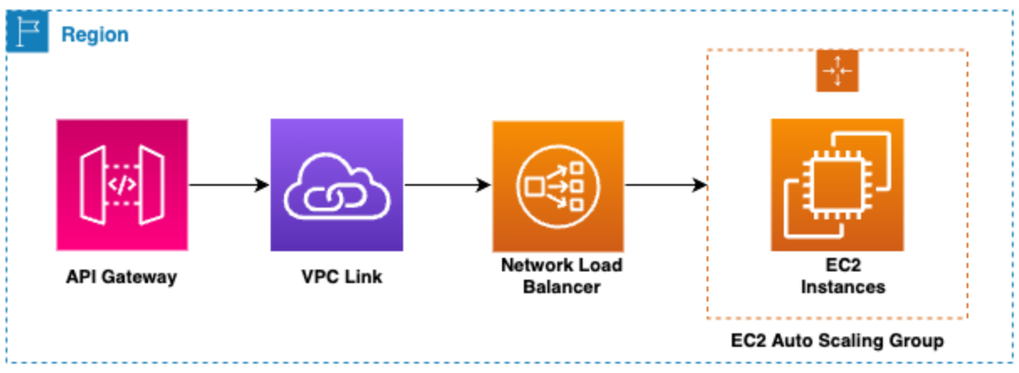
Private Integration

API Gateway Private Integration allows extending access to private HTTP/HTTPs resources within a VPC for clients outside the VPC. This is achieved through the ***VPC Link*** integration type. A VPC link is a resource in Amazon API Gateway that allows for connecting API routes to private resources inside a VPC. A VPC link acts like any other integration endpoint for an API and is an abstraction layer on top of other networking resources. This helps simplify configuring private integrations.

