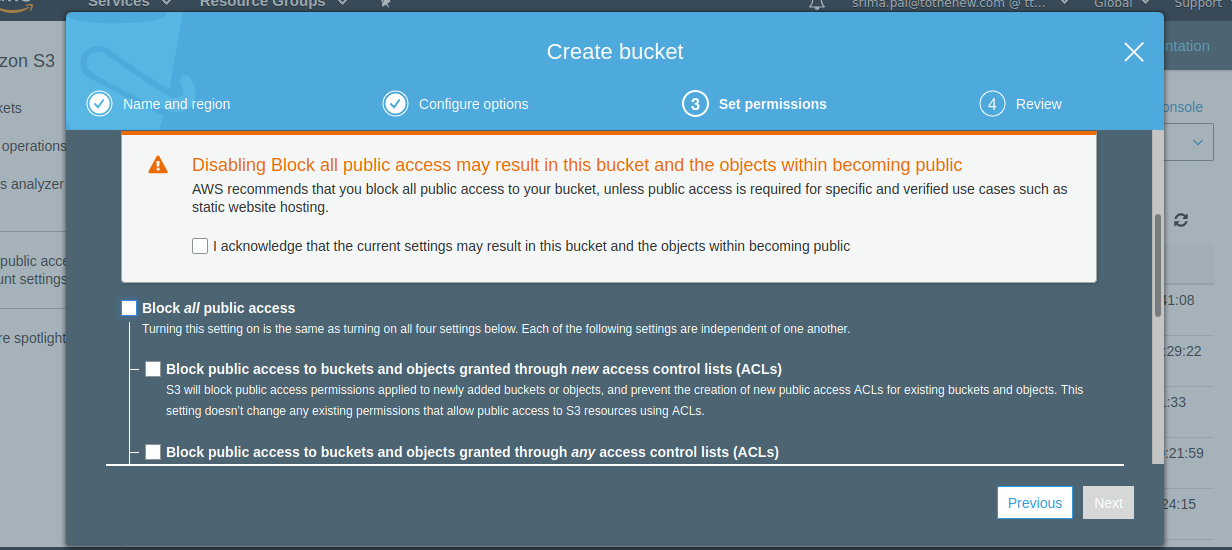
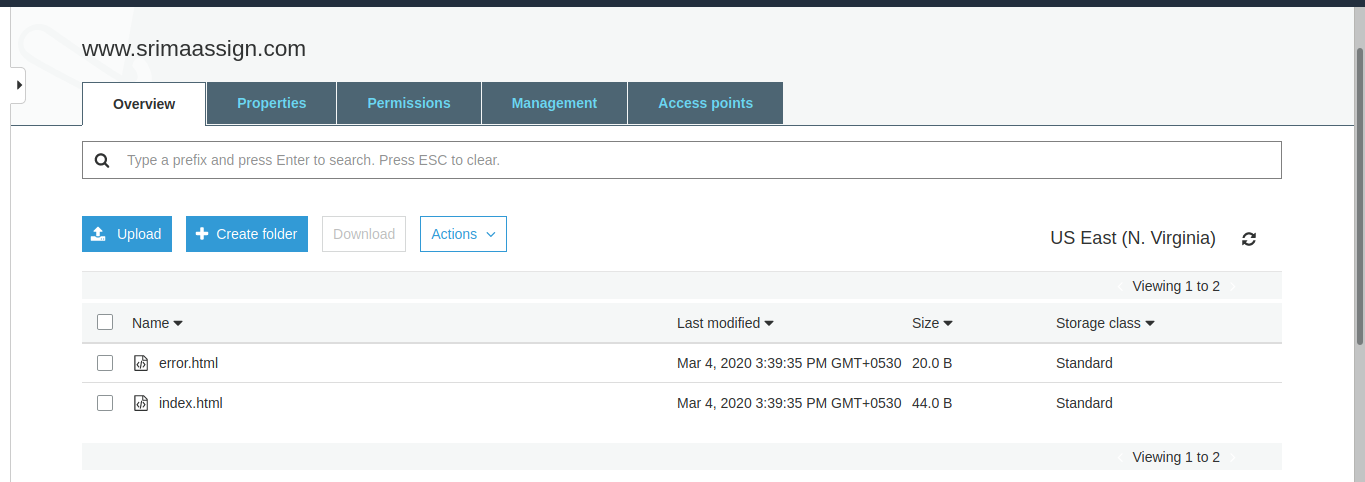
**1. Static website hosting using s3(what is index and error page).**

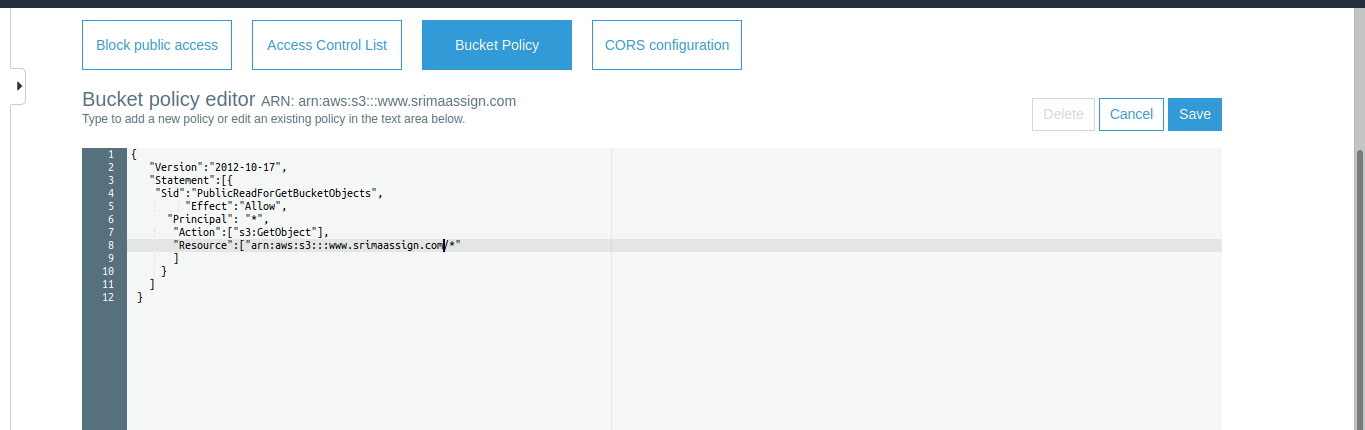
Make a public bucket, with the domain name you want to host the website with.



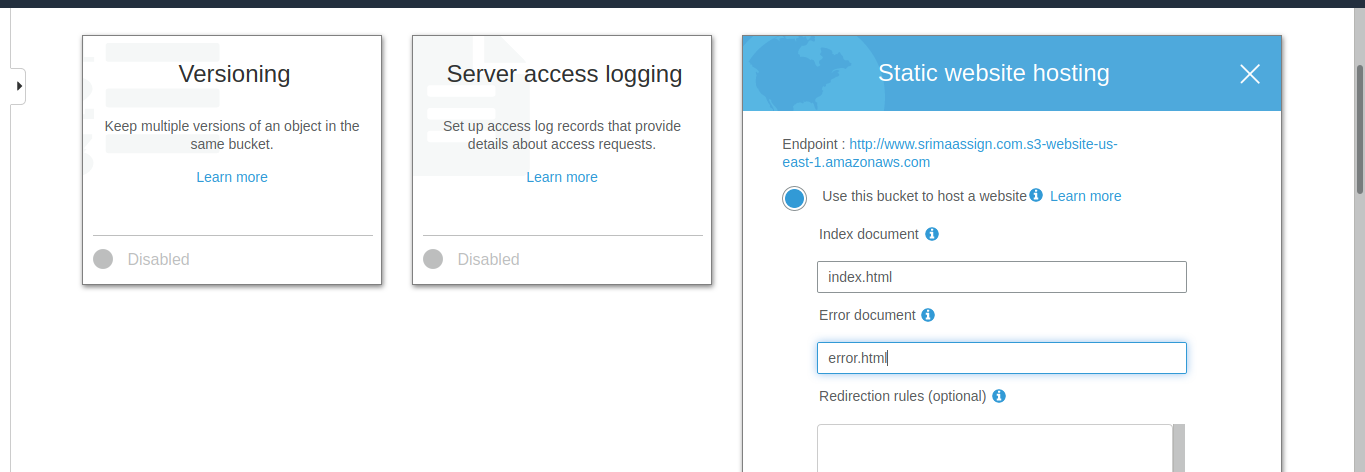
Upload your error page and index page in bucket



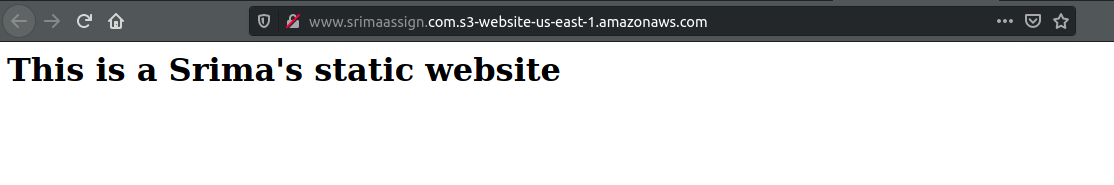
Make the bucket public (After selecting the bucket, go to permissions)

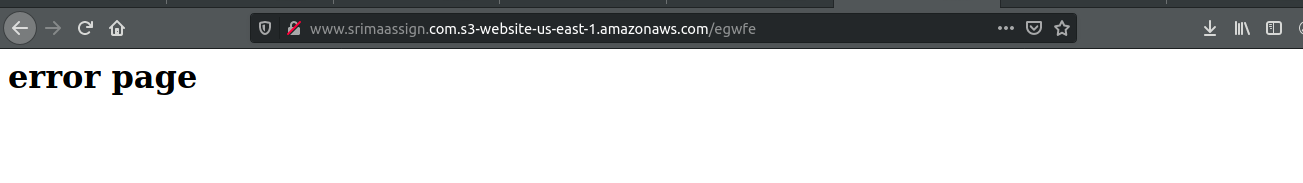


Turn static website hosting on



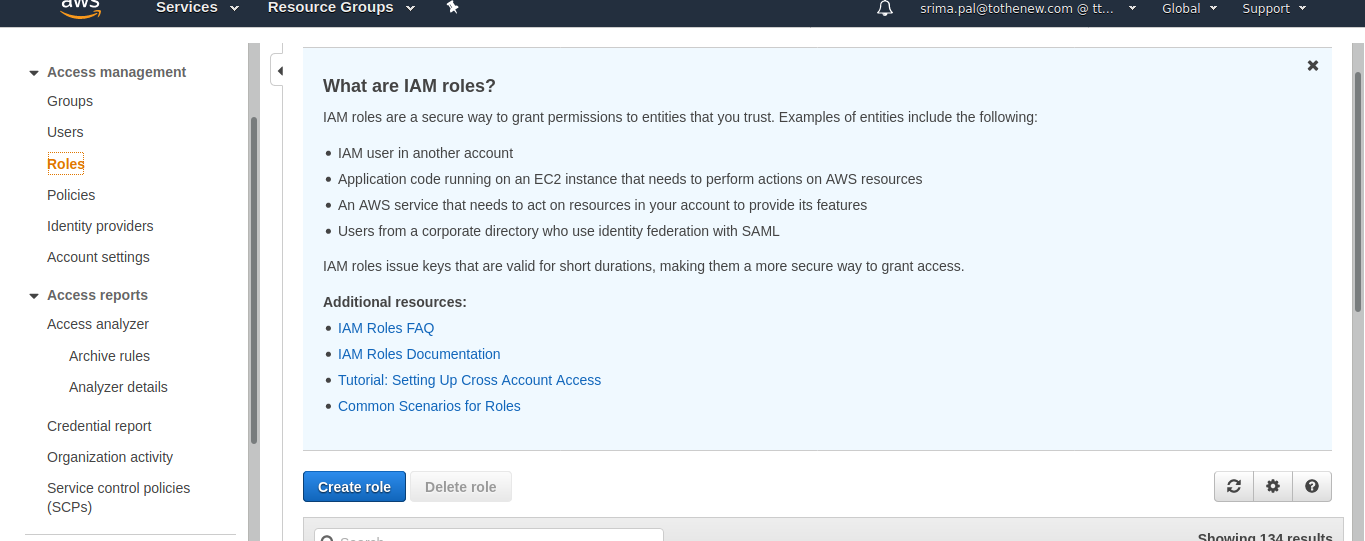
# Click on Static website hosting again ( after everything was saved ) and you should see an endpoint URL

