



guides & tutorials

How to set up a custom domain name for Lambda & API Gateway with Serverless

written by



Alex DeBrie

With Serverless, it's easier than ever to deploy production-ready API endpoints. However, using AWS API Gateway results in odd hostnames for your endpoints. Further, these hostnames will change if you remove and redeploy your service, which can cause problems for existing clients.

In this guide, I'll show you how to map a custom domain name to your endpoints.

This post is the first in a two-part series. Check out the next post to configure [multiple Serverless services on the same domain name](#) for maximum microservice awesomeness.

Before you start

To get started, you'll need the [Serverless Framework](#) installed.

You should also have your desired domain name registered through AWS. Read the documentation on that [here](#).

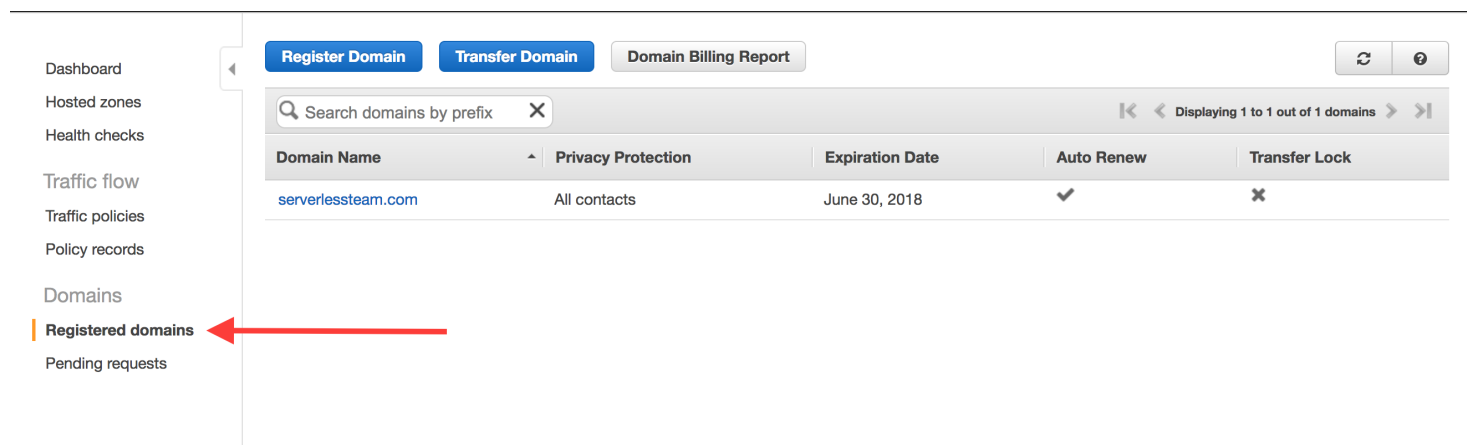
Getting a certificate for your domain

The steps below walk through setting up a certificate for your domain. If you already have a certificate issued, skip to the next section.

API Gateway requests must be served over HTTPS, so you need to get an SSL/TLS certificate. You may manually upload your certificate to Amazon, but I find it easier to use AWS Certificate

