Package ‘paws.networking’

January 14, 2020

Title  Amazon Web Services Networking & Content Delivery APIs
Version  0.1.6
Description  Interface to Amazon Web Services networking and content delivery APIs, including 'Route 53' Domain Name System service, 'CloudFront' content delivery, load balancing, and more <https://aws.amazon.com/>.
License  Apache License (>= 2.0)
Imports  paws.common (>= 0.2.5)
Suggests  testthat
Encoding  UTF-8
LazyData  true
RoxygenNote  7.0.2
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'
NeedsCompilation  no
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

apigateway(config = list())

Arguments

config Optional configuration of credentials, endpoint, and/or region.
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, for an ApiGateway
- `create_usage_plan_key`: 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
- `delete_integration`: Represents a delete integration
- `delete_integration_response`: Represents a delete integration response
- `delete_method`: Deletes an existing Method resource
- `delete_method_response`: Deletes an existing MethodResponse resource
- `delete_model`: Deletes a model
- `delete_request_validator`: Deletes a RequestValidator of a given RestApi
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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 of a model</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_rest_api</td>
<td>Lists information about a collection of Resource resources</td>
</tr>
<tr>
<td>get_rest_apis</td>
<td>Lists the RestApi resource in the collection</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>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>
</tbody>
</table>
get_usage
get_usage_plan
get_usage_plan_key
get_usage_plan_keys
get_usage_plans
get_vpc_link
get_vpc_links
import_api_keys
import_documentation_parts
import_rest_api
put_gateway_response
put_integration
put_integration_response
put_method
put_method_response
put_rest_api
tag_resource
test_invoke_authorizer
test_invoke_method
untag_resource
update_account
update_api_key
update_authorizer
update_base_path_mapping
update_client_certificate
update_deployment
update_documentation_part
update_documentation_version
update_domain_name
update_gateway_response
update_integration
update_integration_response
update_method
update_method_response
update_model
update_request_validator
update_resource
update_rest_api
update_stage
update_usage
update_usage_plan
update_vpc_link

Gets the usage data of a usage plan in a specified time interval
Gets a usage plan of a given plan identifier
Gets a usage plan key of a given key identifier
Gets all the usage plan keys representing the API keys added to a specified usage plan
Gets all the usage plans of the caller's account
Gets a specified VPC link under the caller's account in a region
Gets the VpcLinks collection under the caller's account in a selected region
Import API keys from an external source, such as a CSV-formatted file
Import documentation parts
A feature of the API Gateway control service for creating a new API from an external API definition file
Creates a customization of a GatewayResponse of a specified response type and status code
Sets up a method's integration
Represents a put integration
Add a method to an existing Resource resource
Adds a MethodResponse to an existing Method resource
A feature of the API Gateway control service for updating an existing API with an input of external API definitions
Simulate the execution of an Authorizer in your RestApi with headers, parameters, and an incoming request body
Simulate the execution of a Method in your RestApi with headers, parameters, and an incoming request body
Removes a tag from a given resource
Changes information about the current Account resource
Changes information about an ApiKey resource
Updates an existing Authorizer resource
Changes information about the BasePathMapping resource
Changes information about an ClientCertificate resource
Changes information about a Deployment resource
Update documentation part
Update documentation version
Changes information about the DomainName resource
Updates a GatewayResponse of a specified response type on the given RestApi
Represents an update integration
Updates an existing Method resource
Updates an existing MethodResponse resource
Changes information about a model
Updates a RequestValidator of a given RestApi
Changes information about a Resource resource
Changes information about the specified API
Changes information about a Stage resource
Grants a temporary extension to the remaining quota of a usage plan associated with a specified API key
Updates a usage plan of a given plan Id
Updates an existing VpcLink of a specified identifier

Examples

svc <- apigateway()
svc$create_api_key()}
apigatewaymanagementapi

Foo = 123
)

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

apigatewaymanagementapi(config = list())

Arguments

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

Service syntax

tsvc <- apigatewaymanagementapi(
  config = list(
    credentials = list(
      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
apigatewayv2

Examples

```r
svc <- apigatewaymanagementapi()
svc$delete_connection(
  Foo = 123
)
```