****Giving wrong url

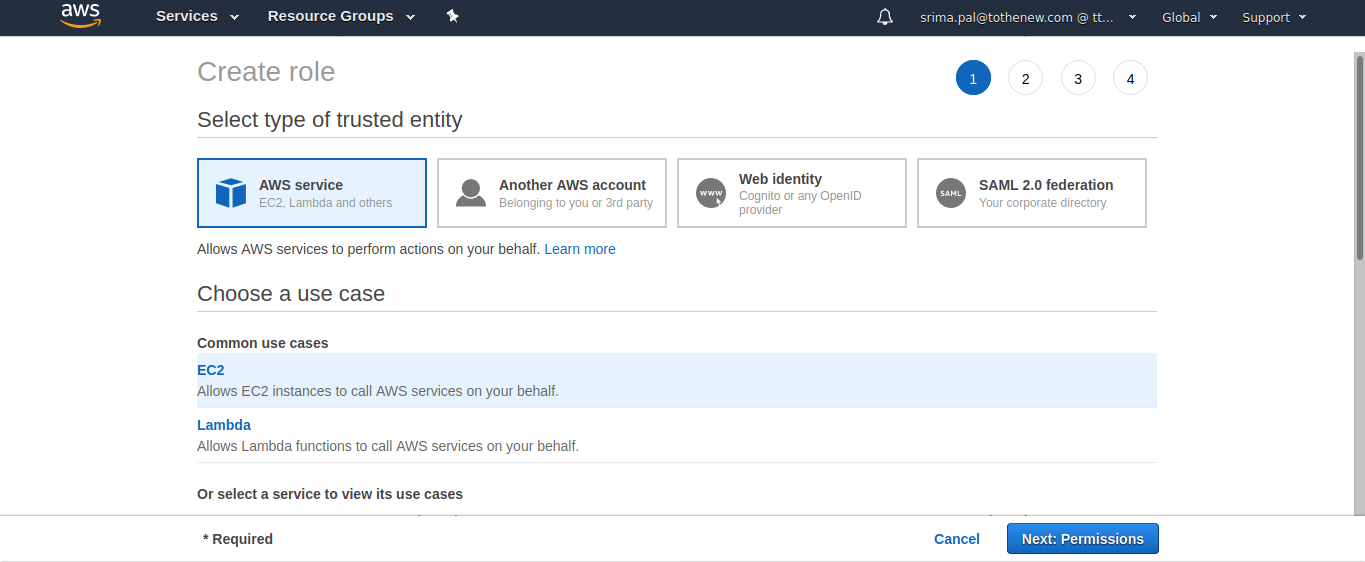
****

**2. Create an assume role to access s3 using ec2.**

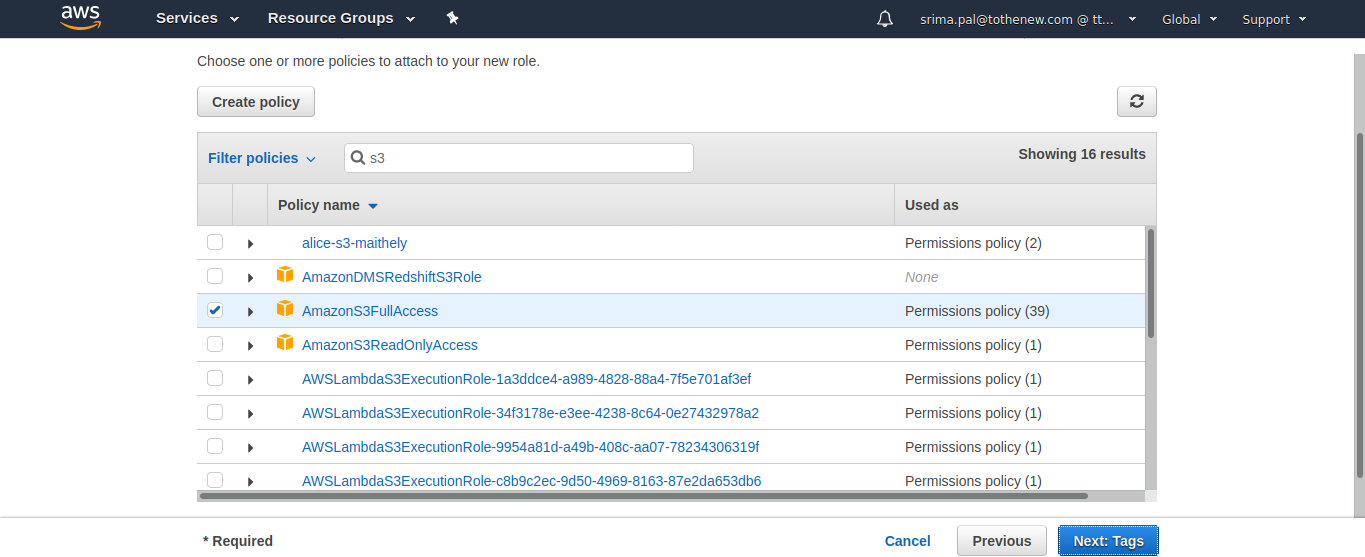
Go to IAM and select create role



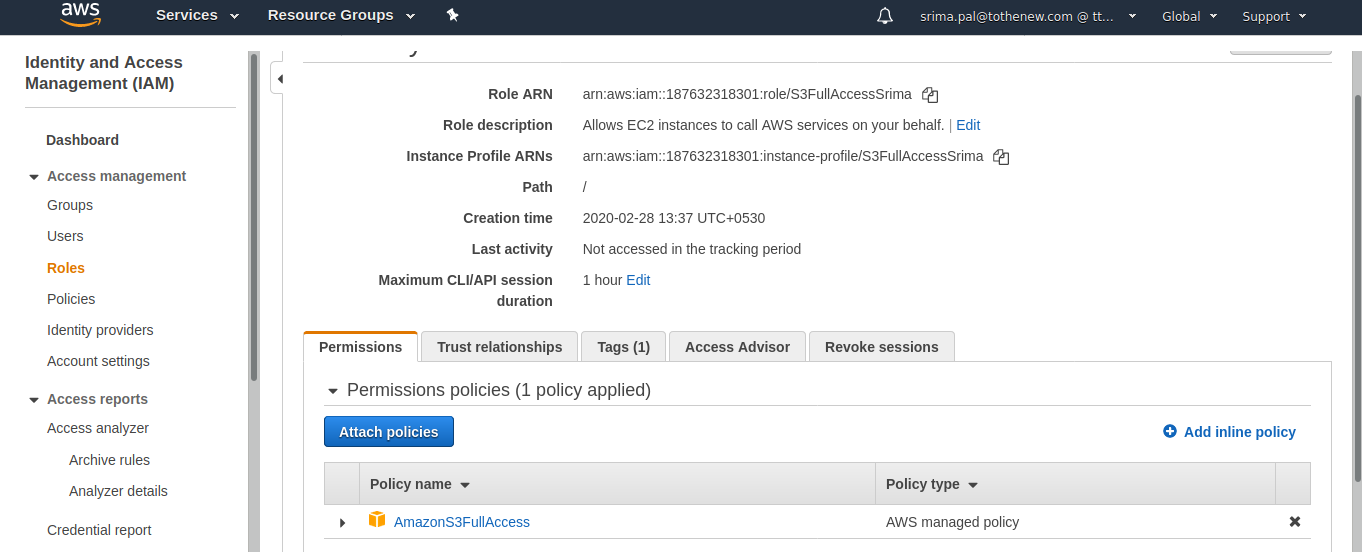
Select the service you want to give access to s3



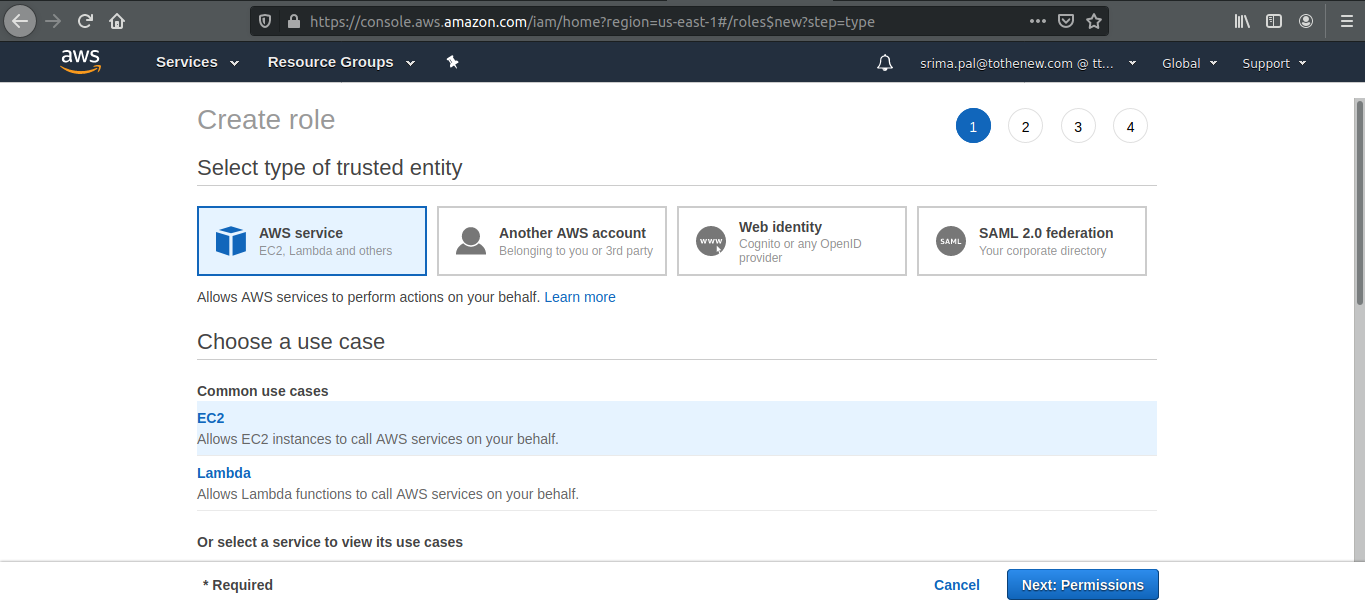
Select a policy with permission you want to give to ec2 (s3 full access)