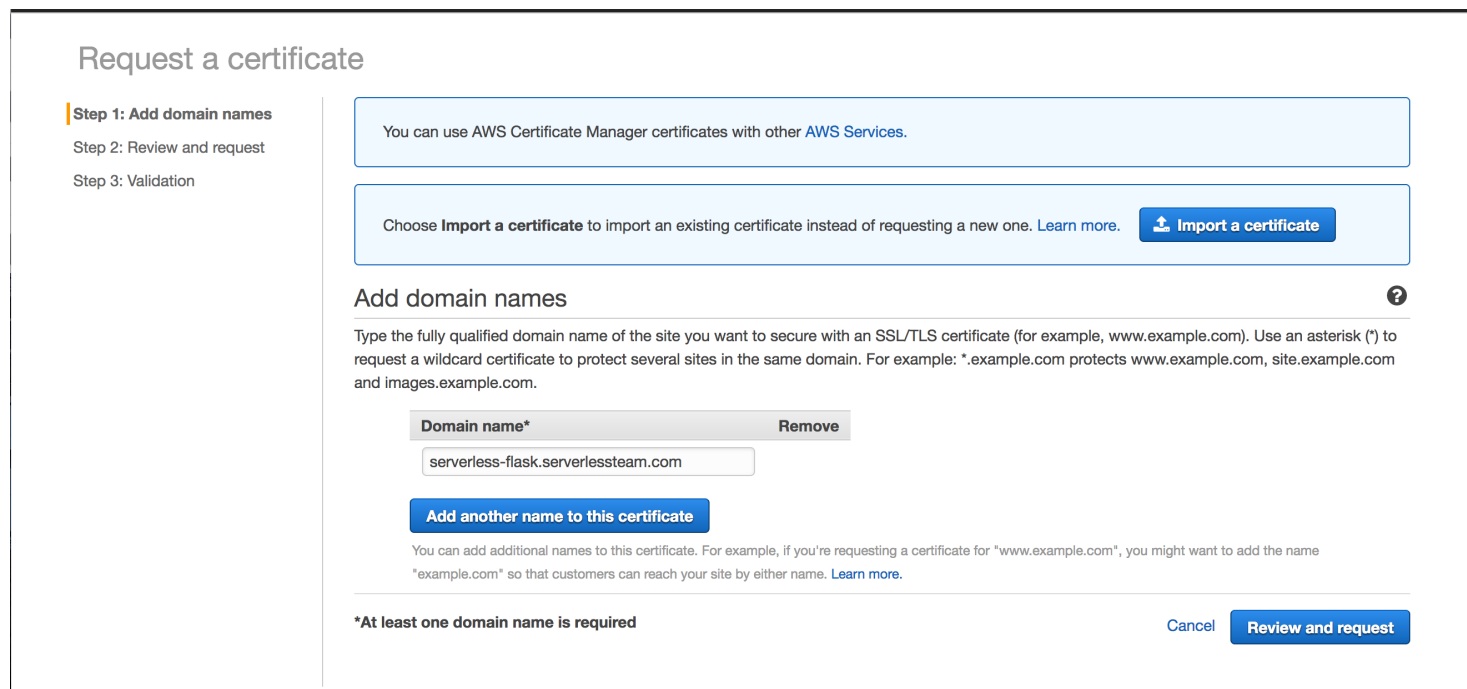
To set up the certificate:

First, make sure you have the domain name in your Registered Domains in Route 53.

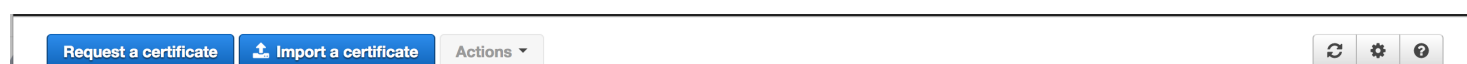


If you have a domain that's registered with a different registrar, you can transfer registration to Route 53. If you don't have a domain yet, you can purchase one through Route 53.

Once you have your domain, request a new certificate with the AWS Certificate Manager. **Note that you'll need to be in region us-east-1.** This is the only region that works with API Gateway.




Add the domain name you want, then hit Review and Request. After you confirm, it will say that a confirmation email has been sent to the registered owner of the domain to confirm the certificate. At this point, the certificate will be in a "Pending validation" status.



	Name ▾	Domain name ▾	Additional names	Status ▾	Type ▾	In use? ▾
<input type="checkbox"/>		serverless-flask.serverlessteam.com		Pending validation	Amazon Issued	No

Status


Validation not complete
 The status of this certificate request is "Pending validation". Further action is needed to validate and approve the certificate. [Learn more.](#)

Status Pending validation

Detailed status Email to validate the request was sent at 2017-09-01T14:27:28UTC but we have not yet received approval to issue the certificate for the following domains:

- ▶ **serverless-flask.serverlessteam.com**

Details

The registered owner of your domain will get a confirmation email from AWS. Click the link in the email to confirm issuance of the certificate. Once you do that, the certificate will change to an "Issued" status.