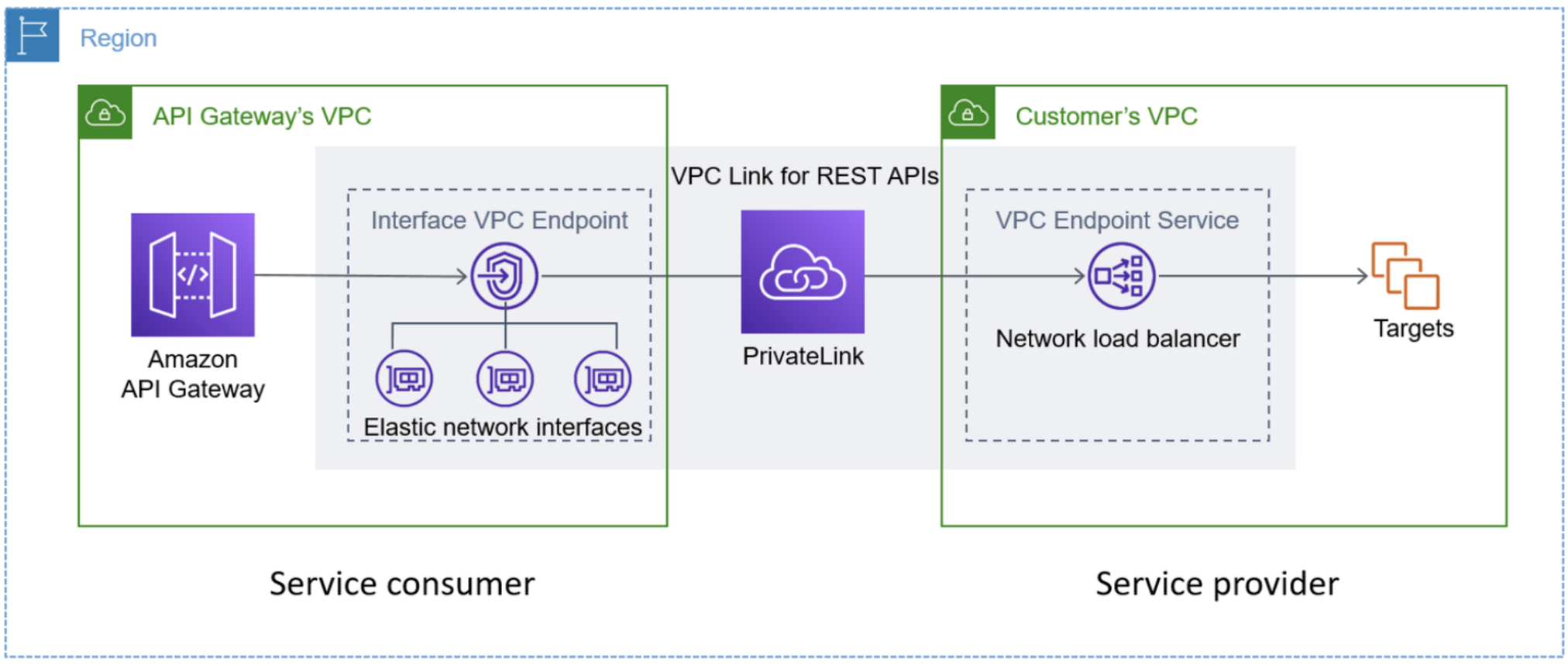


There are two types of VPC links: VPC links for REST APIs and VPC links for HTTP APIs. Both provide access to resources inside a VPC. They are built on top of an internal AWS service called AWS Hyperplane. This is an internal network virtualization platform, which supports inter-VPC connectivity and routing between VPCs. Internally, Hyperplane supports multiple network constructs that AWS services use to connect with the resources in customers' VPCs. One of those constructs is AWS PrivateLink, which is used by API Gateway to support private APIs and private integrations.

AWS PrivateLink allows access to AWS services and services hosted by other AWS customers, while maintaining network traffic within the AWS network. Since the service is exposed via a private IP address, all communication is virtually local and private. This reduces the exposure of data to the public internet.

In AWS PrivateLink, a VPC endpoint service is a networking resource in the service provider side that enables other AWS accounts to access the exposed service from their own VPCs. VPC endpoint services allow sharing a specific service located inside the provider's VPC by extending a virtual connection via an elastic network interface in the consumer's VPC.

An interface VPC endpoint is a networking resource in the service consumer side, which represents a collection of one or more elastic network interfaces. This is the entry point that allows for connecting to services powered by AWS PrivateLink.



As the service provider, you need to create a Network Load Balancer in your VPC as the service front end. The PrivateLink uses this to create a Endpoint Service which makes your service accessible to API Gateway Service by VPC Endpoint in the API Gateway service account.

Thus, to implement this REST API Private Integration, a VPC Link and a Network Load Balancer needs to be created to route traffic from your API to resources in your VPC through your VPC link and Network Load Balancer.

Within this module, you will use AWS SAM and OpenAPI to create and deploy a REST API Gateway that will make request to EC2 instance inside VPC.

Review the documentation:

- Choose an API Gateway API integration type
- Set up API Gateway private integrations

Estimated duration: 20 minutes

Build Amazon API Gateway Private Integration with AWS SAM and OpenAPI

Note
In this module, we will be creating REST API and VPC Link. Creation of REST API VPC Link requires VPC, Network Load Balancer, Target Groups and targets. These resources have already been created for simplicity.

Use AWS SAM and OpenAPI to create an API Gateway REST API with a Private Integration

1. Using AWS Cloud9 console, return to the root folder `module-3/private-integration`
2. This code belongs in your SAM template file `template.yaml`

