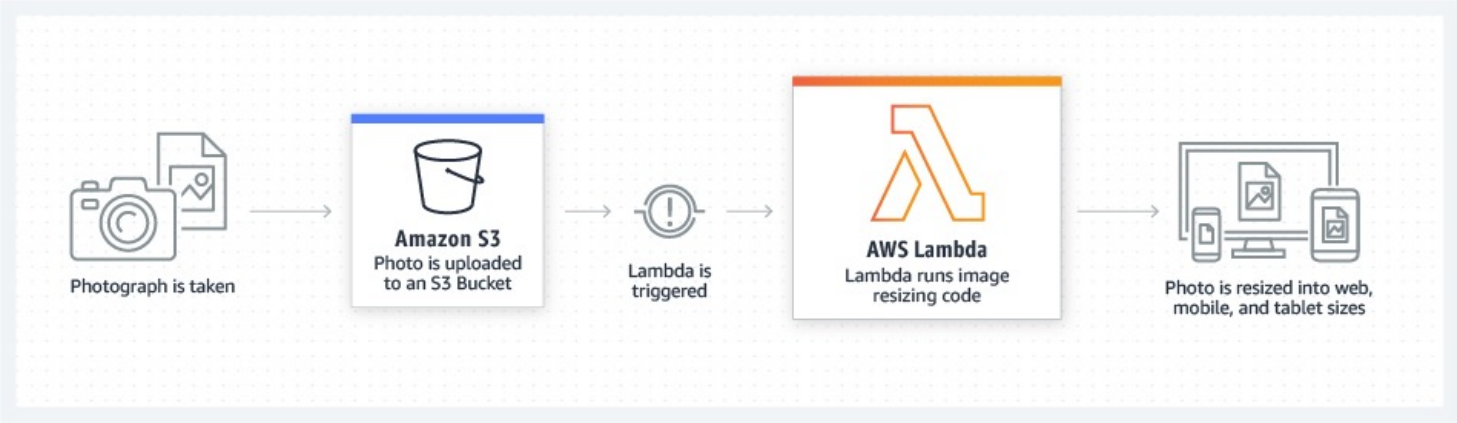
API Gateway



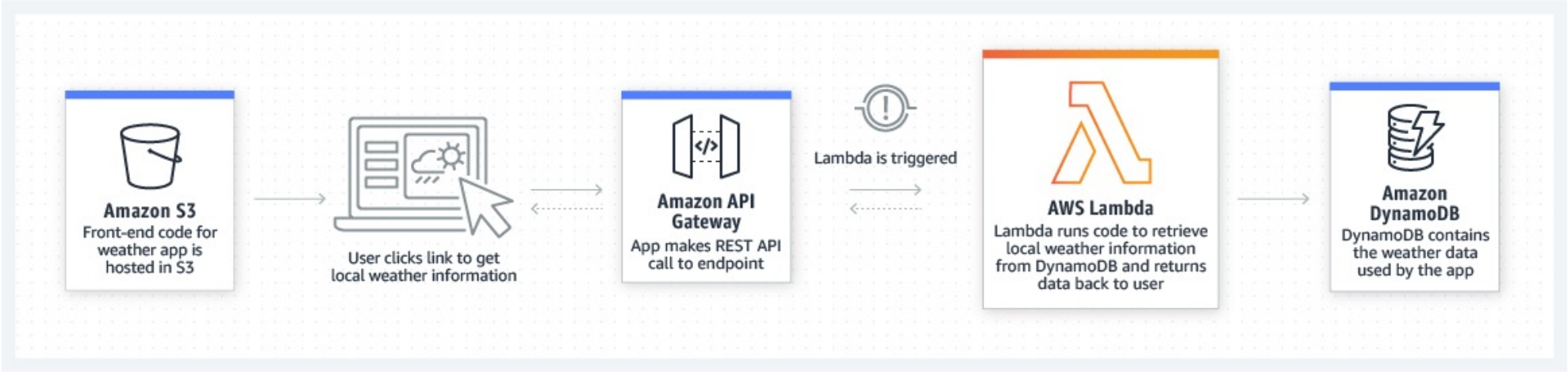
API Proxy (HTTP or REST) as a managed service