Request a certificate

Import a certificate

Actions ▾

↺

⚙

ℹ

☐

Name ▾

Domain name ▾

Additional names

Status ▾

Type ▾

In use? ▾

▶

serverless-flask.serverlessteam.com

Issued

Amazon Issued

No

Your certificate is ready to go! Move on to the next step to create a custom domain in API Gateway.

Create your serverless backend

Before you go any further, you should have a Serverless service with at least one function that has an HTTP event trigger. If you don't have that, you can use the code below. This example is in Python, but any runtime will work.

In a clean directory, add a `handler.py` file with the following contents:

subscribe

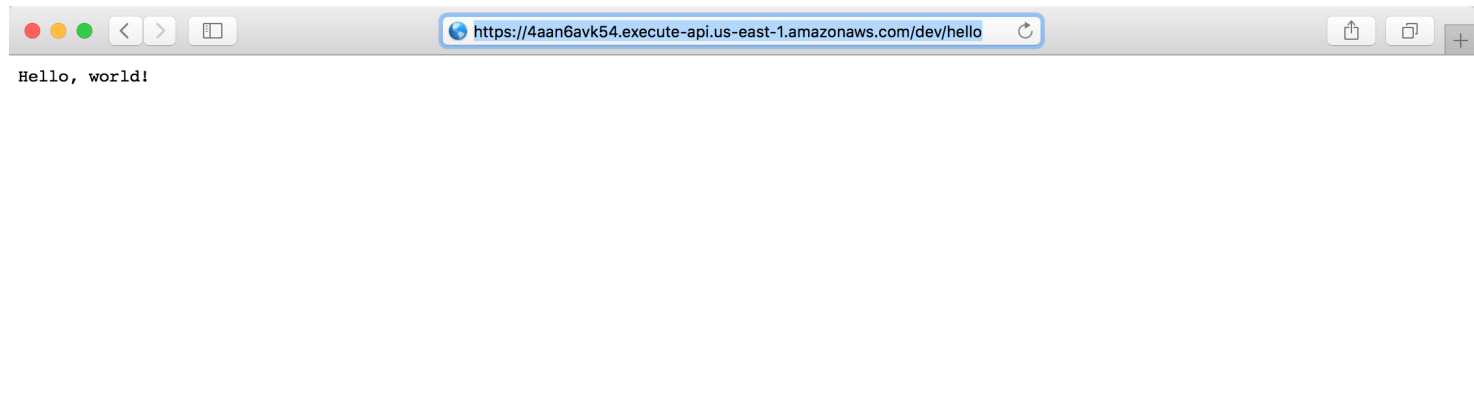
```
1 # handler.py
2
3 def hello(event, context):
4     response = {
5         "statusCode": 200,
6         "body": 'Hello, world!'
7     }
8
9     return response
10
11
12 def goodbye(event, context):
13     response = {
```

```
1 # serverless.yml
2
3 service: serverless-http
4
5 provider:
6     name: aws
7     runtime: python3.6
8
9 functions:
10     hello:
11         handler: handler.hello
12         events:
13             - http:
14                 path: hello
15                 method: get
16     goodbye:
17         handler: handler.goodbye
18         events:
19             - http:
20                 path: goodbye
21                 method: get
```

This `serverless.yml` file configures the functions to respond to HTTP requests. It says that the `hello` function will be triggered on the `/hello` path of your API Gateway, while the `goodbye` function will be triggered on the `/goodbye` path.