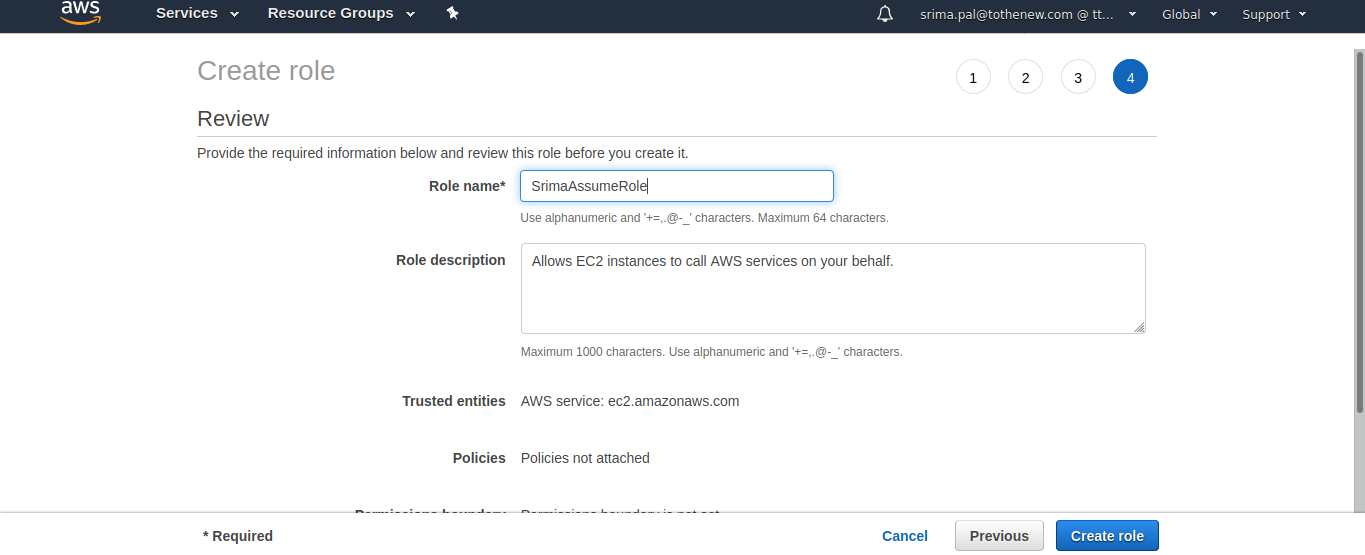
Create a role



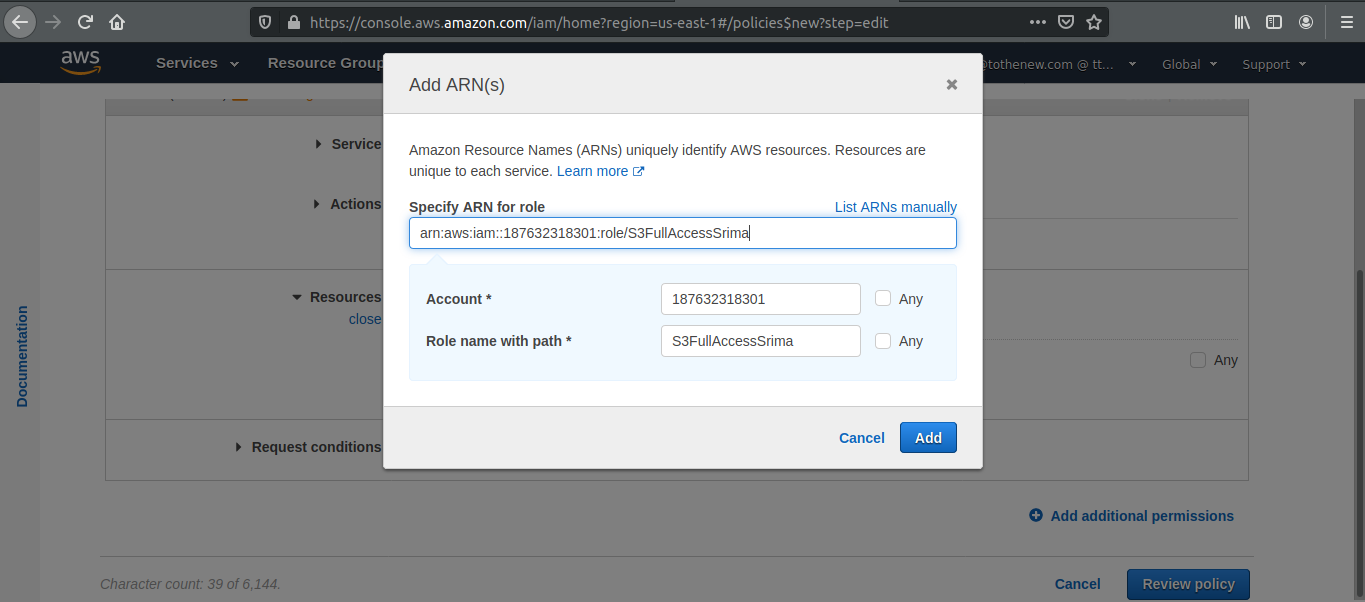
Create a role with service EC2



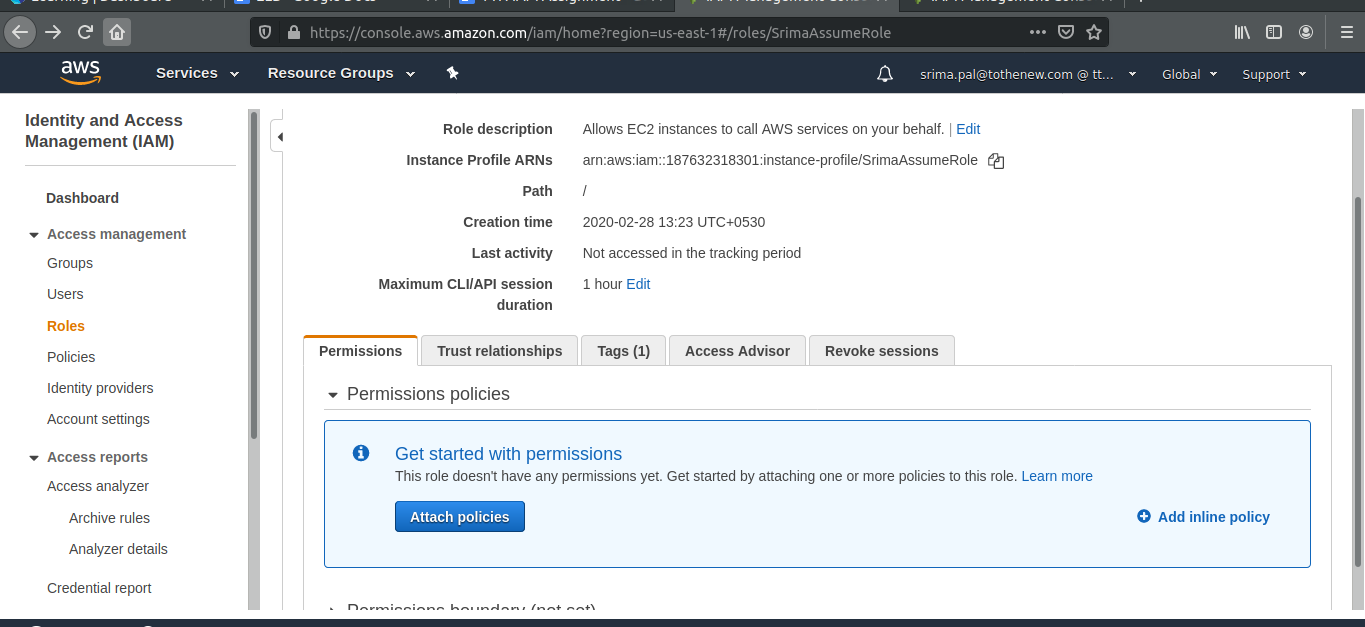
Create an assumed role with no attached policy