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

Lambda



AWS Scripts as a managed service



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

Lambda & API GW

Curl -> API GW -> Lambda v

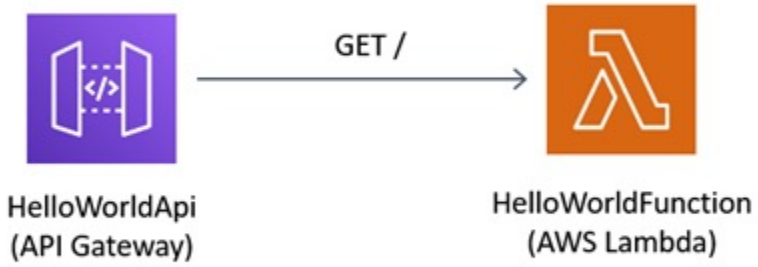
Lambda

Curl <- API GW <- Lambda ^

LAB: Cloud9 & SAM 101

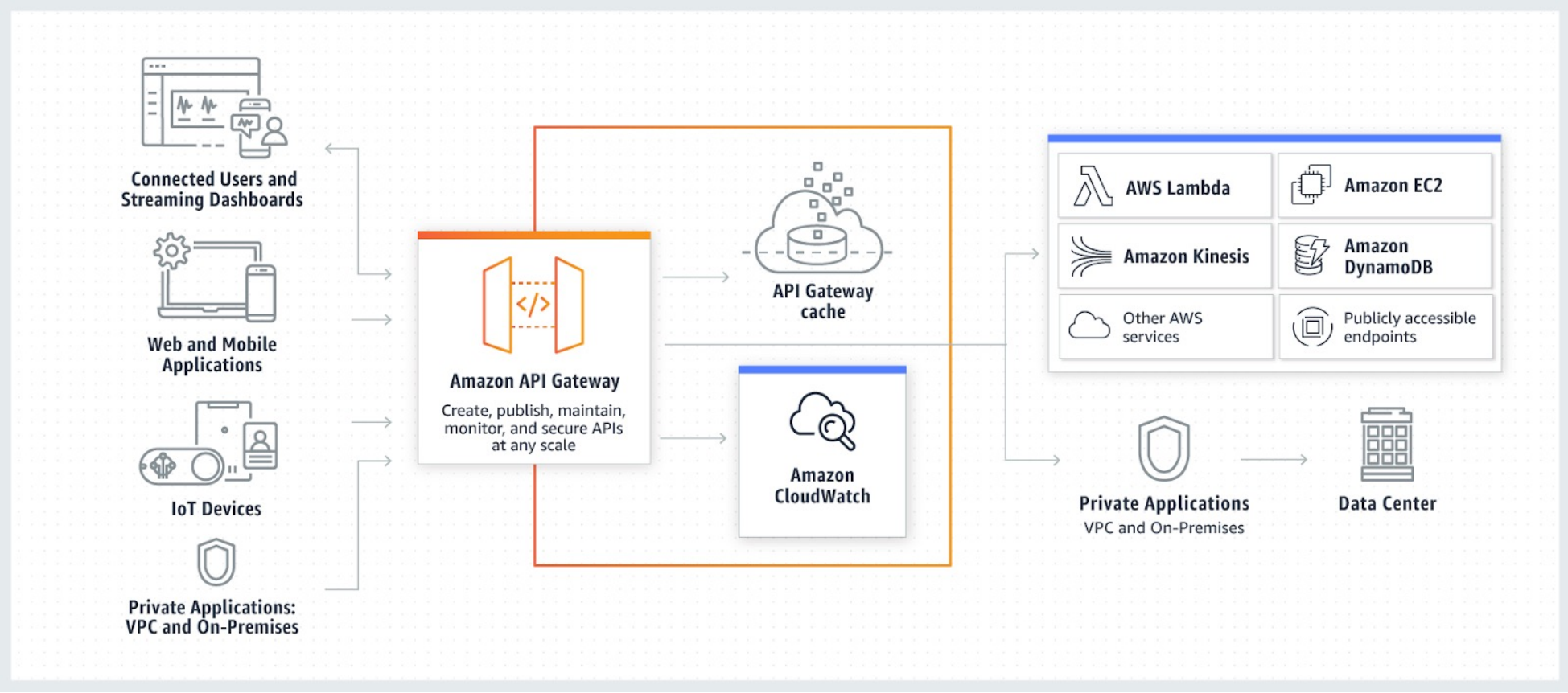
Hands-On Lab:

