Attacking and Defending Serverless Applications Workshop

Ryan Nicholson | Blue Mountain Cyber, LLC

SEC488/SEC541 Author and SANS Certified Instructor

Agenda (1/2)

- Workshop Pre-Requisites
- Evidence-App Overview
  - **Exercise 1: Deploying the Serverless Application**
- Serverless ATT&CK Techniques
  - **Exercise 2: Reconnaissance of Evidence-App**
  - **Exercise 3: Discovering Evidence-App Vulnerability**
  - **Exercise 4: Exploiting Evidence-App and Pivoting to Cloud Account**

Agenda (2/2)

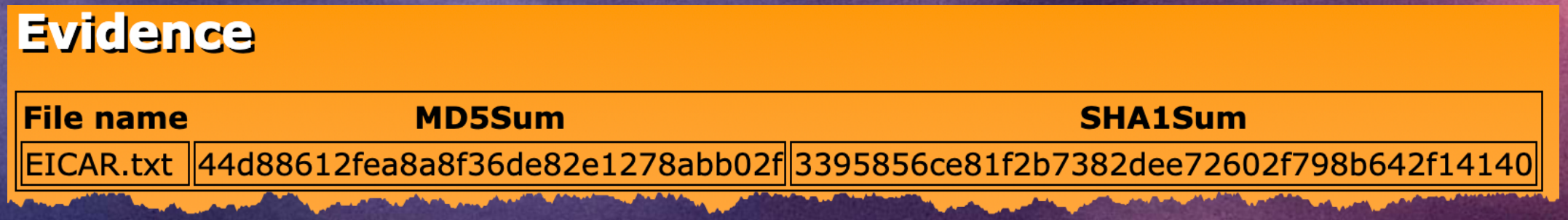
- Investigating Serverless ATT&CK techniques
  - **Exercise 5: Identifying Reconnaissance**
  - **Exercise 6: Identifying Vulnerability Discovery**
  - **Exercise 7: Identifying Exploitation and Pivot**
- Conclusion
  - **Exercise 8: Tearing Down Serverless Application**

Pre-Requisites For This Workshop

- AWS Account with `root` or `AdministratorAccess` privileges
- Supported Web Browser
  - Preferably Chrome or Firefox
- Exercise Instructions
  - https://attack-defend-serverless.sanscloudwars.com
- Have watched Sherlock (the good one...)
  - Just kidding 😄

Evidence-App Overview

- *This serverless web application is used by Sherlock's blue team to import evidence data, generate MD5 and SHA1 hashes of the uploaded files, and save the files in a safe location.*



