Custom domain in AWS API Gateway

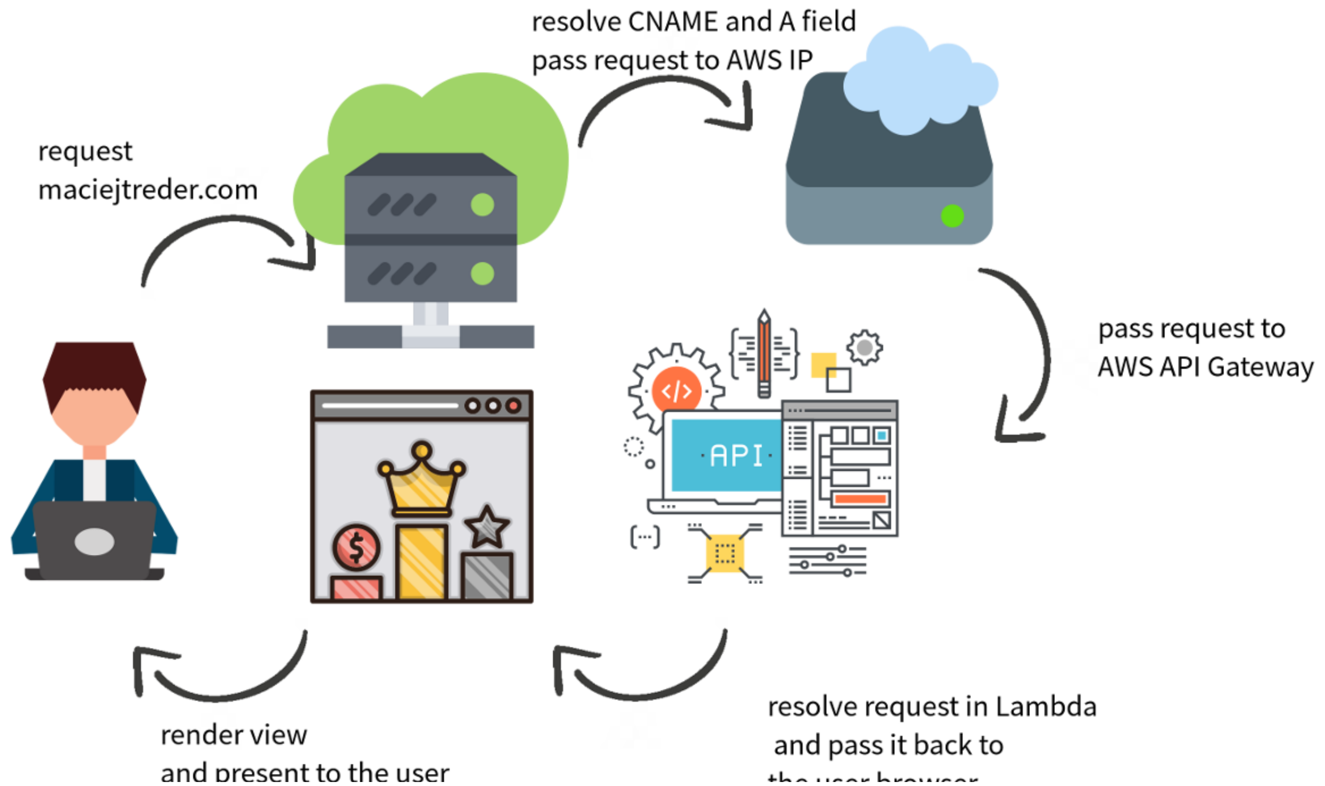
A step by step guide to set up a custom domain in Amazon Web Services API Gateway.

Maciej Treder Follow
May 21, 2018 · 6 min read ★

Serverless environment is getting more popular from day to day. FaaS (Function as a Service) gives you an ability to publish your app in small independent pieces and chain them up together into scalable microservices architecture. Today I would like to cover the process of using the custom domain (let’s say maciejtreder.com) in the AWS API Gateway.

The flow

What happens when a user requests a webpage? How it comes to the AWS? Take a look at high-overview flow-chart:



1. User requests the web-page/API by URL; Request comes to the DNS server
2. DNS server resolves CNAME and A field in the last CNAME chain element
3. The request comes to the AWS CloudFront (CNAME resolved by DNS), AWS passes it to the API Gateway
4. API Gateway looks for a connection between the Host header and declared APIs, passes the request to the Lambda and responds to the browser
5. The user receives the response (i.e. HTML code, which can be rendered by the browser)

CNAME

When the user is typing the domain address in their browser, in fact, they are addressing there request to the specific origin, with the unique IP address. How is it achieved? First of all, users request goes to the DNS server, which is mapping domain names to the IPs. That's the high overview. In fact, DNS server is doing much more, i.e. it is resolving CNAMEs. CNAME is a special kind of redirection. You can think about it as a 30x HTTP redirection, with one difference: user doesn't see that this redirect occurs. Imagine that DNS server is a postman, and the user request is a letter. Postman is taking a look at the letter and sees the address line (Park Lane ave 123), then he is looking into a notebook and sees annotation "All letters to Park Lane ave 123, should be delivered to Forest ave 321". That is what postman does. He doesn't need to inform the sender about that the address has changed, the only important thing is to deliver the dispatch.

This is an important concept which we are going to use later.

