Package ‘paws.networking’

August 23, 2021

Title 'Amazon Web Services' Networking & Content Delivery Services

Version 0.1.12

Description Interface to 'Amazon Web Services' networking and content delivery services, including 'Route 53' Domain Name System service, 'CloudFront' content delivery, load balancing, and more <https://aws.amazon.com/>.

License Apache License (>= 2.0)

URL https://github.com/paws-r/paws

BugReports https://github.com/paws-r/paws/issues

Imports paws.common (>= 0.3.0)

Suggests testthat

Encoding UTF-8

RoxygenNote 7.1.1

Collate 'apigateway_service.R' 'apigateway_interfaces.R'
'apigateway_operations.R' 'apigatewaymanagementapi_service.R'
'apigatewaymanagementapi_interfaces.R'
'apigatewaymanagementapi_operations.R' 'apigatewayv2_service.R'
'apigatewayv2_interfaces.R' 'apigatewayv2_operations.R'
'appmesh_service.R' 'appmesh_interfaces.R'
'appmesh_operations.R' 'cloudfront_service.R'
'cloudfront_interfaces.R' 'cloudfront_operations.R'
directconnect_service.R' 'directconnect_interfaces.R'
directconnect_operations.R' 'elb_service.R' 'elb_interfaces.R'
elb_operations.R' 'elbv2_service.R' 'elbv2_interfaces.R'
elbv2_operations.R' 'globalaccelerator_service.R'
globalaccelerator_interfaces.R'
globalaccelerator_operations.R' 'route53_service.R'
'route53_interfaces.R' 'route53_operations.R'
'route53domains_service.R' 'route53domains_interfaces.R'
'route53domains_operations.R' 'route53resolver_service.R'
'route53resolver_interfaces.R' 'route53resolver_operations.R'
servicediscovery_service.R' 'servicediscovery_interfaces.R'
servicediscovery_operations.R'
Description

Amazon API Gateway helps developers deliver robust, secure, and scalable mobile and web application back ends. API Gateway allows developers to securely connect mobile and web applications to APIs that run on AWS Lambda, Amazon EC2, or other publicly addressable web services that are hosted outside of AWS.

Usage

\[
\text{apigateway}(\text{config} = \text{list}())
\]

Arguments

| config       | Optional configuration of credentials, endpoint, and/or region. |
Value

A client for the service. You can call the service’s operations using syntax like svc$operation(...), where svc is the name you’ve assigned to the client. The available operations are listed in the Operations section.

Service syntax

```r
svc <- apigateway(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string"
    ),
    endpoint = "string",
    region = "string"
  )
)
```

Operations

- `create_api_key` Create an ApiKey resource
- `create_authorizer` Adds a new Authorizer resource to an existing RestApi resource
- `create_base_path_mapping` Creates a new BasePathMapping resource
- `create_deployment` Creates a Deployment resource, which makes a specified RestApi callable over the internet
- `create_documentation_part` Create documentation part
- `create_documentation_version` Create documentation version
- `create_domain_name` Creates a new domain name
- `create_model` Adds a new Model resource to an existing RestApi resource
- `create_request_validator` Creates a ReqeustValidator of a given RestApi
- `create_resource` Creates a Resource resource
- `create_rest_api` Creates a new RestApi resource
- `create_stage` Creates a new Stage resource that references a pre-existing Deployment for the API
- `create_usage_plan` Creates a usage plan with the throttle and quota limits, as well as the associated API stages, and creates a usage plan key for adding an existing API key to a usage plan
- `create_vpc_link` Creates a VPC link, under the caller’s account in a selected region, in an asynchronous operation
- `delete_api_key` Deletes the ApiKey resource
- `delete_authorizer` Deletes an existing Authorizer resource
- `delete_base_path_mapping` Deletes the BasePathMapping resource
- `delete_client_certificate` Deletes the ClientCertificate resource
- `delete_deployment` Deletes a Deployment resource
- `delete_documentation_part` Delete documentation part
- `delete_documentation_version` Delete documentation version
- `delete_domain_name` Deletes the DomainName resource
- `delete_gateway_response` Clears any customization of a GatewayResponse of a specified response type on the given RestApi
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>delete_integration</td>
<td>Represents a delete integration</td>
</tr>
<tr>
<td>delete_integration_response</td>
<td>Represents a delete integration response</td>
</tr>
<tr>
<td>delete_method</td>
<td>Deletes an existing Method resource</td>
</tr>
<tr>
<td>delete_method_response</td>
<td>Deletes an existing MethodResponse resource</td>
</tr>
<tr>
<td>delete_model</td>
<td>Deletes a model</td>
</tr>
<tr>
<td>delete_request_validator</td>
<td>Deletes a RequestValidator of a given RestApi</td>
</tr>
<tr>
<td>delete_resource</td>
<td>Deletes a Resource resource</td>
</tr>
<tr>
<td>delete_rest_api</td>
<td>Deletes the specified API</td>
</tr>
<tr>
<td>delete_stage</td>
<td>Deletes a Stage resource</td>
</tr>
<tr>
<td>delete_usage_plan</td>
<td>Deletes a usage plan of a given plan Id</td>
</tr>
<tr>
<td>delete_usage_plan_key</td>
<td>Deletes a usage plan key and remove the underlying API key from the associated usage plan</td>
</tr>
<tr>
<td>delete_vpc_link</td>
<td>Deletes an existing VpcLink of a specified identifier</td>
</tr>
<tr>
<td>flush_stage_authorizers_cache</td>
<td>Flushes all authorizer cache entries on a stage</td>
</tr>
<tr>
<td>flush_stage_cache</td>
<td>Flushes a stage’s cache</td>
</tr>
<tr>
<td>generate_client_certificate</td>
<td>Generates a ClientCertificate resource</td>
</tr>
<tr>
<td>get_account</td>
<td>Gets information about the current Account resource</td>
</tr>
<tr>
<td>get_api_key</td>
<td>Gets information about the current ApiKey resource</td>
</tr>
<tr>
<td>get_api_keys</td>
<td>Gets information about the current ApiKeys resource</td>
</tr>
<tr>
<td>get_authorizer</td>
<td>Describe an existing Authorizer resource</td>
</tr>
<tr>
<td>get_authorizers</td>
<td>Describe an existing Authorizers resource</td>
</tr>
<tr>
<td>get_base_path_mapping</td>
<td>Describe a BasePathMapping resource</td>
</tr>
<tr>
<td>get_base_path_mappings</td>
<td>Represents a collection of BasePathMapping resources</td>
</tr>
<tr>
<td>get_client_certificate</td>
<td>Gets information about the current ClientCertificate resource</td>
</tr>
<tr>
<td>get_client_certificates</td>
<td>Gets a collection of ClientCertificate resources</td>
</tr>
<tr>
<td>get_deployment</td>
<td>Gets information about a Deployment resource</td>
</tr>
<tr>
<td>get_deployments</td>
<td>Gets information about a Deployments collection</td>
</tr>
<tr>
<td>get_documentation_part</td>
<td>Get documentation part</td>
</tr>
<tr>
<td>get_documentation_parts</td>
<td>Get documentation parts</td>
</tr>
<tr>
<td>get_documentation_version</td>
<td>Get documentation version</td>
</tr>
<tr>
<td>get_documentation_versions</td>
<td>Get documentation versions</td>
</tr>
<tr>
<td>get_domain_name</td>
<td>Represents a domain name that is contained in a simpler, more intuitive URL that can be called</td>
</tr>
<tr>
<td>get_domain_names</td>
<td>Represents a collection of DomainName resources</td>
</tr>
<tr>
<td>get_export</td>
<td>Exports a deployed version of a RestApi in a specified format</td>
</tr>
<tr>
<td>get_gateway_response</td>
<td>Gets a GatewayResponse of a specified response type on the given RestApi</td>
</tr>
<tr>
<td>get_gateway_responses</td>
<td>Gets the GatewayResponses collection on the given RestApi</td>
</tr>
<tr>
<td>get_integration</td>
<td>Get the integration settings</td>
</tr>
<tr>
<td>get_integration_response</td>
<td>Represents a get integration response</td>
</tr>
<tr>
<td>get_method</td>
<td>Describe an existing Method resource</td>
</tr>
<tr>
<td>get_method_response</td>
<td>Describes a MethodResponse resource</td>
</tr>
<tr>
<td>get_model</td>
<td>Describes an existing model defined for a RestApi resource</td>
</tr>
<tr>
<td>get_models</td>
<td>Describes existing Models defined for a RestApi resource</td>
</tr>
<tr>
<td>get_model_template</td>
<td>Generates a sample mapping template that can be used to transform a payload into the structure</td>
</tr>
<tr>
<td>get_request_validator</td>
<td>Gets a RequestValidator of a given RestApi</td>
</tr>
<tr>
<td>get_request_validators</td>
<td>Gets the RequestValidators collection of a given RestApi</td>
</tr>
<tr>
<td>get_resource</td>
<td>Lists information about a resource</td>
</tr>
<tr>
<td>get_resources</td>
<td>Lists information about a collection of Resource resources</td>
</tr>
<tr>
<td>get_rest_api</td>
<td>Lists the RestApi resource in the collection</td>
</tr>
<tr>
<td>get_rest_apis</td>
<td>Lists the RestApis resources for your collection</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>get_sdk</td>
<td>Generates a client SDK for a RestApi and Stage</td>
</tr>
<tr>
<td>get_sdk_type</td>
<td>Get sdk type</td>
</tr>
<tr>
<td>get_sdk_types</td>
<td>Get sdk types</td>
</tr>
<tr>
<td>get_stage</td>
<td>Gets information about a Stage resource</td>
</tr>
<tr>
<td>get_stages</td>
<td>Gets information about one or more Stage resources</td>
</tr>
<tr>
<td>get_tags</td>
<td>Gets the Tags collection for a given resource</td>
</tr>
<tr>
<td>get_usage</td>
<td>Gets the usage data of a usage plan in a specified time interval</td>
</tr>
<tr>
<td>get_usage_plan</td>
<td>Gets a usage plan of a given plan identifier</td>
</tr>
<tr>
<td>get_usage_plan_key</td>
<td>Gets a usage plan key of a given key identifier</td>
</tr>
<tr>
<td>get_usage_plan_keys</td>
<td>Gets all the usage plan keys representing the API keys added to a specified usage plan</td>
</tr>
<tr>
<td>get_usage_plans</td>
<td>Gets all the usage plans of the caller’s account</td>
</tr>
<tr>
<td>get_vpc_link</td>
<td>Gets a specified VPC link under the caller’s account in a region</td>
</tr>
<tr>
<td>get_vpc_links</td>
<td>Gets the VpcLinks collection under the caller’s account in a selected region</td>
</tr>
<tr>
<td>import_api_keys</td>
<td>Import API keys from an external source, such as a CSV-formatted file</td>
</tr>
<tr>
<td>import_documentation_parts</td>
<td>Import documentation parts</td>
</tr>
<tr>
<td>import_rest_api</td>
<td>A feature of the API Gateway control service for creating a new API from an external API definition</td>
</tr>
<tr>
<td>put_gateway_response</td>
<td>Creates a customization of a GatewayResponse of a specified response type and status code</td>
</tr>
<tr>
<td>put_integration</td>
<td>Sets up a method’s integration</td>
</tr>
<tr>
<td>put_integration_response</td>
<td>Represents a put integration</td>
</tr>
<tr>
<td>put_method</td>
<td>Add a method to an existing Resource resource</td>
</tr>
<tr>
<td>put_method_response</td>
<td>Adds a MethodResponse to an existing Method resource</td>
</tr>
<tr>
<td>put_request_validator</td>
<td>A feature of the API Gateway control service for updating an existing API with an input of an external API definition</td>
</tr>
<tr>
<td>put_rest_api</td>
<td>Adds or updates a tag on a given resource</td>
</tr>
<tr>
<td>test_invoke_authorizer</td>
<td>Simulate the execution of an Authorizer in your RestApi with headers, parameters, and an incoming request body</td>
</tr>
<tr>
<td>test_invoke_method</td>
<td>Simulate the execution of a Method in your RestApi with headers, parameters, and an incoming request body</td>
</tr>
<tr>
<td>update_account</td>
<td>Changes information about the current Account resource</td>
</tr>
<tr>
<td>update_api_key</td>
<td>Changes information about an ApiKey resource</td>
</tr>
<tr>
<td>update_authorizer</td>
<td>Updates an existing Authorizer resource</td>
</tr>
<tr>
<td>update_base_path_mapping</td>
<td>Changes information about the BasePathMapping resource</td>
</tr>
<tr>
<td>update_client_certificate</td>
<td>Changes information about an ClientCertificate resource</td>
</tr>
<tr>
<td>update_deployment</td>
<td>Changes information about a Deployment resource</td>
</tr>
<tr>
<td>update_documentation_part</td>
<td>Update documentation part</td>
</tr>
<tr>
<td>update_documentation_version</td>
<td>Update documentation version</td>
</tr>
<tr>
<td>update_domain_name</td>
<td>Changes information about the DomainName resource</td>
</tr>
<tr>
<td>update_gateway_response</td>
<td>Updates a GatewayResponse of a specified response type on the given RestApi</td>
</tr>
<tr>
<td>update_integration</td>
<td>Represents an update integration</td>
</tr>
<tr>
<td>update_integration_response</td>
<td>Represents an update integration response</td>
</tr>
<tr>
<td>update_method</td>
<td>Updates an existing Method resource</td>
</tr>
<tr>
<td>update_method_response</td>
<td>Updates an existing MethodResponse resource</td>
</tr>
<tr>
<td>update_model</td>
<td>Changes information about a model</td>
</tr>
<tr>
<td>update_request_validator</td>
<td>Updates a RequestValidator of a given RestApi</td>
</tr>
<tr>
<td>update_resource</td>
<td>Changes information about a Resource resource</td>
</tr>
<tr>
<td>update_rest_api</td>
<td>Changes information about the specified API</td>
</tr>
<tr>
<td>update_stage</td>
<td>Changes information about a Stage resource</td>
</tr>
<tr>
<td>update_usage</td>
<td>Grants a temporary extension to the remaining quota of a usage plan associated with a specified API</td>
</tr>
<tr>
<td>update_usage_plan</td>
<td>Updates a usage plan of a given plan Id</td>
</tr>
<tr>
<td>update_vpc_link</td>
<td>Updates an existing VpcLink of a specified identifier</td>
</tr>
</tbody>
</table>
Examples

```r
## Not run:
svc <- apigateway()
svc$create_api_key(
  Foo = 123
)

## End(Not run)
```