- URL: https://train.stage2sec.com/



HTTP GET Params

API Gateway



API Proxy (HTTP or REST) as a managed service

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

API Gateway

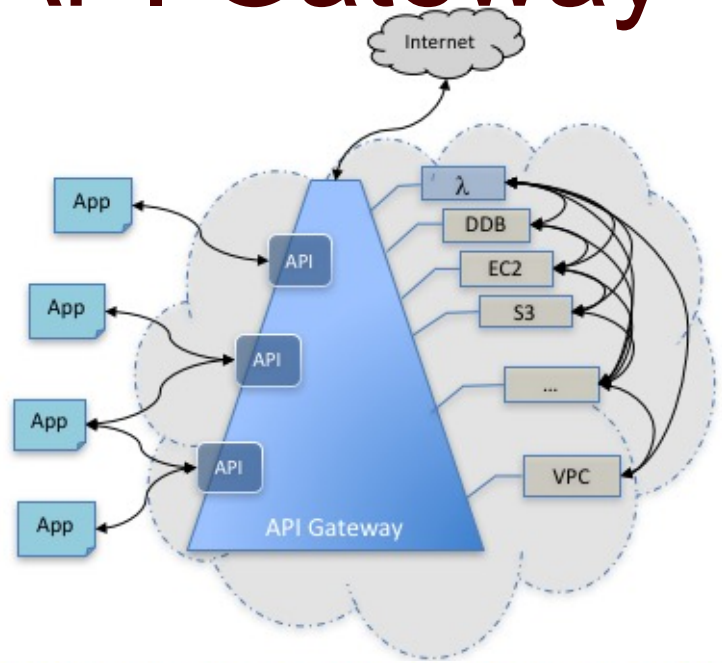
API Gateway is a managed service that creates APIs at scale

- 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



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

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

“Event” Object

Python Dictionary

{'resource': '/hello', 'path': '/hello/', 'httpMethod': 'GET', 'headers': {'Accept': '*/*', 'CloudFront-Forwarded-Proto': 'https', 'CloudFront-Is-Desktop-Viewer': 'true', 'CloudFront-Is-Mobile-Viewer': 'false', 'CloudFront-Is-SmartTV-Viewer': 'false', 'CloudFront-Is-Tablet-Viewer': 'false', 'CloudFront-Viewer-Country': 'US', 'Host': 'jiy58cz051.execute-api.us-east-1.amazonaws.com', 'User-Agent': 'curl/7.58.0', 'Via': '2.0 6ff4697c5089876d94430beacc9a4d5e.cloudfront.net (CloudFront)', 'X-Amz-Cf-Id': 'N2AvPGKjnYO1pmEAEiw9WUFoDpVLAJZJLEir4IVYPiJ1CkCSOncd6Q==', 'X-Amzn-Trace-Id': 'Root=1-614cfd31-70f86876714f678132ccec87', 'X-Forwarded-For': '3.237.255.37, 130.176.133.131', 'X-Forwarded-Port': '443', 'X-Forwarded-Proto': 'https'}, 'multiValueHeaders': {'Accept': ['*/*'], 'CloudFront-Forwarded-Proto': ['https'], 'CloudFront-Is-Desktop-Viewer': ['true'], 'CloudFront-Is-Mobile-Viewer': ['false'], 'CloudFront-Is-SmartTV-Viewer': ['false'], 'CloudFront-Is-Tablet-Viewer': ['false'], 'CloudFront-Viewer-Country': ['US'], 'Host': ['jiy58cz051.execute-api.us-east-1.amazonaws.com'], 'User-Agent': ['curl/7.58.0'], 'Via': ['2.0 6ff4697c5089876d94430beacc9a4d5e.cloudfront.net (CloudFront)'], 'X-Amz-Cf-Id': ['N2AvPGKjnYO1pmEAEiw9WUFoDpVLAJZJLEir4IVYPiJ1CkCSOncd6Q=='], 'X-Amzn-Trace-Id': ['Root=1-614cfd31-70f86876714f678132ccec87'], 'X-Forwarded-For': ['3.237.255.37, 130.176.133.131'], 'X-Forwarded-Port': ['443'], 'X-Forwarded-Proto': ['https']}, 'queryStringParameters': {'AAAA': 'BBBB'}, 'multiValueQueryStringParameters': {'AAAA': ['BBBB']}, 'pathParameters': None, 'stageVariables': None, 'requestContext': {'resourceId': '8978if', 'resourcePath': '/hello', 'httpMethod': 'GET', 'extendedRequestId': 'GIx_yHemoAMFZPg=', 'requestTime': '23/Sep/2021:22:18:25 +0000', 'path': '/Prod/hello/', 'accountId': '580299357056', 'protocol': 'HTTP/1.1', 'stage': 'Prod', 'domainPrefix': 'jiy58cz051', 'requestTimeEpoch': 1632435505617, 'requestId': 'a8ad1156-d894-46c2-8c6d-c54a058ed420', 'identity': {'cognitoIdentityPoolId': None, 'accountId': None, 'cognitoIdentityId': None, 'caller': None, 'sourceIp': '3.237.255.37', 'principalOrgId': None, 'accessKey': None, 'cognitoAuthenticationType': None, 'cognitoAuthenticationProvider': None, 'userArn': None, 'userAgent': 'curl/7.58.0', 'user': None}, 'domainName': 'jiy58cz051.execute-api.us-east-1.amazonaws.com', 'apiId': 'jiy58cz051'}, 'body': None, 'isBase64Encoded': False}