Run `sls deploy` to send your function to production:

```
1 $ sls deploy
2
3 Service Information
4 service: serverless-http
5 stage: dev
6 region: us-east-1
7 stack: serverless-http-dev
8 api keys:
9   None
10 endpoints:
11   GET - https://4aan6avk54.execute-api.us-east-1.amazonaws.com/dev/hello
12   GET - https://4aan6avk54.execute-api.us-east-1.amazonaws.com/dev/goodbye
13 functions:
14   hello: serverless-http-dev-hello
15   goodbye: serverless-http-dev-goodbye
```



and I get my **Hello, world!** response. If I change to the **/goodbye** endpoint, I'll get the **Goodbye, world!** response.

It's nice how easy this is to get a production API endpoint, but this still isn't ideal. My domain is impossible to remember (**4aan6avk54.execute-api.us-east-1.amazonaws.com**). Plus, if I ever remove my service and then redeploy, I'll get a new random domain.

Finally, the path is odd as well -- **/dev/hello** includes my stage as well as my actual page. I'd rather have a cleaner path. This shows the need for using a custom domain.

Create a custom domain in API Gateway

By this point, you should have an issued certificate and a Serverless service with an HTTP event configured. Now you need to create a custom domain in API Gateway that you can use with your

The easiest way to do this with Serverless is with the serverless-domain-manager plugin. Big thanks to the people at Amplify Education for developing this plugin.

To use the plugin, first make sure you have a `package.json` file in your service. Run `npm init -y` to generate one.

Then, you'll need to install the plugin in your service:

```
1 $ npm install serverless-domain-manager --save-dev
```

Then, configure it into your `serverless.yml`:

```
1 plugins:
2   - serverless-domain-manager
3
4 custom:
5   customDomain:
6     domainName: <registered_domain_name>
7     basePath: ""
8     stage: ${self:provider.stage}
9     createRoute53Record: true
```

Make sure you replace the `domainName` value with the domain name that you've configured your certificate for. If you're using a certificate that doesn't exactly match your domain name, such as a wildcard certificate, you'll need to specify the certificate name with a `certificateName` property under `customDomain`.

Once this is ready, you can create your custom domain with a single command:

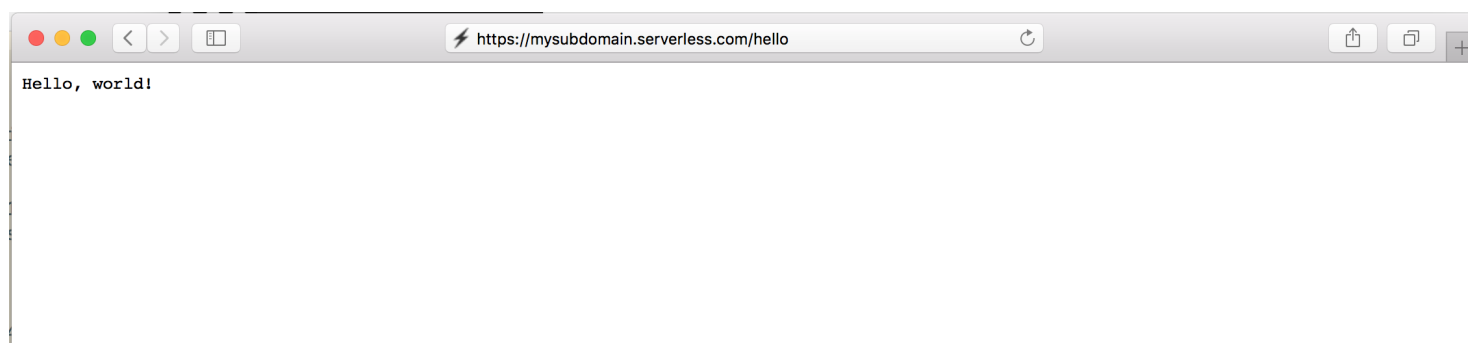
```
1 $ sls create_domain
2 Serverless: Domain was created, may take up to 40 mins to be initialized
```

As the output notes, it can take up to 40 minutes for your domain to be ready. This is how long it takes AWS to provision a CloudFront distribution. In my experience, it generally takes 10-20 minutes.