---

**apigatewaymanagementapi**

*AmazonApiGatewayManagementApi*

---

**Description**

The Amazon API Gateway Management API allows you to directly manage runtime aspects of your deployed APIs. To use it, you must explicitly set the SDK’s endpoint to point to the endpoint of your deployed API. The endpoint will be of the form `https://{api-id}.execute-api.{region}.amazonaws.com/{stage}`, or will be the endpoint corresponding to your API’s custom domain and base path, if applicable.

**Usage**

```r
apigatewaymanagementapi(config = list())
```

**Arguments**

- `config`  
  Optional configuration of credentials, endpoint, and/or region.

**Value**

A client for the service. You can call the service’s operations using syntax like `svc$operation(...)`, where `svc` is the name you’ve assigned to the client. The available operations are listed in the Operations section.

**Service syntax**

```r
svc <- apigatewaymanagementapi(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
```

```r
creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
```
profile = "string"
),
    endpoint = "string",
    region = "string"
)

Operations

delete_connection     Delete the connection with the provided id
get_connection        Get information about the connection with the provided id
post_to_connection    Sends the provided data to the specified connection

Examples

## Not run:
svc <- apigatewaymanagementapi()
svc$delete_connection(
    Foo = 123
)
## End(Not run)

---

**apigatewayv2**        **AmazonApiGatewayV2**

**Description**

Amazon API Gateway V2

**Usage**

apigatewayv2(config = list())

**Arguments**

| config              | Optional configuration of credentials, endpoint, and/or region. |

**Value**

A client for the service. You can call the service’s operations using syntax like svc$operation(...), where svc is the name you’ve assigned to the client. The available operations are listed in the Operations section.
Service syntax

```r
svc <- apigatewayv2(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string"
    ),
    endpoint = "string",
    region = "string"
  )
)
```

Operations

- `create_api` creates an API resource
- `create_api_mapping` creates an API mapping
- `create_authorizer` creates an Authorizer for an API
- `create_deployment` creates a Deployment for an API
- `create_domain_name` creates a domain name
- `create_integration` creates an Integration
- `create_integration_response` creates an IntegrationResponses
- `create_model` creates a Model for an API
- `create_route` creates a Route for an API
- `create_route_response` creates a RouteResponse for a Route
- `create_stage` creates a Stage for an API
- `create_vpc_link` creates a VPC link
- `delete_access_log_settings` deletes the AccessLogSettings for a Stage
- `delete_api` deletes an API resource
- `delete_api_mapping` deletes an API mapping
- `delete_authorizer` deletes an Authorizer
- `delete_cors_configuration` deletes a CORS configuration
- `delete_deployment` deletes a Deployment
- `delete_domain_name` deletes a domain name
- `delete_integration` deletes an Integration
- `delete_integration_response` deletes an IntegrationResponses
- `delete_model` deletes a Model
- `delete_route` deletes a Route
- `delete_route_request_parameter` deletes a route request parameter
- `delete_route_response` deletes a RouteResponse
- `delete_route_settings` deletes the RouteSettings for a Stage
- `delete_stage` deletes a Stage
- `delete_vpc_link` deletes a VPC link
- `export_api` exports an API
- `get_api` gets an API resource
get_api_mapping Gets an API mapping
get_api_mappings Gets API mappings
get_apis Gets a collection of Api resources
get_authorizer Gets an Authorizer
get_authorizers Gets the Authorizers for an API
get_deployment Gets a Deployment
get_deployments Gets the Deployments for an API
get_domain_name Gets a domain name
get_domain_names Gets the domain names for an AWS account
get_integration Gets an Integration
get_integration_response Gets an IntegrationResponses
get_integration_responses Gets the IntegrationResponses for an Integration
get_integrations Gets the Integrations for an API
get_model Gets a Model
get_models Gets the Models for an API
get_model_template Gets a model template
get_route Gets a Route
get_route_response Gets a RouteResponse
get_route_responses Gets the RouteResponses for a Route
get_routes Gets the Routes for an API
get_stage Gets a Stage
get_stages Gets the Stages for an API
get_tags Gets a collection of Tag resources
get_vpc_link Gets a VPC link
get_vpc_links Gets a collection of VPC links
import_api Imports an API
reimport_api Puts an Api resource
reset_authorizers_cache Resets all authorizer cache entries on a stage
tag_resource Creates a new Tag resource to represent a tag
untag_resource Deletes a Tag
update_api Updates an Api resource
update_api_mapping The API mapping
update_authorizer Updates an Authorizer
update_deployment Updates a Deployment
update_domain_name Updates a domain name
update_integration Updates an Integration
update_integration_response Updates an IntegrationResponses
update_model Updates a Model
update_route Updates a Route
update_route_response Updates a RouteResponse
update_stage Updates a Stage
update_vpc_link Updates a VPC link

Examples

## Not run:
svc <- apigatewayv2()
appmesh

Description
AWS App Mesh is a service mesh based on the Envoy proxy that makes it easy to monitor and control microservices. App Mesh standardizes how your microservices communicate, giving you end-to-end visibility and helping to ensure high availability for your applications.

App Mesh gives you consistent visibility and network traffic controls for every microservice in an application. You can use App Mesh with AWS Fargate, Amazon ECS, Amazon EKS, Kubernetes on AWS, and Amazon EC2.

App Mesh supports microservice applications that use service discovery naming for their components. For more information about service discovery on Amazon ECS, see Service Discovery in the Amazon Elastic Container Service Developer Guide. Kubernetes kube-dns and coredns are supported. For more information, see DNS for Services and Pods in the Kubernetes documentation.

Usage
appmesh(config = list())

Arguments
config Optional configuration of credentials, endpoint, and/or region.

Value
A client for the service. You can call the service’s operations using syntax like svc$operation(...), where svc is the name you’ve assigned to the client. The available operations are listed in the Operations section.