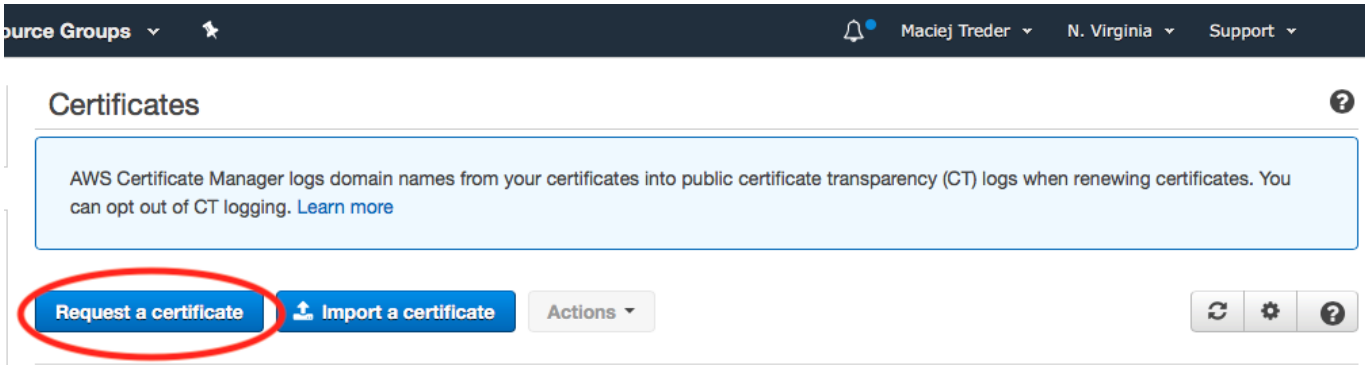
. . .

Let's start configuration!

Issue a certificate

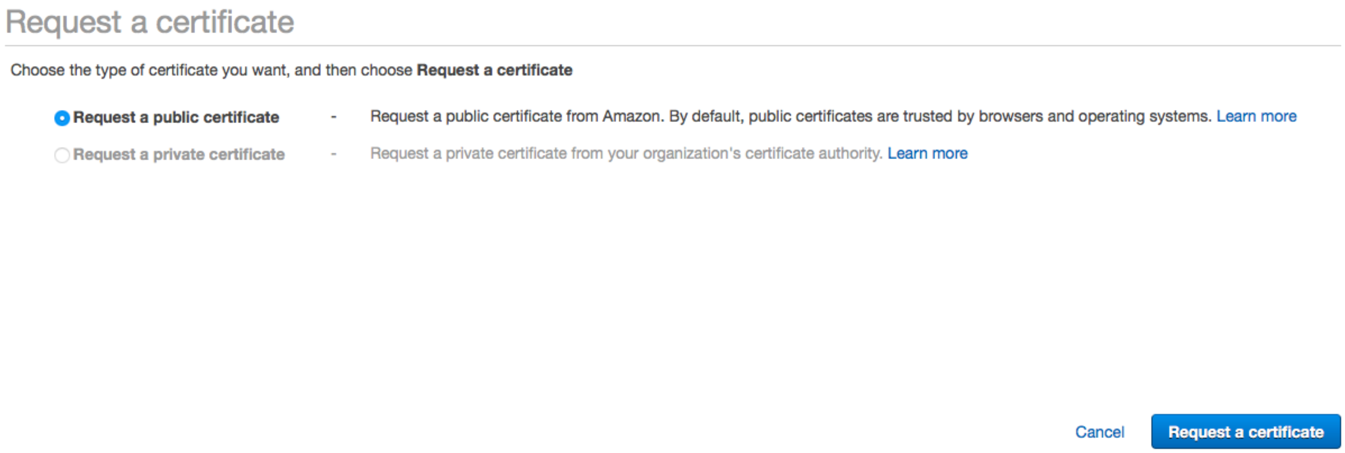
So you have your domain registered within some registrar (it doesn't need to be AWS, i.e. maciejtreder.com is registered within OVH). What we need to do now, is issue an SSL certificate from Amazon. It is necessary to route traffic to the AWS and don't get that traffic rejected by Amazon.

Login to AWS console and navigate to "Certificate Manager". **Make sure you set the region to US East — North Virginia**! That's really important. It doesn't matter in which region you want to deploy your API. SSL certificates must be issued in the **US East (North Virginia) region**. Now click on "Request a certificate" button:



AWS Certificate Manager logs domain names from your certificates into public certificate transparency (CT) logs when renewing certificates. You can opt out of CT logging. Learn more

In the next view, choose "Request a public certificate" and confirm your selection by clicking on "Request certificate" button.



Choose the type of certificate you want, and then choose **Request a certificate**