---

**AmazonApiGatewayV2**

**Description**

Amazon API Gateway V2

**Usage**

```r
apigatewayv2(config = list())
```

**Arguments**

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

**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
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>create_integration_response</td>
<td>Creates an IntegrationResponses</td>
</tr>
<tr>
<td>create_model</td>
<td>Creates a Model for an API</td>
</tr>
<tr>
<td>create_route</td>
<td>Creates a Route for an API</td>
</tr>
<tr>
<td>create_route_response</td>
<td>Creates a RouteResponse for a Route</td>
</tr>
<tr>
<td>create_stage</td>
<td>Creates a Stage for an API</td>
</tr>
<tr>
<td>delete_api</td>
<td>Deletes an Api resource</td>
</tr>
<tr>
<td>delete_api_mapping</td>
<td>Deletes an API mapping</td>
</tr>
<tr>
<td>delete_authorizer</td>
<td>Deletes an Authorizer</td>
</tr>
<tr>
<td>delete_cors_configuration</td>
<td>Deletes a CORS configuration</td>
</tr>
<tr>
<td>delete_deployment</td>
<td>Deletes a Deployment</td>
</tr>
<tr>
<td>delete_domain_name</td>
<td>Deletes a domain name</td>
</tr>
<tr>
<td>delete_integration</td>
<td>Deletes an Integration</td>
</tr>
<tr>
<td>delete_integration_response</td>
<td>Deletes an IntegrationResponses</td>
</tr>
<tr>
<td>delete_model</td>
<td>Deletes a Model</td>
</tr>
<tr>
<td>delete_route</td>
<td>Deletes a Route</td>
</tr>
<tr>
<td>delete_route_response</td>
<td>Deletes a RouteResponse</td>
</tr>
<tr>
<td>delete_route_settings</td>
<td>Deletes the RouteSettings for a stage</td>
</tr>
<tr>
<td>delete_stage</td>
<td>Deletes a Stage</td>
</tr>
<tr>
<td>get_api</td>
<td>Gets an Api resource</td>
</tr>
<tr>
<td>get_api_mapping</td>
<td>Gets an API mapping</td>
</tr>
<tr>
<td>get_api_mappings</td>
<td>Gets API mappings</td>
</tr>
<tr>
<td>get_apis</td>
<td>Gets a collection of Api resources</td>
</tr>
<tr>
<td>get_authorizer</td>
<td>Gets an Authorizer</td>
</tr>
<tr>
<td>get_authorizers</td>
<td>Gets the Authorizers for an API</td>
</tr>
<tr>
<td>get_deployment</td>
<td>Gets a Deployment</td>
</tr>
<tr>
<td>get_deployments</td>
<td>Gets the Deployments for an API</td>
</tr>
<tr>
<td>get_domain_name</td>
<td>Gets a domain name</td>
</tr>
<tr>
<td>get_domain_names</td>
<td>Gets the domain names for an AWS account</td>
</tr>
<tr>
<td>get_integration</td>
<td>Gets an Integration</td>
</tr>
<tr>
<td>get_integration_response</td>
<td>Gets an IntegrationResponses</td>
</tr>
<tr>
<td>get_integration_responses</td>
<td>Gets the IntegrationResponses for an Integration</td>
</tr>
<tr>
<td>get_integrations</td>
<td>Gets the Integrations for an API</td>
</tr>
<tr>
<td>get_model</td>
<td>Gets a Model</td>
</tr>
<tr>
<td>get_models</td>
<td>Gets the Models for an API</td>
</tr>
<tr>
<td>get_model_template</td>
<td>Gets a model template</td>
</tr>
<tr>
<td>get_route</td>
<td>Gets a Route</td>
</tr>
<tr>
<td>get_route_response</td>
<td>Gets a RouteResponse</td>
</tr>
<tr>
<td>get_route_responses</td>
<td>Gets the RouteResponses for a Route</td>
</tr>
<tr>
<td>get_routes</td>
<td>Gets the Routes for an API</td>
</tr>
<tr>
<td>get_stage</td>
<td>Gets a Stage</td>
</tr>
<tr>
<td>get_stages</td>
<td>Gets the Stages for an API</td>
</tr>
<tr>
<td>get_tags</td>
<td>Gets a collection of Tag resources</td>
</tr>
<tr>
<td>import_api</td>
<td>Imports an API</td>
</tr>
<tr>
<td>reimport_api</td>
<td>Puts an Api resource</td>
</tr>
<tr>
<td>tag_resource</td>
<td>Creates a new Tag resource to represent a tag</td>
</tr>
<tr>
<td>untag_resource</td>
<td>Deletes a Tag</td>
</tr>
<tr>
<td>update_api</td>
<td>Updates an Api resource</td>
</tr>
<tr>
<td>update_api_mapping</td>
<td>The API mapping</td>
</tr>
</tbody>
</table>
appmesh

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

Examples

svc <- apigatewayv2()
svc$create_api(
  Foo = 123
)

appmesh

AWS App Mesh

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.
**Service syntax**

```python
csvc <- appmesh(
    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_mesh` Creates a service mesh
- `create_route` Creates a route that is associated with a virtual router
- `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_mesh` Deletes an existing service mesh
- `delete_route` Deletes an existing route
- `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_mesh` Describes an existing service mesh
- `describe_route` Describes an existing route
- `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_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_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_mesh` Updates an existing service mesh
- `update_route` Updates an existing route for a specified service mesh and virtual router
- `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
svc <- appmesh()
svc$create_mesh(
  Foo = 123
)
```

---

### 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

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

### 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_cloud_front_origin_access_identity
create_distribution
create_distribution_with_tags
create_field_level_encryption_config
create_field_level_encryption_profile
delete_cloud_front_origin_access_identity
delete_distribution
delete_field_level_encryption_config
delete_field_level_encryption_profile
delete_public_key
delete_streaming_distribution
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_invalidation
get_public_key
get_public_key_config
get_streaming_distribution
get_streaming_distribution_config
list_cloud_front_origin_access_identities
list_distributions
list_distributions_by_web_acl_id
list_field_level_encryption_configs
list_field_level_encryption_profiles
list_invalidations
list_public_keys
list_streaming_distributions
list_tags_for_resource
tag_resource
untag_resource
update_cloud_front_origin_access_identity
update_distribution
update_field_level_encryption_config
update_field_level_encryption_profile
update_public_key
update_streaming_distribution