Service syntax
svc <- appmesh(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
    )
  )
)

## End(Not run)
profile = "string",
),
    endpoint = "string",
    region = "string"
)
)

Operations

create_gateway_route  Creates a gateway route
create_mesh           Creates a service mesh
create_route          Creates a route that is associated with a virtual router
create_virtual_gateway Creates a virtual gateway
create_virtual_node   Creates a virtual node within a service mesh
create_virtual_router Creates a virtual router within a service mesh
create_virtual_service Creates a virtual service within a service mesh
delete_gateway_route  Deletes an existing gateway route
delete_mesh           Deletes an existing service mesh
delete_route          Deletes an existing route
delete_virtual_gateway Deletes an existing virtual gateway
delete_virtual_node   Deletes an existing virtual node
delete_virtual_router Deletes an existing virtual router
delete_virtual_service Deletes an existing virtual service
describe_gateway_route Describes an existing gateway route
describe_mesh         Describes an existing service mesh
describe_route        Describes an existing route
describe_virtual_gateway Describes an existing virtual gateway
describe_virtual_node  Describes an existing virtual node
describe_virtual_router Describes an existing virtual router
describe_virtual_service Describes an existing virtual service
list_gateway_routes   Returns a list of existing gateway routes that are associated to a virtual gateway
list_meshes           Returns a list of existing service meshes
list_routes           Returns a list of existing routes in a service mesh
list_tags_for_resource List the tags for an App Mesh resource
list_virtual_gateways Returns a list of existing virtual gateways in a service mesh
list_virtual_nodes    Returns a list of existing virtual nodes
list_virtual_routers  Returns a list of existing virtual routers in a service mesh
list_virtual_services Returns a list of existing virtual services in a service mesh
tag_resource          Associates the specified tags to a resource with the specified resourceArn
untag_resource        Deletes specified tags from a resource
update_gateway_route  Updates an existing gateway route that is associated to a specified virtual gateway in a service mesh
update_mesh           Updates an existing service mesh
update_route          Updates an existing route for a specified service mesh and virtual router
update_virtual_gateway Updates an existing virtual gateway in a specified service mesh
update_virtual_node   Updates an existing virtual node in a specified service mesh
update_virtual_router Updates an existing virtual router in a specified service mesh
update_virtual_service Updates an existing virtual service in a specified service mesh
Examples

```r
## Not run:
svc <- appmesh()
svc$create_gateway_route(
   Foo = 123
)

## End(Not run)
```

---

**cloudfront**  
*Amazon CloudFront*

---

Description

This is the *Amazon CloudFront API Reference*. This guide is for developers who need detailed information about CloudFront API actions, data types, and errors. For detailed information about CloudFront features, see the *Amazon CloudFront Developer Guide*.

Usage

```r
cloudfront(config = list())
```

Arguments

<table>
<thead>
<tr>
<th>config</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optional configuration of credentials, endpoint, and/or region.</td>
</tr>
</tbody>
</table>

Value

A client for the service. You can call the service’s operations using syntax like `svc$operation(...)`, where `svc` is the name you’ve assigned to the client. The available operations are listed in the Operations section.

Service syntax

```r
svc <- cloudfront(
   config = list(
      credentials = list(
         creds = list(
            access_key_id = "string",
            secret_access_key = "string",
            session_token = "string"
         ),
         profile = "string"
      ),
      endpoint = "string",
```
region = "string"

Operations

create_cache_policy
create_cloud_front_origin_access_identity
create_distribution
create_distribution_with_tags
create_field_level_encryption_config
create_field_level_encryption_profile
create_invocation
create_key_group
create_monitoring_subscription
create_origin_request_policy
create_public_key
create_realtime_log_config
create_streaming_distribution
create_streaming_distribution_with_tags
delete_cache_policy
delete_cloud_front_origin_access_identity
delete_distribution
delete_field_level_encryption_config
delete_field_level_encryption_profile
delete_key_group
delete_monitoring_subscription
delete_origin_request_policy
delete_public_key
delete_realtime_log_config
delete_streaming_distribution
get_cache_policy
get_cache_policy_config
get_cloud_front_origin_access_identity
get_cloud_front_origin_access_identity_config
get_distribution
get_distribution_config
get_field_level_encryption
get_field_level_encryption_config
get_field_level_encryption_profile
get_field_level_encryption_profile_config
get_invocation
get_key_group
get_key_group_config
get_monitoring_subscription
get_origin_request_policy
get_origin_request_policy_config
get_public_key

Creates a cache policy
Creates a new origin access identity
Creates a new web distribution
Create a new distribution with tags
Create a new field-level encryption configuration
Create a field-level encryption profile
Create a new invalidation
Creates a key group that you can use with CloudFront signed URLs and signed cookies
Enables additional CloudWatch metrics for the specified CloudFront distribution
Creates an origin request policy
Uploads a public key to CloudFront that you can use with signed URLs and signed cookies
Creates a real-time log configuration
This API is deprecated
This API is deprecated
Deletes a cache policy
Delete an origin access identity
Delete a distribution
Remove a field-level encryption configuration
Remove a field-level encryption profile
Deletes a key group
Disables additional CloudWatch metrics for the specified CloudFront distribution
Deletes an origin request policy
Remove a public key you previously added to CloudFront
Deletes a real-time log configuration
Delete a streaming distribution
Gets a cache policy, including the following metadata:
Gets a cache policy configuration
Get the information about an origin access identity
Get the configuration information about an origin access identity
Get the information about a distribution
Get the configuration information about a distribution
Get the field-level encryption configuration information
Get the field-level encryption profile information
Get the field-level encryption profile configuration information
Get the information about an invalidation
Gets a key group, including the date and time when the key group was last modified
Gets a key group configuration
Gets information about whether additional CloudWatch metrics are enabled
Gets an origin request policy, including the following metadata:
Gets an origin request policy configuration
Gets a public key
cloudfront

### Examples

```r
## Not run:
svc <- cloudfront()
svc$create_cache_policy(
  Foo = 123
)

## End(Not run)
```
Direct Connect

Description
AWS Direct Connect links your internal network to an AWS Direct Connect location over a standard Ethernet fiber-optic cable. One end of the cable is connected to your router, the other to an AWS Direct Connect router. With this connection in place, you can create virtual interfaces directly to the AWS cloud (for example, to Amazon EC2 and Amazon S3) and to Amazon VPC, bypassing Internet service providers in your network path. A connection provides access to all AWS Regions except the China (Beijing) and (China) Ningxia Regions. AWS resources in the China Regions can only be accessed through locations associated with those Regions.

Usage

directconnect(config = list())

Arguments
config Optional configuration of credentials, endpoint, and/or region.

Value
A client for the service. You can call the service’s operations using syntax like svc$operation(...), where svc is the name you’ve assigned to the client. The available operations are listed in the Operations section.

Service syntax

svc <- directconnect(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string"
    ),
    endpoint = "string",
    region = "string"
  )
)
Operations

- `accept_direct_connect_gateway_association_proposal` Accepts a proposal request to attach a virtual private gateway or transit gateway to a Direct Connect gateway
- `allocate_connection_on_interconnect` Deprecated
- `allocate_hosted_connection` Creates a hosted connection on the specified interconnect or a link aggregation group (LAG)
- `allocate_private_virtual_interface` Provisions a private virtual interface to be owned by the specified AWS account
- `allocate_public_virtual_interface` Provisions a public virtual interface to be owned by the specified AWS account
- `allocate_transit_virtual_interface` Provisions a transit virtual interface to be owned by the specified AWS account
- `associate_connection_with_lag` Associates an existing connection with a link aggregation group (LAG)
- `associate_hosted_connection` Associates a hosted connection and its virtual interfaces with a link aggregation group (LAG) or interconnect
- `associate_virtual_interface` Associates a virtual interface with a specified link aggregation group (LAG) or connection
- `confirm_connection` Confirms the creation of the specified hosted connection on an interconnect
- `confirm_private_virtual_interface` Accepts ownership of a private virtual interface created by another AWS account
- `confirm_public_virtual_interface` Accepts ownership of a public virtual interface created by another AWS account
- `confirm_transit_virtual_interface` Accepts ownership of a transit virtual interface created by another AWS account
- `create_bgp_peer` Creates a BGP peer on the specified virtual interface
- `create_connection` Creates a connection between a customer network and a specific AWS Direct Connect location
- `create_direct_connect_gateway` Creates a Direct Connect gateway, which is an intermediate object that enables you to connect a set of virtual interfaces and virtual private gateways
- `create_direct_connect_gateway_association` Creates an association between a Direct Connect gateway and a virtual private gateway
- `create_direct_connect_gateway_association_proposal` Creates a proposal to associate the specified virtual private gateway or transit gateway with the specified Direct Connect gateway
- `create_interconnect` Creates an interconnect between an AWS Direct Connect Partner’s network and a specific AWS Direct Connect location
- `create_lag` Creates a link aggregation group (LAG) with the specified number of bundled physical dedicated connections between the customer network and a specific AWS Direct Connect location
- `create_private_virtual_interface` Creates a private virtual interface
- `create_public_virtual_interface` Creates a public virtual interface
- `create_transit_virtual_interface` Creates a transit virtual interface
- `delete_bgp_peer` Deletes the specified BGP peer on the specified virtual interface with the specified customer address and ASN
- `delete_connection` Deletes the specified connection
- `delete_direct_connect_gateway` Deletes the specified Direct Connect gateway
- `delete_direct_connect_gateway_association` Deletes the association between the specified Direct Connect gateway and virtual private gateway
- `delete_direct_connect_gateway_association_proposal` Deletes the association proposal request between the specified Direct Connect gateway and virtual private gateway
- `delete_interconnect` Deletes the specified interconnect
- `delete_lag` Deletes the specified link aggregation group (LAG)
- `delete_virtual_interface` Deletes a virtual interface
- `describe_connection_loa` Deprecated
- `describe_connections` Displays the specified connection or all connections in this Region
- `describe_connections_on_interconnect` Deprecated
- `describe_direct_connect_gateway_association_proposals` Describes one or more association proposals for connection between a Direct Connect gateway and a virtual private gateway
- `describe_direct_connect_gateway_associations` Lists the associations between your Direct Connect gateways and virtual private gateways
- `describe_direct_connect_gateway_attachments` Lists all your Direct Connect gateways or only the specified Direct Connect gateway
- `describe_direct_connect_gateways` Lists the hosted connections that have been provisioned on the specified Direct Connect gateway
- `describe_hosted_connections` Lists the interconnects owned by the AWS account or only the specified interconnect
- `describe_interconnect_loa` Deprecated
- `describe_interconnects` Describes all your link aggregation groups (LAG) or the specified link aggregation group
- `describe_loa` Gets the LOA-CFA for a connection, interconnect, or link aggregation group
- `describe_locations` Lists the AWS Direct Connect locations in the current AWS Region
- `describe_tags` Describes the tags associated with the specified AWS Direct Connect resource
- `describe_virtual_gateways` Lists the virtual private gateways owned by the AWS account
**elb**

Elastic Load Balancing

**Description**

A load balancer can distribute incoming traffic across your EC2 instances. This enables you to increase the availability of your application. The load balancer also monitors the health of its registered instances and ensures that it routes traffic only to healthy instances. You configure your load balancer to accept incoming traffic by specifying one or more listeners, which are configured with a protocol and port number for connections from clients to the load balancer and a protocol and port number for connections from the load balancer to the instances.