Once your domain name is ready, run `sls deploy` again to redeploy your service:

```
1    $ sls deploy
2
3    Service Information
4    service: serverless-http
5    stage: dev
6    region: us-east-1
7    stack: serverless-http-dev
8    api keys:
9      None
10   endpoints:
11     GET - https://4aan6avk54.execute-api.us-east-1.amazonaws.com/dev/hello
12     GET - https://4aan6avk54.execute-api.us-east-1.amazonaws.com/dev/goodbye
13   functions:
14     hello: serverless-http-dev-hello
15     goodbye: serverless-http-dev-goodbye
16
17   Serverless Domain Manager Summary
18   Domain Name
19     mysubdomain.serverless.com
20   Distribution Domain Name
21     a2fcnefljuqlt1.cloudfront.net
```

At the end of the `Service Information` block, you'll also get a `Serverless Domain Manager Summary` that shows the domain name associated with your domain. Now you can visit that domain in your browser with the cleaner path that you've assigned to your functions:



Voila! You have a much cleaner URL for your endpoints.

If you want to put multiple services on the same domain, be sure to check out the [follow up post!](#)



About Alex DeBrie

Alex DeBrie is a data engineer at Serverless.

How to use Serverless and Twilio to automate your communication channels

We're happy to announce that you can now deploy Twilio Functions using the Serverless Framework. Here's how!

written by Stefan Judis

How to use AWS Fargate and Lambda for long-running processes in a Serverless app

We'll show you how to process a video file that extracts a thumbnail in Amazon ECS using Fargate and Lambda

written by Rupak Ganguly

32 Comments

serverless-inc


1 Login

Recommend 11

Tweet

Share


Sort by Best



Join the discussion...

LOG IN WITH

OR SIGN UP WITH DISQUS ?



Cristian Sepulveda • 2 years ago

in my case a I have a wildcard certificate *.mydomain.com and I try to create the domain api.mydomain.com... you advise: "If you're using a certificate that doesn't exactly match your domain name, such as a wildcard certificate, you'll need to specify the certificate name with a certificateName property under customDomain." I tried it but do not worked, but the plugin page says: "If certificateName is not provided, the certificate will be chosen using the domain name." so I erase the



Joseph Astrahan → Cristian Sepulveda • 2 years ago

There is a tag under certificates that says name, you have to fill the name their correctly.

1 ^ | v • Reply • Share ›



Joseph Astrahan • 2 years ago

Something that is extremely important that is not noted in this tutorial is that this will ONLY WORK with subdomains NOT the root domain. I think this should be added to the post. I spent 6 hours figuring this out the hard way :P.

5 ^ | v • Reply • Share ›



Kareem Khattab → Joseph Astrahan • 2 years ago • edited

how do we host our serverless applications with our root domain? and the backend should always be a subdomain correct?

^ | v • Reply • Share ›



Ryan McMahon • 2 years ago

Hi, Is there a way to have a different domains for different stages? I want to have service-name.prod.X.X and service-name.beta.X.X for my beta and prod stages and the dev stage not linked to a custom domain.

3 ^ | v • Reply • Share ›



Arek Jaworski → Ryan McMahon • 2 years ago • edited

You can't do it with their plugin. Yet, you can do this with following 'hack' :)

Please check this StackOverflow question and answer for details:

<https://stackoverflow.com/q...>

2 ^ | v • Reply • Share ›



Cristian Alexis Araya • a year ago

I've tried this and everything seems to work but when I try out our ping method "mydomain.com/ping" I get "missing authentication token". Any ideas?

2 ^ | v • Reply • Share ›



Shadi Akiki → Cristian Alexis Araya • 25 days ago • edited

Actually, never mind me, I just figured it out :s It turns out that the stage name shouldn't be included in the new custom domain URL. So for example the URL <https://4aan6avk54.execute-api.us-east-1.amazonaws.com/dev/hello> with the custom domain api.example.com would become <https://api.example.com/hello> (notice no dev)

^ | v • Reply • Share ›



Shadi Akiki → Cristian Alexis Araya • 25 days ago

hey, did you figure this out?