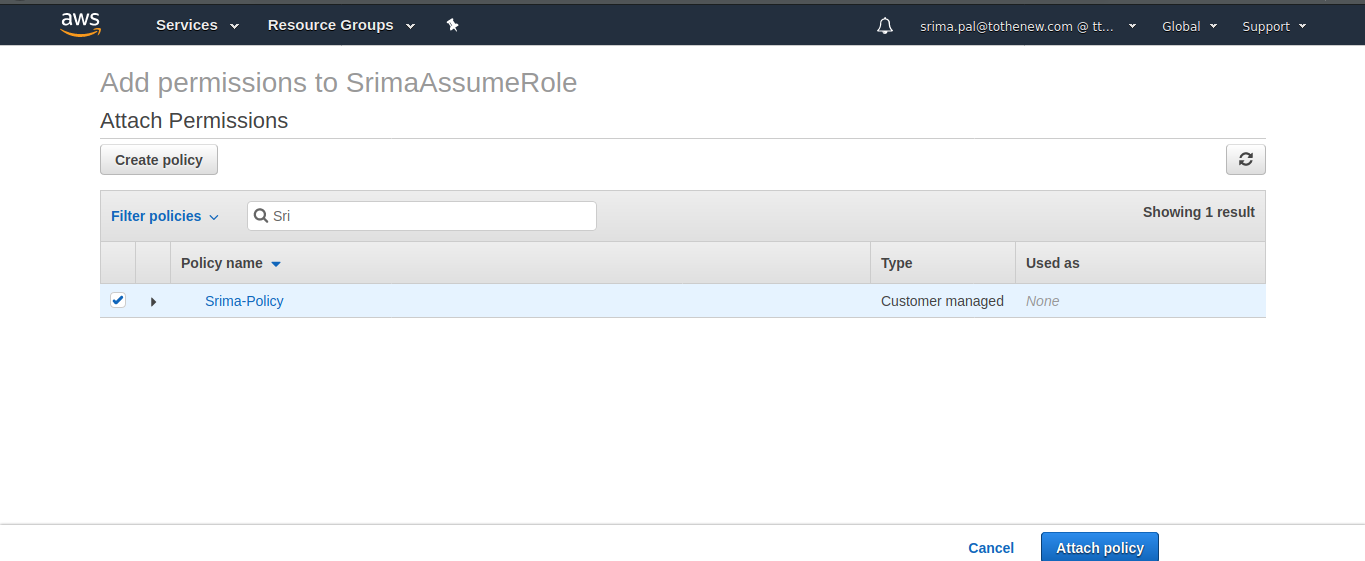


Create a policy to attach with the above role

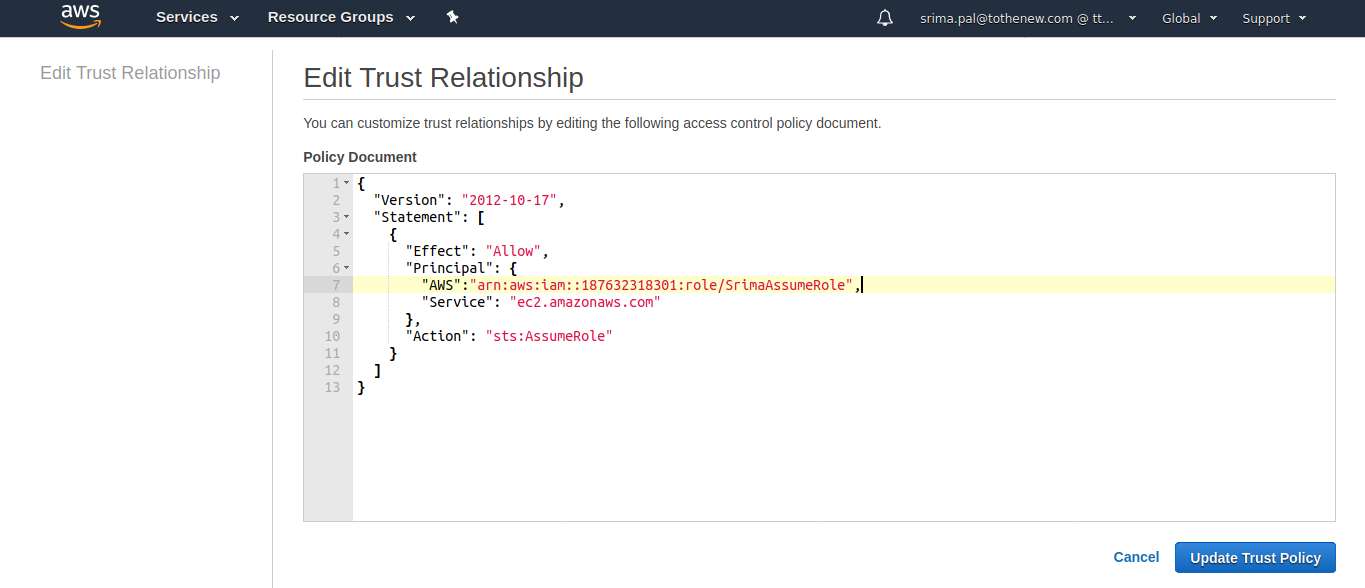


Attach the created policy

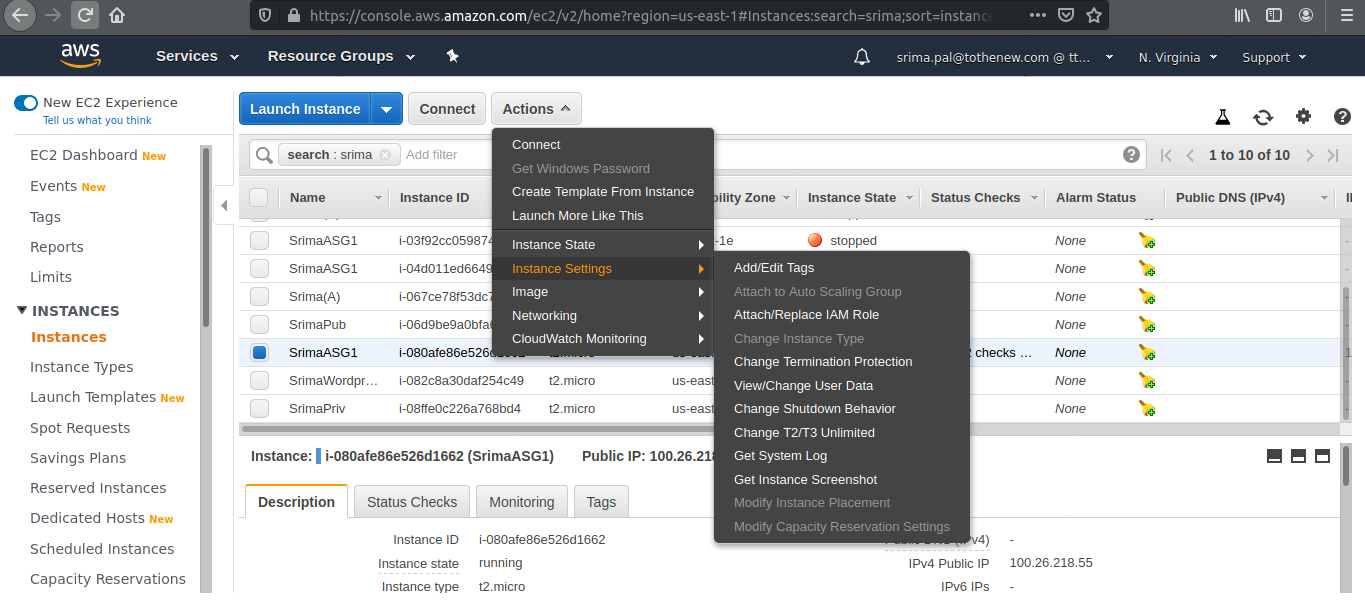


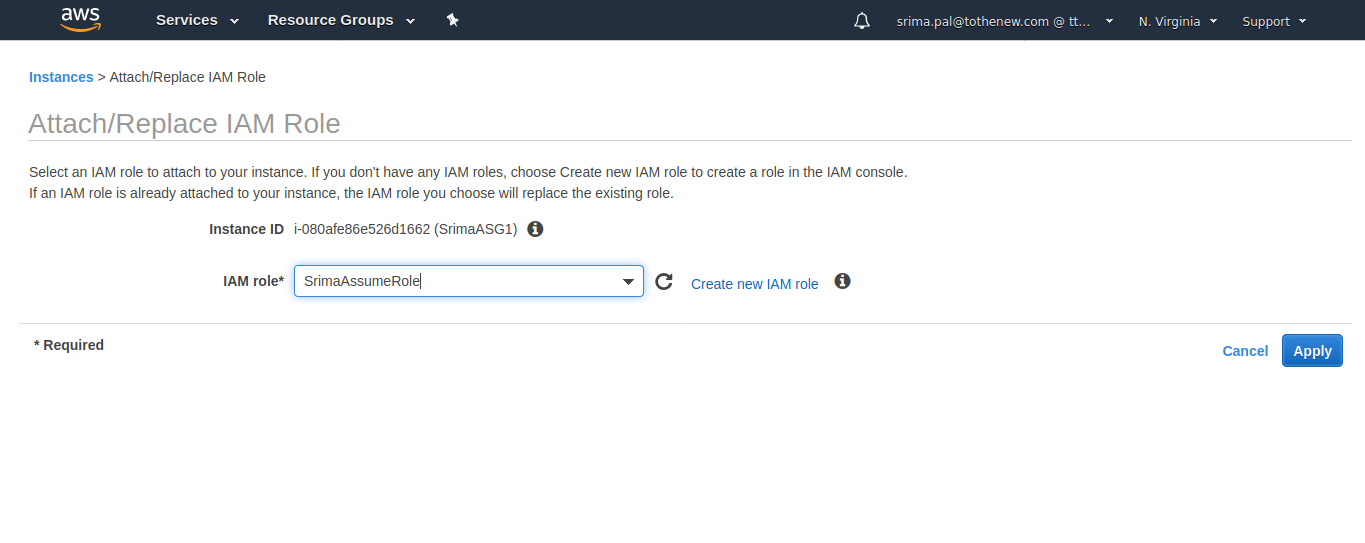


Update trust relationship of S3fullaccess role



Attach the assume role to the EC2 instance

****

****

****

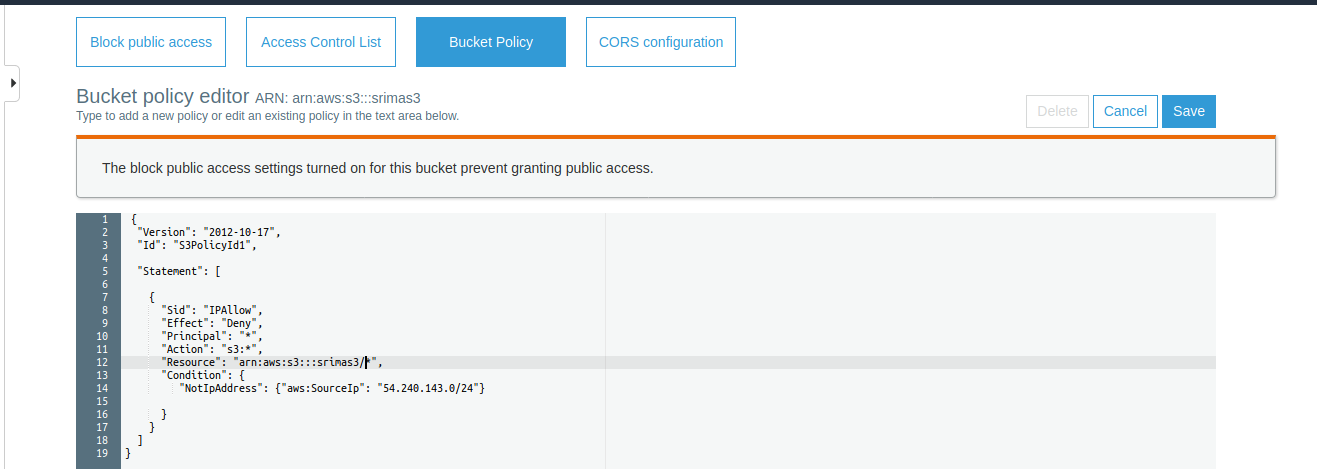
**3. Block s3 access on the basis of**

**i. IP**

**ii. Domain**

**iii. Pre-signed URL(Time based)**

Edit the bucket policy (based on IP)



Edit the bucket policy (based on Domain)

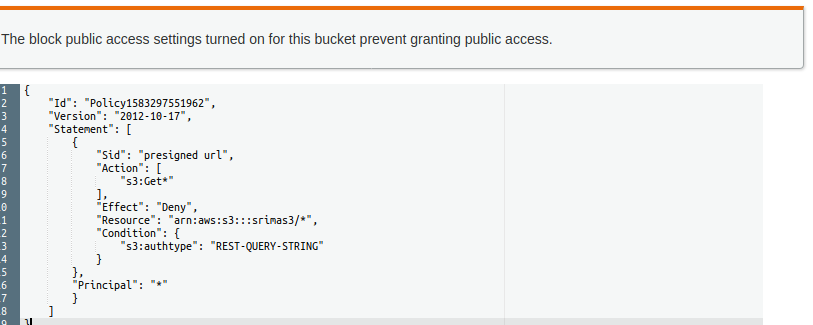


Edit bucket policy , presigned URL

A presigned URL is a URL that you can provide to your users to grant temporary access to a specific S3 object.

A pre-signed URL uses three parameters to limit the access to the user;

* Bucket: The bucket that the object is in (or will be in)
* Key : The name of the object.
* Expires: The amount of time that the URL is valid.