“Event” Object

Python Dictionary

…

'queryStringParameters': {'AAAA': 'BBBB’},

…

“Event” Object

Python Dictionary

…

'queryStringParameters': {'AAAA': 'BBBB’},

…

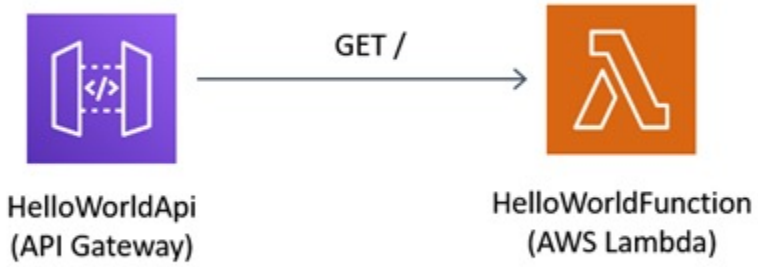
event['queryStringParameters']

event['queryStringParameters']['AAAA']

LAB: HTTP GET Parameters

Hands-On Lab:

- URL: https://train.stage2sec.com/



Debug & Test

Local Debug

```
if __name__=="__main__":
    event = dict({…'queryStringParameters': {'AAAA': 'BBBB'},…})
    context = ''
    lambda_handler(event, context)
```

Locally Hosted API

sam local start-api

curl http://127.0.0.1:3000/hello?AAAA=BBBB

Lambda Function Execution

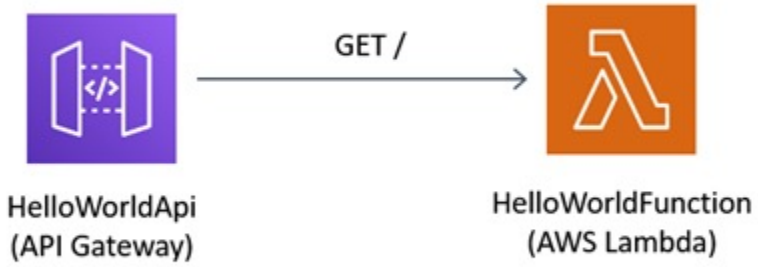
sam local invoke "HelloWorldFunction" -e events/event.json

sam local generate-event apigateway aws-proxy --body "" --path "hello" --method GET > /home/ubuntu/environment/debug-app-001/events/api-event.json

LAB: Local Debug & Testing

Hands-On Lab:

- URL: https://train.stage2sec.com/



Open Port Check

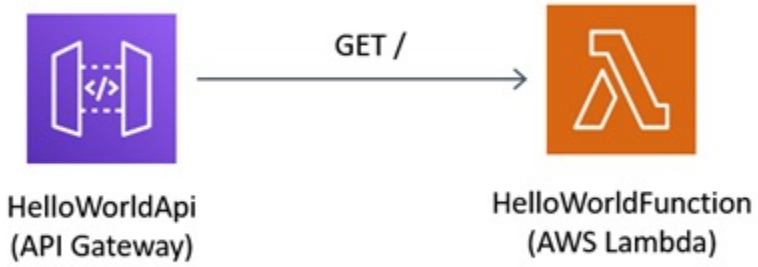
Open Port Check

```
sock = socket.socket(socket.AF_INET,
socket.SOCK_STREAM)
sock.settimeout(2) #2 Second Timeout
result = sock.connect_ex((sTargetIp,int(sTcpPort)))
if result == 0:
        print("Port is open")
        sReturn = sTargetIp + ":" + sTcpPort + "/TCP is open“
else:
        print("Port is not open")
        sReturn = sTargetIp + ":" + sTcpPort + "/TCP is closed“
…
```

LAB: Open Port Check

Hands-On Lab:

- URL: https://train.stage2sec.com/



Subdomain Discovery

Subdomain Discovery

```
filepath = 'namelist.txt'
        with open(filepath) as f:
…
 addrs_List = [str(i[4][0]) for i in socket.getaddrinfo(sFqdn, 65535)]
...
 return {
          "body": sReturn
    }
```

Subdomain Discovery

template.yaml

```
Policies:
    - S3FullAccessPolicy:
        BucketName: batmanisbestman001
```

app.py

```
import boto3

client.put_object(Body=sFileContents, Bucket=sBucketName, Key=sFileName)
```

Subdomain Discovery

[+] START

[+] Subdomain discovered: cdn.lizardblue.com -> 52.217.194.113

[+] Subdomain discovered: cdn2.lizardblue.com -> 52.219.105.19

[+] Subdomain discovered: hash.lizardblue.com -> 52.85.61.12, 52.85.61.64, 52.85.61.65, 52.85.61.94

[+] Subdomain discovered: images.lizardblue.com -> 52.219.142.28

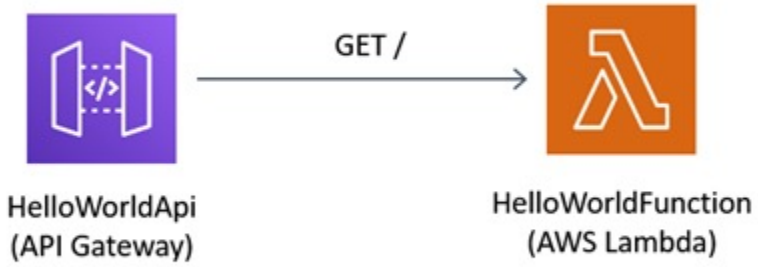
[+] Subdomain discovered: ixhash.lizardblue.com -> 3.226.63.115, 3.228.53.222, 3.230.230.89, 3.232.236.175, 34.193.24.255, 52.73.45.196, 54.172.93.56, 54.80.73.136