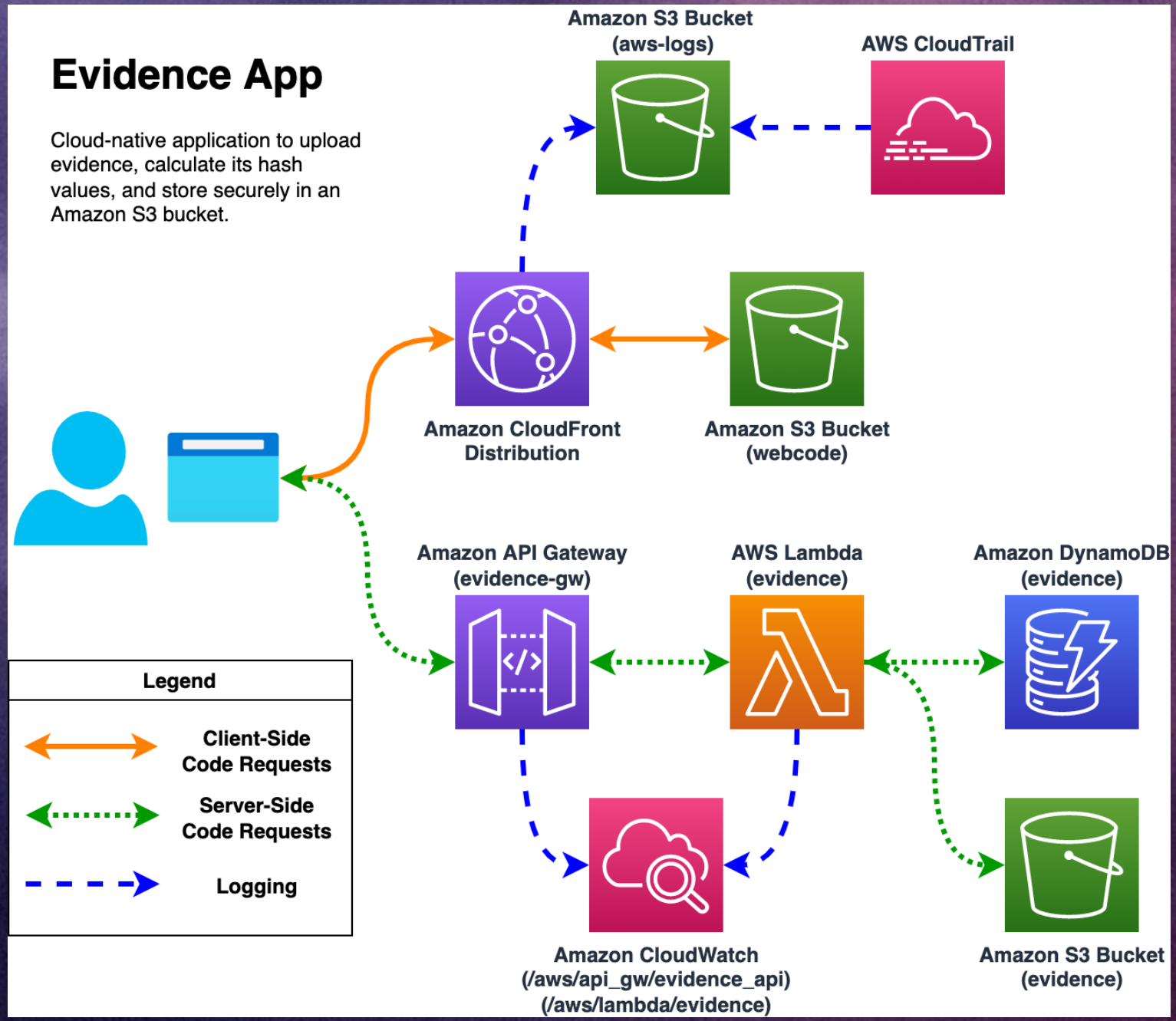
- Source Code: https://github.com/bluemountaincyber/evidence-app

So... what's in that repo?

- **EVERYTHING** as Code
  - Application **Source Code** (`HTML` , `CSS` , `JavaScript` , and `Python 3`)
  - **Infrastructure as Code (IaC)** to build cloud resources and deploy application (`Terraform` and `CloudFormation`)
  - **Exercise documentation** for this workshop (`mkdocs`)
    - In case you want to work on this afterwards or share with your friends/co-workers
  - This presentation (`marp`)
- **LOTS** of opportunity for a coding mistake...

Evidence App

Cloud-native application to upload evidence, calculate its hash values, and store securely in an Amazon S3 bucket.