**5. ACL, Bucket policy, IAM Policy.**

**The Access Control List (ACL**): is used to define other users' access permissions for your file and folder objects. The Access Permissions that you set using the ACL determine what a user can and cannot do with your file and folder objects. For example, you can set permissions on a file object to let one user read the contents of a file (read access) and let another user make changes to the file (write access). In Amazon S3 you will first add grants to objects and then set the permissions for the grant.

There are 4 types of grants:

1. An Owner grant - which defines the permissions the owner of the object has.

2. Authenticated Users – which are all Amazon S3 storage users that have an account with S3.

3. Public – which means any anonymous user that you have provided the URL to.

4. Email-ID – which is an email address of specific S3 customers that have S3 accounts, not general public emails. The email given must match exactly the email address the S3 user signed up with and can only match one user account.

**Bucket Policies:** bucket Policies are similar to IAM policies in that they allow access to resources via a JSON script. However, Bucket policies are applied to Buckets in S3, where as IAM policies are assigned to user/groups/roles and are used to govern access to any AWS resource through the IAM service.

When a bucket policy is applied the permissions assigned apply to all objects within the Bucket. The policy will specify which ‘principles’ (users) are allowed to access which resources. The use of Principles within a Bucket policy differs from IAM policies, Principles within IAM policies are defined by who is associated to that policy via the user and group element. As Bucket policies are assigned to Buckets, there is this need of an additional requirement of ‘Principles’.

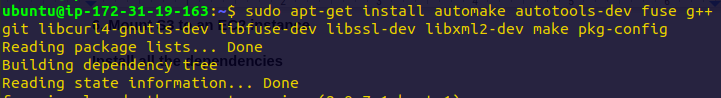
**IAM POLICY :**A [policy](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html) is an entity that, when attached to an identity or resource, defines their permissions. A policy that is attached to an identity in IAM is known as an *identity-based policy*. Identity-based policies can include AWS managed policies, customer managed policies, and inline policies. AWS managed policies are created and managed by AWS. You can use them, but you can't manage them. An inline policy is one that you create and embed directly to an IAM group, user, or role. Inline policies can't be reused on other identities or managed outside of the identity where it exists.

**6. Mount S3 to an EC2 instance**

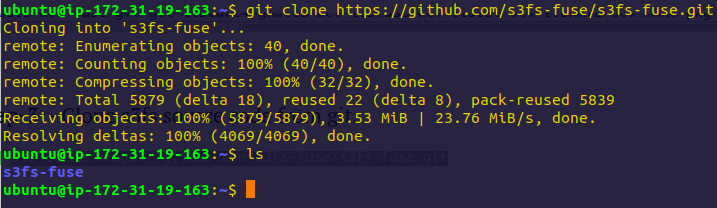
A S3 bucket can be mounted in a AWS instance as a file system known as S3fs. S3fs is a FUSE file-system that allows you to mount an Amazon S3 bucket as a local file-system. It behaves like a network attached drive, as it does not store anything on the Amazon EC2, but user can access the data on S3 from EC2 instance.

Filesystem in Userspace (FUSE) is a simple interface for userspace programs to export a virtual file-system to the Linux kernel. It also aims to provide a secure method for non privileged users to create and mount their own file-system implementations.

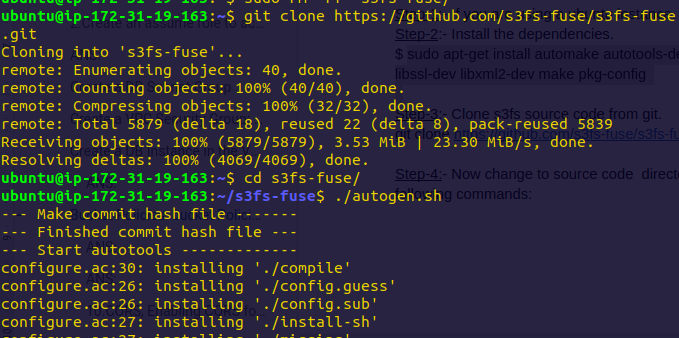
Install all the dependencies

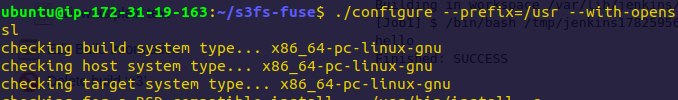
****

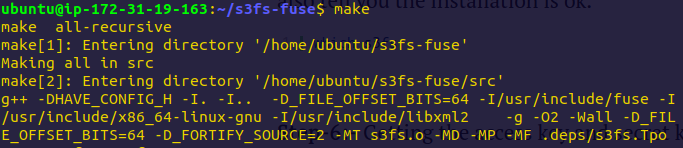
Clone s3fs code from git

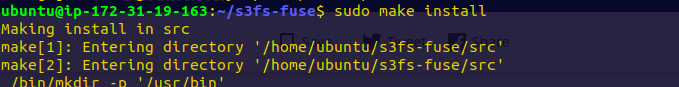


Now change to source code directory, and compile and install the code



****

****

****

Installation successful

****

Create a new file in /etc with the name passwd-s3fs and Paste the access key and secret key in the below format .

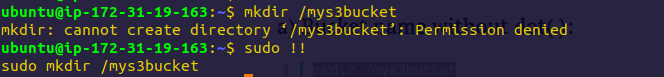
****

****

Change the permission of your file

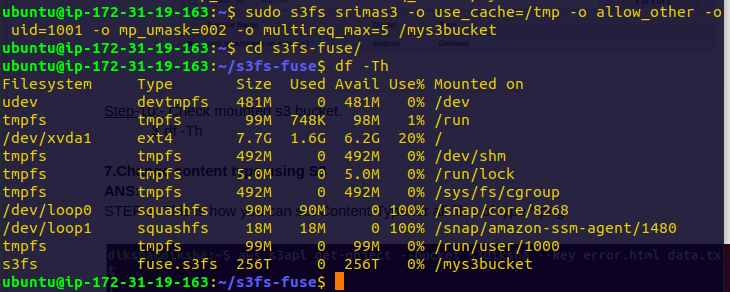
****

Now create a directory or provide the path of an existing directory and mount S3bucket in it.



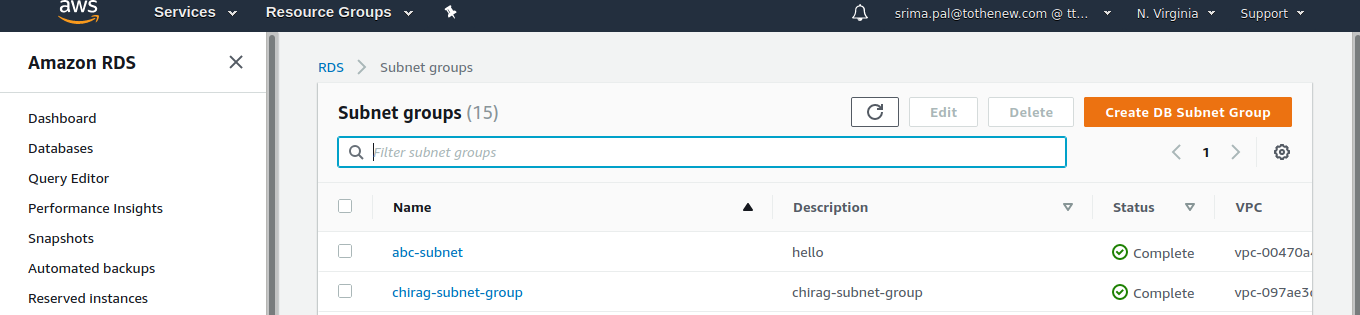


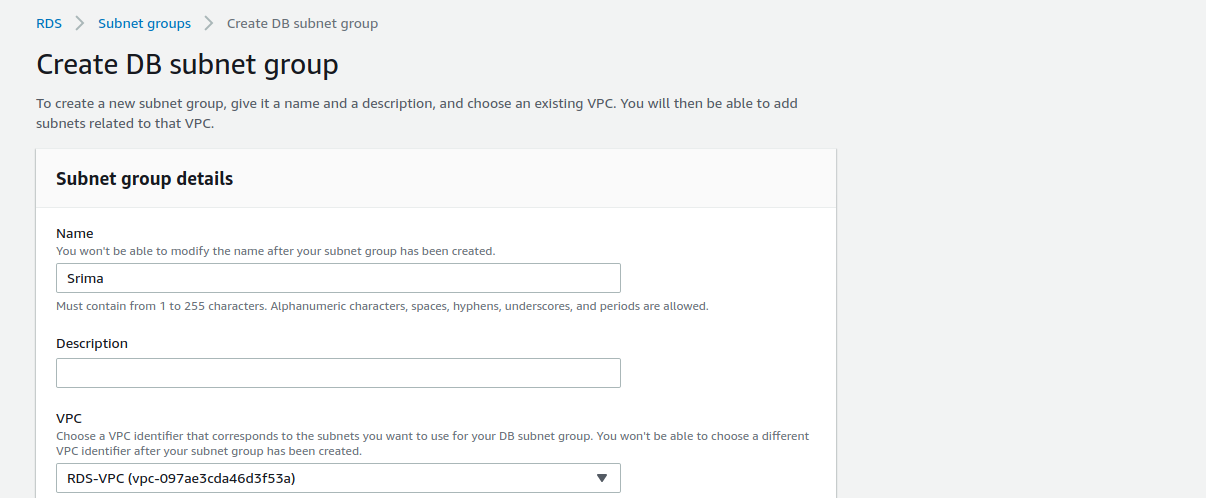
Check the mounted S3 bucket



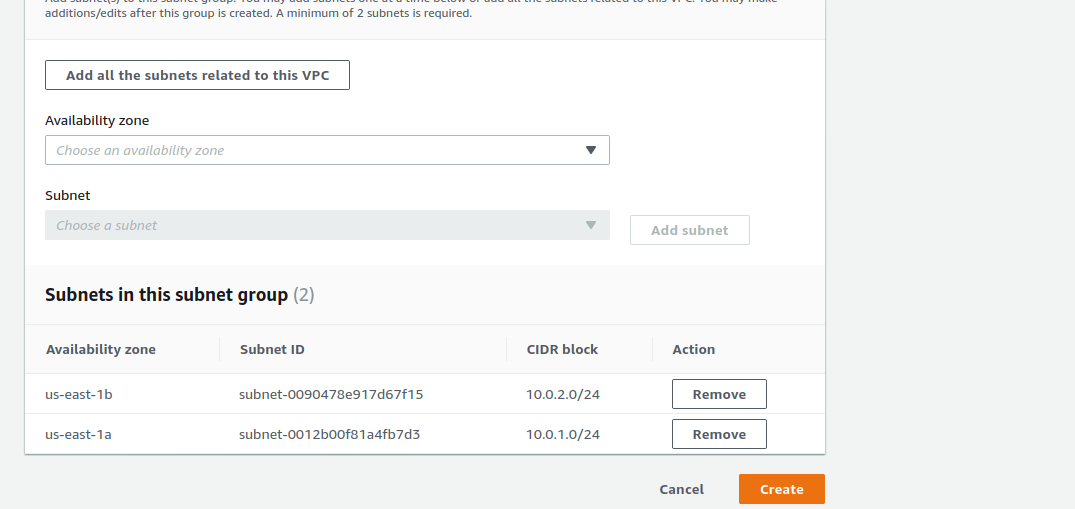
**4. Create RDS subnet and launch RDS instance, what is parameter group and option group?**

Go to Amazon RDS, build a db-subnet group

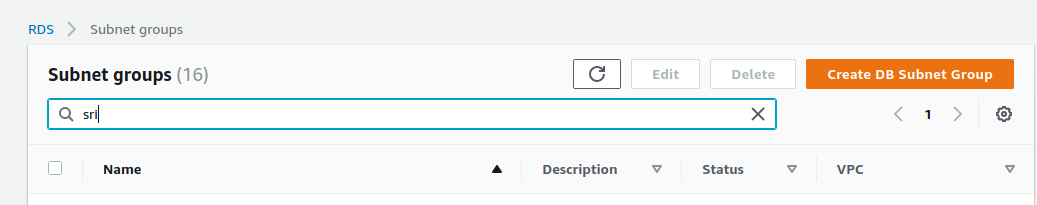




In the Add subnets section, choose Add all the subnets related to this VPC.

