Package 'paws.networking'

July 23, 2025

Title 'Amazon Web Services' Networking & Content Delivery Services

Version 0.9.0

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, https://paws-r.r-universe.dev/paws.networking

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

Imports paws.common (>= 0.8.0)

Suggests testthat

Encoding UTF-8

RoxygenNote 7.3.2

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' 'appfabric_service.R' 'appfabric_interfaces.R' 'appfabric_operations.R' 'appmesh_service.R' 'appmesh_interfaces.R' 'appmesh_operations.R' 'arczonalshift_service.R' 'arczonalshift_interfaces.R' 'arczonalshift_operations.R' 'backupgateway_service.R' 'backupgateway_interfaces.R' 'backupgateway_operations.R' 'cloudfront_service.R' 'cloudfront_interfaces.R' 'cloudfront_operations.R' 'cloudfrontkeyvaluestore_service.R' 'cloudfrontkeyvaluestore_interfaces.R' 'cloudfrontkeyvaluestore_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' 'networkfirewall_service.R'
'networkfirewall_interfaces.R' 'networkfirewall_operations.R'
'networkmanager_service.R' 'networkmanager_interfaces.R'
'networkmanager_operations.R' 'reexports_paws.common.R'
'route53_service.R' 'route53_interfaces.R'
'route53_operations.R' 'route53domains_service.R'
'route53domains_interfaces.R' 'route53domains_operations.R'
'route53profiles_service.R' 'route53profiles_interfaces.R'
'route53profiles_operations.R'
'route53recoverycluster_service.R'
'route53recoverycluster_interfaces.R'
'route53recoverycluster_operations.R'
'route53recoverycontrolconfig_service.R'
'route53recoverycontrolconfig_interfaces.R'
'route53recoverycontrolconfig_operations.R'
'route53recoveryreadiness_service.R'
'route53recoveryreadiness_interfaces.R'
'route53recoveryreadiness_operations.R'
'route53resolver_service.R' 'route53resolver_interfaces.R'
'route53resolver_operations.R' 'servicediscovery_service.R'
'servicediscovery_interfaces.R' 'servicediscovery_operations.R'
'telconetworkbuilder_service.R'
'telconetworkbuilder_interfaces.R'
'telconetworkbuilder_operations.R' 'vpclattice_service.R'
'vpclattice_interfaces.R' 'vpclattice_operations.R'

NeedsCompilation no

Author David Kretch [aut],
Adam Banker [aut],
Dyfan Jones [cre],
Amazon.com, Inc. [cph]

Maintainer Dyfan Jones <dyfan.r.jones@gmail.com>

Repository CRAN

Date/Publication 2025-03-14 16:30:02 UTC

Contents

`apigateway` *Amazon API Gateway*

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 Lambda, Amazon EC2, or other publicly addressable web services that are hosted outside of AWS.

Usage

```
apigateway(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token

  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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

```
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  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 th
create_documentation_part	Creates a documentation part
create_documentation_version	Creates a documentation version
create_domain_name	Creates a new domain name
create_domain_name_access_association	Creates a domain name access association resource between an access association
create_model	Adds a new Model resource to an existing RestApi resource
create_request_validator	Creates a RequestValidator 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 AI
create_usage_plan	Creates a usage plan with the throttle and quota limits, as well as the associated A
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 asynchro
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	Deletes a documentation part
delete_documentation_version	Deletes a documentation version
delete_domain_name	Deletes the DomainName resource
delete_domain_name_access_association	Deletes the DomainNameAccessAssociation resource
delete_gateway_response	Clears any customization of a GatewayResponse of a specified response type on th
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
delete_resource	Deletes a Resource resource
delete_rest_api	Deletes the specified API
delete_stage	Deletes a Stage resource
delete_usage_plan	Deletes a usage plan of a given plan Id
delete_usage_plan_key	Deletes a usage plan key and remove the underlying API key from the associated u
delete_vpc_link	Deletes an existing VpcLink of a specified identifier
flush_stage_authorizers_cache	Flushes all authorizer cache entries on a stage
flush_stage_cache	Flushes a stage's cache
generate_client_certificate	Generates a ClientCertificate resource
get_account	Gets information about the current Account resource
get_api_key	Gets information about the current ApiKey resource
get_api_keys	Gets information about the current ApiKeys resource
get_authorizer	Describe an existing Authorizer resource
get_authorizers	Describe an existing Authorizers resource
get_base_path_mapping	Describe a BasePathMapping resource
get_base_path_mappings	Represents a collection of BasePathMapping resources
get_client_certificate	Gets information about the current ClientCertificate resource
get_client_certificates	Gets a collection of ClientCertificate resources
get_deployment	Gets information about a Deployment resource
get_deployments	Gets information about a Deployments collection
get_documentation_part	Gets a documentation part
get_documentation_parts	Gets documentation parts
get_documentation_version	Gets a documentation version
get_documentation_versions	Gets documentation versions
get_domain_name	Represents a domain name that is contained in a simpler, more intuitive URL that
get_domain_name_access_associations	Represents a collection on DomainNameAccessAssociations resources
get_domain_names	Represents a collection of DomainName resources
get_export	Exports a deployed version of a RestApi in a specified format
get_gateway_response	Gets a GatewayResponse of a specified response type on the given RestApi
get_gateway_responses	Gets the GatewayResponses collection on the given RestApi
get_integration	Get the integration settings
get_integration_response	Represents a get integration response
get_method	Describe an existing Method resource
get_method_response	Describes a MethodResponse resource
get_model	Describes an existing model defined for a RestApi resource
get_models	Describes existing Models defined for a RestApi resource
get_model_template	Generates a sample mapping template that can be used to transform a payload into
get_request_validator	Gets a RequestValidator of a given RestApi
get_request_validators	Gets the RequestValidators collection of a given RestApi
get_resource	Lists information about a resource
get_resources	Lists information about a collection of Resource resources
get_rest_api	Lists the RestApi resource in the collection
get_rest_apis	Lists the RestApis resources for your collection
get_sdk	Generates a client SDK for a RestApi and Stage
get_sdk_type	Gets an SDK type

get_sdk_types	Gets SDK types
get_stage	Gets information about a Stage resource
get_stages	Gets information about one or more Stage resources
get_tags	Gets the Tags collection for a given resource
get_usage	Gets the usage data of a usage plan in a specified time interval
get_usage_plan	Gets a usage plan of a given plan identifier
get_usage_plan_key	Gets a usage plan key of a given key identifier
get_usage_plan_keys	Gets all the usage plan keys representing the API keys added to a specified usage p
get_usage_plans	Gets all the usage plans of the caller's account
get_vpc_link	Gets a specified VPC link under the caller's account in a region
get_vpc_links	Gets the VpcLinks collection under the caller's account in a selected region
import_api_keys	Import API keys from an external source, such as a CSV-formatted file
import_documentation_parts	Imports documentation parts
import_rest_api	A feature of the API Gateway control service for creating a new API from an exte
put_gateway_response	Creates a customization of a GatewayResponse of a specified response type and st
put_integration	Sets up a method's integration
put_integration_response	Represents a put integration
put_method	Add a method to an existing Resource resource
put_method_response	Adds a MethodResponse to an existing Method resource
put_rest_api	A feature of the API Gateway control service for updating an existing API with an
reject_domain_name_access_association	Rejects a domain name access association with a private custom domain name
tag_resource	Adds or updates a tag on a given resource
test_invoke_authorizer	Simulate the execution of an Authorizer in your RestApi with headers, parameters
test_invoke_method	Simulate the invocation of a Method in your RestApi with headers, parameters, an
untag_resource	Removes a tag from a given resource
update_account	Changes information about the current Account resource
update_api_key	Changes information about an ApiKey resource
update_authorizer	Updates an existing Authorizer resource
update_base_path_mapping	Changes information about the BasePathMapping resource
update_client_certificate	Changes information about an ClientCertificate resource
update_deployment	Changes information about a Deployment resource
update_documentation_part	Updates a documentation part
update_documentation_version	Updates a documentation version
update_domain_name	Changes information about the DomainName resource
update_gateway_response	Updates a GatewayResponse of a specified response type on the given RestApi
update_integration	Represents an update integration
update_integration_response	Represents an update integration response
update_method	Updates an existing Method resource
update_method_response	Updates an existing MethodResponse resource
update_model	Changes information about a model
update_request_validator	Updates a RequestValidator of a given RestApi
update_resource	Changes information about a Resource resource
update_rest_api	Changes information about the specified API
update_stage	Changes information about a Stage resource
update_usage	Grants a temporary extension to the remaining quota of a usage plan associated wi
update_usage_plan	Updates a usage plan of a given plan Id
update_vpc_link	Updates an existing VpcLink of a specified identifier