   Review the code and then **copy/paste** it into the `template.yaml` file.

```yaml
AWSTemplateFormatVersion: '2010-09-09'
Transform: 'AWS::Serverless-2016-10-31'
Description: >
    module3-private-integration-rest-api: Sample SAM Template for module3-private-integration-rest-api

Globals:
    # Enable Logs
    Api:
        MethodSettings:
            - ResourcePath: "/*"
              HttpMethod: "*"
              DataTraceEnabled: True
              LoggingLevel: INFO
              MetricsEnabled: True

Parameters:
    InternalNlbArn:
        Type: String
        Description: ARN of the Internal Network Load Balancer to link to the REST API
    InternalNlbDns:
        Type: String
        Description: DNS of the Internal Network Load Balancer to use in to the REST API Integration

Resources:
    # Private Integration Example API
    PrivateIntegrationExampleAPI:
        Type: AWS::Serverless::Api
        Properties:
            StageName: dev
            OpenApiVersion: 3.0.3
            DefinitionBody: # an OpenApi definition
                "Fn::Transform":
                    Name: "AWS::Include"
                    Parameters:
                        Location: "openapi.yaml"
            EndpointConfiguration:
                Type: REGIONAL

    VpcLink:
        Type: 'AWS::ApiGateway::VpcLink'
        Properties:
            Name: RestAPILinkInternalNLB
            Description: VPC Link for REST API
            TargetArns:
            - Fn::Sub:
                - '${InternalNlbArn}'
                - InternalNlbArn:
                    Ref: InternalNlbArn

Outputs:
  RestApiEndpoint:
      Description: Rest API endpoint URL for Dev stage (Private Integration Example API)
      Value: !Sub "https://${PrivateIntegrationExampleAPI}.execute-api.${AWS::Region}.amazonaws.com/${PrivateIntegrationExampleAPI.Stage}/"
  VpcLinkId:
      Description: ID of the created VPC Link
      Value: !Ref VpcLink
```

   The `VpcLink` resource **(lines 39-48)** defines a API Gateway **VPC Link** resource which will be used by REST API to connect to private resources.

3. This code belongs in your OpenAPI definition file `openapi.yaml`

   Review the code and then **copy/paste** it into the `openapi.yaml` file

```yaml
openapi: "3.0.1"
info:
  title: "module-3-private-integration"
  description: "API Gateway example for Private Integration"
  version: "1.0"
paths:
  /internal-nlb:
    get:
      x-amazon-apigateway-integration:
        connectionId:
            Fn::Sub:
            - '${VpcLink}'
            - VpcLink:
                Ref: VpcLink
        httpMethod: "GET"
        uri:
            Fn::Sub:
            - 'http://${InternalNlbDns}'
            - InternalNlbDns:
                Ref: InternalNlbDns
        responses:
          default:
            statusCode: "200"
        passthroughBehavior: "when_no_match"
        connectionType: "VPC_LINK"
        type: "http_proxy"
components: {}
```

   The x-amazon-apigateway-integration on Line (9) is an OpenAPI extension that allows us to integrate our APIs to AWS backends. **connectionId** on Line (10) related to the VPC Link Id. The URI on Line (16) needs to be the URL of NLB.

Deploy the project

1. The stack deployment using AWS SAM will require information about Network Load Balancer. Run the following command from Cloud9 termimal get **Output** of the main stack which contains DNS Name and ARN of the Network Load Balancer.

Note
Change the region in the command below to the one where you are running the workshop

```
aws cloudformation describe-stacks --region us-east-1 --query "Stacks[?contains(StackName,'APIGWModule3')][].Outputs"
```

2. To deploy the API Gateway and VPC Link resources to your AWS account, run the following commands from the application root `module-3/private-integration`, where the `template.yaml` file for the sample application is located:

```
sam build && sam deploy --guided
```

   The first time that you run the `sam deploy --guided` command, AWS SAM starts an AWS CloudFormation deployment. In this case, you need to say what are the configurations that you want SAM to have in order to get the guided deployment. You can configure it as below.