Elastic Load Balancing supports three types of load balancers: Application Load Balancers, Network Load Balancers, and Classic Load Balancers. You can select a load balancer based on your application needs. For more information, see the Elastic Load Balancing User Guide.

This reference covers the 2012-06-01 API, which supports Classic Load Balancers. The 2015-12-01 API supports Application Load Balancers and Network Load Balancers.

To get started, create a load balancer with one or more listeners using `create_load_balancer`. Register your instances with the load balancer using `register_instances_with_load_balancer`.

All Elastic Load Balancing operations are *idempotent*, which means that they complete at most one time. If you repeat an operation, it succeeds with a 200 OK response code.

---

**Examples**

```r
## Not run:
svc <- directconnect()
svc$accept_direct_connect_gateway_association_proposal(
  Foo = 123
)

## End(Not run)
```

---

## Not run:

```r
svc <- directconnect()
svc$accept_direct_connect_gateway_association_proposal(
  Foo = 123
)

## End(Not run)
```
Usage

elb(config = list())

Arguments

config Optional configuration of credentials, endpoint, and/or region.

Value

A client for the service. You can call the service’s operations using syntax like svc$operation(...), where svc is the name you’ve assigned to the client. The available operations are listed in the Operations section.

Service syntax

svc <- elb(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string"
    ),
    endpoint = "string",
    region = "string"
  )
)

Operations

add_tags Adds the specified tags to the specified load balancer
apply_security_groups_to_load_balancer Associates one or more security groups with your load balancer in a virtual private cloud (VPC)
attach_load_balancer_to_subnets Adds one or more subnets to the set of configured subnets for the specified load balancer
configure_health_check Specifies the health check settings to use when evaluating the health state of your EC2 instances
create_app_cookie_stickiness_policy Generates a stickiness policy with sticky session lifetimes that follow that of an application-generated cookie
create_lb_cookie_stickiness_policy Generates a stickiness policy with sticky session lifetimes controlled by the browser (user-agent) or a specified expiration period
create_load_balancer Creates a Classic Load Balancer
create_load_balancer_listeners Creates one or more listeners for the specified load balancer
create_load_balancer_policy Creates a policy with the specified attributes for the specified load balancer
delete_load_balancer Deletes the specified load balancer
delete_load_balancer_listeners Deletes the specified listeners from the specified load balancer
delete_load_balancer_policy Deletes the specified policy from the specified load balancer
deregister_instances_from_load_balancer Deregisters the specified instances from the specified load balancer
describe_account_limits Describes the current Elastic Load Balancing resource limits for your AWS account
describe_instance_health Describes the state of the specified instances with respect to the specified load balancer
describe_load_balancer_attributes Describes the attributes for the specified load balancer
**Description**

A load balancer distributes incoming traffic across targets, such as your EC2 instances. This enables you to increase the availability of your application. The load balancer also monitors the health of its registered targets and ensures that it routes traffic only to healthy targets. You configure your
load balancer to accept incoming traffic by specifying one or more listeners, which are configured with a protocol and port number for connections from clients to the load balancer. You configure a target group with a protocol and port number for connections from the load balancer to the targets, and with health check settings to be used when checking the health status of the targets.

Elastic Load Balancing supports the following types of load balancers: Application Load Balancers, Network Load Balancers, Gateway Load Balancers, and Classic Load Balancers. This reference covers the following load balancer types:

- **Application Load Balancer** - Operates at the application layer (layer 7) and supports HTTP and HTTPS.
- **Network Load Balancer** - Operates at the transport layer (layer 4) and supports TCP, TLS, and UDP.
- **Gateway Load Balancer** - Operates at the network layer (layer 3).

For more information, see the Elastic Load Balancing User Guide.

All Elastic Load Balancing operations are idempotent, which means that they complete at most one time. If you repeat an operation, it succeeds.

**Usage**

```python
elbv2(config = list())
```

**Arguments**

- `config` - Optional configuration of credentials, endpoint, and/or region.

**Value**

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

**Service syntax**

```python
svc <- elbv2(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string"
    ),
    endpoint = "string",
    region = "string"
  )
)
```
**Operations**

- add_listener_certificates: Adds the specified SSL server certificate to the certificate list for the specified HTTPS or TLS listener.
- add_tags: Adds the specified tags to the specified Elastic Load Balancing resource.
- create_listener: Creates a listener for the specified Application Load Balancer, Network Load Balancer, or Gateway Load Balancer.
- create_load_balancer: Creates an Application Load Balancer, Network Load Balancer, or Gateway Load Balancer.
- create_rule: Creates a rule for the specified listener.
- create_target_group: Creates a target group.
- delete_listener: Deletes the specified listener.
- delete_load_balancer: Deletes the specified Application Load Balancer, Network Load Balancer, or Gateway Load Balancer.
- delete_rule: Deletes the specified rule.
- delete_target_group: Deletes the specified target group.
- deregister_targets: Deregisters the specified targets from the specified target group.
- describe_account_limits: Describes the current Elastic Load Balancing resource limits for your AWS account.
- describe_listener_certificates: Describes the default certificate and the certificate list for the specified HTTPS or TLS listener.
- describe_listeners: Describes the specified listeners or the listeners for the specified Application Load Balancer, Network Load Balancer, or Gateway Load Balancer.
- describe_load_balancer_attributes: Describes the attributes for the specified Application Load Balancer, Network Load Balancer, or Gateway Load Balancer.
- describe_load_balancers: Describes the specified load balancers or all of your load balancers.
- describe_rules: Describes the specified rules or the rules for the specified listener.
- describe_ssl_policies: Describes the specified policies or all policies used for SSL negotiation.
- describe_tags: Describes the tags for the specified Elastic Load Balancing resources.
- describe_target_group_attributes: Describes the attributes for the specified target group.
- describe_target_groups: Describes the specified target groups or all of your target groups.
- describe_target_health: Describes the health of the specified targets or all of your targets.
- modify_listener: Modifies the specified properties of the specified listener.
- modify_load_balancer_attributes: Modifies the specified attributes of the specified Application Load Balancer, Network Load Balancer, or Gateway Load Balancer.
- modify_rule: Modifies the specified properties of the specified rule.
- modify_target_group: Modifies the health checks used when evaluating the health state of the targets in the specified target group.
- modify_target_group_attributes: Modifies the specified attributes of the specified target group.
- register_targets: Registers the specified targets with the specified target group.
- remove_listener_certificates: Removes the specified certificate from the certificate list for the specified HTTPS or TLS listener.
- remove_tags: Removes the specified tags from the specified Elastic Load Balancing resources.
- set_ip_address_type: Sets the type of IP addresses used by the subnets of the specified Application Load Balancer or Network Load Balancer.
- set_rule_priorities: Sets the priorities of the specified rules.
- set_security_groups: Associates the specified security groups with the specified Application Load Balancer.
- set_subnets: Enables the Availability Zones for the specified public subnets for the specified Application Load Balancer.

**Examples**

```r
## Not run:
svc <- elbv2()
# This example adds the specified tags to the specified load balancer.
svc$add_tags(
    ResourceArns = list(
        "arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/m..."
    ),
    Tags = list(
```
list(
    Key = "project",
    Value = "lima"
),
list(
    Key = "department",
    Value = "digital-media"
)

## End(Not run)

globalaccelerator  AWS Global Accelerator

Description

This is the AWS Global Accelerator API Reference. This guide is for developers who need detailed information about AWS Global Accelerator API actions, data types, and errors. For more information about Global Accelerator features, see the AWS Global Accelerator Developer Guide.

AWS Global Accelerator is a service in which you create accelerators to improve the performance of your applications for local and global users. Depending on the type of accelerator you choose, you can gain additional benefits.

- By using a standard accelerator, you can improve availability of your internet applications that are used by a global audience. With a standard accelerator, Global Accelerator directs traffic to optimal endpoints over the AWS global network.
- For other scenarios, you might choose a custom routing accelerator. With a custom routing accelerator, you can use application logic to directly map one or more users to a specific endpoint among many endpoints.

Global Accelerator is a global service that supports endpoints in multiple AWS Regions but you must specify the US West (Oregon) Region to create or update accelerators.

By default, Global Accelerator provides you with two static IP addresses that you associate with your accelerator. With a standard accelerator, instead of using the IP addresses that Global Accelerator provides, you can configure these entry points to be IPv4 addresses from your own IP address ranges that you bring to Global Accelerator. The static IP addresses are anycast from the AWS edge network. For a standard accelerator, they distribute incoming application traffic across multiple endpoint resources in multiple AWS Regions, which increases the availability of your applications. Endpoints for standard accelerators can be Network Load Balancers, Application Load Balancers, Amazon EC2 instances, or Elastic IP addresses that are located in one AWS Region or multiple Regions. For custom routing accelerators, you map traffic that arrives to the static IP addresses to specific Amazon EC2 servers in endpoints that are virtual private cloud (VPC) subnets.

The static IP addresses remain assigned to your accelerator for as long as it exists, even if you disable the accelerator and it no longer accepts or routes traffic. However, when you delete an
accelerator, you lose the static IP addresses that are assigned to it, so you can no longer route traffic by using them. You can use IAM policies like tag-based permissions with Global Accelerator to limit the users who have permissions to delete an accelerator. For more information, see Tag-based policies.

For standard accelerators, Global Accelerator uses the AWS global network to route traffic to the optimal regional endpoint based on health, client location, and policies that you configure. The service reacts instantly to changes in health or configuration to ensure that internet traffic from clients is always directed to healthy endpoints.

For a list of the AWS Regions where Global Accelerator and other services are currently supported, see the AWS Region Table.