Examples

```
## 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

```
apigatewaymanagementapi(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.

- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- apigatewaymanagementapi(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
```

```
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  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(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- apigatewayv2(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
```

```
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string",
  close_connection = "logical",
  timeout = "numeric",
  s3_force_path_style = "logical",
  sts_regional_endpoint = "string"
),
credentials = list(
  creds = list(
    access_key_id = "string",
    secret_access_key = "string",
    session_token = "string"
  ),
  profile = "string",
  anonymous = "logical"
),
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	Export 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()
svc$create_api(
  Foo = 123
)

## End(Not run)
```

`appfabric` *AppFabric*

Description

Amazon Web Services AppFabric quickly connects software as a service (SaaS) applications across your organization. This allows IT and security teams to easily manage and secure applications using a standard schema, and employees can complete everyday tasks faster using generative artificial intelligence (AI). You can use these APIs to complete AppFabric tasks, such as setting up audit log ingestions or viewing user access. For more information about AppFabric, including the required permissions to use the service, see the Amazon Web Services AppFabric Administration Guide. For more information about using the Command Line Interface (CLI) to manage your AppFabric resources, see the AppFabric section of the CLI Reference.

Usage

```
appfabric(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID

    * **secret_access_key**: AWS secret access key
    * **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- appfabric(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
```

```
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

batch_get_user_access_tasks	Gets user access details in a batch request
connect_app_authorization	Establishes a connection between Amazon Web Services AppFabric and an application, which
create_app_authorization	Creates an app authorization within an app bundle, which allows AppFabric to connect to an a
create_app_bundle	Creates an app bundle to collect data from an application using AppFabric
create_ingestion	Creates a data ingestion for an application
create_ingestion_destination	Creates an ingestion destination, which specifies how an application's ingested data is process
delete_app_authorization	Deletes an app authorization
delete_app_bundle	Deletes an app bundle
delete_ingestion	Deletes an ingestion
delete_ingestion_destination	Deletes an ingestion destination
get_app_authorization	Returns information about an app authorization
get_app_bundle	Returns information about an app bundle
get_ingestion	Returns information about an ingestion
get_ingestion_destination	Returns information about an ingestion destination
list_app_authorizations	Returns a list of all app authorizations configured for an app bundle
list_app_bundles	Returns a list of app bundles
list_ingestion_destinations	Returns a list of all ingestion destinations configured for an ingestion
list_ingestions	Returns a list of all ingestions configured for an app bundle
list_tags_for_resource	Returns a list of tags for a resource
start_ingestion	Starts (enables) an ingestion, which collects data from an application
start_user_access_tasks	Starts the tasks to search user access status for a specific email address
stop_ingestion	Stops (disables) an ingestion
tag_resource	Assigns one or more tags (key-value pairs) to the specified resource
untag_resource	Removes a tag or tags from a resource
update_app_authorization	Updates an app authorization within an app bundle, which allows AppFabric to connect to an
update_ingestion_destination	Updates an ingestion destination, which specifies how an application's ingested data is process

Examples

```
## Not run:
svc <- appfabric()
svc$batch_get_user_access_tasks(
  Foo = 123
)

## End(Not run)
```

`appmesh` *AWS App Mesh*

Description

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 Amazon Web Services Fargate, Amazon ECS, Amazon EKS, Kubernetes on Amazon Web Services, 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(), credentials = list(), endpoint = NULL, region = NULL)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.

- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
```

```
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  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 mes
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

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

## End(Not run)
```

`arczonalshift` *AWS ARC - Zonal Shift*

Description

Welcome to the API Reference Guide for zonal shift and zonal autoshift in Amazon Route 53 Application Recovery Controller (Route 53 ARC).

You can start a zonal shift to move traffic for a load balancer resource away from an Availability Zone to help your application recover quickly from an impairment in an Availability Zone. For example, you can recover your application from a developer's bad code deployment or from an Amazon Web Services infrastructure failure in a single Availability Zone.

You can also configure zonal autoshift for supported load balancer resources. Zonal autoshift is a capability in Route 53 ARC where you authorize Amazon Web Services to shift away application resource traffic from an Availability Zone during events, on your behalf, to help reduce your time to recovery. Amazon Web Services starts an autoshift when internal telemetry indicates that there is an Availability Zone impairment that could potentially impact customers.

To help make sure that zonal autoshift is safe for your application, you must also configure practice runs when you enable zonal autoshift for a resource. Practice runs start weekly zonal shifts for a resource, to shift traffic for the resource away from an Availability Zone. Practice runs help you to make sure, on a regular basis, that you have enough capacity in all the Availability Zones in an Amazon Web Services Region for your application to continue to operate normally when traffic for a resource is shifted away from one Availability Zone.

Before you configure practice runs or enable zonal autoshift, we strongly recommend that you prescale your application resource capacity in all Availability Zones in the Region where your application resources are deployed. You should not rely on scaling on demand when an autoshift or practice run starts. Zonal autoshift, including practice runs, works independently, and does not wait for auto scaling actions to complete. Relying on auto scaling, instead of pre-scaling, can result in loss of availability.

If you use auto scaling to handle regular cycles of traffic, we strongly recommend that you configure the minimum capacity of your auto scaling to continue operating normally with the loss of an Availability Zone.

Be aware that Route 53 ARC does not inspect the health of individual resources. Amazon Web Services only starts an autoshift when Amazon Web Services telemetry detects that there is an Availability Zone impairment that could potentially impact customers. In some cases, resources might be shifted away that are not experiencing impact.

For more information about using zonal shift and zonal autoshift, see the Amazon Route 53 Application Recovery Controller Developer Guide.

Usage

```
arczonalshift(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e
html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token

- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- arczonalshift(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

cancel_zonal_shift	Cancel a zonal shift in Amazon Route 53 Application Recovery Controller
create_practice_run_configuration	A practice run configuration for zonal autoshift is required when you enable z
delete_practice_run_configuration	Deletes the practice run configuration for a resource
get_autoshift_observer_notification_status	Returns the status of autoshift observer notification
get_managed_resource	Get information about a resource that's been registered for zonal shifts with A
list_autoshifts	Returns a list of autoshifts for an Amazon Web Services Region
list_managed_resources	Lists all the resources in your Amazon Web Services account in this Amazon
list_zonal_shifts	Lists all active and completed zonal shifts in Amazon Route 53 Application R
start_zonal_shift	You start a zonal shift to temporarily move load balancer traffic away from an
update_autoshift_observer_notification_status	Update the status of autoshift observer notification
update_practice_run_configuration	Update a practice run configuration to change one or more of the following: a
update_zonal_autoshift_configuration	The zonal autoshift configuration for a resource includes the practice run con
update_zonal_shift	Update an active zonal shift in Amazon Route 53 Application Recovery Cont

Examples

```
## Not run:
svc <- arczonalshift()
svc$cancel_zonal_shift(
  Foo = 123
)

## End(Not run)
```

`backupgateway` *AWS Backup Gateway*

Description

Backup gateway

Backup gateway connects Backup to your hypervisor, so you can create, store, and restore backups of your virtual machines (VMs) anywhere, whether on-premises or in the VMware Cloud (VMC) on Amazon Web Services.