^ | v • Reply • Share ›



Ronald Burgandy • 4 months ago

Am I understanding correctly that to have a domain in front of an API Gateway, you *must* transfer it to Route 53? Is that not a bit strange? Is it not possible to have the domain managed else where and then use DNS to configure the API Gateway?

1 ^ | v • Reply • Share ›

No you don't need to. All AWS does when you setup the custom API GW domain, is to create a CloudFront distribution behind the scenes and it shows you the domain name of that. If you CNAME that from Route53 or from your DNS provider of choice shouldn't matter at the end. To see the "Target Domain Name" of the Cloudfront distribution just open the API GW console and navigate to Custom Domains section here you see the list of custom domain distributions.

^ | v • Reply • Share ›



Bhavik Shah • a year ago • edited

I feel like I'm doing something off. Please help.

1. Our certificate is name *.myurl.com
2. Name Tag assigned to our certificate in Certificate Manager is: myurl.com
3. We are not hosting the certificate on route 53 but I have the certificate in Certificate Manager and it is issued so it should work properly.
4. My settings:

custom:

customDomain:

domainName: "api.myurl.com"

certificateName: "*.myurl.com"

basePath: "sls_test"

stage: \${self:provider.stage}

createRoute53Record: true

Error: 'api.myurl.com' was not created in API Gateway.

Error: Error: Could not find the certificate myurl.com.

Also, our Certificate is in "us-west-2" and I HAVE to use it for my functions. If it's us-west-2, is it still not gonna work?

1 ^ | v • Reply • Share ›



Chirag Rajkarnikar ➔ **Bhavik Shah** • 10 months ago

You need to set `endpointType: 'regional'` to make it work in region us-west-2

By default endpointType is set to edge which require certificates to be in us-east-1.

^ | v • Reply • Share ›



Bill • 2 years ago • edited

For anyone who use this plugin, it is not compatible with `serverless-plugin-split-stacks` :

<https://github.com/amplify-...>

1 ^ | v • Reply • Share ›



Vadorequest • 2 years ago

You say `Once your domain name is ready`, but how do we know when it's ready?

1 ^ | v • Reply • Share ›



Rob Weaver • 3 months ago

OK, this is nice, but it would be better if the base serverless.yml supported the domain name. In order to get this to sort of work, I have to run the 'sls deploy' (to create the certificate), then run the 'sls create_domain' to register the domain, then run 'sls deploy' again (and it only sometimes works)

So I do create the zone separately, which forces me to hard-code the zone ID like:

customDomain:

domainName: "accountnext-cloud-\${self:custom.stage}-workfront.com"

```
stage: ${self:provider.stage}
createRoute53Record: true
endpointType: 'regional'
hostedZoneId: 'XXXXXXXXXX'
```

And of course in my resources, I have the certificate:

```
# SSL certificate for API Gateway
SSLCertificate:
  Type: 'AWS::CertificateManager::Certificate'
  Properties:
    DomainName:
      "accountsrest.cloud-${self:custom.stage}.workfront.com"
    ValidationMethod: DNS
```

So by running `sls deploy`, I get a certificate. Then running the `sls create_domain` it seems to create the DNS record, and then I run `sls deploy` again just for safety.

^ | v • Reply • Share ›



Jiangshui Yu • 7 months ago

For those who encounter an error like `Error: Could not find the certificate xxx`, you need to understand the differences of API Gateway regional and Edge optimized. The default option of this plugin is Edge optimized, which means you will use cloud front services and you may be waiting for 20-40 minutes to apply, and in this case, your custom domain must be verified in us-east-1 ACM. So, if your custom domain in us-west-2 regional ACM, the plugin couldn't find it, but this error message is not friendly. If you want to choose regional, you can add `endpointType: 'regional'` to your serverless.yml and everything will be fine. After that, you may encounter some errors about IAM limitations, you need to make sure the serverless IAM user have the right policy for Route53, ACM, etc.