Creating 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
Add a new public key to CloudFront to use, for example, for field-level encryption
Create a new RTMP distribution
Create a new streaming distribution with tags
Delete an origin access identity
Delete a distribution
Remove a field-level encryption configuration
Remove a field-level encryption profile
Remove a public key you previously added to CloudFront
Delete a streaming distribution
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 configuration information
Get the field-level encryption profile information
Get the field-level encryption profile configuration information
Get the information about an invalidation
Get the public key information
Return public key configuration information
Get the configuration information about a streaming distribution
Get the configuration information about a streaming distribution
List CloudFront distributions
List the distributions that are associated with a specified AWS WAF web ACL
List all field-level encryption configurations that have been created in CloudFront
Request a list of field-level encryption profiles that have been created in CloudFront
List invalidation batches
List all public keys that have been added to CloudFront for this account
List streaming distributions
List tags for a CloudFront resource
Add tags to a CloudFront resource
Remove tags from a CloudFront resource
Update an origin access identity
Updates the configuration for a web distribution
Update a field-level encryption configuration
Update a field-level encryption profile
Update public key information
Update a streaming distribution
Examples

```r
svc <- cloudfront()
svc$create_cloud_front_origin_access_identity(
  Foo = 123
)
```

---

directconnect  

**AWS 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**

```r
directconnect(config = list())
```

**Arguments**

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

**Service syntax**

```r
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
allocate_connection_on_interconnect
allocate_hosted_connection
allocate_private_virtual_interface
allocate_public_virtual_interface
allocate_transit_virtual_interface
associate_connection_with_lag
associate_hosted_connection
associate_virtual_interface
confirm_connection
confirm_private_virtual_interface
confirm_public_virtual_interface
confirm_transit_virtual_interface
create_bgp_peer
create_connection
create_direct_connect_gateway
create_direct_connect_gateway_association
create_direct_connect_gateway_association_proposal
create_interconnect
create_lag
create_private_virtual_interface
create_public_virtual_interface
create_transit_virtual_interface
delete_bgp_peer
delete_connection
delete_direct_connect_gateway
delete_direct_connect_gateway_association
delete_direct_connect_gateway_association_proposal
delete_interconnect
delete_lag
delete_virtual_interface
describe_connection_loa
describe_connections
describe_connections_on_interconnect
describe_direct_connect_gateway_association_proposals
describe_direct_connect_gateway_associations
describe_direct_connect_gateway_attachments
describe_direct_connect_gateways
describe_hosted_connections
describe_interconnect_loa
describe_interconnects
describe_lags
describe_loa
describe_locations
describe_tags
describe_virtual_gateways
describe_virtual_interfaces
disassociate_connection_from_lag

Accepts a proposal request to attach a virtual private gateway or transit gateway to a Direct Connect gateway
Deprecated
Creates a hosted connection on the specified interconnect or a link aggregation group (LAG)
Provisions a private virtual interface to be owned by the specified AWS account
Provisions a public virtual interface to be owned by the specified AWS account
Provisions a transit virtual interface to be owned by the specified AWS account
Associates an existing connection with a link aggregation group (LAG)
Associates a hosted connection and its virtual interfaces with a link aggregation group (LAG)
Associates a virtual interface with a specified link aggregation group (LAG)
Confirms the creation of the specified hosted connection on an interconnect
Accepts ownership of a private virtual interface created by another AWS account
Accepts ownership of a public virtual interface created by another AWS account
Accepts ownership of a transit virtual interface created by another AWS account
Creates a BGP peer on the specified virtual interface
Creates a connection between a customer network and a specific AWS Direct Connect location
Creates a Direct Connect gateway, which is an intermediate object that enables you to connect a set of virtual interfaces and virtual private gateways
Creates an association between a Direct Connect gateway and a virtual private gateway
Creates a proposal to associate the specified virtual private gateway with the specified Direct Connect gateway
Creates an interconnect between an AWS Direct Connect Partner’s network and a specific AWS Direct Connect location
Creates a link aggregation group (LAG) with the specified number of bundled physical connections
Creates a private virtual interface
Creates a public virtual interface
Creates a transit virtual interface
Deletes the specified BGP peer on the specified virtual interface with the specified customer address and Autonomous System Number (ASN)
Deletes the specified connection
Deletes the specified Direct Connect gateway
Deletes the association between the specified Direct Connect gateway and virtual private gateway
Deletes the association proposal request between the specified Direct Connect gateway and virtual private gateway
Deletes the specified interconnect
Deletes the specified link aggregation group (LAG)
Deletes a virtual interface
Deprecated
Displays the specified connection or all connections in this Region
Deprecated
Describes one or more association proposals for connection between a virtual private gateway or transit gateway and a Direct Connect gateway
Lists the associations between your Direct Connect gateways and virtual private gateways or transit gateways
Lists the attachments between your Direct Connect gateways and virtual private gateways or transit gateways
Lists all your Direct Connect gateways or only the specified Direct Connect gateway
Lists the hosted connections that have been provisioned on the specified Direct Connect gateway
Lists the interconnects owned by the AWS account or only the specified interconnect
Describes all your link aggregation groups (LAG) or the specified LAG
Gets the LOA-CFA for a connection, interconnect, or link aggregation group
Lists the AWS Direct Connect locations in the current AWS Region
Describes the tags associated with the specified AWS Direct Connect gateway
Lists the virtual private gateways owned by the AWS account
Displays all virtual interfaces for an AWS account
Disassociates a connection from a link aggregation group (LAG)
tag_resource
untag_resource
update_direct_connect_gateway_association
update_lag
update_virtual_interface_attributes

Examples

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

---

**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 CreateLoadBalancer. Register your instances with the load balancer using RegisterInstancesWithLoadBalancer.

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.

**Usage**