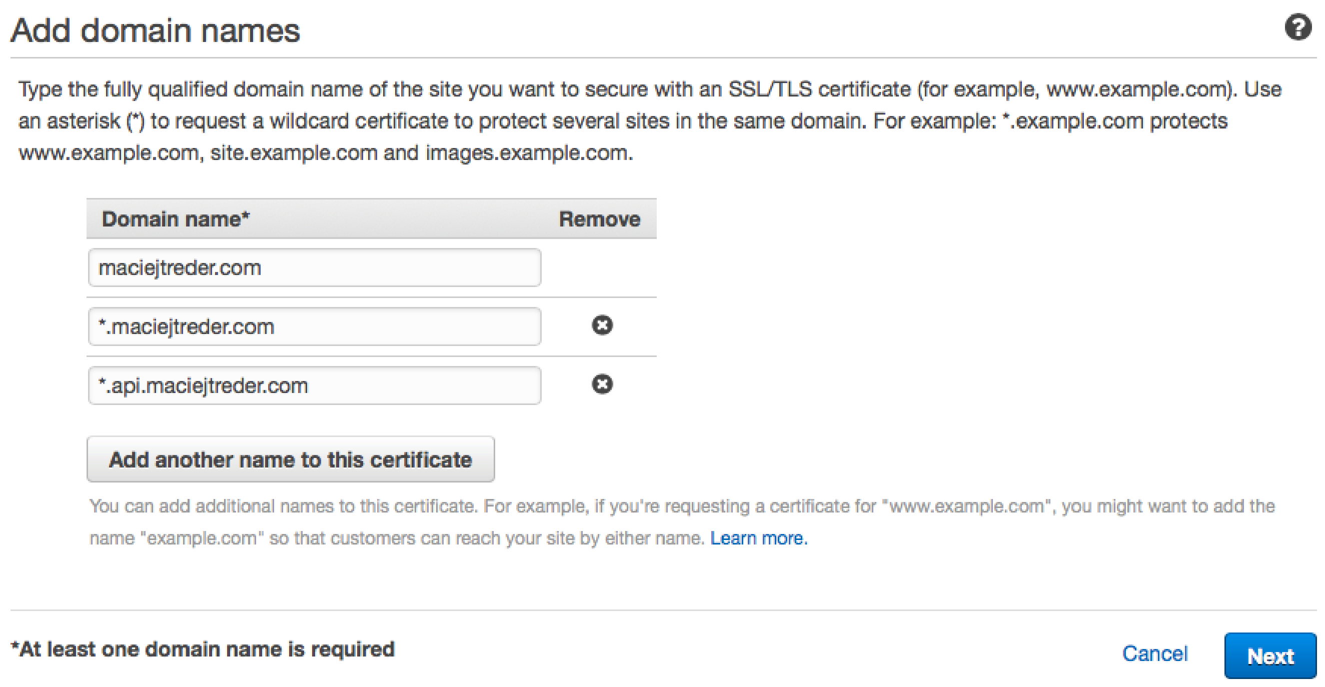
● **Request a public certificate** - Request a public certificate from Amazon. By default, public certificates are trusted by browsers and operating systems. Learn more

○ **Request a private certificate** - Request a private certificate from your organization's certificate authority. Learn more

Now you need to choose domain names for which you want to issue a certificate. Keep in mind that www.maciejtreder.com and maciejtreder.com **are two different domains**. The other important thing is, that a wildcard (*) matches **only one** "section" (from a dot . to dot .). So, if I want to issue the certificate which will work with following URLs:

- maciejtreder.com
- www.maciejtreder.com
- api.maciejtreder.com
- www.api.maciejtreder.com

My settings are:



Add domain names

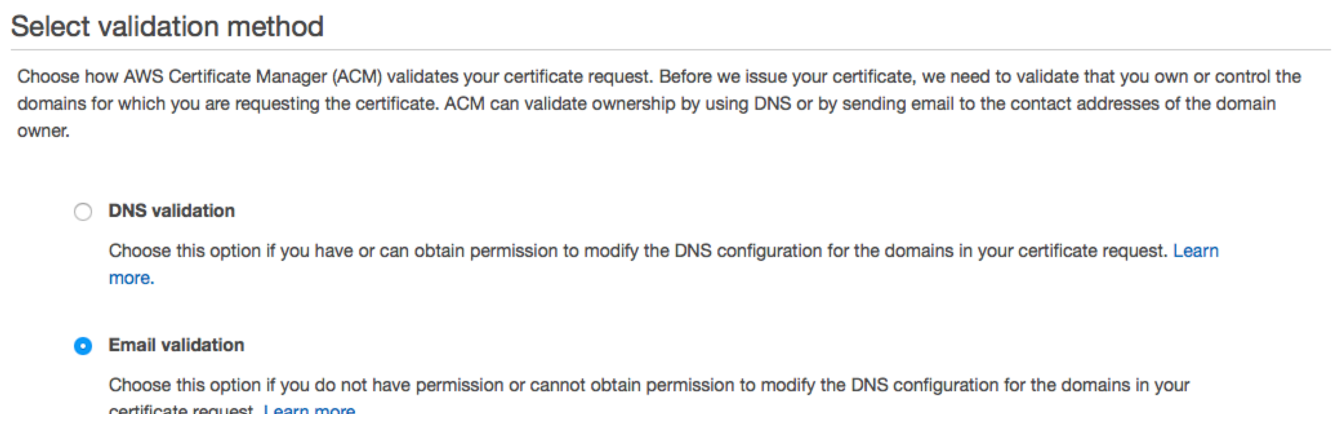
Type the fully qualified domain name of the site you want to secure with an SSL/TLS certificate (for example, www.example.com). Use an asterisk (*) to request a wildcard certificate to protect several sites in the same domain. For example: *.example.com protects www.example.com, site.example.com and images.example.com.

Domain name*	Remove
maciejtreder.com	
*.maciejtreder.com	⊗
*.api.maciejtreder.com	⊗

You can add additional names to this certificate. For example, if you're requesting a certificate for "www.example.com", you might want to add the name "example.com" so that customers can reach your site by either name. Learn more.

*At least one domain name is required

After clicking “next”, you will need to choose validation method. Amazon needs to makes sure, that you are the owner of the domain for which you are requesting the certificate. There are two ways for that. First one is manipulation of DNS entries, the second one is an e-mail validation. I prefer the second one:



Select validation method

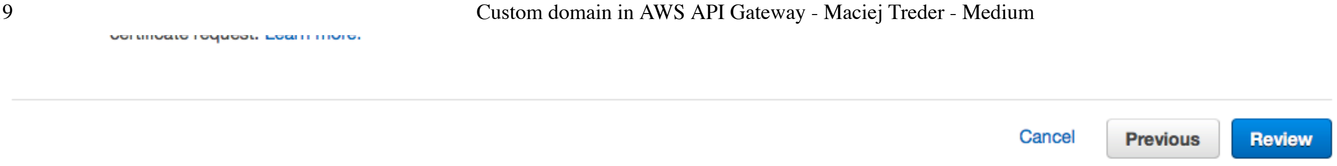
Choose how AWS Certificate Manager (ACM) validates your certificate request. Before we issue your certificate, we need to validate that you own or control the domains for which you are requesting the certificate. ACM can validate ownership by using DNS or by sending email to the contact addresses of the domain owner.