^ | v • Reply • Share ›



kuldeep yadav • a year ago

Can I create custom domain without ACM certificate? Such that my client can run on http rather than https.

^ | v • Reply • Share ›



Andy Groff • a year ago

When I try this I'm getting an access denied error at the URL. It is creating an alias to cloudfront and the cloudfronts origin is an S3 bucket. Shouldn't the origin be the API gateway? I'm not sure where i'm going wrong.

^ | v • Reply • Share ›



Tom Yip • a year ago

I tried installing this plug in with global install and serverless does not recognized it. Local install works. Is there a way to get server with this plug in installed globally? thanks!

^ | v • Reply • Share ›



Rob • a year ago

Could you finish out the details of the last part - don't really know how to connect the dots on the missing pieces and would love an example to work off of

^ | v • Reply • Share ›



Chris Shenton • 2 years ago

^ | v • Reply • Share ›



MoonJong • 2 years ago

I miss one point 'This is only working us-east-01'

If your terminal throw just 'Error: Could not find the certificate doamin.com.'

Please check your sertificart region.

^ | v • Reply • Share ›



Andrew Hansen • 2 years ago

Am I missing something?

To follow this process it looks like we need to have some pretty wide open IAM permissions (PutRolePolicy & DeleteRolePolicy) permissions on the provisioning IAM role which essentially empower the pipeline to grant allow *.* to anything it builds.

The only reason I can see that it needs those permissions is to create a role with two inline policies

- apigateway-permissions
- logs-permissions

These inline policies only allow:

```
"Action": ["apigateway:*"],
"Resource": "arn:aws:apigateway:*:*:*",
"Effect": "Allow"
&
"Action": ["logs:*"],
"Resource": "arn:aws:logs:*:*:*",
"Effect": "Allow"
```

Wouldn't it be better if we required (at least optionally allowed) a role to be passed in with

- apigateway-permissions
- logs-permissions

Especially since we may want to restrict apigateway resources instead of always allow *.*

^ | v • Reply • Share ›



sihan cen • 2 years ago

I have this error.

when I tried serverless create_domain, I got Serverless Error: Serverless command "create_domain" not found.

^ | v • Reply • Share ›



jad • 2 years ago • edited

im having this error

An error occurred: pathmapping - Only one base path mapping is allowed if the base path is empty..

^ | v • Reply • Share ›



Sam Ray • 2 years ago

Hi...is there a way to create custom domain regional endpoint using serverless framework or is there a plugin with which I can create one

^ | v • Reply • Share ›



Steven Wade • 2 years ago

^ | v • Reply • Share ›



Joseph Astrahan ➔ Steven Wade • 2 years ago • edited

Yes it does, I verified by testing.

^ | v • Reply • Share ›



Leo • 2 years ago

Thanks!

^ | v • Reply • Share ›



Luciano Panepucci • 2 years ago

Hi. I've just tried it and got "Error: Could not find the certificate" after running "sls create_domain"...

Any idea?

^ | v • Reply • Share ›



Luciano Panepucci ➔ Luciano Panepucci • 2 years ago

Nevermind.. I figured it out. I had to add the www domain as certificateName property under customDomain.

^ | v • Reply • Share ›

New to serverless?

To get started, pop open your terminal & run:

```
npm i
```

documentation

examples

plugins

Join our monthly newsletter

e-mail address

Product

CLI
Plugins
Components
Dashboard
Monitoring
CI/CD
Alerts
Policies
Debugging
Integrations

Docs

Pricing

Learn

Courses
What?
Why?
Use cases
Examples
Case studies
Comparisons

Services

Training
Support

Company

About us
Join us
Contact
Terms of service
Privacy Policy

Community

Blog
GitHub
Forum
Slack
Meetups
Partners