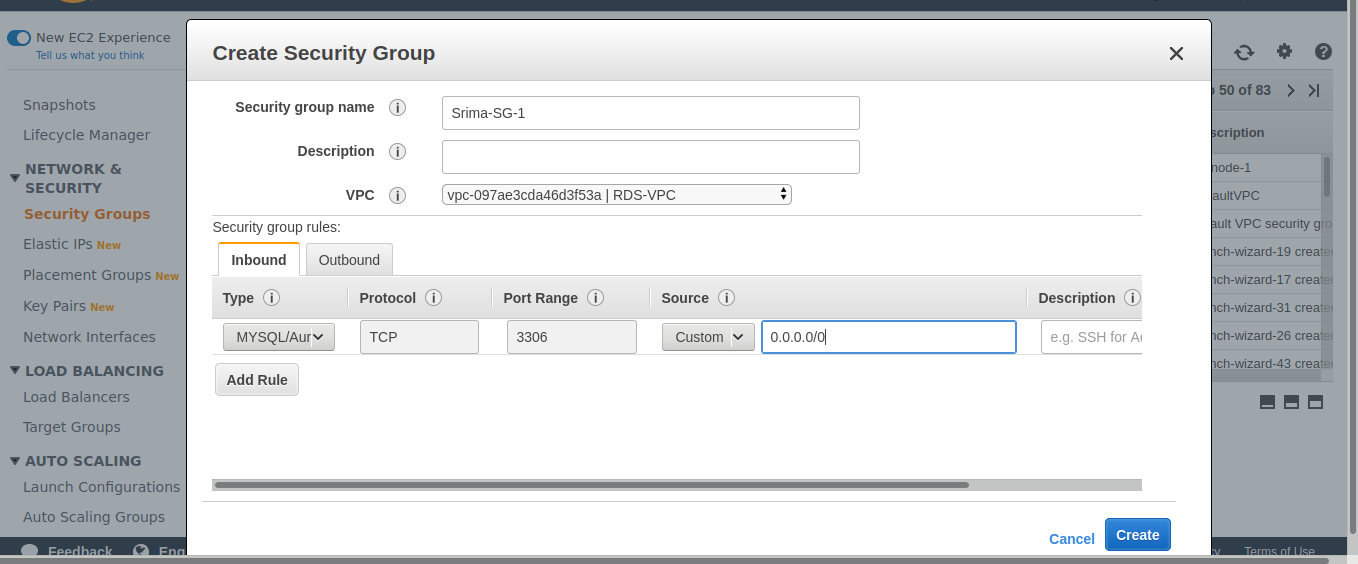


Choose Create.



### Create a VPC Security Group: Before you create your DB instance, you must create a VPC security group to associate with your DB instance. Choose the security group you created and edit inbound rules.\

### Set the following values for your new inbound rule to allow MySQL traffic on port 3306 from your EC2 instance. If you do this, you can connect from your web server to your DB instance to store and retrieve data from your web application to your database.

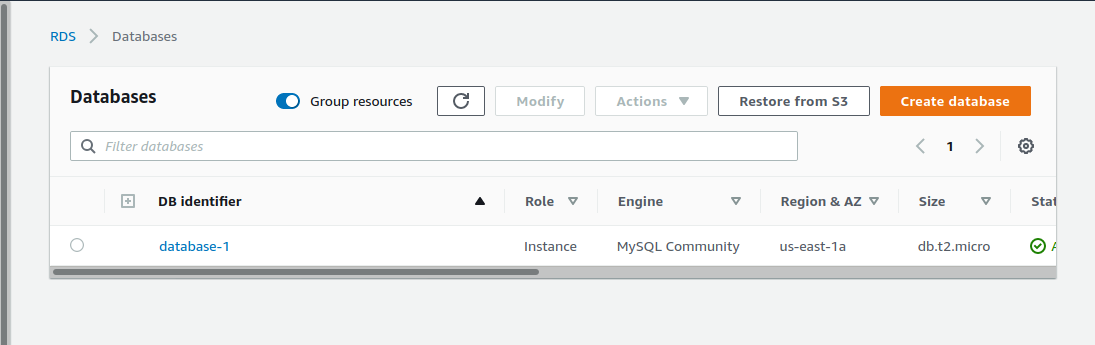


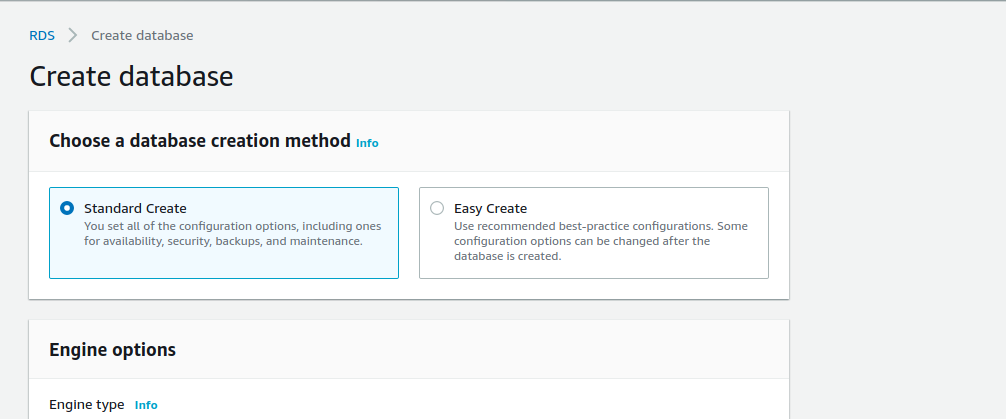
### Create a DB Instance in the VPC

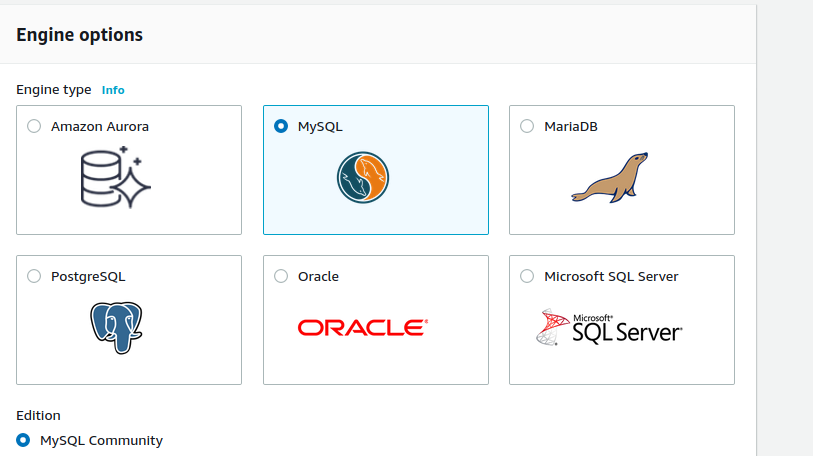
Databases > Choose Create database > In Choose a database creation method, choose Standard Create

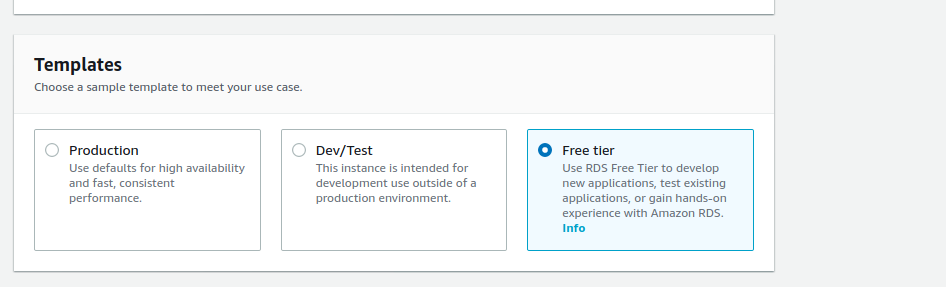
\*Use the VPC name, the DB subnet group, and the VPC security group you created in the previous steps.

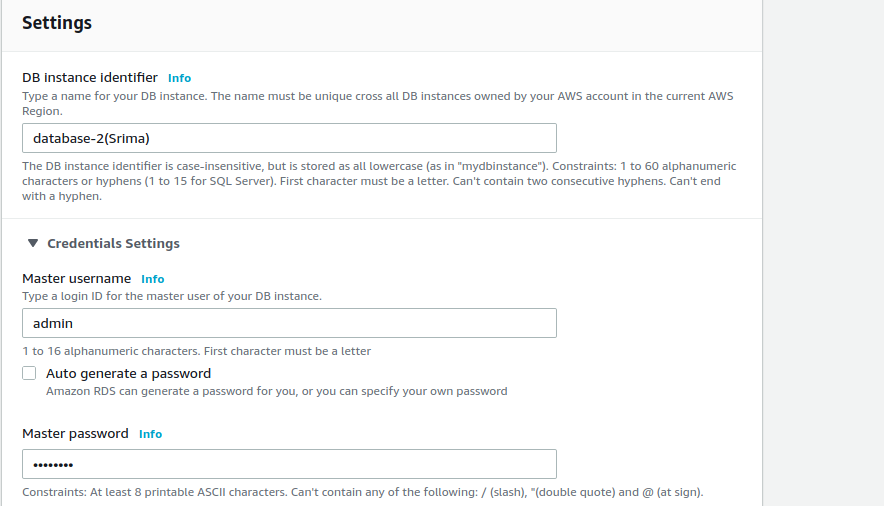
\*If you want your DB instance in the VPC to be publicly accessible, you must enable the VPC attributes *DNS hostnames* and *DNS resolution*.