AWS Global Accelerator includes the following components:

**Static IP addresses:**
Global Accelerator provides you with a set of two static IP addresses that are anycast from the AWS edge network. If you bring your own IP address range to AWS (BYOIP) to use with a standard accelerator, you can instead assign IP addresses from your own pool to use with your accelerator. For more information, see Bring your own IP addresses (BYOIP) in AWS Global Accelerator.

The IP addresses serve as single fixed entry points for your clients. If you already have Elastic Load Balancing load balancers, Amazon EC2 instances, or Elastic IP address resources set up for your applications, you can easily add those to a standard accelerator in Global Accelerator. This allows Global Accelerator to use static IP addresses to access the resources.

The static IP addresses remain assigned to your accelerator for as long as it exists, even if you disable the accelerator and it no longer accepts or routes traffic. However, when you delete an accelerator, you lose the static IP addresses that are assigned to it, so you can no longer route traffic by using them. You can use IAM policies like tag-based permissions with Global Accelerator to delete an accelerator. For more information, see Tag-based policies.

**Accelerator:**
An accelerator directs traffic to endpoints over the AWS global network to improve the performance of your internet applications. Each accelerator includes one or more listeners.

There are two types of accelerators:

- A **standard** accelerator directs traffic to the optimal AWS endpoint based on several factors, including the user’s location, the health of the endpoint, and the endpoint weights that you configure. This improves the availability and performance of your applications. Endpoints can be Network Load Balancers, Application Load Balancers, Amazon EC2 instances, or Elastic IP addresses.

- A **custom routing** accelerator directs traffic to one of possibly thousands of Amazon EC2 instances running in a single or multiple virtual private clouds (VPCs). With custom routing, listener ports are mapped to statically associate port ranges with VPC subnets, which allows Global Accelerator to determine an EC2 instance IP address at the time of connection. By default, all port mapping destinations in a VPC subnet can’t receive traffic. You can choose to configure all destinations in the subnet to receive traffic, or to specify individual port mappings that can receive traffic.

For more information, see Types of accelerators.

**DNS name:**
Global Accelerator assigns each accelerator a default Domain Name System (DNS) name, similar to `a1234567890abcdef.awsglobalaccelerator.com`, that points to the static IP addresses that Global Accelerator assigns to you or that you choose from your own IP address range. Depending on the use case, you can use your accelerator’s static IP addresses or DNS name to route traffic to your accelerator, or set up DNS records to route traffic using your own custom domain name.

**Network zone:**
A network zone services the static IP addresses for your accelerator from a unique IP subnet. Similar to an AWS Availability Zone, a network zone is an isolated unit with its own set of physical infrastructure. When you configure an accelerator, by default, Global Accelerator allocates two IPv4 addresses for it. If one IP address from a network zone becomes unavailable due to IP address blocking by certain client networks, or network disruptions, then client applications can retry on the healthy static IP address from the other isolated network zone.

**Listener:**
A listener processes inbound connections from clients to Global Accelerator, based on the port (or port range) and protocol (or protocols) that you configure. A listener can be configured for TCP, UDP, or both TCP and UDP protocols. Each listener has one or more endpoint groups associated with it, and traffic is forwarded to endpoints in one of the groups. You associate endpoint groups with listeners by specifying the Regions that you want to distribute traffic to. With a standard accelerator, traffic is distributed to optimal endpoints within the endpoint groups associated with a listener.

**Endpoint group:**
Each endpoint group is associated with a specific AWS Region. Endpoint groups include one or more endpoints in the Region. With a standard accelerator, you can increase or reduce the percentage of traffic that would be otherwise directed to an endpoint group by adjusting a setting called a *traffic dial*. The traffic dial lets you easily do performance testing or blue/green deployment testing, for example, for new releases across different AWS Regions.

**Endpoint:**
An endpoint is a resource that Global Accelerator directs traffic to.
Endpoints for standard accelerators can be Network Load Balancers, Application Load Balancers, Amazon EC2 instances, or Elastic IP addresses. An Application Load Balancer endpoint can be internet-facing or internal. Traffic for standard accelerators is routed to endpoints based on the health of the endpoint along with configuration options that you choose, such as endpoint weights. For each endpoint, you can configure weights, which are numbers that you can use to specify the proportion of traffic to route to each one. This can be useful, for example, to do performance testing within a Region.
Endpoints for custom routing accelerators are virtual private cloud (VPC) subnets with one or many EC2 instances.

**Usage**

```python
globalaccelerator(config = list())
```

**Arguments**

- `config` Optional configuration of credentials, endpoint, and/or region.
A client for the service. You can call the service’s operations using syntax like `svc$operation(...)`, where `svc` is the name you’ve assigned to the client. The available operations are listed in the Operations section.

Service syntax

```r
svc <- globalaccelerator(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string"
    ),
    endpoint = "string",
    region = "string"
  )
)
```

Operations

- `add_custom_routing_endpoints`: Associate a virtual private cloud (VPC) subnet endpoint with your custom routing accelerator.
- `advertise_byoip_cidr`: Advertises an IPv4 address range that is provisioned for use with your AWS resources through bring your own IP addresses (BYOIP).
- `allow_custom_routing_traffic`: Specify the Amazon EC2 instance (destination) IP addresses and ports that can receive traffic for a custom routing accelerator.
- `create_accelerator`: Create an accelerator.
- `create_custom_routing_accelerator`: Create a custom routing accelerator.
- `create_custom_routing_endpoint_group`: Create an endpoint group for the specified listener for a custom routing accelerator.
- `create_custom_routing_listener`: Create a listener to process inbound connections from clients to a custom routing accelerator.
- `create_endpoint_group`: Create an endpoint group for the specified listener.
- `create_listener`: Create a listener to process inbound connections from clients to an accelerator.
- `delete_accelerator`: Delete an accelerator.
- `delete_custom_routing_accelerator`: Delete a custom routing accelerator.
- `delete_custom_routing_endpoint_group`: Delete an endpoint group from a listener for a custom routing accelerator.
- `delete_custom_routing_listener`: Delete a listener for a custom routing accelerator.
- `delete_endpoint_group`: Delete an endpoint group from a listener.
- `delete_listener`: Delete a listener from an accelerator.
- `deny_custom_routing_traffic`: Specify the Amazon EC2 instance (destination) IP addresses and ports that cannot receive traffic for a custom routing accelerator.
- `deprovision_byoip_cidr`: Releases the specified address range that you provisioned to use with your BYOIP address pool.
- `describe_accelerator`: Describe an accelerator.
- `describe_accelerator_attributes`: Describe the attributes of an accelerator.
- `describe_custom_routing_accelerator`: Describe a custom routing accelerator.
- `describe_custom_routing_accelerator_attributes`: Describe the attributes of a custom routing accelerator.
- `describe_custom_routing_endpoint_group`: Describe an endpoint group for a custom routing accelerator.
- `describe_custom_routing_listener`: The description of a listener for a custom routing accelerator.
- `describe_endpoint_group`: Describe an endpoint group.
describe_listener
list_accelerators
list_byoip_cidrs
list_custom_routing_accelerators
list_custom_routing_endpoint_groups
list_custom_routing_listeners
list_custom_routing_port_mappings
list_custom_routing_port_mappings_by_destination
list_endpoint_groups
list_listeners
list_tags_for_resource
provision_byoip_cidr
remove_custom_routing_endpoints
tag_resource
untag_resource
update_accelerator
update_accelerator_attributes
update_custom_routing_accelerator
update_custom_routing_accelerator_attributes
update_custom_routing_endpoint_group
update_listener
withdraw_byoip_cidr

Describe a listener
List the accelerators for an AWS account
Lists the IP address ranges that were specified in calls to ProvisionByoipCidr
List the custom routing accelerators for an AWS account
List the endpoint groups that are associated with a listener for a custom routing accelerator
List the listeners for a custom routing accelerator
Provides a complete mapping from the public accelerator IP address and port to destination EC2 instance and port in a VPC subnet endpoint
List the port mappings for a specific EC2 instance (destination) in a VPC subnet endpoint
List the endpoint groups that are associated with a listener for a custom routing accelerator
List the listeners for a custom routing accelerator
List all tags for an accelerator
Provisions an IP address range to use with your AWS resources through bring your own IP addresses (BYOIP) and creates a corresponding address pool
Remove endpoints from a custom routing accelerator
Add tags to an accelerator resource
Remove tags from a Global Accelerator resource
Update an accelerator
Update the attributes for an accelerator
Update a custom routing accelerator
Update the attributes for a custom routing accelerator
Update a listener for a custom routing accelerator
Update an endpoint group
Update a listener
Stops advertising an address range that is provisioned as an address pool

Examples

```r
## Not run:
svc <- globalaccelerator()
svc$add_custom_routing_endpoints(
  Foo = 123
)

## End(Not run)
```

route53

Amazon Route 53

Description

Amazon Route 53 is a highly available and scalable Domain Name System (DNS) web service.

Usage

```
route53(config = list())
```
Arguments

config Optional configuration of credentials, endpoint, and/or region.