○ **DNS validation**

Choose this option if you have or can obtain permission to modify the DNS configuration for the domains in your certificate request. Learn more.

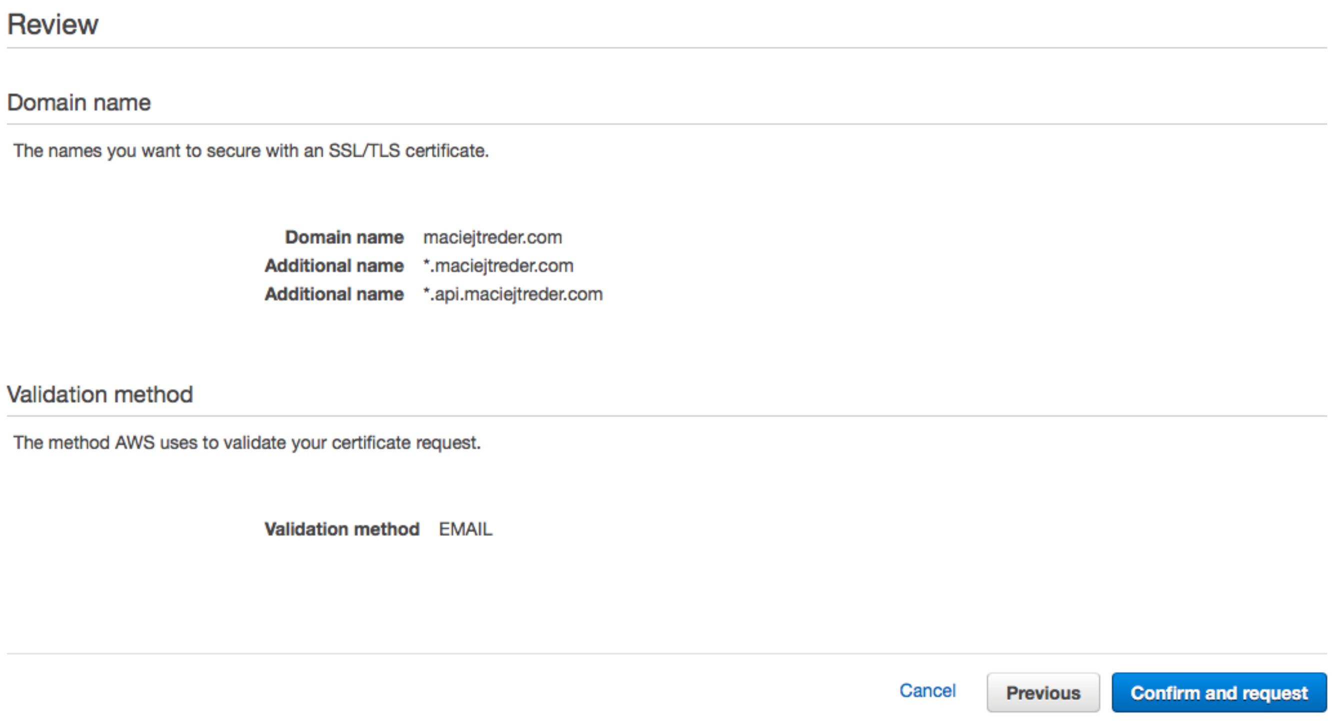
● **Email validation**

Choose this option if you do not have permission or cannot obtain permission to modify the DNS configuration for the domains in your certificate request. Learn more.


certificate request. Learn more.

What Amazon will do now, it will look up at DNS registry and send an e-mail with the activation link to the e-mail addresses listed in the “Registrant Contact”, “Admin contact” and “Tech Contact”. You can check your domain whois information on the ICANN whois page.

Ok. Now we are ready to click on the “Review” button, and then, finally “Confirm and Request”:


Review

Domain name

The names you want to secure with an SSL/TLS certificate.

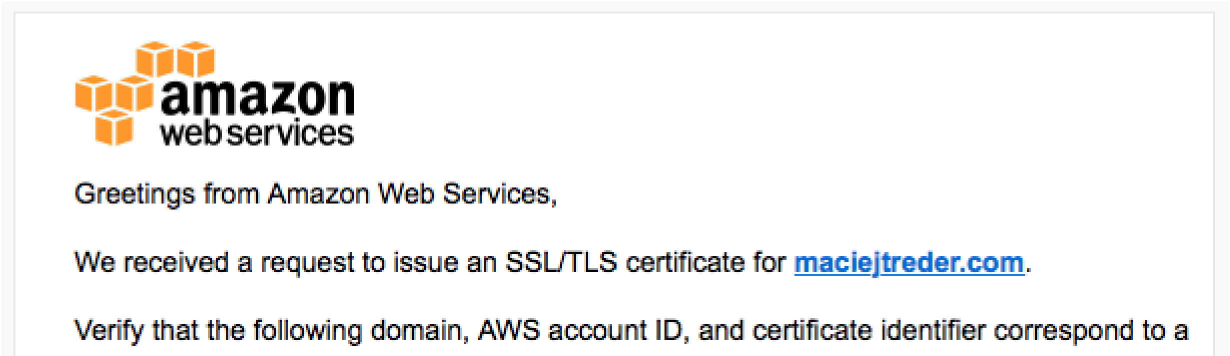
Domain name	maciejtreder.com
Additional name	*.maciejtreder.com
Additional name	*.api.maciejtreder.com

Validation method

The method AWS uses to validate your certificate request.