   - **Stack Name**: `module-3-private-integration`
   - **AWS Region**: Put the chosen region to run the workshop. **e.g.** `us-east-1`
   - **Parameter InternalNlbArn**: Copy from step 1
   - **Parameter InternalNlbDns**: Copy from step 1
   - **Confirm changes before deploy**: `Y`
   - **Allow SAM CLI IAM role creation**: `Y`
   - **Disable rollback**: `N`
   - **Save arguments to configuration file**: `Y`
   - **SAM configuration file** and **SAM configuration environment** leave blank

```
Configuring SAM deploy
======================

        Looking for config file [samconfig.toml] :  Not found

        Setting default arguments for 'sam deploy'
        =========================================
        Stack Name [sam-app]: module-3-private-integration
        AWS Region [ap-south-1]:
        Parameter InternalNlbArn []: arn:aws:elasticloadbalancing:ap-south-1:           :loadbalancer/net/    -NLB-Internal/
        Parameter InternalNlbDns []:                                .elb.ap-south-1.amazonaws.com
        #Shows you resources changes to be deployed and require a 'Y' to initiate deploy
        Confirm changes before deploy [y/N]:
        #SAM needs permission to be able to create roles to connect to the resources in your template
        Allow SAM CLI IAM role creation [Y/n]:
        #Preserves the state of previously provisioned resources when an operation fails
        Disable rollback [y/N]:
        Save arguments to configuration file [Y/n]:
        SAM configuration file [samconfig.toml]:
        SAM configuration environment [default]:
```

3. After configuring the deployment, AWS SAM will display assets that will be created. But first, it will automatically upload the template to a temporary bucket it creates. Then, it will ask you to confirm the changes. Type `y` to confirm.

```
CREATE_COMPLETE                  AWS::ApiGateway::Stage          PrivateIntegrationExampleAPIdevStage  -
CREATE_COMPLETE                  AWS::CloudFormation::Stack      module-3-private-integration          -
-----------------------------------------------------------------------------------------------------------

CloudFormation outputs from deployed stack
-----------------------------------------------------------------------------------------------------------
Outputs
-----------------------------------------------------------------------------------------------------------
Key                 RestApiEndpoint
Description         Rest API endpoint URL for Dev stage (Private Integration Example API)
Value               https://          .execute-api.ap-south-1.amazonaws.com/dev/

Key                 VpcLinkId
Description         ID of the created VPC Link
Value               ******
-----------------------------------------------------------------------------------------------------------


Successfully created/updated stack - module-3-private-integration in ap-south-1
```

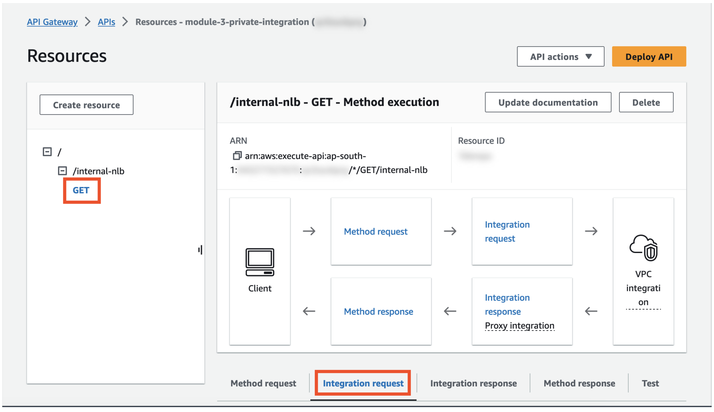
3. Once the deployment has been successful, you will see an **'Outputs'** section that contains the API Gateway invoke URL

Test the Integration

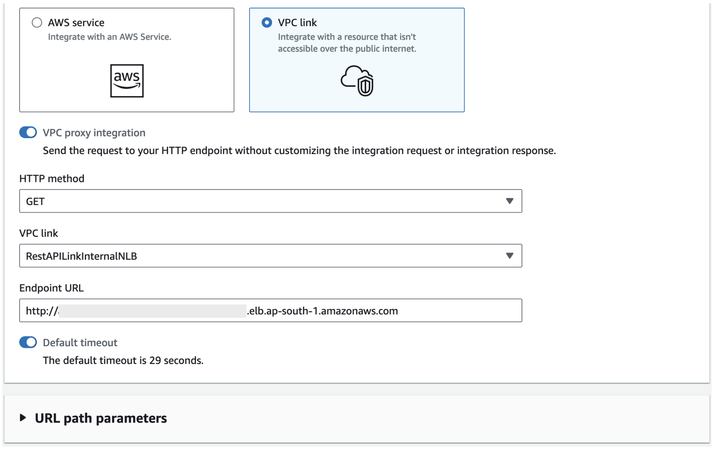
Now the VPC Link is integrated with API and the API has been deployed, it is time to test the private integration.