Value

A client for the service. You can call the service’s operations using syntax like `svc$operation(...)`, where `svc` is the name you’ve assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- route53(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string"
    ),
    endpoint = "string",
    region = "string"
  )
)
```

Operations

- **activate_key_signing_key**: Activates a key signing key (KSK) so that it can be used for signing by DNSSEC.
- **associate_vpc_with_hosted_zone**: Associates an Amazon VPC with a private hosted zone.
- **change_resource_record_sets**: Creates, changes, or deletes a resource record set, which contains authoritative DNS information for a specified domain name or subdomain name.
- **change_tags_for_resource**: Adds, edits, or deletes tags for a health check or a hosted zone.
- **create_health_check**: Creates a new health check.
- **create_hosted_zone**: Creates a new public or private hosted zone.
- **create_key_signing_key**: Creates a new key signing key (KSK) associated with a hosted zone.
- **create_query_logging_config**: Creates a configuration for DNS query logging.
- **create_reusable_delegation_set**: Creates a delegation set (a group of four name servers) that can be reused by multiple hosted zones that were created by the same AWS account.
- **create_traffic_policy**: Creates a traffic policy, which you use to create multiple DNS resource record sets for one domain name (such as example.com).
- **create_traffic_policy_instance**: Creates resource record sets in a specified hosted zone based on the settings in a specified traffic policy version.
- **create_traffic_policy_version**: Creates a new version of an existing traffic policy.
- **create_vpc_association_authorization**: Authorizes the AWS account that created a specified VPC to submit an AssociateVPCWithHostedZone request to associate the VPC with a specified hosted zone created by a different account.
- **deactivate_key_signing_key**: Deactivates a key signing key (KSK) so that it will not be used for signing by DNSSEC.
- **delete_health_check**: Deletes a health check.
- **delete_hosted_zone**: Deletes a hosted zone.
- **delete_key_signing_key**: Deletes a key signing key (KSK).
- **delete_query_logging_config**: Deletes a configuration for DNS query logging.
- **delete_reusable_delegation_set**: Deletes a reusable delegation set.
- **delete_traffic_policy**: Deletes a traffic policy.
delete_traffic_policy_instance
delete_vpc_association_authorization
disable_hosted_zone_dnssec
disassociate_vpc_from_hosted_zone
enable_hosted_zone_dnssec
get_account_limit
get_change
get_checker_ip_ranges
get_dnssec
get_geo_location
get_health_check
get_health_check_count
get_health_check_last_failure_reason
get_health_check_status
get_hosted_zone
get_hosted_zone_count
get_hosted_zone_limit
get_query_logging_config
get_reusable_delegation_set
get_reusable_delegation_set_limit
get_traffic_policy
get_traffic_policy_instance
get_traffic_policy_instance_count
list_geo_locations
list_health_checks
list_hosted_zones
list_hosted_zones_by_name
list_hosted_zones_by_vpc
list_query_logging_configs
list_resource_record_sets
list_reusable_delegation_sets
list_tags_for_resource
list_tags_for_resources
list_traffic_policies
list_traffic_policy_instances
list_traffic_policy_instances_by_hosted_zone
list_traffic_policy_instances_by_policy
list_traffic_policy_versions
list_vpc_association_authorizations
test_dns_answer
update_health_check
update_hosted_zone_comment
update_traffic_policy_comment
update_traffic_policy_instance

Deletes a traffic policy instance and all of the resource record sets that Amazon Route 53 created when you created the instance.

Removes authorization to submit an AssociateVPCWithHostedZone request to associate a specified VPC with a hosted zone that was created by a different account.

Disables DNSSEC signing in a specific hosted zone.

Disassociates an Amazon Virtual Private Cloud (Amazon VPC) from an Amazon Route 53 private hosted zone.

Enables DNSSEC signing in a specific hosted zone.

Gets the specified limit for the current account, for example, the maximum number of health checks that you can create using the account.

Returns the current status of a change batch request.

GetCheckerIpRanges still works, but we recommend that you download ip-ranges from the Amazon Route 53 console.

Returns information about DNSSEC for a specific hosted zone, including the key signing keys (KSKs) and zone signing keys (ZSKs) in the hosted zone.

Gets information about whether a specified geographic location is supported for Amazon Route 53 geolocation resource record sets.

Retrieves the number of health checks that are associated with the current AWS account.

Gets the reason that a specified health check failed most recently.

Gets status of a specified health check.

Gets information about a specified hosted zone including the four name servers assigned to the hosted zone.

Retrieves the number of hosted zones that are associated with the current AWS account.

Gets the specified limit for a specified hosted zone, for example, the maximum number of records that you can create in the hosted zone.

Gets information about a specified configuration for DNS query logging.

Retrieves information about a specified reusable delegation set, including the four name servers that are assigned to the delegation set.

Gets information about a specific traffic policy version.

Gets information about a specified traffic policy instance.

Gets the number of traffic policy instances that are associated with the current AWS account.

Retrieves a list of supported geographic locations.

Retrieves a list of the health checks that are associated with the current AWS account.

Retrieves a list of the public and private hosted zones that are associated with the current AWS account.

Retrieves a list of your hosted zones in lexicographic order.

Lists all the private hosted zones that a specified VPC is associated with, regardless of which AWS account or AWS service owns the hosted zones.

Lists the configurations for DNS query logging that are associated with the current AWS account.

Lists the resource record sets in a specified hosted zone.

Retrieves a list of the reusable delegation sets that are associated with the current AWS account.

Lists tags for one health check or hosted zone.

Lists tags for up to 10 health checks or hosted zones.

Gets information about the latest version for every traffic policy that is associated with the current AWS account.

Gets information about the traffic policy instances that you created by using the specified traffic policy version.

Gets information about the traffic policy instances that you created by using a specified traffic policy.

Gets information about all of the versions for a specified traffic policy.

Gets a list of the VPCs that were created by other accounts and that can be associated with the current hosted zone.

Gets the value that Amazon Route 53 returns in response to a DNS request for a specified resource record set.

Updates an existing health check.

Updates the comment for a specified hosted zone.

Updates the comment for a specified traffic policy version.

Updates the resource record sets in a specified hosted zone that were created by using the specified traffic policy instance.
Examples

```r
## Not run:
svc <- route53()
# The following example associates the VPC with ID vpc-1a2b3c4d with the
# hosted zone with ID Z3M3LMPEXAMPLE.
svc$associate_vpc_with_hosted_zone(
  Comment = "",
  HostedZoneId = "Z3M3LMPEXAMPLE",
  VPC = list(
    VPCId = "vpc-1a2b3c4d",
    VPCRegion = "us-east-2"
  )
)
## End(Not run)
```

Description

Amazon Route 53 API actions let you register domain names and perform related operations.

Usage

```r
route53domains(config = list())
```

Arguments

```r
config
```
Optional configuration of credentials, endpoint, and/or region.

Value

A client for the service. You can call the service’s operations using syntax like `svc$operation(...),` where `svc` is the name you’ve assigned to the client. The available operations are listed in the Operations section.

Service syntax

```r
csvc <- route53domains(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      )
    )
  )
)
```
profile = "string"
),
    endpoint = "string",
    region = "string"
)

Operations

accept_domain_transfer_from_another_aws_account
cancel_domain_transfer_to_another_aws_account
check_domain_availability
check_domain_transferability
delete_tags_for_domain
disable_domain_auto_renew
disable_domain_transfer_lock
enable_domain_auto_renew
enable_domain_transfer_lock
get_contact_reachability_status
get_domain_detail
get_domain_suggestions
get_operation_detail
list_domains
list_operations
list_tags_for_domain
register_domain
reject_domain_transfer_from_another_aws_account
renew_domain
resend_contact_reachability_email
retrieve_domain_auth_code
transfer_domain
transfer_domain_to_another_aws_account
update_domain_contact
update_domain_contact_privacy
update_domain_nameservers
update_tags_for_domain
view_billing

Accepts the transfer of a domain from another AWS account to the current AWS account.
Cancels the transfer of a domain from the current AWS account to another AWS account.
This operation checks the availability of one domain name.
Checks whether a domain name can be transferred to Amazon Route 53.
This operation deletes the specified tags for a domain.
This operation disables automatic renewal of domain registration for the specified domain.
This operation removes the transfer lock on the domain (specifically the clientTransferProhibited status).
This operation configures Amazon Route 53 to automatically renew the specified domain.
This operation sets the transfer lock on the domain (specifically the clientTransferProhibited status).
For operations that require confirmation that the email address for the registrant contact is valid for a new domain registration, this operation returns information about whether the registrant contact has responded.
This operation returns detailed information about a specified domain.
The GetDomainSuggestions operation returns a list of suggested domain names.
This operation returns the current status of an operation that is not completed.
This operation returns all the domain names registered with Amazon Route 53.
Returns information about all of the operations that return an operation ID and that have ever been performed on domains that were registered by the current account.
This operation returns all of the tags that are associated with the specified domain.
This operation registers a domain.
Rejects the transfer of a domain from another AWS account to the current AWS account.
This operation renews a domain for the specified number of years.
For operations that require confirmation that the email address for the registrant contact is valid for a new domain registration, this operation returns information about whether the registrant contact has responded.
This operation returns the AuthCode for the domain.
Transfers a domain from another registrar to Amazon Route 53.
Transfers a domain from the current AWS account to another AWS account.
This operation updates the contact information for a particular domain.
This operation updates the specified domain contact’s privacy setting.
This operation replaces the current set of name servers for the domain with a new set of name servers.
This operation adds or updates tags for a specified domain.
Returns all the domain-related billing records for the current AWS account.

Examples

```r
## Not run:
svc <- route53domains()
svc$accept_domain_transfer_from_another_aws_account(
    Foo = 123
)