Add on-premises resources by connecting to a hypervisor through a gateway. Backup will automatically discover the resources in your hypervisor.

Use Backup to assign virtual or on-premises resources to a backup plan, or run on-demand backups. Once you have backed up your resources, you can view them and restore them like any resource supported by Backup.

To download the Amazon Web Services software to get started, navigate to the Backup console, choose **Gateways**, then choose **Create gateway**.

Usage

```
backupgateway(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e
html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- backupgateway(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

associate_gateway_to_server	Associates a backup gateway with your server
create_gateway	Creates a backup gateway
delete_gateway	Deletes a backup gateway
delete_hypervisor	Deletes a hypervisor
disassociate_gateway_from_server	Disassociates a backup gateway from the specified server
get_bandwidth_rate_limit_schedule	Retrieves the bandwidth rate limit schedule for a specified gateway
get_gateway	By providing the ARN (Amazon Resource Name), this API returns the gateway
get_hypervisor	This action requests information about the specified hypervisor to which the gateway
get_hypervisor_property_mappings	This action retrieves the property mappings for the specified hypervisor
get_virtual_machine	By providing the ARN (Amazon Resource Name), this API returns the virtual machi
import_hypervisor_configuration	Connect to a hypervisor by importing its configuration
list_gateways	Lists backup gateways owned by an Amazon Web Services account in an Amazon W
list_hypervisors	Lists your hypervisors

list_tags_for_resource	Lists the tags applied to the resource identified by its Amazon Resource Name (ARN
list_virtual_machines	Lists your virtual machines
put_bandwidth_rate_limit_schedule	This action sets the bandwidth rate limit schedule for a specified gateway
put_hypervisor_property_mappings	This action sets the property mappings for the specified hypervisor
put_maintenance_start_time	Set the maintenance start time for a gateway
start_virtual_machines_metadata_sync	This action sends a request to sync metadata across the specified virtual machines
tag_resource	Tag the resource
test_hypervisor_configuration	Tests your hypervisor configuration to validate that backup gateway can connect witl
untag_resource	Removes tags from the resource
update_gateway_information	Updates a gateway's name
update_gateway_software_now	Updates the gateway virtual machine (VM) software
update_hypervisor	Updates a hypervisor metadata, including its host, username, and password

Examples

```
## Not run:
svc <- backupgateway()
svc$associate_gateway_to_server(
  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

```
cloudfront(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- cloudfront(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
```

```
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string",
  close_connection = "logical",
  timeout = "numeric",
  s3_force_path_style = "logical",
  sts_regional_endpoint = "string"
),
credentials = list(
  creds = list(
    access_key_id = "string",
    secret_access_key = "string",
    session_token = "string"
  ),
  profile = "string",
  anonymous = "logical"
),
endpoint = "string",
region = "string"
)
```

Operations

associate_alias	Associates an alias (also known as a CNAME or an alternate domain nam
copy_distribution	Creates a staging distribution using the configuration of the provided prin
create_anycast_ip_list	Creates an Anycast static IP list
create_cache_policy	Creates a cache policy
create_cloud_front_origin_access_identity	Creates a new origin access identity
create_continuous_deployment_policy	Creates a continuous deployment policy that distributes traffic for a custo
create_distribution	Creates a CloudFront distribution
create_distribution_with_tags	Create a new distribution with tags
create_field_level_encryption_config	Create a new field-level encryption configuration
create_field_level_encryption_profile	Create a field-level encryption profile
create_function	Creates a CloudFront function
create_invalidation	Create a new invalidation
create_key_group	Creates a key group that you can use with CloudFront signed URLs and s
create_key_value_store	Specifies the key value store resource to add to your account
create_monitoring_subscription	Enables additional CloudWatch metrics for the specified CloudFront distr
create_origin_access_control	Creates a new origin access control in CloudFront
create_origin_request_policy	Creates an origin request policy
create_public_key	Uploads a public key to CloudFront that you can use with signed URLs a
create_realtime_log_config	Creates a real-time log configuration
create_response_headers_policy	Creates a response headers policy

create_streaming_distribution	This API is deprecated
create_streaming_distribution_with_tags	This API is deprecated
create_vpc_origin	Create an Amazon CloudFront VPC origin
delete_anycast_ip_list	Deletes an Anycast static IP list
delete_cache_policy	Deletes a cache policy
delete_cloud_front_origin_access_identity	Delete an origin access identity
delete_continuous_deployment_policy	Deletes a continuous deployment policy
delete_distribution	Delete a distribution
delete_field_level_encryption_config	Remove a field-level encryption configuration
delete_field_level_encryption_profile	Remove a field-level encryption profile
delete_function	Deletes a CloudFront function
delete_key_group	Deletes a key group
delete_key_value_store	Specifies the key value store to delete
delete_monitoring_subscription	Disables additional CloudWatch metrics for the specified CloudFront dist
delete_origin_access_control	Deletes a CloudFront origin access control
delete_origin_request_policy	Deletes an origin request policy
delete_public_key	Remove a public key you previously added to CloudFront
delete_realtime_log_config	Deletes a real-time log configuration
delete_response_headers_policy	Deletes a response headers policy
delete_streaming_distribution	Delete a streaming distribution
delete_vpc_origin	Delete an Amazon CloudFront VPC origin
describe_function	Gets configuration information and metadata about a CloudFront functior
describe_key_value_store	Specifies the key value store and its configuration
get_anycast_ip_list	Gets an Anycast static IP list
get_cache_policy	Gets a cache policy, including the following metadata:
get_cache_policy_config	Gets a cache policy configuration
get_cloud_front_origin_access_identity	Get the information about an origin access identity
get_cloud_front_origin_access_identity_config	Get the configuration information about an origin access identity
get_continuous_deployment_policy	Gets a continuous deployment policy, including metadata (the policy's ide
get_continuous_deployment_policy_config	Gets configuration information about a continuous deployment policy
get_distribution	Get the information about a distribution
get_distribution_config	Get the configuration information about a distribution
get_field_level_encryption	Get the field-level encryption configuration information
get_field_level_encryption_config	Get the field-level encryption configuration information
get_field_level_encryption_profile	Get the field-level encryption profile information
get_field_level_encryption_profile_config	Get the field-level encryption profile configuration information
get_function	Gets the code of a CloudFront function
get_invalidation	Get the information about an invalidation
get_key_group	Gets a key group, including the date and time when the key group was las
get_key_group_config	Gets a key group configuration
get_monitoring_subscription	Gets information about whether additional CloudWatch metrics are enabl
get_origin_access_control	Gets a CloudFront origin access control, including its unique identifier
get_origin_access_control_config	Gets a CloudFront origin access control configuration
get_origin_request_policy	Gets an origin request policy, including the following metadata:
get_origin_request_policy_config	Gets an origin request policy configuration
get_public_key	Gets a public key
get_public_key_config	Gets a public key configuration
get_realtime_log_config	Gets a real-time log configuration