Test Private Integration using API Gateway console

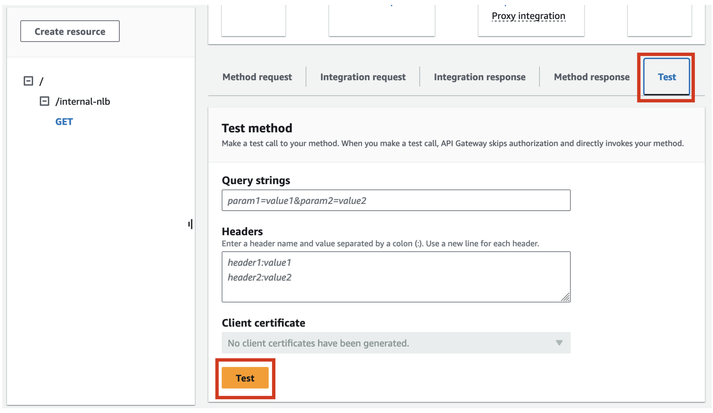
1. Open the Amazon API Gateway console and sign in.
2. Choose your REST API named, `module-3-private-integration`.
3. In the *Resources* pane, select `/internal-nlb` resource and the select `GET` method.



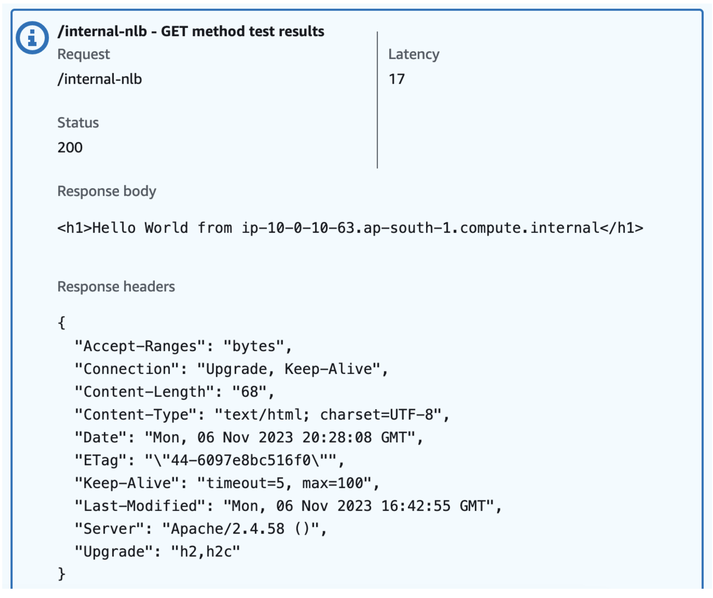
4. Click the **Integration Request** and then **Edit** to review the configuration details for integration.



5. Click on **Test** tab. Leave all fields as-is and click on **TEST** button.



6. The test result shows:



Test the deployed API using cURL

1. Open a new terminal window in your AWS Cloud9 environment.
2. Copy the below cURL command and paste it into the terminal window, replacing `<api-id>` with your API ID and `<region>` with the region where your API is deployed. You can also get the full URL from the Outputs section from your SAM deployment. You can also find the full invoke URL in the API Gateway console by navigating to **Stages > dev**.

```
curl -X GET \
  'https://<api-id>.execute-api.<region>.amazonaws.com/dev/internal-nlb'
```

3. The **Response Body** output should be:

```
<h1>Hello World from ip-10-0-10-63.ap-south-1.compute.internal</h1>
```