```r
elb(config = list())
```

**Arguments**

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

```r
svc <- elb(
  config = 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 lifetime of 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.
- `describe_load_balancer_policies` Describes the specified policies for the specified load balancer.
- `describe_load_balancer_policies_of_listener` Describes the specified policies for the specified load balancer port.
- `describe_load_balancer_policy_types` Describes the load balancer policy types or all load balancer policy types.
- `describe_load_balancers` Describes the specified the load balancers.
- `describe_tags` Describes the tags associated with the specified load balancers.
- `detach_load_balancer_from_subnets` Removes the specified subnets from the set of configured subnets for the load balancer.
- `disable_availability_zones_for_load_balancer` Removes the specified Availability Zones from the set of Availability Zones for the load balancer.
- `enable_availability_zones_for_load_balancer` Adds the specified Availability Zones to the set of Availability Zones for the load balancer.
- `modify_load_balancer_attributes` Modifies the attributes of the specified load balancer.
- `register_instances_with_load_balancer` Adds the specified instances to the specified load balancer.
- `remove_tags` Removes one or more tags from the specified load balancer.
- `set_load_balancer_listener.ssl_certificate` Sets the certificate that terminates the specified listener’s SSL connections.
- `set_load_balancer_policies_for_backend_server` Replaces the set of policies associated with the specified port on which the load balancer is receiving client traffic.
- `set_load_balancer_policies_of_listener` Replaces the current set of policies for the specified load balancer port with new policies.
Examples

```r
# This example adds two tags to the specified load balancer.
svc <- elb()
svc$add_tags(
  LoadBalancerNames = list("my-load-balancer"),
  Tags = list(
    list(Key = "project", Value = "lima"),
    list(Key = "department", Value = "digital-media")
  )
)
```

**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, and Classic Load Balancers. This reference covers Application Load Balancers and Network Load Balancers.

An Application Load Balancer makes routing and load balancing decisions at the application layer (HTTP/HTTPS). A Network Load Balancer makes routing and load balancing decisions at the transport layer (TCP/TLS). Both Application Load Balancers and Network Load Balancers can route requests to one or more ports on each EC2 instance or container instance in your virtual private cloud (VPC). 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**

```r
elbv2(config = list())
```
Arguments

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

Service syntax

drv <- 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 or Network Load Balancer
create_load_balancer  Creates an Application Load Balancer or a Network 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 or Network Load Balancer and its attached listeners
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 or Network Load Balancer
describe_load_balancer_attributes  Describes the attributes for the specified Application Load Balancer or Network 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 resource
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 attributes of the specified Application Load Balancer or Network Load Balancer
modify_load_balancer_attributes  Replaces the specified properties of the specified load balancer
modify_rule  Replaces 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
register_targets
remove_listener_certificates
remove_tags
set_ip_address_type
set_rule_priorities
set_security_groups
set_subnets

Examples

# This example adds the specified tags to the specified load balancer.
svc <- elbv2()
svc$add_tags(
  ResourceArns = list(
  ),
  Tags = list(
    list(
      Key = "project",
      Value = "lima"
    ),
    list(
      Key = "department",
      Value = "digital-media"
    )
  )
)
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 directed to only healthy endpoints.

Global Accelerator includes components that work together to help you improve performance and availability for your applications:

Static IP address

AWS Global Accelerator provides you with a set of static IP addresses which are anycast from the AWS edge network and serve as the single fixed entry points for your clients. If you already have Elastic Load Balancing or Elastic IP address resources set up for your applications, you can easily add those to Global Accelerator to allow the resources to be accessed by a Global Accelerator static IP address.

Accelerator

An accelerator directs traffic to optimal endpoints over the AWS global network to improve availability and performance for your internet applications that have a global audience. Each accelerator includes one or more listeners.

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, 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 protocol and port that you configure. 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. 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. 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 new releases across different AWS Regions, for example.

Endpoint

An endpoint is an Elastic IP address, Network Load Balancer, or Application Load Balancer. Traffic is routed to endpoints based on several factors, including the geo-proximity to the user, the health of the endpoint, and the 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.
Usage

`globalaccelerator(config = list())`

Arguments

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

Service syntax

```r
csvc <- 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

- `create_accelerator` Create an 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_endpoint_group` Delete an endpoint group from a listener
- `delete_listener` Delete a listener from an accelerator
- `describe_accelerator` Describe an accelerator
- `describe_accelerator_attributes` Describe the attributes of an accelerator
- `describe_endpoint_group` Describe an endpoint group
- `describe_listener` Describe a listener
- `list_accelerators` List the accelerators for an AWS account
- `list_endpoint_groups` List the endpoint groups that are associated with a listener
- `list_listeners` List the listeners for an accelerator
- `update_accelerator` Update an accelerator
- `update_accelerator_attributes` Update the attributes for an accelerator
- `update_endpoint_group` Update an endpoint group
- `update_listener` Update a listener
Examples

```r
svc <- globalaccelerator()
svc$create_accelerator(
  Foo = 123
)
```

Amazon Route 53

Description

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

Usage

```r
route53(config = list())
```

Arguments

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

Service syntax