get_response_headers_policy	Gets a response headers policy, including metadata (the policy's identifier
get_response_headers_policy_config	Gets a response headers policy configuration
get_streaming_distribution	Gets information about a specified RTMP distribution, including the distr
get_streaming_distribution_config	Get the configuration information about a streaming distribution
get_vpc_origin	Get the details of an Amazon CloudFront VPC origin
list_anycast_ip_lists	Lists your Anycast static IP lists
list_cache_policies	Gets a list of cache policies
list_cloud_front_origin_access_identities	Lists origin access identities
list_conflicting_aliases	Gets a list of aliases (also called CNAMEs or alternate domain names) th
list_continuous_deployment_policies	Gets a list of the continuous deployment policies in your Amazon Web Se
list_distributions	List CloudFront distributions
list_distributions_by_anycast_ip_list_id	Lists the distributions in your account that are associated with the specifie
list_distributions_by_cache_policy_id	Gets a list of distribution IDs for distributions that have a cache behavior
list_distributions_by_key_group	Gets a list of distribution IDs for distributions that have a cache behavior
list_distributions_by_origin_request_policy_id	Gets a list of distribution IDs for distributions that have a cache behavior
list_distributions_by_realtime_log_config	Gets a list of distributions that have a cache behavior that's associated wit
list_distributions_by_response_headers_policy_id	Gets a list of distribution IDs for distributions that have a cache behavior
list_distributions_by_vpc_origin_id	List CloudFront distributions by their VPC origin ID
list_distributions_by_web_acl_id	List the distributions that are associated with a specified WAF web ACL
list_field_level_encryption_configs	List all field-level encryption configurations that have been created in Clo
list_field_level_encryption_profiles	Request a list of field-level encryption profiles that have been created in C
list_functions	Gets a list of all CloudFront functions in your Amazon Web Services acc
list_invalidations	Lists invalidation batches
list_key_groups	Gets a list of key groups
list_key_value_stores	Specifies the key value stores to list
list_origin_access_controls	Gets the list of CloudFront origin access controls (OACs) in this Amazon
list_origin_request_policies	Gets a list of origin request policies
list_public_keys	List all public keys that have been added to CloudFront for this account
list_realtime_log_configs	Gets a list of real-time log configurations
list_response_headers_policies	Gets a list of response headers policies
list_streaming_distributions	List streaming distributions
list_tags_for_resource	List tags for a CloudFront resource
list_vpc_origins	List the CloudFront VPC origins in your account
publish_function	Publishes a CloudFront function by copying the function code from the D
tag_resource	Add tags to a CloudFront resource
test_function	Tests a CloudFront function
untag_resource	Remove tags from a CloudFront resource
update_cache_policy	Updates a cache policy configuration
update_cloud_front_origin_access_identity	Update an origin access identity
update_continuous_deployment_policy	Updates a continuous deployment policy
update_distribution	Updates the configuration for a CloudFront distribution
update_distribution_with_staging_config	Copies the staging distribution's configuration to its corresponding prima
update_field_level_encryption_config	Update a field-level encryption configuration
update_field_level_encryption_profile	Update a field-level encryption profile
update_function	Updates a CloudFront function
update_key_group	Updates a key group
update_key_value_store	Specifies the key value store to update
update_origin_access_control	Updates a CloudFront origin access control

update_origin_request_policy	Updates an origin request policy configuration
update_public_key	Update public key information
update_realtime_log_config	Updates a real-time log configuration
update_response_headers_policy	Updates a response headers policy
update_streaming_distribution	Update a streaming distribution
update_vpc_origin	Update an Amazon CloudFront VPC origin in your account

Examples

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

## End(Not run)
```

---

cloudfrontkeyvaluestore

Amazon CloudFront KeyValueStore

Description

Amazon CloudFront KeyValueStore Service to View and Update Data in a KVS Resource

Usage

```
cloudfrontkeyvaluestore(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.

  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e
html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- cloudfrontkeyvaluestore(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
```

```
      sts_regional_endpoint = "string"
    ),
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string"
  )
```

Operations

delete_key	Deletes the key value pair specified by the key
describe_key_value_store	Returns metadata information about Key Value Store
get_key	Returns a key value pair
list_keys	Returns a list of key value pairs
put_key	Creates a new key value pair or replaces the value of an existing key
update_keys	Puts or Deletes multiple key value pairs in a single, all-or-nothing operation

Examples

```
## Not run:
svc <- cloudfrontkeyvaluestore()
svc$delete_key(
  Foo = 123
)

## End(Not run)
```

`directconnect` *AWS Direct Connect*

Description

Direct Connect links your internal network to an Direct Connect location over a standard Ethernet fiber-optic cable. One end of the cable is connected to your router, the other to an Direct Connect router. With this connection in place, you can create virtual interfaces directly to the Amazon Web Services 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 Amazon Web

Services Regions except the China (Beijing) and (China) Ningxia Regions. Amazon Web Services resources in the China Regions can only be accessed through locations associated with those Regions.

Usage

```
directconnect(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e
html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

accept_direct_connect_gateway_association_proposal	Accepts a proposal request to attach a virtual private gateway or tra
allocate_connection_on_interconnect	Deprecated
allocate_hosted_connection	Creates a hosted connection on the specified interconnect or a link
allocate_private_virtual_interface	Provisions a private virtual interface to be owned by the specified A
allocate_public_virtual_interface	Provisions a public virtual interface to be owned by the specified A
allocate_transit_virtual_interface	Provisions a transit virtual interface to be owned by the specified A
associate_connection_with_lag	Associates an existing connection with a link aggregation group (L
associate_hosted_connection	Associates a hosted connection and its virtual interfaces with a link

associate_mac_sec_key	Associates a MAC Security (MACsec) Connection Key Name (CK
associate_virtual_interface	Associates a virtual interface with a specified link aggregation gro
confirm_connection	Confirms the creation of the specified hosted connection on an inte
confirm_customer_agreement	The confirmation of the terms of agreement when creating the con
confirm_private_virtual_interface	Accepts ownership of a private virtual interface created by another
confirm_public_virtual_interface	Accepts ownership of a public virtual interface created by another
confirm_transit_virtual_interface	Accepts ownership of a transit virtual interface created by another
create_bgp_peer	Creates a BGP peer on the specified virtual interface
create_connection	Creates a connection between a customer network and a specific D
create_direct_connect_gateway	Creates a Direct Connect gateway, which is an intermediate object
create_direct_connect_gateway_association	Creates an association between a Direct Connect gateway and a vi
create_direct_connect_gateway_association_proposal	Creates a proposal to associate the specified virtual private gatewa
create_interconnect	Creates an interconnect between an Direct Connect Partner's netw
create_lag	Creates a link aggregation group (LAG) with the specified number
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 w
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 gatev
delete_direct_connect_gateway_association_proposal	Deletes the association proposal request between the specified Dir
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_customer_metadata	Get and view a list of customer agreements, along with their signe
describe_direct_connect_gateway_association_proposals	Describes one or more association proposals for connection betwe
describe_direct_connect_gateway_associations	Lists the associations between your Direct Connect gateways and v
describe_direct_connect_gateway_attachments	Lists the attachments between your Direct Connect gateways and v
describe_direct_connect_gateways	Lists all your Direct Connect gateways or only the specified Direct
describe_hosted_connections	Lists the hosted connections that have been provisioned on the spe
describe_interconnect_loa	Deprecated
describe_interconnects	Lists the interconnects owned by the Amazon Web Services accou
describe_lags	Describes all your link aggregation groups (LAG) or the specified
describe_loa	Gets the LOA-CFA for a connection, interconnect, or link aggrega
describe_locations	Lists the Direct Connect locations in the current Amazon Web Ser
describe_router_configuration	Details about the router
describe_tags	Describes the tags associated with the specified Direct Connect res
describe_virtual_gateways	Deprecated
describe_virtual_interfaces	Displays all virtual interfaces for an Amazon Web Services accour
disassociate_connection_from_lag	Disassociates a connection from a link aggregation group (LAG)
disassociate_mac_sec_key	Removes the association between a MAC Security (MACsec) secu
list_virtual_interface_test_history	Lists the virtual interface failover test history
start_bgp_failover_test	Starts the virtual interface failover test that verifies your configurat
stop_bgp_failover_test	Stops the virtual interface failover test