Validation method EMAIL

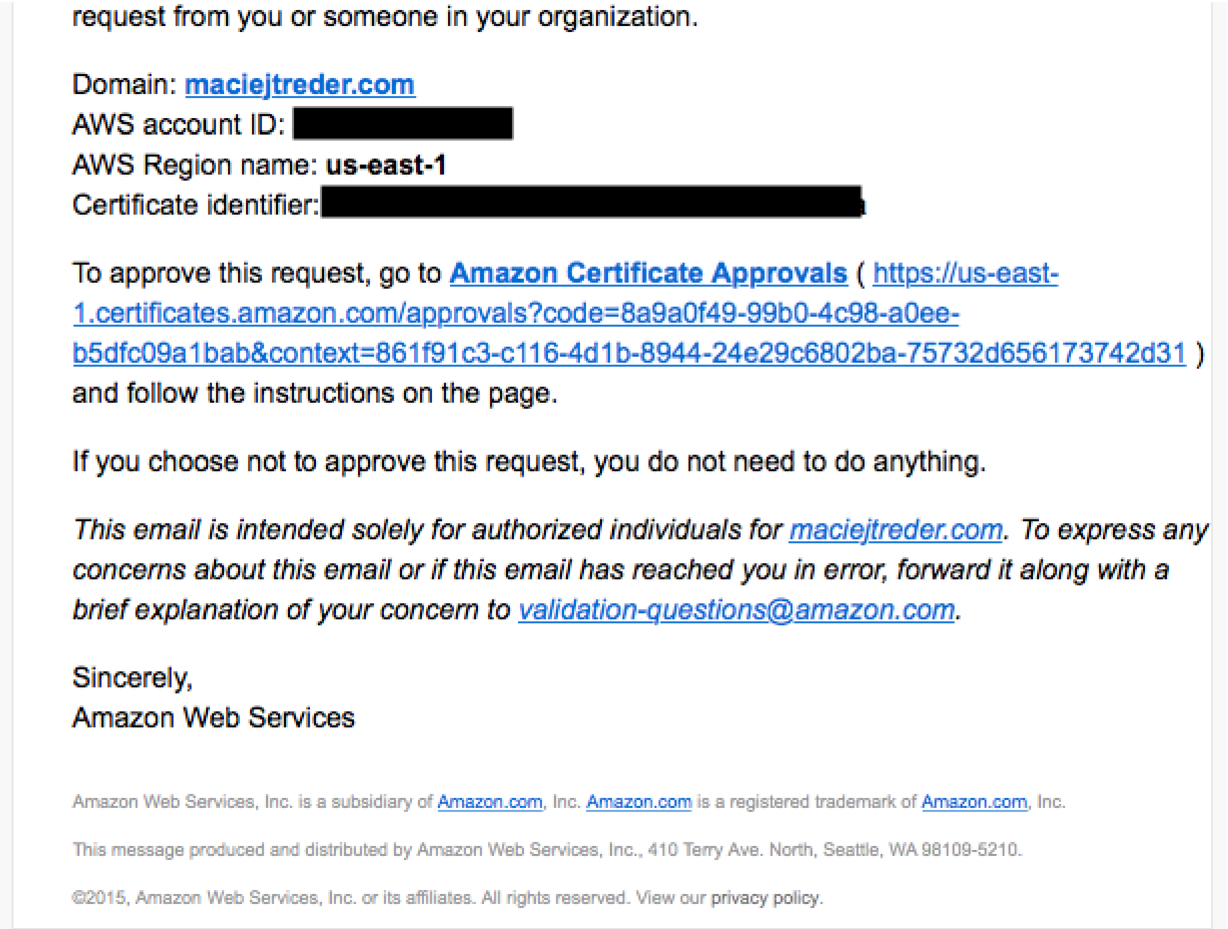
At the next page, you will get some information that your Certificate is requested and that the verification e-mail has been sent. After a while you should get it:



Greetings from Amazon Web Services,

We received a request to issue an SSL/TLS certificate for maciejtreder.com.

Verify that the following domain, AWS account ID, and certificate identifier correspond to a



request from you or someone in your organization.

Domain: maciejtreder.com
AWS account ID: [redacted]
AWS Region name: **us-east-1**
Certificate identifier: [redacted]

To approve this request, go to **Amazon Certificate Approvals** (https://us-east-1.certificates.amazon.com/approvals?code=8a9a0f49-99b0-4c98-a0ee-b5dfc09a1bab&context=861f91c3-c116-4d1b-8944-24e29c6802ba-75732d656173742d31) and follow the instructions on the page.

If you choose not to approve this request, you do not need to do anything.

This email is intended solely for authorized individuals for maciejtreder.com. To express any concerns about this email or if this email has reached you in error, forward it along with a brief explanation of your concern to validation-questions@amazon.com.