```r
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

- **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_query_logging_config**: Creates a configuration for DNS query logging
create_reusable_delegation_set
create_traffic_policy
create_traffic_policy_instance
create_traffic_policy_version
create_vpc_association_authorization
delete_health_check
delete_hosted_zone
delete_query_logging_config
delete_reusable_delegation_set
delete_traffic_policy
delete_traffic_policy_instance
disassociate_vpc_from_hosted_zone
get_account_limit
get_change
get_checker_ip_ranges
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_hosted_zones
list_health_checks
list_hosted_zones
list_hosted_zones_by_name
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

create_reusable_delegation_set
Creates a delegation set (a group of four name servers) that can be reused by multiple hosted zones.
create_traffic_policy
Creates a traffic policy, which you use to create multiple DNS resource record sets.
create_traffic_policy_instance
Creates resource record sets in a specified hosted zone based on the settings in a specified traffic policy version.
create_vpc_association_authorization
Creates a new version of an existing traffic policy.
delete_health_check
Deletes a health check.
delete_hosted_zone
Deletes a hosted zone.
delete_query_logging_config
Deletes a configuration for DNS query logging.
delete_reusable_delegation_set
Deletes a re usable delegation set.
delete_traffic_policy
Deletes a traffic policy.
delete_traffic_policy_instance
Deletes a traffic policy instance and all of the resource record sets that Amazon Route 53 creates when you create the instance.
disassociate_vpc_from_hosted_zone
Disassociates a VPC from a Amazon Route 53 private hosted zone.
get_account_limit
Gets the specified limit for the current account, for example, the maximum number of health checks that you can create using the account.
get_change
Returns the current status of a change batch request.
get_checker_ip_ranges
GetCheckerIpRanges still works, but we recommend that you download ip-ranges.
get_geo_location
Gets information about whether a specified geographic location is supported for Amazon Route 53 geolocation resource record sets.
get_health_check
Gets information about a specified health check.
get_health_check_count
Retrieves the number of health checks that are associated with the current AWS account.
get_health_check_last_failure_reason
Gets the reason that a specified health check failed most recently.
get_health_check_status
Gets status of a specified health check.
get_hosted_zone
Gets information about a specified hosted zone including the four name servers assigned to the hosted zone.
get_hosted_zone_count
Retrieves the number of hosted zones that are associated with the current AWS account.
get_hosted_zone_limit
Gets the specified limit for a specified hosted zone, for example, the maximum number of records that you can create in the hosted zone.
get_query_logging_config
Gets information about a specified configuration for DNS query logging.
get_reusable_delegation_set
Gets information about a specified re usable delegation set, including the four name servers that are assigned to the delegation set.
get_reusable_delegation_set_limit
Gets the maximum number of hosted zones that you can associate with the specified reusable delegation set.
get_traffic_policy
Gets information about a specific traffic policy version.
get_traffic_policy_instance
Gets information about a specified traffic policy instance.
get_traffic_policy_instance_count
Retrieves a list of supported geographic locations.
get_traffic_policy_instance_count
Retrieves a list of the health checks that are associated with the current AWS account.
get_traffic_policy_instances
Retrieves a list of the public and private hosted zones that are associated with the current AWS account.
get_traffic_policy_instances
Retrieves a list of your hosted zones in lexicographic order.
get_traffic_policy_versions
Lists the configurations for DNS query logging that are associated with the current AWS account.
get_traffic_policy_versions
Lists the resource record sets in a specified hosted zone.
get_traffic_policy_versions
Retrieves a list of the reusable delegation sets that are associated with the current AWS account.
get_traffic_policy_versions
Lists tags for one health check or hosted zone.
get_traffic_policy_versions
Lists tags for up to 10 health checks or hosted zones.
get_traffic_policy_versions
Gets information about the latest version for every traffic policy that is associated with the current AWS account.
get_traffic_policy_versions
Gets information about the traffic policy instances that you created by using the current AWS account.
get_traffic_policy_versions
Gets information about all of the versions for a specified traffic policy.
get_traffic_policy_versions
Gets a list of the VPCs that were created by other accounts and that can be associated with a hosted zone.
get_traffic_policy_versions
Gets the value that Amazon Route 53 returns in response to a DNS request for a DNS record.
get_traffic_policy_versions
Updates an existing health check.
get_traffic_policy_versions
Updates the comment for a specified hosted zone.
**update_traffic_policy_comment** Updates the comment for a specified traffic policy version

**update_traffic_policy_instance** Updates the resource record sets in a specified hosted zone that were created based on the settings in a specified traffic policy version

**Examples**

```
# The following example associates the VPC with ID vpc-1a2b3c4d with the
# hosted zone with ID Z3M3LMPEXAMPLE.
svc <- route53()
svc$associate_vpc_with_hosted_zone(
    Comment = "",
    HostedZoneId = "Z3M3LMPEXAMPLE",
    VPC = list(
        VPCId = "vpc-1a2b3c4d",
        VPCRegion = "us-east-2"
    )
)
```