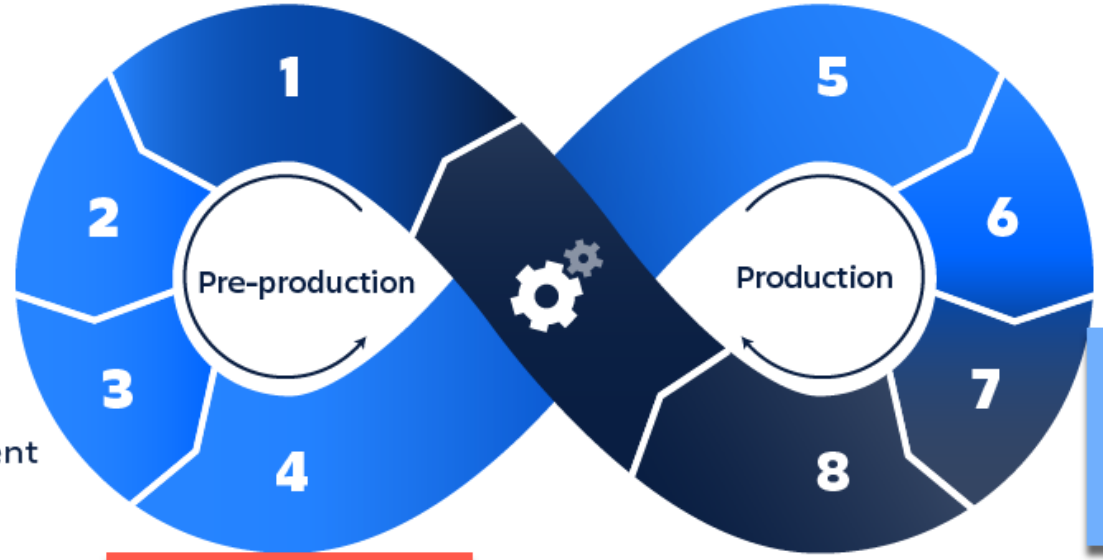


DevSecOps

Plan
Threat landscape, change impact analysis

Release
Access and configuration management

Code
Pre-commit hooks, SAST



Deploy
Chaos engineering, pen test

Build
Software component analysis, SAST

Monitor
Log collection, SIEM, RASP

and then here.

You are here...

Test
Authentication, SQL injection, DAST

Respond
Block attacks, roll back

Deploying the Evidence-App

From **AWS CloudShell**:

- Download Source Code:

```
git clone https://github.com/bluemountaincyber/evidence-app.git
cd /home/cloudshell-user/evidence-app
```

- Execute `cloudformation-deploy.sh`:

```
./cloudformation-deploy.sh
```

- Sit back and relax for ~5 mins

Now It's Your Turn!

https://attack-defend-serverless.sanscloudwars.com

Complete *Exercise 1* and then STOP!

MITRE | ATT&CK®

- *MITRE ATT&CK® is a globally-accessible knowledge base of adversary tactics and techniques based on real-world observations*
- Tactics include:

Reconnaissance
Resource Development
Initial Access
Execution
Persistence
Privilege Escalation
Defense Evasion
Credential Access
Discovery
Lateral Movement
Collection
Command and Control
Exfiltration
Impact

MITRE | ATT&CK®

You will leverage and analyze common ATT&CK techniques

- **Active Scanning (T1595)**
- **Cloud Infrastructure Discovery (T1580)**
- **Exploit Public-Facing Application (T1190)**
- **Command and Scripting Interpreter: Unix Shell (T1059.003)**
- **Unsecured Credentials (T1552)**
- **Data Destruction (T1485)**
- **Defacement: External Defacement (T1491.002)**

Custom Tooling

- Custom Python script to **fuzz** this application
  - `/home/cloudshell-user/evidence-app/scripts/fuzz_evidence_app.py`
- Tests popular command injection payloads
  - https://github.com/payloadbox/command-injection-payload-list

```
$ /home/cloudshell-user/evidence-app/scripts/fuzz_evidence_app.py --target $TARGET/api/
;id; worked as command injection for the file_name parameter!
  Here is a curl command:
  curl -X POST https://djm72vhitom0v.cloudfront.net/api/ -H 'Content-Type: application/x-www-form-urlencoded; charset=UTF-8' -d '{"file_name":";id;","file_data":"dGVzdAo="}'
```

Attack Sequence

1. Use application as a *normal* user would
2. **Spider** web application
3. **Interact** with and learn more about newly-discovered endpoints
4. **Fuzz** the application to discover **command injection**
5. Use **remote code execution** to uncover cloud credentials
6. **Pivot** to cloud account and perform **discovery**
7. Be extra evil by **destroying data** and **defacing** the application

Now It's Your Turn!