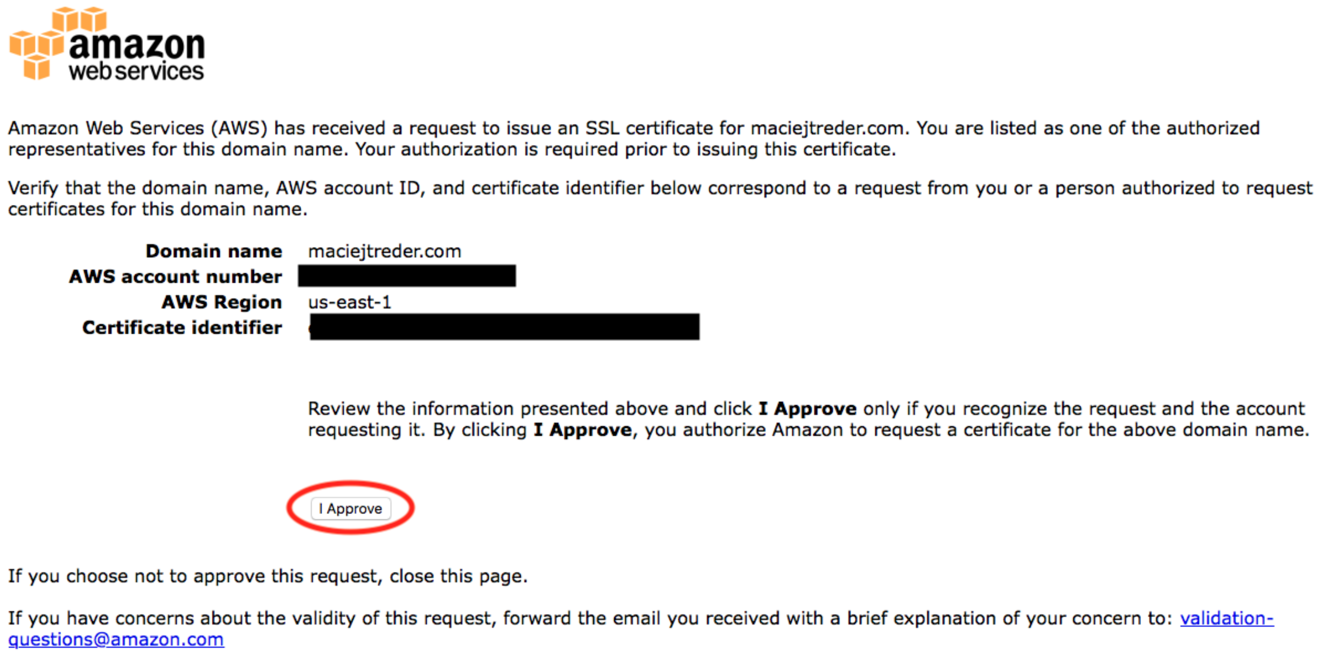
Sincerely,
Amazon Web Services

Amazon Web Services, Inc. is a subsidiary of Amazon.com, Inc. Amazon.com is a registered trademark of Amazon.com, Inc.

This message produced and distributed by Amazon Web Services, Inc., 410 Terry Ave. North, Seattle, WA 98109-5210.

©2015, Amazon Web Services, Inc. or its affiliates. All rights reserved. View our privacy policy.

Click on the link in the message and “approve your certificate”:



Amazon Web Services (AWS) has received a request to issue an SSL certificate for maciejtreder.com. You are listed as one of the authorized representatives for this domain name. Your authorization is required prior to issuing this certificate.

Verify that the domain name, AWS account ID, and certificate identifier below correspond to a request from you or a person authorized to request certificates for this domain name.

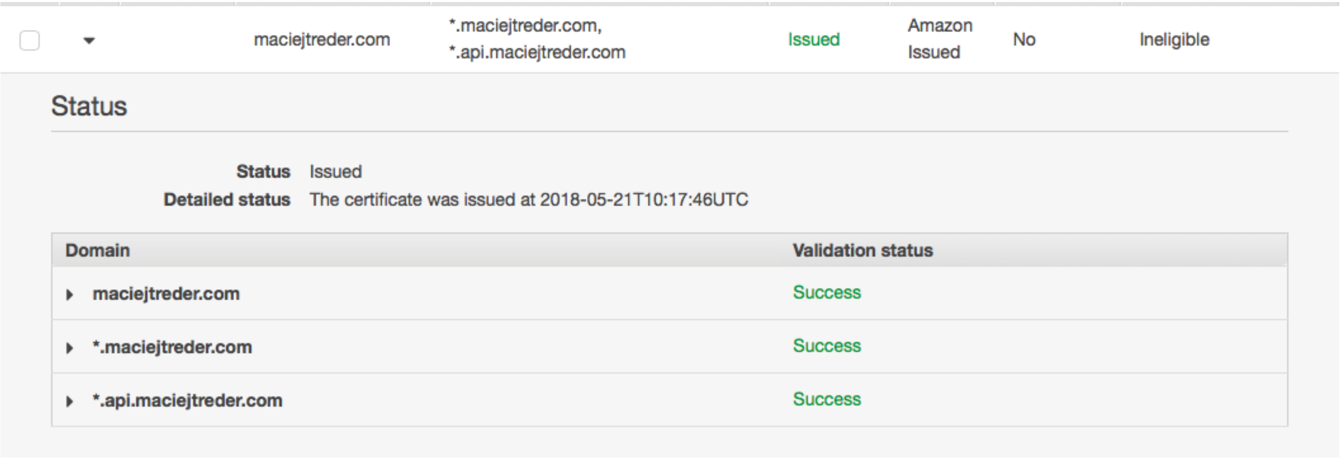
Domain name	maciejtreder.com
AWS account number	[redacted]
AWS Region	us-east-1
Certificate identifier	[redacted]

Review the information presented above and click **I Approve** only if you recognize the request and the account requesting it. By clicking **I Approve**, you authorize Amazon to request a certificate for the above domain name.

If you choose not to approve this request, close this page.