---

**route53domains**  
*Amazon Route 53 Domains*

**Description**

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

**Usage**

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

**Arguments**

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

**Service syntax**

```
svc <- 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

- 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
- renew_domain
- resend_contact_reachability_email
- retrieve_domain_auth_code
- transfer_domain
- update_domain_contact
- update_domain_contact_privacy
- update_domain_nameservers
- update_tags_for_domain
- view_billing

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:
This operation returns detailed information about a specified domain that is associated with the current AWS account
The GetDomainSuggestions operation returns a list of suggested domain names given a string
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 for the current AWS account
This operation returns the operation IDs of operations that are not yet complete
This operation returns all of the tags that are associated with the specified domain
This operation registers a domain
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:
This operation returns the AuthCode for the domain
This operation transfers a domain from another registrar to Amazon Route 53
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 the specified set
This operation adds or updates tags for a specified domain
Returns all the domain-related billing records for the current AWS account for a specified period

Examples

```r
svc <- route53domains()
svc$check_domain_availability(
  Foo = 123
)
```
Description

Here's how you set up to query an Amazon Route 53 private hosted zone from your network:

1. Connect your network to a VPC using AWS Direct Connect or a VPN.
2. Run the following AWS CLI command to create a Resolver endpoint:
   ```
   create-resolver-endpoint --name \[endpoint_name\] --direction INBOUND --creator-request-id \[unique_string\] --security-group-ids \[security_group_with_inbound_rules\] --ip-addresses SubnetId=\[subnet_id\] SubnetId=\[subnet_id_in_different_AZ\]
   ```
   Note the resolver endpoint ID that appears in the response. You'll use it in step 3.
3. Get the IP addresses for the Resolver endpoints:
   ```
   get-resolver-endpoint --resolver-endpoint-id \[resolver_endpoint_id\]
   ```
4. In your network configuration, define the IP addresses that you got in step 3 as DNS servers. You can now query instance names in your VPCs and the names of records in your private hosted zone.

You can also perform the following operations using the AWS CLI:

- `list-resolver-endpoints`: List all endpoints. The syntax includes options for pagination and filtering.
- `update-resolver-endpoints`: Add IP addresses to an endpoint or remove IP addresses from an endpoint.

To delete an endpoint, use the following AWS CLI command:
```
delete-resolver-endpoint --resolver-endpoint-id \[resolver_endpoint_id\]
```

Usage

```python
route53resolver(config = list())
```

Arguments

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

Service syntax

```python
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_rule`: Associates a resolver rule with a VPC.
- `create_resolver_endpoint`: Creates a resolver endpoint.
- `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_rule`: Deletes a resolver rule.
- `disassociate_resolver_endpoint_ip_address`: Removes IP addresses from an inbound or an outbound resolver endpoint.
- `disassociate_resolver_rule`: Removes the association between a specified resolver rule and a specified VPC.
- `get_resolver_endpoint`: Gets information about a specified resolver endpoint, such as whether it’s an inbound or outbound resolver endpoint.
- `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 VPC.
- `get_resolver_rule_policy`: Gets information about a resolver rule policy.
- `list_resolver_endpoint_ip_addresses`: Gets the IP addresses for a specified resolver endpoint.
- `list_resolver_endpoints`: Lists all the resolver endpoints that were created using the current AWS account.
- `list_resolver_rule_associations`: Lists the associations that were created between resolver rules and VPCs using the current AWS account.
- `list_resolver_rules`: Lists the resolver rules that were created using the current AWS account.
- `list_tags_for_resource`: Lists the tags that you associated with the specified resource.
- `put_resolver_rule_policy`: Specifies the Resolver operations and resources that you want to allow another AWS account to be able to use.
- `tag_resource`: Adds one or more tags to a specified resource.
- `untag_resource`: Removes one or more tags from a specified resource.
- `update_resolver_endpoint`: Updates the name of an inbound or an outbound resolver endpoint.
- `update_resolver_rule`: Updates settings for a specified resolver rule.

**Examples**

```r
csvc <- route53resolver()
csvc$associate_resolver_endpoint_ip_address(
  Foo = 123
)
```

---

**servicediscovery**

AWS Cloud Map

**Description**

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.
servicediscovery

Usage

servicediscovery(config = list())

Arguments

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

Service syntax

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 specified 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: - For public and private DNS namespaces, it defines the DNS names that can be used to access instances - For HTTP namespaces, it defines the URL pattern to which instances can be accessed
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 that are associated with a specified service
get_namespace Gets information about a 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
get_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
register_instance Creates or updates one or more records and, optionally, creates a health check based on the settings in a specified service
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: - Add or delete DnsRecords configurations - Add or remove namespaces - Add or remove instances
Examples

```r
svc <- servicediscovery()
svc$create_http_namespace(
  Foo = 123
)
```
Index