## End(Not run)
```
Amazon Route 53 Resolver

Description

When you create a VPC using Amazon VPC, you automatically get DNS resolution within the VPC from Route 53 Resolver. By default, Resolver answers DNS queries for VPC domain names such as domain names for EC2 instances or ELB load balancers. Resolver performs recursive lookups against public name servers for all other domain names.

You can also configure DNS resolution between your VPC and your network over a Direct Connect or VPN connection:

Forward DNS queries from resolvers on your network to Route 53 Resolver

DNS resolvers on your network can forward DNS queries to Resolver in a specified VPC. This allows your DNS resolvers to easily resolve domain names for AWS resources such as EC2 instances or records in a Route 53 private hosted zone. For more information, see How DNS Resolvers on Your Network Forward DNS Queries to Route 53 Resolver in the Amazon Route 53 Developer Guide.

Conditionally forward queries from a VPC to resolvers on your network

You can configure Resolver to forward queries that it receives from EC2 instances in your VPCs to DNS resolvers on your network. To forward selected queries, you create Resolver rules that specify the domain names for the DNS queries that you want to forward (such as example.com), and the IP addresses of the DNS resolvers on your network that you want to forward the queries to. If a query matches multiple rules (example.com, acme.example.com), Resolver chooses the rule with the most specific match (acme.example.com) and forwards the query to the IP addresses that you specified in that rule. For more information, see How Route 53 Resolver Forwards DNS Queries from Your VPCs to Your Network in the Amazon Route 53 Developer Guide.

Like Amazon VPC, Resolver is regional. In each region where you have VPCs, you can choose whether to forward queries from your VPCs to your network (outbound queries), from your network to your VPCs (inbound queries), or both.

Usage

route53resolver(config = list())

Arguments

config Optional configuration of credentials, endpoint, and/or region.

Value

A client for the service. You can call the service’s operations using syntax like svc$operation(...), where svc is the name you’ve assigned to the client. The available operations are listed in the Operations section.
Service syntax

svc <- route53resolver(
    config = list(
        credentials = list(
            creds = list(
                access_key_id = "string",
                secret_access_key = "string",
                session_token = "string"
            ),
            profile = "string"
        ),
        endpoint = "string",
        region = "string"
    )
)

Operations

associate_resolver_endpoint_ip_address Adds IP addresses to an inbound or an outbound Resolver endpoint
associate_resolver_query_log_config Associates an Amazon VPC with a specified query logging configuration
associate_resolver_rule Associates a Resolver rule with a VPC
create_resolver_endpoint Creates a Resolver endpoint
create_resolver_query_log_config Creates a Resolver query logging configuration, which defines where you want Resolver to save DNS query logs that originate in your VPCs
create_resolver_rule For DNS queries that originate in your VPCs, specifies which Resolver endpoint the queries pass through, one domain name that you want to forward to your network, and the IP addresses of the DNS resolvers in your network
delete_resolver_endpoint Deletes a Resolver endpoint
delete_resolver_query_log_config Deletes a query logging configuration
delete_resolver_rule Deletes a Resolver rule
disassociate_resolver_endpoint_ip_address Removes IP addresses from an inbound or an outbound Resolver endpoint
disassociate_resolver_query_log_config Disassociates a VPC from a query logging configuration
disassociate_resolver_rule Removes the association between a specified Resolver rule and a VPC
get_resolver_dnssec_config Gets DNSSEC validation information for a specified resource
get_resolver_endpoint Gets information about a specified Resolver endpoint, such as whether it’s an inbound or an outbound endpoint
get_resolver_query_log_config Gets information about a specified Resolver query logging configuration, such as the number of VPCs that the configuration is logging queries for and the location that logs are sent to
get_resolver_query_log_config_association Gets information about a specified association between a Resolver query logging configuration and an Amazon VPC
get_resolver_query_log_config_policy Gets information about a query logging policy
get_resolver_rule Gets information about a specified Resolver rule, such as the domain name that the rule forwards DNS queries for and the ID of the outbound Resolver endpoint that the rule is associated with
get_resolver_rule_association Gets information about an association between a specified Resolver rule and a specified VPC
get_resolver_rule_policy Gets information about the Resolver rule policy for a specified rule
get_resolvers endpoint_ip_addresses Gets the IP addresses for a specified Resolver endpoint
get_resolvers endpoints Gets all the Resolver endpoints that were created using the current AWS account
get_resolvers query_log_config Gets information about the specified query logging configurations
get_resolvers query_log_config_associations Lists the associations that were created between Resolver rules and VPCs using the current AWS account
get_resolvers rules Lists the Resolver rules that were created using the current AWS account
list_resolvers Lists the tags that you associated with the specified resource
list_tags_for_resource Specifies an AWS account that you want to share a query logging configuration with
put_resolvers query_log_config_policy Specifies an AWS rule that you want to share with another account, the account
put_resolvers rule_policy
### AWS Cloud Map

AWS Cloud Map lets you configure public DNS, private DNS, or HTTP namespaces that your microservice applications run in. When an instance of the service becomes available, you can call the AWS Cloud Map API to register the instance with AWS Cloud Map. For public or private DNS namespaces, AWS Cloud Map automatically creates DNS records and an optional health check. Clients that submit public or private DNS queries, or HTTP requests, for the service receive an answer that contains up to eight healthy records.

### Usage

```
servicediscovery(config = list())
```

### Arguments

- **config**
  
  Optional configuration of credentials, endpoint, and/or region.

### Value

A client for the service. You can call the service’s operations using syntax like `svc$operation(...)`, where `svc` is the name you’ve assigned to the client. The available operations are listed in the Operations section.
Service syntax

```r
svc <- servicediscovery(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string"
    ),
    endpoint = "string",
    region = "string"
  )
)
```

Operations

- `create_http_namespace`: Creates an HTTP namespace
- `create_private_dns_namespace`: Creates a private namespace based on DNS, which will be visible only inside a specific Amazon VPC
- `create_public_dns_namespace`: Creates a public namespace based on DNS, which will be visible on the internet
- `create_service`: Creates a service, which defines the configuration for the following entities:
- `delete_namespace`: Deletes a namespace from the current account
- `delete_service`: Deletes a specified service
- `deregister_instance`: Deletes the Amazon Route 53 DNS records and health check, if any, that AWS Cloud Map created for the specified instance
- `discover_instances`: Discovers registered instances for a specified namespace and service
- `get_instance`: Gets information about a specified instance
- `get_instances_health_status`: Gets the current health status (Healthy, Unhealthy, or Unknown) of one or more instances
- `get_namespace`: Gets information about a specified namespace
- `get_operation`: Gets information about any operation that returns an operation ID in the response, such as a CreateService request
- `get_service`: Gets the settings for a specified service
- `list_instances`: Lists summary information about the instances that you registered by using a specified service
- `list_namespaces`: Lists summary information about the namespaces that were created by the current AWS account
- `list_operations`: Lists operations that match the criteria that you specify
- `list_services`: Lists summary information for all the services that are associated with one or more specified namespaces
- `list_tags_for_resource`: Lists tags for the specified resource
- `register_instance`: Creates or updates one or more records and, optionally, creates a health check based on the settings in a specified service
- `tag_resource`: Adds one or more tags to the specified resource
- `untag_resource`: Removes one or more tags from the specified resource
- `update_instance_custom_health_status`: Submits a request to change the health status of a custom health check to healthy or unhealthy
- `update_service`: Submits a request to perform the following operations:

Examples