If you have concerns about the validity of this request, forward the email you received with a brief explanation of your concern to: validation-questions@amazon.com

Finally! We have it:



□	▾	maciejtreder.com	*.maciejtreder.com, *.api.maciejtreder.com	Issued	Amazon Issued	No	Ineligible

Status

Status Issued
Detailed status The certificate was issued at 2018-05-21T10:17:46UTC

Domain	Validation status
▸ maciejtreder.com	Success
▸ *.maciejtreder.com	Success
▸ *.api.maciejtreder.com	Success

Setup custom domain on your API

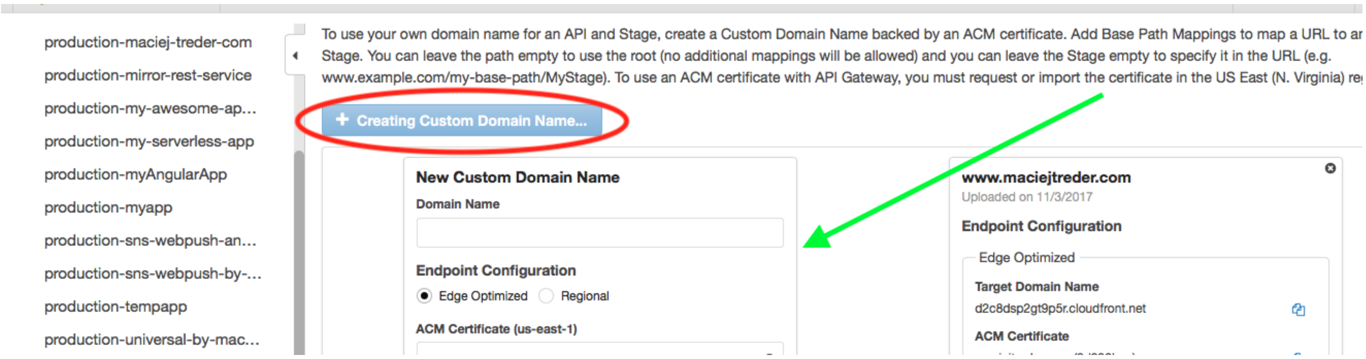
Now you can Navigate to the API Gateway (**remember to change region to this one where your API is located**).

. . .

I assume that you have your API set up, and you have the “maintenace link” (i.e., https://rfi6svv20f.execute-api.eu-central-1.amazonaws.com/production). If not check out my article How to Deploy JavaScript & Node.js Applications to AWS Lambda for Twillio. You can also (if you have some Angular application) take a look at @ng-toolkit/serverless library for Angular app (An article about usage of this plugin can be found here: Angular & Serverless)

. . .

Now in the left-side menu choose "Custom Domain Names" link and click on "Create Custom Domain Name" button. New, blank custom domain name form should appear on your screen:



production-maciej-treder-com
production-mirror-rest-service
production-my-awesome-ap...
production-my-serverless-app
production-myAngularApp
production-myapp
production-sns-webpush-an...
production-sns-webpush-by-...
production-tempapp
production-universal-by-mac...
+ Creating Custom Domain Name...

To use your own domain name for an API and Stage, create a Custom Domain Name backed by an ACM certificate. Add Base Path Mappings to map a URL to ar Stage. You can leave the path empty to use the root (no additional mappings will be allowed) and you can leave the Stage empty to specify it in the URL (e.g. www.example.com/my-base-path/MyStage). To use an ACM certificate with API Gateway, you must request or import the certificate in the US East (N. Virginia) re

New Custom Domain Name

Domain Name

Endpoint Configuration

● Edge Optimized ○ Regional

ACM Certificate (us-east-1)

www.maciejtreder.com

Uploaded on 11/3/2017

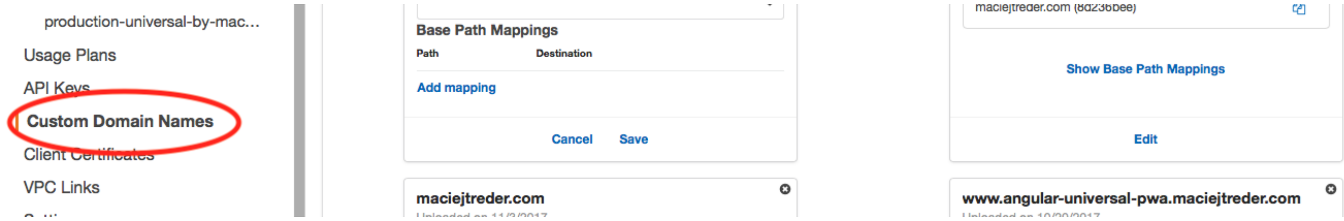
Endpoint Configuration

Edge Optimized

Target Domain Name

d2c8dsp2gt9p5r.cloudfront.net

ACM Certificate



production-universal-by-mac...
Usage Plans
API Keys
Custom Domain Names
Client Certificates
VPC Links

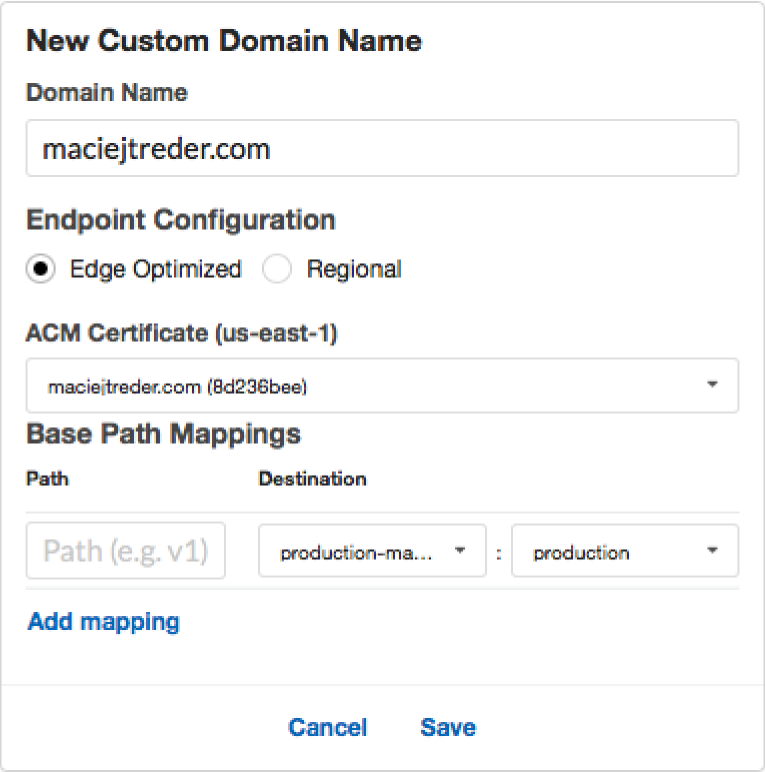
Base Path Mappings
Path Destination
Add mapping
Cancel Save