tag_resource	Adds the specified tags to the specified Direct Connect resource
untag_resource	Removes one or more tags from the specified Direct Connect resou
update_connection	Updates the Direct Connect dedicated connection configuration
update_direct_connect_gateway	Updates the name of a current Direct Connect gateway
update_direct_connect_gateway_association	Updates the specified attributes of the Direct Connect gateway ass
update_lag	Updates the attributes of the specified link aggregation group (LA
update_virtual_interface_attributes	Updates the specified attributes of the specified virtual private inte

Examples

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

## End(Not run)
```

---

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.

Usage

```
elb(config = list(), credentials = list(), endpoint = NULL, region = NULL)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string",
  close_connection = "logical",
  timeout = "numeric",
  s3_force_path_style = "logical",
  sts_regional_endpoint = "string"
),
credentials = list(
  creds = list(
    access_key_id = "string",
    secret_access_key = "string",
    session_token = "string"
  ),
  profile = "string",
  anonymous = "logical"
),
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
attach_load_balancer_to_subnets	Adds one or more subnets to the set of configured subnets for the specified
configure_health_check	Specifies the health check settings to use when evaluating the health state o
create_app_cookie_stickiness_policy	Generates a stickiness policy with sticky session lifetimes that follow that c
create_lb_cookie_stickiness_policy	Generates a stickiness policy with sticky session lifetimes controlled by the
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
describe_instance_health	Describes the state of the specified instances with respect to the specified lo
describe_load_balancer_attributes	Describes the attributes for the specified load balancer
describe_load_balancer_policies	Describes the specified policies
describe_load_balancer_policy_types	Describes the specified load balancer policy types or all load balancer polic
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 lo
disable_availability_zones_for_load_balancer	Removes the specified Availability Zones from the set of Availability Zones
enable_availability_zones_for_load_balancer	Adds the specified Availability Zones to the set of Availability Zones for th
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
set_load_balancer_policies_of_listener	Replaces the current set of policies for the specified load balancer port with

Examples

```
## Not run:
svc <- elb()
# This example adds two tags to the specified load balancer.
svc$add_tags(
  LoadBalancerNames = list(
    "my-load-balancer"
  ),
  Tags = list(
    list(
      Key = "project",
      Value = "lima"
    ),
    list(
      Key = "department",
      Value = "digital-media"
    )
  )
)

## End(Not run)
```

`elbv2` *Elastic Load Balancing*

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

```
elbv2(config = list(), credentials = list(), endpoint = NULL, region = NULL)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy `https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html`

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.

- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- elbv2(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

add_listener_certificates	Adds the specified SSL server certificate to the certificate list for the specified HTTP
add_tags	Adds the specified tags to the specified Elastic Load Balancing resource
add_trust_store_revocations	Adds the specified revocation file to the specified trust store

create_listener	Creates a listener for the specified Application Load Balancer, Network Load Balanc
create_load_balancer	Creates an Application Load Balancer, Network Load Balancer, or Gateway Load Ba
create_rule	Creates a rule for the specified listener
create_target_group	Creates a target group
create_trust_store	Creates a trust store
delete_listener	Deletes the specified listener
delete_load_balancer	Deletes the specified Application Load Balancer, Network Load Balancer, or Gatewa
delete_rule	Deletes the specified rule
delete_shared_trust_store_association	Deletes a shared trust store association
delete_target_group	Deletes the specified target group
delete_trust_store	Deletes a trust store
deregister_targets	Deregisters the specified targets from the specified target group
describe_account_limits	Describes the current Elastic Load Balancing resource limits for your Amazon Web S
describe_capacity_reservation	Describes the capacity reservation status for the specified load balancer
describe_listener_attributes	Describes the attributes for the specified listener
describe_listener_certificates	Describes the default certificate and the certificate list for the specified HTTPS or TL
describe_listeners	Describes the specified listeners or the listeners for the specified Application Load Ba
describe_load_balancer_attributes	Describes the attributes for the specified Application Load Balancer, Network Load B
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
describe_trust_store_associations	Describes all resources associated with the specified trust store
describe_trust_store_revocations	Describes the revocation files in use by the specified trust store or revocation files
describe_trust_stores	Describes all trust stores for the specified account
get_resource_policy	Retrieves the resource policy for a specified resource
get_trust_store_ca_certificates_bundle	Retrieves the ca certificate bundle
get_trust_store_revocation_content	Retrieves the specified revocation file
modify_capacity_reservation	Modifies the capacity reservation of the specified load balancer
modify_listener	Replaces the specified properties of the specified listener
modify_listener_attributes	Modifies the specified attributes of the specified listener
modify_load_balancer_attributes	Modifies the specified attributes of the specified Application Load Balancer, Network
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
modify_target_group_attributes	Modifies the specified attributes of the specified target group
modify_trust_store	Update the ca certificate bundle for the specified trust store
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
remove_tags	Removes the specified tags from the specified Elastic Load Balancing resources
remove_trust_store_revocations	Removes the specified revocation file from the specified trust store
set_ip_address_type	Sets the type of IP addresses used by the subnets of the specified 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 Balance
set_subnets	Enables the Availability Zones for the specified public subnets for the specified Appl

Examples

```
## 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

Global Accelerator

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

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 Amazon Web Services 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 Amazon Web Services Regions but you must specify the US West (Oregon) Region to create, update, or otherwise work

with accelerators. That is, for example, specify `--region us-west-2` on Amazon Web Services CLI commands.

By default, Global Accelerator provides you with static IP addresses that you associate with your accelerator. The static IP addresses are anycast from the Amazon Web Services edge network. For IPv4, Global Accelerator provides two static IPv4 addresses. For dual-stack, Global Accelerator provides a total of four addresses: two static IPv4 addresses and two static IPv6 addresses. With a standard accelerator for IPv4, instead of using the 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 (BYOIP).

For a standard accelerator, they distribute incoming application traffic across multiple endpoint resources in multiple Amazon Web Services 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 Amazon Web Services Region or multiple Amazon Web Services 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 Amazon Web Services 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 more information about understanding and using Global Accelerator, see the Global Accelerator Developer Guide.