accept_direct_connect_gateway_association_proposal, 15
add_listener_certificates, 19
add_tags, 17, 19
allocate_connection_on_interconnect, 15
allocate_hosted_connection, 15
allocate_private_virtual_interface, 15
allocate_public_virtual_interface, 15
allocate_transit_virtual_interface, 15
apigateway, 2
apigatewaymanagementapi, 6
apigatewayv2, 7
apply_security_groups_to_load_balancer, 17
appmesh, 9
associate_connection_with_lag, 15
associate_hosted_connection, 15
associate_resolver_endpoint_ip_address, 28
associate_resolver_rule, 28
associate_virtual_interface, 15
associate_vpc_with_hosted_zone, 23
attach_load_balancer_to_subnets, 17
change_resource_record_sets, 23
change_tags_for_resource, 23
check_domain_availability, 26
check_domain_transferability, 26
cloudfront, 11
configure_health_check, 17
confirm_connection, 15
confirm_private_virtual_interface, 15
confirm_public_virtual_interface, 15
confirm_transit_virtual_interface, 15
create_accelerator, 22
create_api, 7
create_api_key, 3
create_api_mapping, 7
create_authorizer, 3, 7
create_base_path_mapping, 3
create_bgp_peer, 15
create_cloud_front_origin_access_identity, 12
create_connection, 15
create_deployment, 3, 7
create_direct_connect_gateway, 15
create_direct_connect_gateway_association, 15
create_direct_connect_gateway_association_proposal, 15
create_distribution, 12
create_distribution_with_tags, 12
create_documentation_part, 3
create_documentation_version, 3
create_domain_name, 3, 7
create_endpoint_group, 22
create_field_level_encryption_config, 12
create_field_level_encryption_profile, 12
create_health_check, 23
create_hosted_zone, 23
create_http_namespace, 29
create_integration, 7
create_integration_response, 8
create_interconnect, 15
create_invalidation, 12
create_lag, 15
create_lb_cookie_stickiness_policy, 17
create_listener, 19, 22
create_load_balancer, 17, 19
create_load_balancer_listeners, 17
create_load_balancer_policy, 17
create_mesh, 10
create_model, 3, 8
create_private_dns_namespace, 29
create_private_virtual_interface, 15
create_public_dns_namespace, 29
create_public_key, 12
create_public_virtual_interface, 15
create_query_logging_config, 23
create_request_validator, 3
create_resolver_endpoint, 28
create_resolver_rule, 28
create_resource, 3
create_rest_api, 3
create_reusable_delegation_set, 24
create_route, 8, 10
create_route_response, 8
create_rule, 19
create_service, 29
create_stage, 3, 8
create_streaming_distribution, 12
create_streaming_distribution_with_tags, 12
create_target_group, 19
create_traffic_policy, 24
create_traffic_policy_instance, 24
create_traffic_policy_version, 24
create_transit_virtual_interface, 15
create_usage_plan, 3
create_usage_plan_key, 3
create_virtual_node, 10
create_virtual_router, 10
create_virtual_service, 10
create_vpc_association_authorization, 24
create_vpc_link, 3
delete_accelerator, 22
delete_api, 8
delete_api_key, 3
delete_api_mapping, 8
delete_authorizer, 3, 8
delete_base_path_mapping, 3
delete_bgp_peer, 15
delete_client_certificate, 3
delete_cloud_front_origin_access_identity, 12
create_connection, 6, 15
delete_cors_configuration, 8
delete_deployment, 3, 8
delete_direct_connect_gateway, 15
delete_direct_connect_gateway_association, 15
delete_direct_connect_gateway_association_proposal, 15
delete_distribution, 12
delete_documentation_part, 3
delete_documentation_version, 3
delete_domain_name, 3, 8
delete_endpoint_group, 22
delete_field_level_encryption_config, 12
delete_field_level_encryption_profile, 12
delete_gateway_response, 3
delete_health_check, 24
delete_hosted_zone, 24
delete_integration, 3, 8
delete_integration_response, 3, 8
delete_interconnect, 15
delete_lag, 15
delete_listener, 19, 22
delete_load_balancer, 17, 19
delete_load_balancer_listeners, 17
delete_load_balancer_policy, 17
delete_mesh, 10
delete_method, 3
delete_method_response, 3
delete_model, 3, 8
delete_namespace, 29
delete_public_key, 12
delete_query_logging_config, 24
delete_request_validator, 3
delete_resolver_endpoint, 28
delete_resolver_rule, 28
delete_resource, 4
delete_rest_api, 4
delete_reusable_delegation_set, 24
delete_route, 8, 10
delete_route_response, 8
delete_route_settings, 8
delete_rule, 19
delete_service, 29
delete_stage, 4, 8
delete_streaming_distribution, 12
delete_tags_for_domain, 26
delete_target_group, 19
delete_traffic_policy, 24
INDEX

- delete_usage_plan, 4
- delete_usage_plan_key, 4
- delete_virtual_interface, 15
- delete_virtual_node, 10
- delete_virtual_router, 10
- delete_virtual_service, 10
- delete_vpc_association_authorization, 24
- delete_vpc_link, 4
- deregister_instance, 29
- deregister_instances_from_load_balancer, 17
- deregister_targets, 19
- describe_accelerator, 22
- describe_accelerator_attributes, 22
- describe_account_limits, 17, 19
- describe_connection_loa, 15
- describe_connections, 15
- describe_connections_on_interconnect, 15
- describe_direct_connect_gateway_association_proposals, 15
- describe_direct_connect_gateway_associations, 15
- describe_direct_connect_gateway_attachments, 15
- describe_direct_connect_gateways, 15
- describe_endpoint_group, 22
- describe_hosted_connections, 15
- describe_instance_health, 17
- describe_interconnect_loa, 15
- describe_interconnects, 15
- describe_lags, 15
- describe_listener, 22
- describe_listener_certificates, 19
- describe_listeners, 19
- describe_loa, 15
- describe_load_balancer_attributes, 17, 19
- describe_load_balancer_policies, 17
- describe_load_balancer_policy_types, 17
- describe_load_balancers, 17, 19
- describe_locations, 15
- describe_mesh, 10
- describe_route, 10
- describe_rules, 19
- describe_ssl_policies, 19
- describe_tags, 15, 17, 19
- describe_target_group_attributes, 19
- describe_target_groups, 19
- describe_target_health, 19
- describe_virtual_gateways, 15
- describe_virtual_interfaces, 15
- describe_virtual_node, 10
- describe_virtual_router, 10
- describe_virtual_service, 10
- detach_load_balancer_from_subnets, 17
- directconnect, 13
- disable_availability_zones_for_load_balancer, 17
- disable_domain_auto_renew, 26
- disable_domain_transfer_lock, 26
- disassociate_connection_from_lag, 15
- disassociate_resolver_endpoint_ip_address, 28
- disassociate_resolver_rule, 28
- disassociate_vpc_from_hosted_zone, 24
- disconnect_instances, 29
- elb, 16
- elbv2, 18
- enable_availability_zones_for_load_balancer, 17
- enable_domain_auto_renew, 26
- enable_domain_transfer_lock, 26
- flush_stage_authorizers_cache, 4
- flush_stage_cache, 4
- generate_client_certificate, 4
- get_account, 4
- get_account_limit, 24
- get_api, 8
- get_api_key, 4
- get_api_keys, 4
- get_api_mapping, 8
- get_api_mappings, 8
- get_apis, 8
- get_authorizer, 4, 8
- get_authorizers, 4, 8
- get_base_path_mapping, 4
- get_base_path_mappings, 4
- get_change, 24
- get_checker_ip_ranges, 24
- get_client_certificate, 4
- get_client_certificates, 4
INDEX