maciejtreder.com

maciejtreder.com (8d236bee)
Show Base Path Mappings
Edit

www.angular-universal-pwa.maciejtreder.com

Fill up the form, and remember: www.maciejtreder.com and maciejtreder.com are two different domains, so you need to customize them separately (both of them can point to the same lambda — no problem). Choose the certificate which you created in the previous step, add mapping to your lambda application, and choose the stage which you want to use:



New Custom Domain Name

Domain Name: maciejtreder.com

Endpoint Configuration: ● Edge Optimized ○ Regional

ACM Certificate (us-east-1): maciejtreder.com (8d236bee)

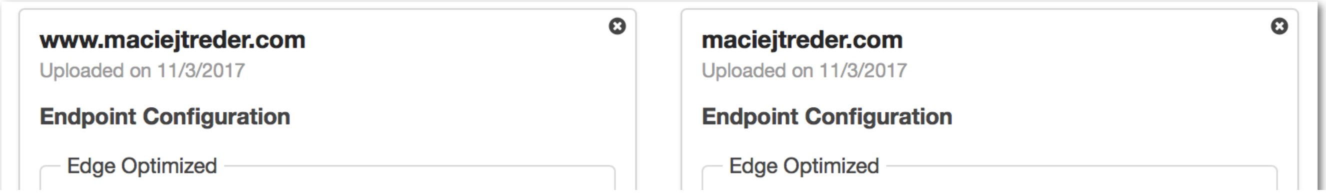
Base Path Mappings

Path	Destination	
Path (e.g. v1)	production-ma...	production

Add mapping

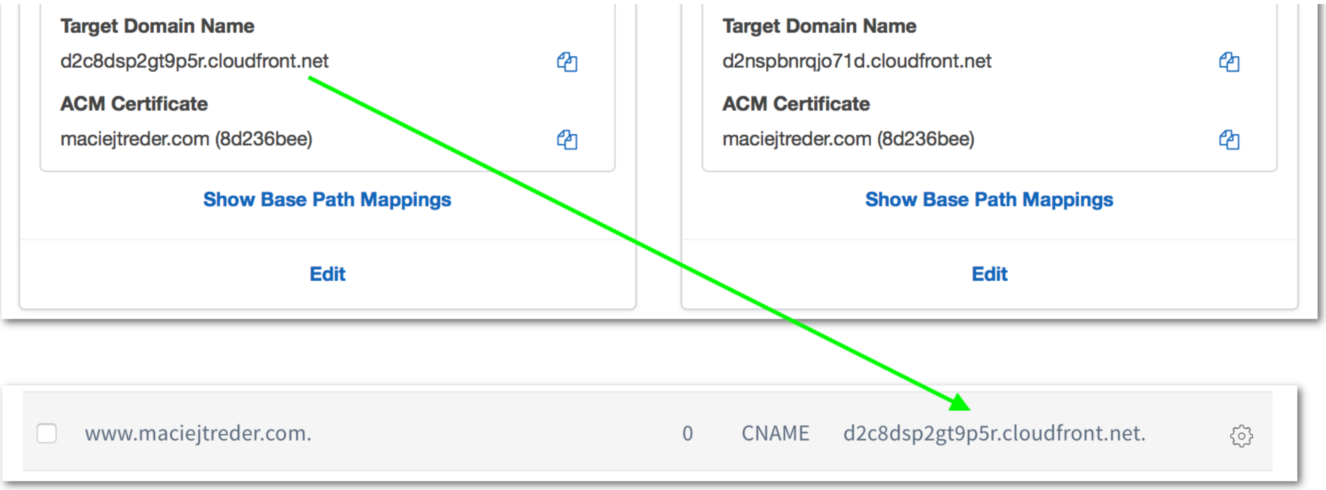
Cancel Save

After adding your custom domains, you should see the “Target Domain Name”, set up in CloudFront (AWS is doing that automatically). Do you remember section about CNAME? We are going to use that target domain in the CNAME field of your domain entry in the DNS. This is how the configuration of my domain looks like:



www.maciejtreder.com
Uploaded on 11/3/2017
Endpoint Configuration
Edge Optimized

maciejtreder.com
Uploaded on 11/3/2017
Endpoint Configuration
Edge Optimized



Target Domain Name
d2c8dsp2gt9p5r.cloudfront.net
ACM Certificate
maciejtreder.com (8d236bee)

Target Domain Name
d2nspbnrqjo71d.cloudfront.net
ACM Certificate
maciejtreder.com (8d236bee)

□	www.maciejtreder.com.	0	CNAME	d2c8dsp2gt9p5r.cloudfront.net.

You’re done! Now you need only to wait for DNS data propagation across the web! You can check if the data is propagated already using whatsmydns tool.

. . .

Thanks for reading!

Maciej Treder, https://www.maciejtreder.com

Follow me on Twitter and GitHub to be notified about my new activites.

If you want, you can donate my activity on OpenCollective, DonorBox or LiberaPay.

AWS Lambda Serverless Amazon Web Services Web Development Microservices

About Help Legal