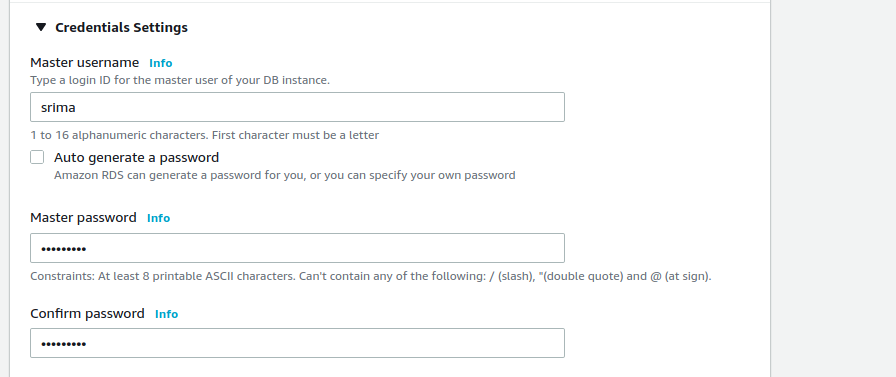


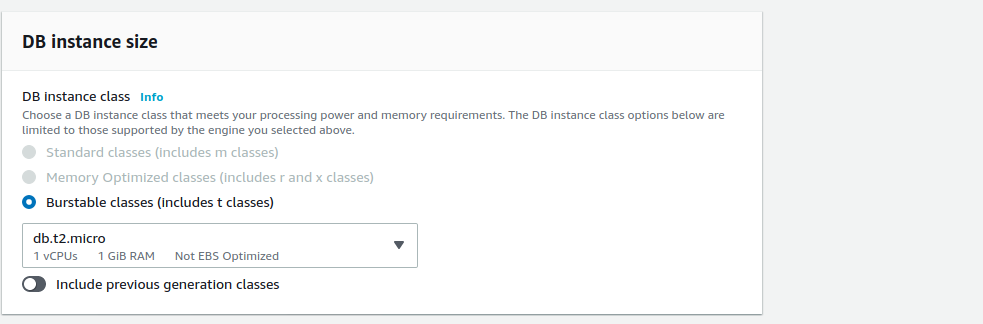


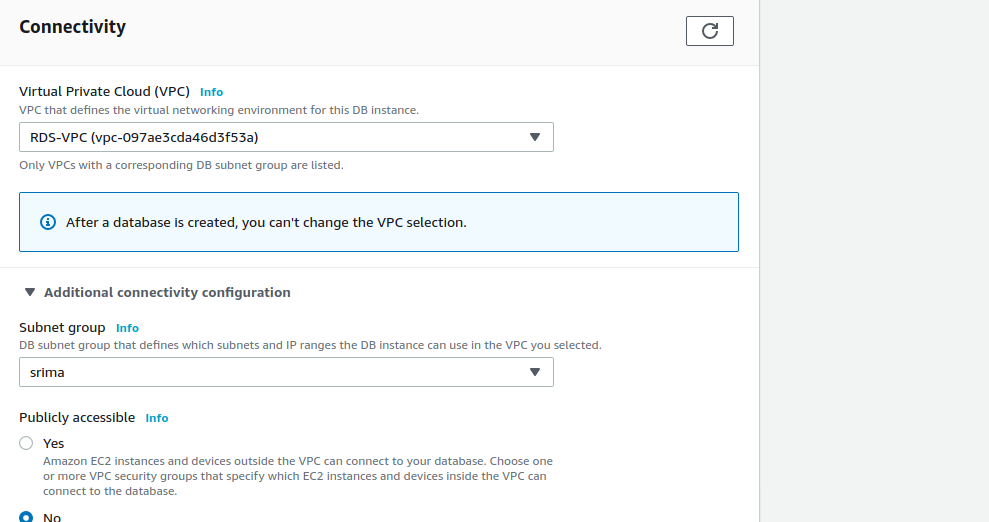












**Parameter group :**For AWS RDS instances, you manage your database engine configuration through the use of parameters in a DB parameter group. DB parameter groups act as a container for engine configuration values that are applied to one or more DB instances.

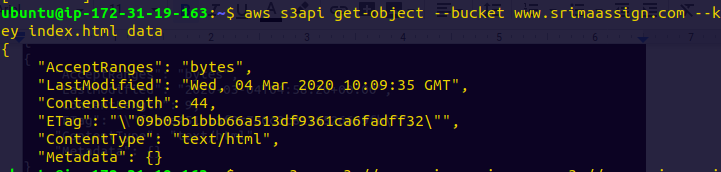
**Option Group:**An *option group* can specify features, called options, that are available for a particular Amazon RDS DB instance. Options can have settings that specify how the option works. When you associate a DB instance with an option group, the specified options and option settings are enabled for that DB instance.

Amazon RDS supports options for the following database engines:

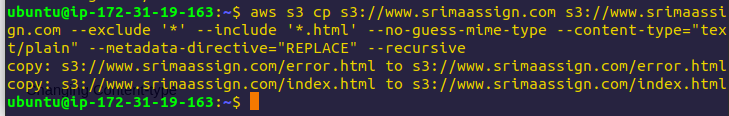
|  |  |
| --- | --- |
| Database Engine | Relevant Documentation |
| MariaDB | [Options for MariaDB Database Engine](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Appendix.MariaDB.Options.html) |
| Microsoft SQL Server | [Options for the Microsoft SQL Server Database Engine](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Appendix.SQLServer.Options.html) |
| MySQL | [Options for MySQL DB Instances](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Appendix.MySQL.Options.html) |
| Oracle | [Options for Oracle DB Instances](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Appendix.Oracle.Options.html) |

**7.Change content type using S3**

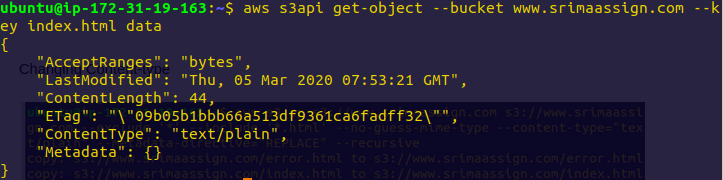
Earlier, content type was:

****

Changing Content-type

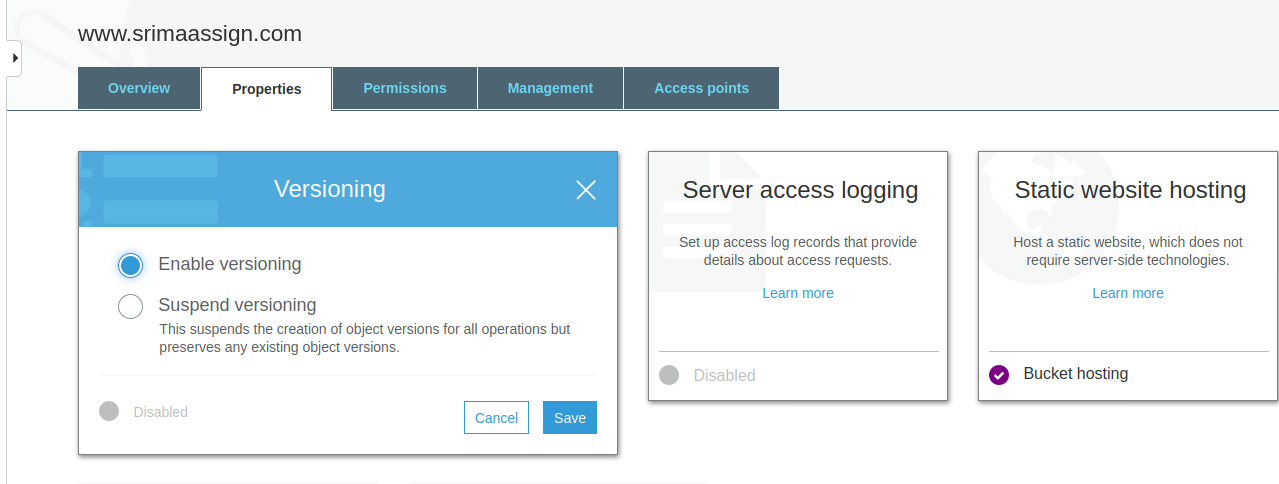


Now the content type is

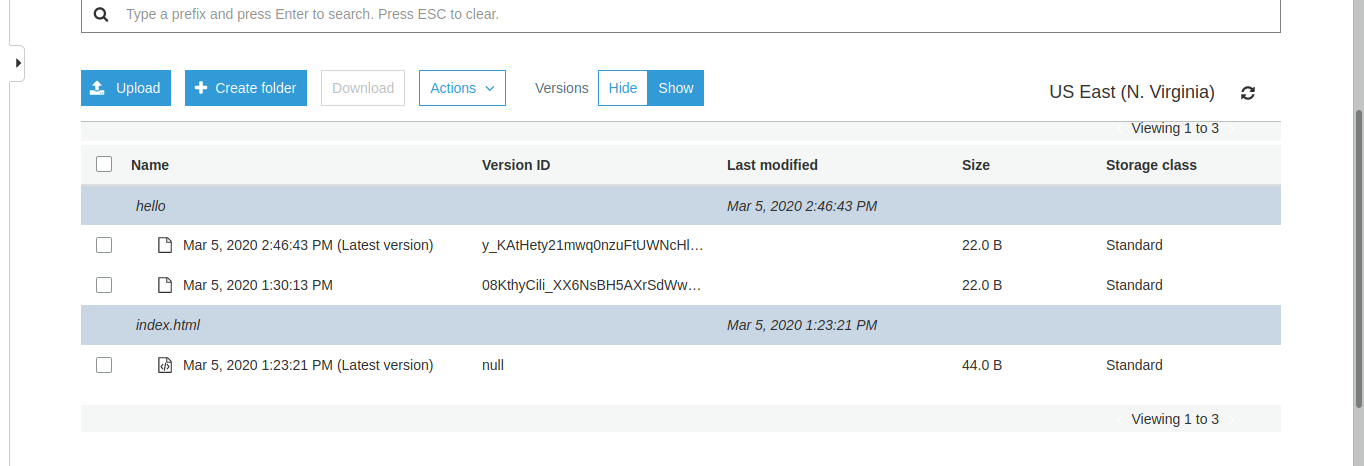


**8. Retrieve previous version of S3 (enable versioning).**

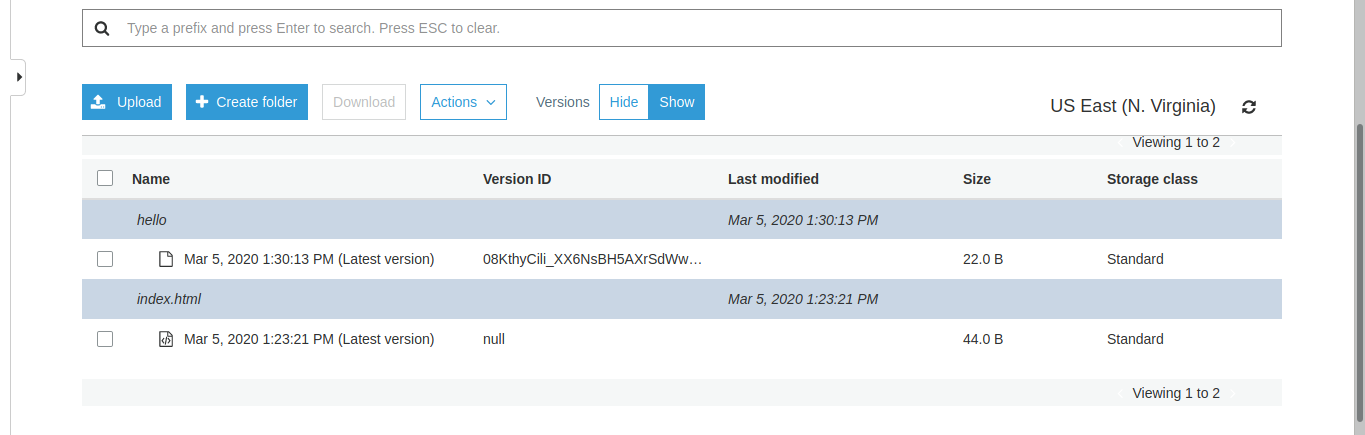
Enable versioning in the bucket



We can see two versions uploaded



Delete the latest version to retrieve previous version.