Usage

```
globalaccelerator(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token

  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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

```
      timeout = "numeric",
      s3_force_path_style = "logical",
      sts_regional_endpoint = "string"
    ),
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string"
  )
```

Operations

add_custom_routing_endpoints	Associate a virtual private cloud (VPC) subnet endpoint with your cust
add_endpoints	Add endpoints to an endpoint group
advertise_byoip_cidr	Advertises an IPv4 address range that is provisioned for use with your A
allow_custom_routing_traffic	Specify the Amazon EC2 instance (destination) IP addresses and ports
create_accelerator	Create an accelerator
create_cross_account_attachment	Create a cross-account attachment in Global 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
create_custom_routing_listener	Create a listener to process inbound connections from clients to a custo
create_endpoint_group	Create an endpoint group for the specified listener
create_listener	Create a listener to process inbound connections from clients to an acce
delete_accelerator	Delete an accelerator
delete_cross_account_attachment	Delete a cross-account attachment
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 accelerat
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
deprovision_byoip_cidr	Releases the specified address range that you provisioned to use with y
describe_accelerator	Describe an accelerator
describe_accelerator_attributes	Describe the attributes of an accelerator
describe_cross_account_attachment	Gets configuration information about a cross-account attachment
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	Describe a listener

list_accelerators	List the accelerators for an Amazon Web Services account
list_byoip_cidrs	Lists the IP address ranges that were specified in calls to ProvisionByoi
list_cross_account_attachments	List the cross-account attachments that have been created in Global Ac
list_cross_account_resource_accounts	List the accounts that have cross-account resources
list_cross_account_resources	List the cross-account resources available to work with
list_custom_routing_accelerators	List the custom routing accelerators for an Amazon Web Services acco
list_custom_routing_endpoint_groups	List the endpoint groups that are associated with a listener for a custom
list_custom_routing_listeners	List the listeners for a custom routing accelerator
list_custom_routing_port_mappings	Provides a complete mapping from the public accelerator IP address an
list_custom_routing_port_mappings_by_destination	List the port mappings for a specific EC2 instance (destination) in a VP
list_endpoint_groups	List the endpoint groups that are associated with a listener
list_listeners	List the listeners for an accelerator
list_tags_for_resource	List all tags for an accelerator
provision_byoip_cidr	Provisions an IP address range to use with your Amazon Web Services
remove_custom_routing_endpoints	Remove endpoints from a custom routing accelerator
remove_endpoints	Remove endpoints from an endpoint group
tag_resource	Add tags to an accelerator resource
untag_resource	Remove tags from a Global Accelerator resource
update_accelerator	Update an accelerator to make changes, such as the following:
update_accelerator_attributes	Update the attributes for an accelerator
update_cross_account_attachment	Update a cross-account attachment to add or remove principals or resou
update_custom_routing_accelerator	Update a custom routing accelerator
update_custom_routing_accelerator_attributes	Update the attributes for a custom routing accelerator
update_custom_routing_listener	Update a listener for a custom routing accelerator
update_endpoint_group	Update an endpoint group
update_listener	Update a listener
withdraw_byoip_cidr	Stops advertising an address range that is provisioned as an address poo

Examples

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

## End(Not run)
```

`networkfirewall` *AWS Network Firewall*

Description

This is the API Reference for Network Firewall. This guide is for developers who need detailed information about the Network Firewall API actions, data types, and errors.

The REST API requires you to handle connection details, such as calculating signatures, handling request retries, and error handling. For general information about using the Amazon Web Services REST APIs, see Amazon Web Services APIs.

To view the complete list of Amazon Web Services Regions where Network Firewall is available, see Service endpoints and quotas in the *Amazon Web Services General Reference*.

To access Network Firewall using the IPv4 REST API endpoint: `https://network-firewall.<region>.amazonaws.com`

To access Network Firewall using the Dualstack (IPv4 and IPv6) REST API endpoint: `https://network-firewall.<regio`

Alternatively, you can use one of the Amazon Web Services SDKs to access an API that's tailored to the programming language or platform that you're using. For more information, see Amazon Web Services SDKs.

For descriptions of Network Firewall features, including and step-by-step instructions on how to use them through the Network Firewall console, see the Network Firewall Developer Guide.

Network Firewall is a stateful, managed, network firewall and intrusion detection and prevention service for Amazon Virtual Private Cloud (Amazon VPC). With Network Firewall, you can filter traffic at the perimeter of your VPC. This includes filtering traffic going to and coming from an internet gateway, NAT gateway, or over VPN or Direct Connect. Network Firewall uses rules that are compatible with Suricata, a free, open source network analysis and threat detection engine. Network Firewall supports Suricata version 7.0.3. For information about Suricata, see the Suricata website and the Suricata User Guide.

You can use Network Firewall to monitor and protect your VPC traffic in a number of ways. The following are just a few examples:

- Allow domains or IP addresses for known Amazon Web Services service endpoints, such as Amazon S3, and block all other forms of traffic.
- Use custom lists of known bad domains to limit the types of domain names that your applications can access.
- Perform deep packet inspection on traffic entering or leaving your VPC.
- Use stateful protocol detection to filter protocols like HTTPS, regardless of the port used.

To enable Network Firewall for your VPCs, you perform steps in both Amazon VPC and in Network Firewall. For information about using Amazon VPC, see Amazon VPC User Guide.

To start using Network Firewall, do the following:

1. (Optional) If you don't already have a VPC that you want to protect, create it in Amazon VPC.
2. In Amazon VPC, in each Availability Zone where you want to have a firewall endpoint, create a subnet for the sole use of Network Firewall.
3. In Network Firewall, create stateless and stateful rule groups, to define the components of the network traffic filtering behavior that you want your firewall to have.
4. In Network Firewall, create a firewall policy that uses your rule groups and specifies additional default traffic filtering behavior.

5. In Network Firewall, create a firewall and specify your new firewall policy and VPC subnets. Network Firewall creates a firewall endpoint in each subnet that you specify, with the behavior that's defined in the firewall policy.
6. In Amazon VPC, use ingress routing enhancements to route traffic through the new firewall endpoints.