```r
## Not run:
svc <- servicediscovery()
```
# This example creates an HTTP namespace.
svc$create_http_namespace(
    CreatorRequestId = "example-creator-request-id-0001",
    Description = "Example.com AWS Cloud Map HTTP Namespace",
    Name = "example-http.com"
)

## End(Not run)
Index

accept_direct_connect_gateway_association_proposal, 16
accept_domain_transfer_from_another_aws_account, 30
activate_key_signing_key, 27
add_custom_routing_endpoints, 25
add_listener_certificates, 21
add_tags, 18, 21
advertise_byoip_cidr, 25
associate_connection_with_lag, 16
allocate_hosted_connection, 16
allocate_private_virtual_interface, 16
allocate_public_virtual_interface, 16
allocate_transit_virtual_interface, 16
allow_custom_routing_traffic, 25
apigateway, 2
apigatewaymanagementapi, 6
apigatewayv2, 7
apply_security_groups_to_load_balancer, 18
appmesh, 10
associate_connection_with_lag, 16
associate_hosted_connection, 16
associate_resolver_endpoint_ip_address, 32
associate_resolver_query_log_config, 32
associate_resolver_rule, 32
associate_virtual_interface, 16
associate_vpc_with_hosted_zone, 27
attach_load_balancer_to_subnets, 18
cancel_domain_transfer_to_another_aws_account, 30
change_resource_record_sets, 27
change_tags_for_resource, 27
check_domain_availability, 30
check_domain_transferability, 30
cloudfront, 12
configure_health_check, 18
confirm_connection, 16
confirm_private_virtual_interface, 16
confirm_public_virtual_interface, 16
confirm_transit_virtual_interface, 16
create_accelerator, 25
create_api, 8
create_api_key, 3
create_api_mapping, 8
create_app_cookie_stickiness_policy, 18
create_authorizer, 3, 8
create_base_path_mapping, 3
create_bgp_peer, 16
create_cache_policy, 13
create_cloud_front_origin_access_identity, 13
create_connection, 16
create_custom_routing_accelerator, 25
create_custom_routing_endpoint_group, 25
create_custom_routing_listener, 25
create_deployment, 3, 8
create_direct_connect_gateway, 16
create_direct_connect_gateway_association, 16
create_direct_connect_gateway_association_proposal, 16
create_distribution, 13
create_distribution_with_tags, 13
create_documentation_part, 3
create_documentation_version, 3
cancel_domain_transfer_to_another_aws_account, 30
create_health_check, 27
create_hosted_zone, 27
create_http_namespace, 34
create_integration, 8
create_integration_response, 8
create_interconnect, 16
create_invalidation, 13
create_key_group, 13
create_key_signing_key, 27
create_lag, 16
create_lb_cookie_stickiness_policy, 18
create_listener, 21, 25
create_load_balancer, 17, 18, 21
create_load_balancer_listeners, 18
create_load_balancer_policy, 18
create_mesh, 11
create_model, 3, 8
create_monitoring_subscription, 13
create_origin_request_policy, 13
create_private_dns_namespace, 34
create_private_virtual_interface, 16
create_public_dns_namespace, 34
create_public_key, 13
create_public_virtual_interface, 16
create_query_logging_config, 27
create_realtime_log_config, 13
create_request_validator, 3
create_resolver_endpoint, 32
create_resolver_query_log_config, 32
create_resolver_rule, 32
create_resource, 3
create_rest_api, 3
create_reusable_delegation_set, 27
create_route, 8, 11
create_route_response, 8
create_rule, 21
create_service, 34
create_stage, 3, 8
create_streaming_distribution, 13
create_streaming_distribution_with_tags, 13
create_target_group, 21
create_traffic_policy, 27
create_traffic_policy_instance, 27
create_traffic_policy_version, 27
create_transit_virtual_interface, 16
create_usage_plan, 3
create_usage_plan_key, 3
create_virtual_gateway, 11
create_virtual_node, 11
create_virtual_router, 11
create_virtual_service, 11
create_vpc_association_authorization, 27
create_vpc_link, 3, 8
deactivate_key_signing_key, 27
delete_accelerator, 25
delete_access_log_settings, 8
delete_api, 8
delete_api_key, 3
delete_api_mapping, 8
delete_authorizer, 3, 8
delete_base_path_mapping, 3
delete_bgp_peer, 16
delete_cache_policy, 13
delete_client_certificate, 3
delete_cloud_front_origin_access_identity, 13
delete_connection, 7, 16
delete_cors_configuration, 8
delete_custom_routing_accelerator, 25
delete_custom_routing_endpoint_group, 25
delete_custom_routing_listener, 25
delete_deployment, 3, 8
delete_direct_connect_gateway, 16
delete_direct_connect_gateway_association, 16
delete_direct_connect_gateway_association_proposal, 16
delete_distribution, 13
delete_documentation_part, 3
delete_documentation_version, 3
delete_domain_name, 3, 8
delete_endpoint_group, 25
delete_field_level_encryption_config, 13
delete_field_level_encryption_profile, 13
delete_gateway_response, 3
delete_gateway_route, 11
delete_health_check, 27
delete_hosted_zone, 27
delete_integration, 4, 8
delete_integration_response, 4, 8
delete_interconnect, 16
delete_key_group, 13
delete_key_signing_key, 27
delete_lag, 16
delete_listener, 21, 25
delete_load_balancer, 18, 21
delete_load_balancer_listeners, 18
delete_load_balancer_policy, 18
delete_mesh, 11
delete_method, 4
delete_method_response, 4
delete_model, 4, 8
delete_monitoring_subscription, 13
delete_namespace, 34
delete_origin_request_policy, 13
delete_public_key, 13
delete_query_logging_config, 27
delete_realtime_log_config, 13
delete_request_validator, 4
delete_resolver_endpoint, 32
delete_resolver_query_log_config, 32
delete_resolver_rule, 32
delete_resource, 4
delete_rest_api, 4
delete_reusable_delegation_set, 27
delete_route, 8, 11
delete_route_request_parameter, 8
delete_route_response, 8
delete_route_settings, 8
delete_rule, 21
delete_service, 34
delete_stage, 4, 8
delete_streaming_distribution, 13
delete_tags_for_domain, 30
delete_target_group, 21
delete_traffic_policy, 27
delete_traffic_policy_instance, 28
delete_usage_plan, 4
delete_usage_plan_key, 4
delete_virtual_gateway, 11
delete_virtual_interface, 16
delete_virtual_node, 11
delete_virtual_router, 11
delete_virtual_service, 11
delete_vpc_association_authorization, 28
delete_vpc_link, 4, 8
deny_custom_routing_traffic, 25
deprovision_byoip_cidr, 25
deregister_instance, 34
deregister_instances_from_load_balancer, 18
deregister_targets, 21
describe_accelerator, 25
describe_accelerator_attributes, 25
describe_account_limits, 18, 21
describe_connection_loa, 16
describe_connections, 16
describe_connections_on_interconnect, 16
describe_custom_routing_accelerator, 25
describe_custom_routing_accelerator_attributes, 25
describe_custom_routing_endpoint_group, 25
describe_custom_routing_listener, 25
describe_direct_connect_gateway_association_proposals, 16
describe_direct_connect_gateway_associations, 16
describe_direct_connect_gateway_attachments, 16
describe_direct_connect_gateways, 16
describe_endpoint_group, 25
describe_gateway_route, 11
describe_hosted_connections, 16
describe_instance_health, 18
describe_interconnect_loa, 16
describe_interconnects, 16
describe_lags, 16
describe_listener, 26
describe_listener_certificates, 21
describe_listeners, 21
describe_loa, 16
describe_load_balancer_attributes, 18, 21
describe_load_balancer_policies, 19
describe_load_balancer_policy_types, 19
describe_load_balancers, 19, 21
describe_locations, 16
describe_mesh, 11
describe_route, 11
describe_rules, 21
describe.ssl_policies, 21
describe.tags, 16, 19, 21
get_integrations, 9
get_invalidation, 13
get_key_group, 13
get_key_group_config, 13
get_method, 4
get_method_response, 4
get_model, 4, 9
get_model_template, 4, 9
get_models, 4, 9
get_monitoring_subscription, 13
get_namespace, 34
get_operation, 34
get_operation_detail, 30
get_origin_request_policy, 13
get_origin_request_policy_config, 13
get_public_key, 13
get_public_key_config, 14
get_query_logging_config, 28
get_realtime_log_config, 14
get_request_validator, 4
get_request_validators, 4
get_resolver_dnssec_config, 32
get_resolver_endpoint, 32
get_resolver_query_log_config, 32
get_resolver_query_log_config_association, 32
get_resolver_query_log_config_policy, 32
get_resolver_rule, 32
get_resolver_rule_association, 32
get_resolver_rule_policy, 32
get_resource, 4
get_resources, 4
get_rest_api, 4
get_rest_apis, 4
get_reusable_delegation_set, 28
get_reusable_delegation_set_limit, 28
get_route, 9
get_route_response, 9
get_route_responses, 9
get_routes, 9
get_sdk, 5
get_sdk_type, 5
get_sdk_types, 5
get_service, 34
get_stage, 5, 9
get_stages, 5, 9
get_streaming_distribution, 14
get_streaming_distribution_config, 14
get_tags, 5, 9
get_traffic_policy, 28
get_traffic_policy_instance, 28
get_traffic_policy_instance_count, 28
get_usage, 5
get_usage_plan, 5
get_usage_plan_key, 5
get_usage_plan_keys, 5
get_usage_plans, 5
get_vpc_link, 5, 9
get_vpc_links, 5, 9
globalaccelerator, 22
import_api, 9
import_api_keys, 5
import_documentation_parts, 5
import_rest_api, 5
list_accelerators, 26
list_byoip_cidrs, 26
list_cache_policies, 14
list_cloud_front_origin_access_ids, 14
list_custom_routing_accelerators, 26
list_custom_routing_endpoint_groups, 26
list_custom_routing_listeners, 26
list_custom_routing_port_mappings, 26
list_custom_routing_port_mappings_by_destination, 26
list_distributions, 14
list_distributions_by_cache_policy_id, 14
list_distributions_by_key_group, 14
list_distributions_by_origin_request_policy_id, 14
list_distributions_by_realtime_log_config, 14
list_distributions_by_web_acl_id, 14
list_domains, 30
list_endpoint_groups, 26
list_field_level_encryption_configs, 14
list_field_level_encryption_profiles, 14
list_gateway_routes, 11
list_geo_locations, 28
list_health_checks, 28
list_hosted_zones, 28
list_hosted_zones_by_name, 28
list_hosted_zones_by_vpc, 28
list_instances, 34
list_invalidations, 14
list_key_groups, 14
list_listeners, 26
list_meshes, 11
list_namespaces, 34
list_operations, 30, 34
list_origin_request_policies, 14
list_public_keys, 14
list_query_logging_configs, 28
list_realtime_log_configs, 14
list_resolver_dnssec_configs, 32
list_resolver_endpoint_ip_addresses, 32
list_resolver_endpoints, 32
list_resolver_query_log_config_associations, 32
list_resolver_query_log_configs, 32
list_resolver_rule_associations, 32
list_resolver_rules, 32
list_resource_record_sets, 28
list_reusable_delegation_sets, 28
list_routes, 11
list_services, 34
list_streaming_distributions, 14
list_tags_for_domain, 30
list_tags_for_resource, 11, 14, 26, 28, 32, 34
list_tags_for_resources, 28
list_traffic_policies, 28
list_traffic_policy_instances, 28
list_traffic_policy_instances_by_hosted_zone, route53resolver, 31
list_traffic_policy_instances_by_policy, 28
list_traffic_policy_versions, 28
list_virtual_gateways, 11
list_virtual_interface_test_history, 17
list_virtual_nodes, 11
list_virtual_routers, 11
list_virtual_services, 11
list_vpc_association_authorizations, 28
modify_listener, 21
modify_load_balancer_attributes, 19, 21
modify_rule, 21
modify_target_group, 21
modify_target_group_attributes, 21
post_to_connection, 7
provision_byoip_cidr, 26
put_gateway_response, 5
put_integration, 5
put_integration_response, 5
put_method, 5
put_method_response, 5
put_resolver_query_log_config_policy, 32
put_resolver_rule_policy, 32
put_rest_api, 5
register_domain, 30
register_instance, 34
register_instances_with_load_balancer, 17, 19
register_targets, 21
reimport_api, 9
reject_domain_transfer_from_another_aws_account, 30
remove_custom_routing_endpoints, 26
remove_listener_certificates, 21
remove_tags, 19, 21
renew_domain, 30
resend_contact_reachability_email, 30
reset_authorizers_cache, 9
retrieve_domain_auth_code, 30
route53, 26
route53domains, 29
servicediscovery, 33
set_ip_address_type, 21
set_load_balancer_listener_ssl_certificate, 19
set_load_balancer_policies_for_backend_server, 19
set_load_balancer_policies_of_listener, 19
set_rule_priorities, 21
set_security_groups, 21
set_subnets, 21
start_bgp_failover_test, 17
stop_bgp_failover_test, 17
tag_resource, 5, 9, 11, 14, 17, 26, 33, 34

test_dns_answer, 28
test_invoke_authorizer, 5
test_invoke_method, 5
transfer_domain, 30
transfer_domain_to_another_aws_account, 30
untag_resource, 5, 9, 11, 14, 17, 26, 33, 34
update_accelerator, 26
update_accelerator_attributes, 26
update_account, 5
update_api, 9
update_api_key, 5
update_api_mapping, 9
update_authorizer, 5, 9
update_base_path_mapping, 5
update_cache_policy, 14
update_client_certificate, 5
update_cloud_front_origin_access_identity, 14
update_custom_routing_accelerator, 26
update_custom_routing_accelerator_attributes, 26
update_custom_routing_listener, 26
update_deployment, 5, 9
update_direct_connect_gateway_association, 17
update_distribution, 14
update_documentation_part, 5
update_documentation_version, 5
update_domain_contact, 30
update_domain_contact_privacy, 30
update_domain_name, 5, 9
update_domain_nameservers, 30
update_endpoint_group, 26
update_field_level_encryption_config, 14
update_field_level_encryption_profile, 14
update_gateway_response, 5
update_gateway_route, 11
update_health_check, 28
update_hosted_zone_comment, 28
update_instance_zone_comment, 28
update_instance_custom_health_status, 34
update_integration, 5, 9
update_integration_response, 5, 9
update_key_group, 14
update_lag, 17
update_listener, 26
update_mesh, 11
update_method, 5
update_method_response, 5
update_model, 5, 9
update_origin_request_policy, 14
update_public_key, 14
update_realtime_log_config, 14
update_request_validator, 5
update_resolver_dnssec_config, 33
update_resolver_endpoint, 33
update_resolver_rule, 33
update_resource, 5
update_rest_api, 5
update_route, 9, 11
update_route_response, 9
update_service, 34
update_stage, 5, 9
update_streaming_distribution, 14
update_tags_for_domain, 30
update_traffic_policy_comment, 28
update_traffic_policy_instance, 28
update_usage, 5
update_usage_plan, 5
update_virtual_gateway, 11
update_virtual_interface_attributes, 17
update_virtual_node, 11
update_virtual_router, 11
update_virtual_service, 11
update_vpc_link, 5, 9
view_billing, 30
withdraw_byoip_cidr, 26