[+] END

LAB: Subdomain Discovery

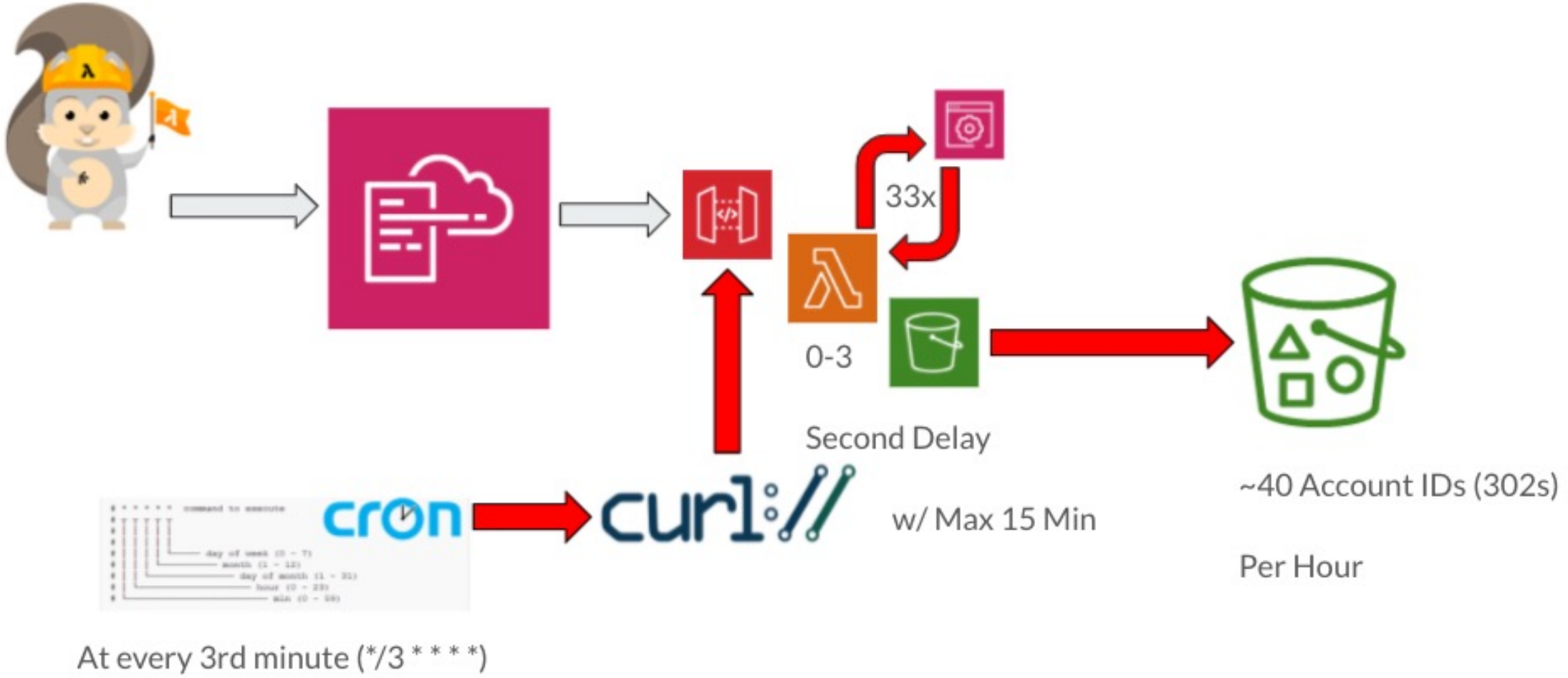
Hands-On Lab:

- URL: https://train.stage2sec.com/



Account ID Discovery

Finding Accounts

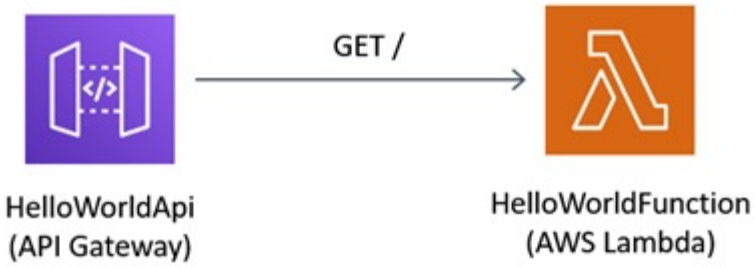


https://docs.aws.amazon.com/serverless-application-model/latest/developerguide/sam-property-function-schedule.html

LAB: Account ID Discovery

Hands-On Lab:

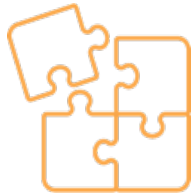
- Code: https://github.com/Stage2Sec/CaptureTheCloud/tree/master/sam_discover_account_ids



Conclusion

S2 Secure Cloud MSS

Red Team-as-a-Service, MDR and Risk management in a single platform

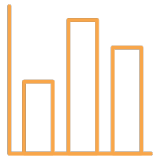


Orchestrated platform of managed services that work together

Create consistent, automated processes and slash discovery & response times whether in cloud, container or on-premise