Usage

```
networkfirewall(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e
html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- networkfirewall(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

associate_firewall_policy	Associates a FirewallPolicy to a Firewall
associate_subnets	Associates the specified subnets in the Amazon VPC to the firewall
create_firewall	Creates an Network Firewall Firewall and accompanying FirewallStatus for a VP
create_firewall_policy	Creates the firewall policy for the firewall according to the specifications
create_rule_group	Creates the specified stateless or stateful rule group, which includes the rules for
create_tls_inspection_configuration	Creates an Network Firewall TLS inspection configuration
delete_firewall	Deletes the specified Firewall and its FirewallStatus
delete_firewall_policy	Deletes the specified FirewallPolicy

delete_resource_policy	Deletes a resource policy that you created in a PutResourcePolicy request
delete_rule_group	Deletes the specified RuleGroup
delete_tls_inspection_configuration	Deletes the specified TLSInspectionConfiguration
describe_firewall	Returns the data objects for the specified firewall
describe_firewall_policy	Returns the data objects for the specified firewall policy
describe_logging_configuration	Returns the logging configuration for the specified firewall
describe_resource_policy	Retrieves a resource policy that you created in a PutResourcePolicy request
describe_rule_group	Returns the data objects for the specified rule group
describe_rule_group_metadata	High-level information about a rule group, returned by operations like create and
describe_tls_inspection_configuration	Returns the data objects for the specified TLS inspection configuration
disassociate_subnets	Removes the specified subnet associations from the firewall
get_analysis_report_results	The results of a COMPLETED analysis report generated with StartAnalysisRepo
list_analysis_reports	Returns a list of all traffic analysis reports generated within the last 30 days
list_firewall_policies	Retrieves the metadata for the firewall policies that you have defined
list_firewalls	Retrieves the metadata for the firewalls that you have defined
list_rule_groups	Retrieves the metadata for the rule groups that you have defined
list_tags_for_resource	Retrieves the tags associated with the specified resource
list_tls_inspection_configurations	Retrieves the metadata for the TLS inspection configurations that you have defin
put_resource_policy	Creates or updates an IAM policy for your rule group or firewall policy
start_analysis_report	Generates a traffic analysis report for the timeframe and traffic type you specify
tag_resource	Adds the specified tags to the specified resource
untag_resource	Removes the tags with the specified keys from the specified resource
update_firewall_analysis_settings	Enables specific types of firewall analysis on a specific firewall you define
update_firewall_delete_protection	Modifies the flag, DeleteProtection, which indicates whether it is possible to dele
update_firewall_description	Modifies the description for the specified firewall
update_firewall_encryption_configuration	A complex type that contains settings for encryption of your firewall resources
update_firewall_policy	Updates the properties of the specified firewall policy
update_firewall_policy_change_protection	Modifies the flag, ChangeProtection, which indicates whether it is possible to ch
update_logging_configuration	Sets the logging configuration for the specified firewall
update_rule_group	Updates the rule settings for the specified rule group
update_subnet_change_protection	Update subnet change protection
update_tls_inspection_configuration	Updates the TLS inspection configuration settings for the specified TLS inspecti

Examples

```
## Not run:
svc <- networkfirewall()
svc$associate_firewall_policy(
  Foo = 123
)

## End(Not run)
```

networkmanager *AWS Network Manager*

Description

Amazon Web Services enables you to centrally manage your Amazon Web Services Cloud WAN core network and your Transit Gateway network across Amazon Web Services accounts, Regions, and on-premises locations.

Usage

```
networkmanager(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e
html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token

- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- networkmanager(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

accept_attachment	Accepts a core network attachment request
associate_connect_peer	Associates a core network Connect peer with a device and optionally, with a
associate_customer_gateway	Associates a customer gateway with a device and optionally, with a link
associate_link	Associates a link to a device
associate_transit_gateway_connect_peer	Associates a transit gateway Connect peer with a device, and optionally, wi
create_connect_attachment	Creates a core network Connect attachment from a specified core network a
create_connection	Creates a connection between two devices
create_connect_peer	Creates a core network Connect peer for a specified core network connect a
create_core_network	Creates a core network as part of your global network, and optionally, with
create_device	Creates a new device in a global network
create_direct_connect_gateway_attachment	Creates an Amazon Web Services Direct Connect gateway attachment
create_global_network	Creates a new, empty global network
create_link	Creates a new link for a specified site
create_site	Creates a new site in a global network
create_site_to_site_vpn_attachment	Creates an Amazon Web Services site-to-site VPN attachment on an edge lo
create_transit_gateway_peering	Creates a transit gateway peering connection
create_transit_gateway_route_table_attachment	Creates a transit gateway route table attachment
create_vpc_attachment	Creates a VPC attachment on an edge location of a core network
delete_attachment	Deletes an attachment
delete_connection	Deletes the specified connection in your global network
delete_connect_peer	Deletes a Connect peer
delete_core_network	Deletes a core network along with all core network policies
delete_core_network_policy_version	Deletes a policy version from a core network
delete_device	Deletes an existing device
delete_global_network	Deletes an existing global network
delete_link	Deletes an existing link
delete_peering	Deletes an existing peering connection
delete_resource_policy	Deletes a resource policy for the specified resource
delete_site	Deletes an existing site
deregister_transit_gateway	Deregisters a transit gateway from your global network
describe_global_networks	Describes one or more global networks
disassociate_connect_peer	Disassociates a core network Connect peer from a device and a link
disassociate_customer_gateway	Disassociates a customer gateway from a device and a link
disassociate_link	Disassociates an existing device from a link
disassociate_transit_gateway_connect_peer	Disassociates a transit gateway Connect peer from a device and link
execute_core_network_change_set	Executes a change set on your core network
get_connect_attachment	Returns information about a core network Connect attachment
get_connections	Gets information about one or more of your connections in a global networ
get_connect_peer	Returns information about a core network Connect peer
get_connect_peer_associations	Returns information about a core network Connect peer associations
get_core_network	Returns information about the LIVE policy for a core network
get_core_network_change_events	Returns information about a core network change event
get_core_network_change_set	Returns a change set between the LIVE core network policy and a submitte
get_core_network_policy	Returns details about a core network policy
get_customer_gateway_associations	Gets the association information for customer gateways that are associated
get_devices	Gets information about one or more of your devices in a global network
get_direct_connect_gateway_attachment	Returns information about a specific Amazon Web Services Direct Connect
get_link_associations	Gets the link associations for a device or a link

get_links	Gets information about one or more links in a specified global network
get_network_resource_counts	Gets the count of network resources, by resource type, for the specified glol
get_network_resource_relationships	Gets the network resource relationships for the specified global network
get_network_resources	Describes the network resources for the specified global network
get_network_routes	Gets the network routes of the specified global network
get_network_telemetry	Gets the network telemetry of the specified global network
get_resource_policy	Returns information about a resource policy
get_route_analysis	Gets information about the specified route analysis
get_sites	Gets information about one or more of your sites in a global network
get_site_to_site_vpn_attachment	Returns information about a site-to-site VPN attachment
get_transit_gateway_connect_peer_associations	Gets information about one or more of your transit gateway Connect peer a
get_transit_gateway_peering	Returns information about a transit gateway peer
get_transit_gateway_registrations	Gets information about the transit gateway registrations in a specified globa
get_transit_gateway_route_table_attachment	Returns information about a transit gateway route table attachment
get_vpc_attachment	Returns information about a VPC attachment
list_attachments	Returns a list of core network attachments
list_connect_peers	Returns a list of core network Connect peers
list_core_network_policy_versions	Returns a list of core network policy versions
list_core_networks	Returns a list of owned and shared core networks
list_organization_service_access_status	Gets the status of the Service Linked Role (SLR) deployment for the accou
list_peerings	Lists the peerings for a core network
list_tags_for_resource	Lists the tags for a specified resource
put_core_network_policy	Creates a new, immutable version of a core network policy
put_resource_policy	Creates or updates a resource policy
register_transit_gateway	Registers a transit gateway in your global network
reject_attachment	Rejects a core network attachment request
restore_core_network_policy_version	Restores a previous policy version as a new, immutable version of a core ne
start_organization_service_access_update	Enables the Network Manager service for an Amazon Web Services Organi
start_route_analysis	Starts analyzing the routing path between the specified source and destinati
tag_resource	Tags a specified resource
untag_resource	Removes tags from a specified resource
update_connection	Updates the information for an existing connection
update_core_network	Updates the description of a core network
update_device	Updates the details for an existing device
update_direct_connect_gateway_attachment	Updates the edge locations associated with an Amazon Web Services Direc
update_global_network	Updates an existing global network
update_link	Updates the details for an existing link
update_network_resource_metadata	Updates the resource metadata for the specified global network
update_site	Updates the information for an existing site
update_vpc_attachment	Updates a VPC attachment