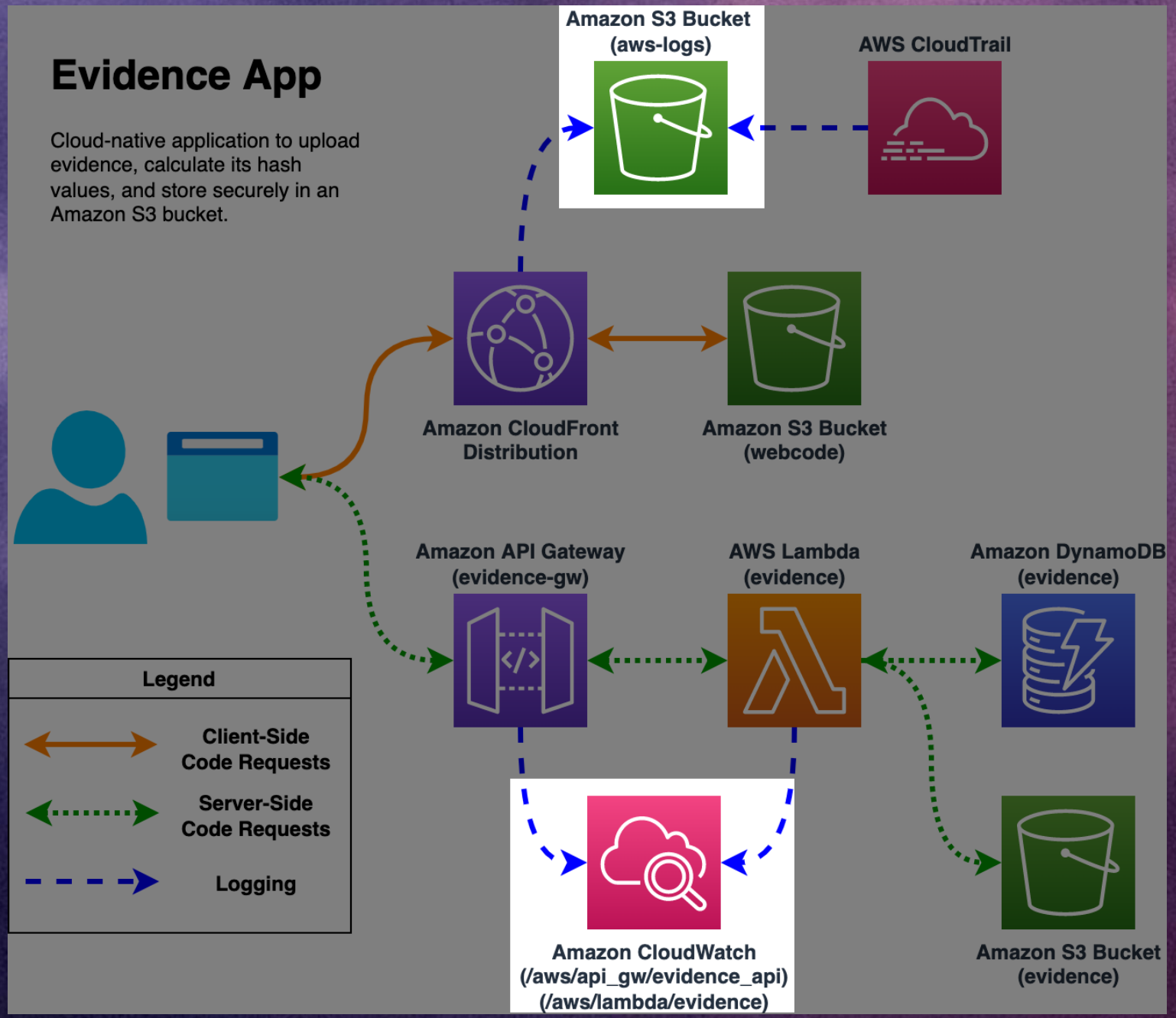
Complete *Exercise 2, 3, and 4* and then STOP!

Investigating the Attack

- **Web interactions** can be found in a few places:
  - **CloudFront** logs in `aws-logs` S3 bucket
  - **API Gateway** (requests to `/api`) logs in `/aws/api_gw/evidence_api` CloudWatch log group
- Server-side execution data (Lambda) found in `/aws/lambda/evidence` CloudWatch log group
- API calls by stolen credentials found in `aws-logs` S3 bucket
  - **S3 data events** also enabled and stored here
- **AWS CLI** and **Linux Kung-Fu** for the win!

Evidence App

Cloud-native application to upload evidence, calculate its hash values, and store securely in an Amazon S3 bucket.



Now It's Your Turn!

Complete Exercise *5, 6, and 7* and then STOP!

Conclusion

You did a **LOT** in this workshop by attacking and detecting:

- **Active Scanning (T1595)**
- **Cloud Infrastructure Discovery (T1580)**
- **Exploit Public-Facing Application (T1190)**
- **Command and Scripting Interpreter: Unix Shell (T1059.003)**
- **Unsecured Credentials (T1552)**
- **Data Destruction (T1485)**
- **Defacement: External Defacement (T1491.002)**

Now It's Your Turn!

Complete *Exercise 8* and... you're done!

Thanks for attending and please feel free to ask any questions!