get_cloud_front_origin_access_identity, 12
get_cloud_front_origin_access_identity_config, 12
get_connection, 6
get_contact_reachability_status, 26
get_deployment, 4, 8
get_deployments, 4, 8
get_distribution, 12
get_distribution_config, 12
get_documentation_part, 4
get_documentation_parts, 4
get_documentation_version, 4
get_documentation_versions, 4
get_domain_detail, 26
get_domain_name, 4, 8
get_domain_names, 4, 8
get_domain_suggestions, 26
get_export, 4
get_field_level_encryption, 12
get_field_level_encryption_config, 12
get_field_level_encryption_profile, 12
get_field_level_encryption_profile_config, 12
get_gateway_response, 4
get_gateway_responses, 4
get_geo_location, 24
get_health_check, 24
get_health_check_count, 24
get_health_check_last_failure_reason, 24
get_health_check_status, 24
get_hosted_zone, 24
get_hosted_zone_count, 24
get_hosted_zone_limit, 24
get_instance, 29
get_instances_health_status, 29
get_integration, 4, 8
get_integration_response, 4, 8
get_integration_responses, 8
get_integrations, 8
get_invalidation, 12
get_method, 4
get_method_response, 4
get_model, 4, 8
get_model_template, 4, 8
get_models, 4, 8
get_namespace, 29
get_operation, 29
get_operation_detail, 26
get_public_key, 12
get_public_key_config, 12
get_query_logging_config, 24
get_request_validator, 4
get_request_validators, 4
get_resolver_endpoint, 28
get_resolver_rule, 28
get_resolver_rule_association, 28
get_resolver_rule_policy, 28
get_resource, 4
get_resources, 4
get_rest_api, 4
get_rest_apis, 4
get_reusable_delegation_set, 24
get_reusable_delegation_set_limit, 24
get_route, 8
get_route_response, 8
get_route_responses, 8
get_routes, 8
get_sdk, 4
get_sdk_type, 4
get_sdk_types, 4
get_service, 29
get_stage, 4, 8
get_stages, 4, 8
get_streaming_distribution, 12
get_streaming_distribution_config, 12
get_tags, 4, 8
get_traffic_policy, 24
get_traffic_policy_instance, 24
get_traffic_policy_instance_count, 24
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
globalaccelerator, 20
import_api, 8
import_api_keys, 5
import_documentation_parts, 5
import_rest_api, 5
list_accelerators, 22
update_accelerator_attributes, 22
update_account, 5
update_api, 8
update_api_key, 5
update_api_mapping, 8
update_authorizer, 5, 9
update_base_path_mapping, 5
update_client_certificate, 5
update_cloud_front_origin_access_identity, 12
update_deployment, 5, 9
update_direct_connect_gateway_association, 16
update_distribution, 12
update_documentation_part, 5
update_documentation_version, 5
update_domain_contact, 26
update_domain_contact_privacy, 26
update_domain_name, 5, 9
update_domain_nameservers, 26
update_endpoint_group, 22
update_field_level_encryption_config, 12
update_field_level_encryption_profile, 12
update_gateway_response, 5
update_health_check, 24
update_hosted_zone_comment, 24
update_instance_custom_health_status, 29
update_integration, 5, 9
update_integration_response, 5, 9
update_lag, 16
update_listener, 22
update_mesh, 10
update_method, 5
update_method_response, 5
update_model, 5, 9
update_public_key, 12
update_request_validator, 5
update_resolver_endpoint, 28
update_resolver_rule, 28
update_resource, 5
update_rest_api, 5
update_route, 9, 10
update_route_response, 9
update_service, 29
update_stage, 5, 9
update_streaming_distribution, 12
update_tags_for_domain, 26
update_traffic_policy_comment, 25
update_traffic_policy_instance, 25
update_usage, 5
update_usage_plan, 5
update_virtual_interface_attributes, 16
update_virtual_node, 10
update_virtual_router, 10
update_virtual_service, 10
update_vpc_link, 5
view_billing, 26