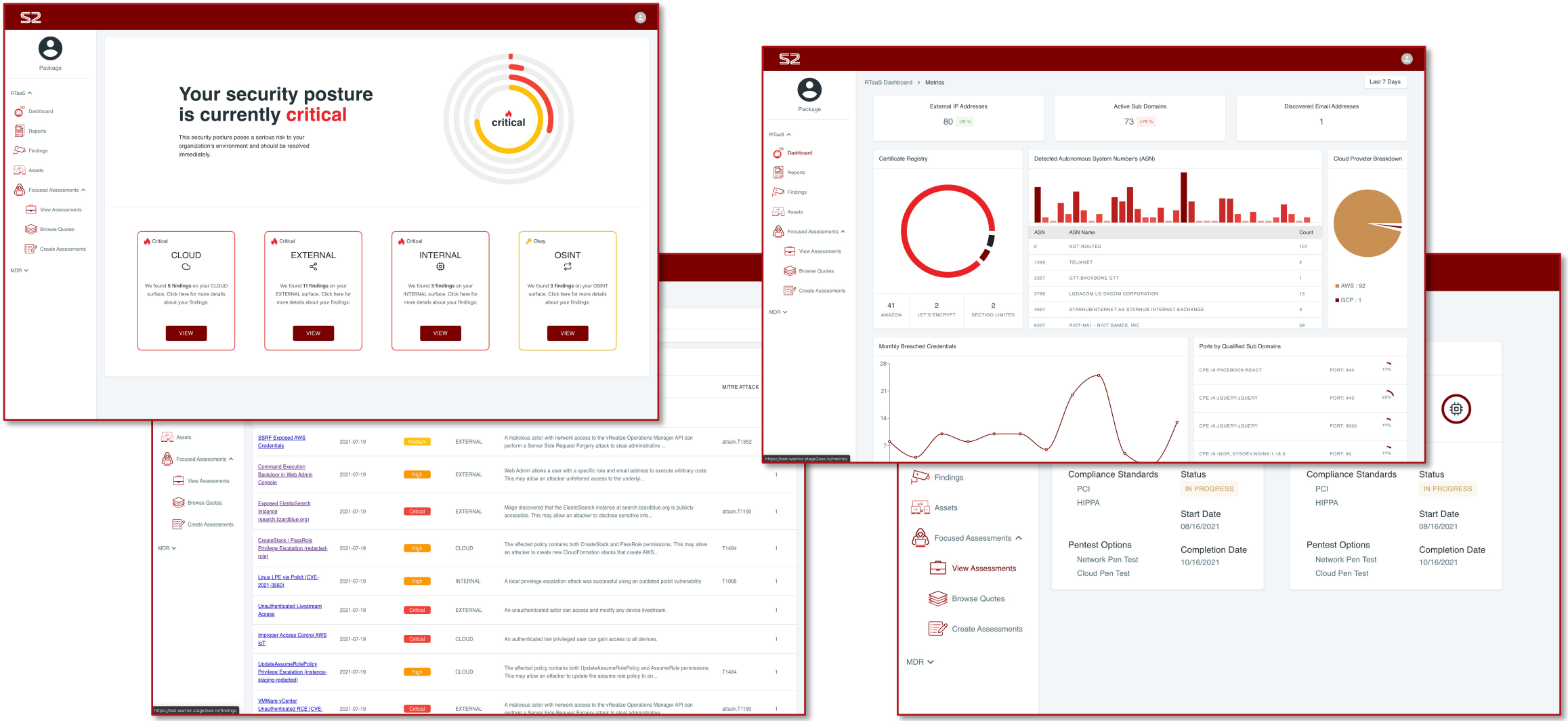
Focus on reducing IMMIENT RISK

Outcomes. Prioritized vulnerabilities, alerts and IMMINENT RISK



Track, measure and improve your security risk posture

PTaaS/RTaaS – Continuous Testing



Thank You! Bryce@Stage2Sec.com

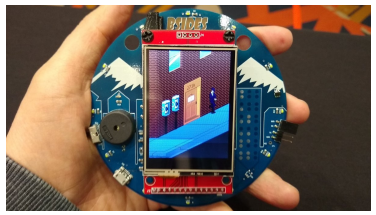
Trainings: Hands-On Cloud Red Teaming

Code: Github.com/Stage2Sec/CaptureTheCloud

Slides: SpeakerDeck.com/TweekFawkes

Bryce Kunz

@TweekFawkes



Thank You

Bryce@Stage2Sec.com