**9. S3 VPC endpoint.**

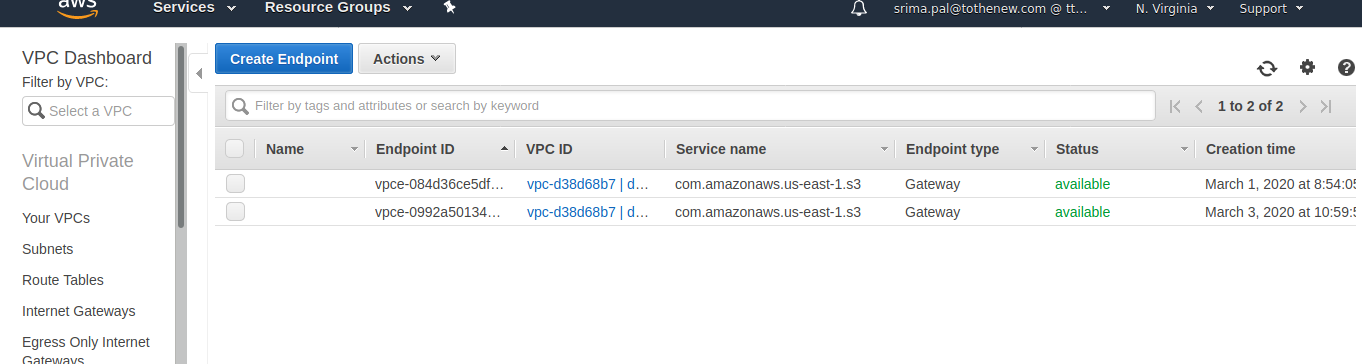
VPC endpoint enables a user to connect with AWS services that are outside the VPC through a private link. VPC endpoints use AWS PrivateLinks in the backend with which users will be able to connect to AWS services without using public IP’s. Thus the traffic will not leave the Amazon network. AWS PrivateLinks are highly available, redundant and scalable technology.

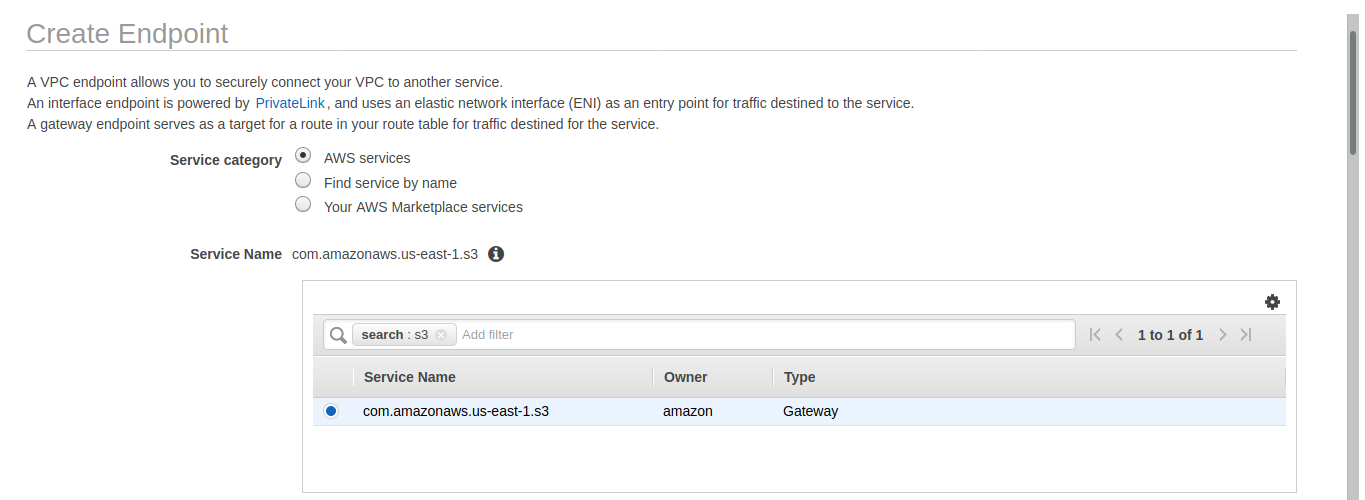
There are two types of VPC endpoints Interface Endpoints and Gateway Endpoints:

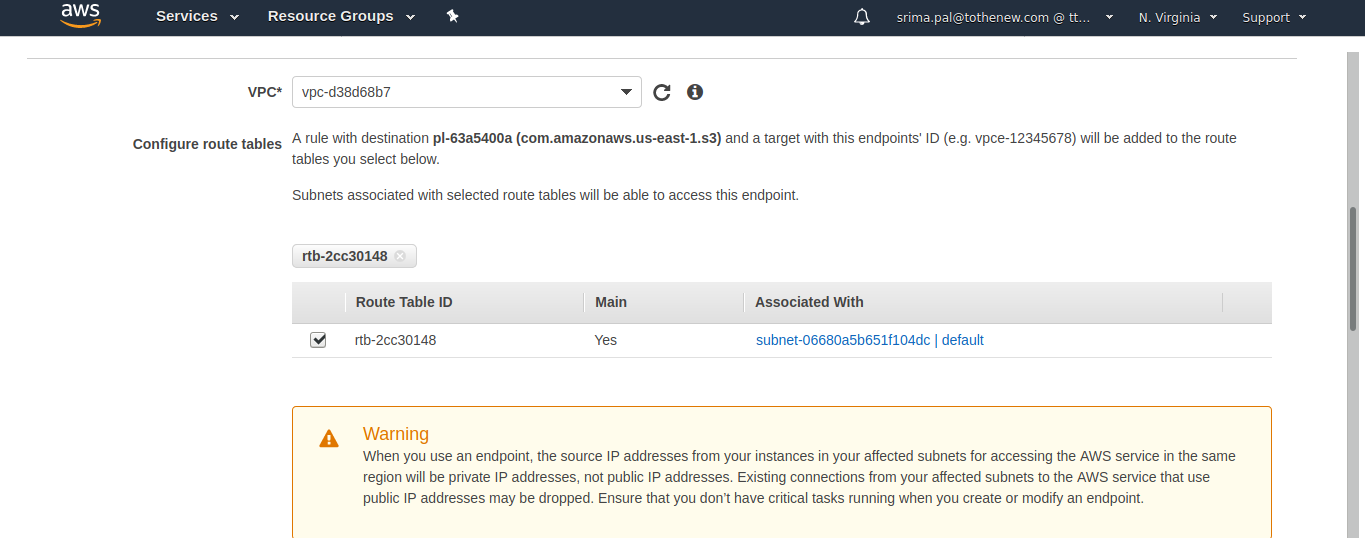
Interface Endpoints are Elastic Network Interfaces (ENI) with private IP addresses. ENI will act as the entry point for the traffic that is destined to a particular service. Services such as Amazon CloudWatch Logs, Amazon SNS, etc. are supported.

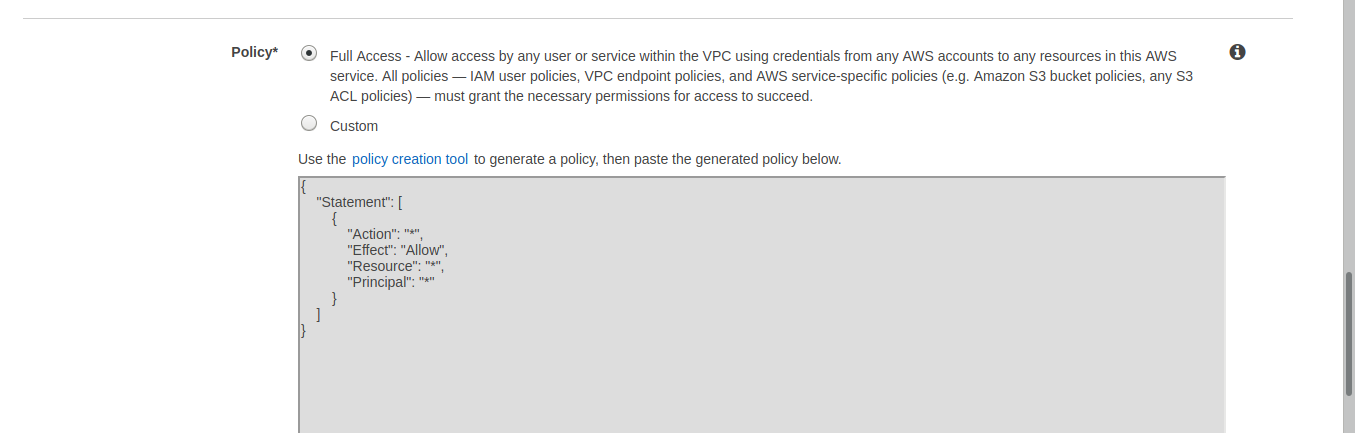
Gateway endpoints is a gateway targeted for a specific route in the routCreate another role which has the policy to assume the previous Roleeing table. They can be used to route traffic to a destined AWS service. As of now, Amazon S3 and DynamoDB are the only services that are supported by gateway endpoints.

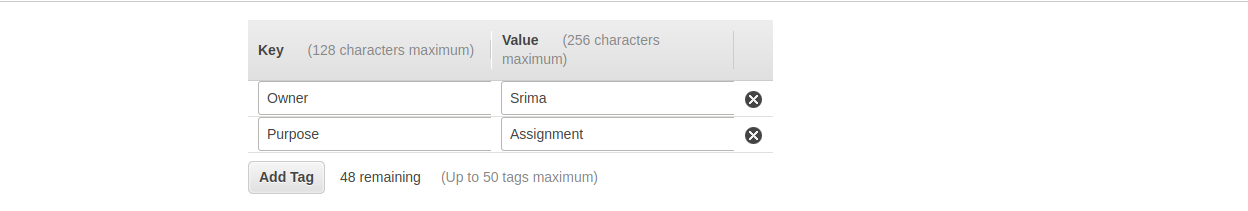
Go to VPC > Endpoint > Create Endpoint and mention the service eg. S3











**10.CORS, Enabling CORS for 2 specific website**

Cross-Origin Resource Sharing ([CORS](https://developer.mozilla.org/en-US/docs/Glossary/CORS)) is a mechanism that uses additional [HTTP](https://developer.mozilla.org/en-US/docs/Glossary/HTTP) headers to tell browsers to give a web application running at one [origin](https://developer.mozilla.org/en-US/docs/Glossary/origin), access to selected resources from a different origin. A web application executes a cross-origin HTTP request when it requests a resource that has a different origin (domain, protocol, or port) from its own.