Examples

```
## Not run:
svc <- networkmanager()
svc$accept_attachment(
  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.

You can use Route 53 to:

- Register domain names.
  For more information, see How domain registration works.
- Route internet traffic to the resources for your domain
  For more information, see How internet traffic is routed to your website or web application.
- Check the health of your resources.
  For more information, see How Route 53 checks the health of your resources.

Usage

```
route53(config = list(), credentials = list(), endpoint = NULL, region = NULL)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy `https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e`
  `html`

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
```

```
    region = "string"
  )
```

Operations

activate_key_signing_key	Activates a key-signing key (KSK) so that it can be used for signing by DNSS
associate_vpc_with_hosted_zone	Associates an Amazon VPC with a private hosted zone
change_cidr_collection	Creates, changes, or deletes CIDR blocks within a collection
change_resource_record_sets	Creates, changes, or deletes a resource record set, which contains authoritative
change_tags_for_resource	Adds, edits, or deletes tags for a health check or a hosted zone
create_cidr_collection	Creates a CIDR collection in the current Amazon Web Services account
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 r
create_traffic_policy	Creates a traffic policy, which you use to create multiple DNS resource record
create_traffic_policy_instance	Creates resource record sets in a specified hosted zone based on the settings in
create_traffic_policy_version	Creates a new version of an existing traffic policy
create_vpc_association_authorization	Authorizes the Amazon Web Services account that created a specified VPC to
deactivate_key_signing_key	Deactivates a key-signing key (KSK) so that it will not be used for signing by
delete_cidr_collection	Deletes a CIDR collection in the current Amazon Web Services account
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	Deletes a traffic policy instance and all of the resource record sets that Amazo
delete_vpc_association_authorization	Removes authorization to submit an AssociateVPCWithHostedZone request t
disable_hosted_zone_dnssec	Disables DNSSEC signing in a specific hosted zone
disassociate_vpc_from_hosted_zone	Disassociates an Amazon Virtual Private Cloud (Amazon VPC) from an Ama
enable_hosted_zone_dnssec	Enables DNSSEC signing in a specific hosted zone
get_account_limit	Gets the specified limit for the current account, for example, the maximum nu
get_change	Returns the current status of a change batch request
get_checker_ip_ranges	Route 53 does not perform authorization for this API because it retrieves infor
get_dnssec	Returns information about DNSSEC for a specific hosted zone, including the
get_geo_location	Gets information about whether a specified geographic location is supported f
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 Am
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 server
get_hosted_zone_count	Retrieves the number of hosted zones that are associated with the current Ama
get_hosted_zone_limit	Gets the specified limit for a specified hosted zone, for example, the maximun
get_query_logging_config	Gets information about a specified configuration for DNS query logging
get_reusable_delegation_set	Retrieves information about a specified reusable delegation set, including the
get_reusable_delegation_set_limit	Gets the maximum number of hosted zones that you can associate with the sp

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	Gets the number of traffic policy instances that are associated with the current
list_cidr_blocks	Returns a paginated list of location objects and their CIDR blocks
list_cidr_collections	Returns a paginated list of CIDR collections in the Amazon Web Services acc
list_cidr_locations	Returns a paginated list of CIDR locations for the given collection (metadata
list_geo_locations	Retrieves a list of supported geographic locations
list_health_checks	Retrieve a list of the health checks that are associated with the current Amazo
list_hosted_zones	Retrieves a list of the public and private hosted zones that are associated with
list_hosted_zones_by_name	Retrieves a list of your hosted zones in lexicographic order
list_hosted_zones_by_vpc	Lists all the private hosted zones that a specified VPC is associated with, rega
list_query_logging_configs	Lists the configurations for DNS query logging that are associated with the cu
list_resource_record_sets	Lists the resource record sets in a specified hosted zone
list_reusable_delegation_sets	Retrieves a list of the reusable delegation sets that are associated with the curr
list_tags_for_resource	Lists tags for one health check or hosted zone
list_tags_for_resources	Lists tags for up to 10 health checks or hosted zones
list_traffic_policies	Gets information about the latest version for every traffic policy that is associa
list_traffic_policy_instances	Gets information about the traffic policy instances that you created by using th
list_traffic_policy_instances_by_hosted_zone	Gets information about the traffic policy instances that you created in a specifi
list_traffic_policy_instances_by_policy	Gets information about the traffic policy instances that you created by using a
list_traffic_policy_versions	Gets information about all of the versions for a specified traffic policy
list_vpc_association_authorizations	Gets a list of the VPCs that were created by other accounts and that can be ass
test_dns_answer	Gets the value that Amazon Route 53 returns in response to a DNS request for
update_health_check	Updates an existing health check
update_hosted_zone_comment	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	After you submit a UpdateTrafficPolicyInstance request, there's a brief delay

Examples

```
## 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)
```

route53domains *Amazon Route 53 Domains*

Description

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

Usage

```
route53domains(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.

- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- route53domains(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

accept_domain_transfer_from_another_aws_account	Accepts the transfer of a domain from another Amazon Web Services a
associate_delegation_signer_to_domain	Creates a delegation signer (DS) record in the registry zone for this dor
cancel_domain_transfer_to_another_aws_account	Cancels the transfer of a domain from the current Amazon Web Service

check_domain_availability	This operation checks the availability of one domain name
check_domain_transferability	Checks whether a domain name can be transferred to Amazon Route 5
delete_domain	This operation deletes the specified domain
delete_tags_for_domain	This operation deletes the specified tags for a domain
disable_domain_auto_renew	This operation disables automatic renewal of domain registration for th
disable_domain_transfer_lock	This operation removes the transfer lock on the domain (specifically th
disassociate_delegation_signer_from_domain	Deletes a delegation signer (DS) record in the registry zone for this dor
enable_domain_auto_renew	This operation configures Amazon Route 53 to automatically renew the
enable_domain_transfer_lock	This operation sets the transfer lock on the domain (specifically the clie
get_contact_reachability_status	For operations that require confirmation that the email address for the r
get_domain_detail	This operation returns detailed information about a specified domain th
get_domain_suggestions	The GetDomainSuggestions operation returns a list of suggested doma
get_operation_detail	This operation returns the current status of an operation that is not com
list_domains	This operation returns all the domain names registered with Amazon R
list_operations	Returns information about all of the operations that return an operation
list_prices	Lists the following prices for either all the TLDs supported by Route 5
list_tags_for_domain	This operation returns all of the tags that are associated with the specifi
push_domain	Moves a domain from Amazon Web Services to another registrar
register_domain	This operation registers a domain
reject_domain_transfer_from_another_aws_account	Rejects the transfer of a domain from another Amazon Web Services ac
renew_domain	This operation renews a domain for the specified number of years
resend_contact_reachability_email	For operations that require confirmation that the email address for the r
resend_operation_authorization	Resend the form of authorization email for this operation
retrieve_domain_auth_code	This operation returns the authorization code for the domain
transfer_domain	Transfers a domain from another registrar to Amazon Route 53
transfer_domain_to_another_aws_account	Transfers a domain from the current Amazon Web Services account to
update_domain_contact	This operation updates the contact information for a particular domain
update_domain_contact_privacy	This operation updates the specified domain contact's privacy setting
update_domain_nameservers	This operation replaces the current set of name servers for the domain
update_tags_for_domain	This operation adds or updates tags for a specified domain
view_billing	Returns all the domain-related billing records for the current Amazon

Examples

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

## End(Not run)
```

route53profiles *Route 53 Profiles*

Description

With Amazon Route 53 Profiles you can share Route 53 configurations with VPCs and AWS accounts

Usage

```
route53profiles(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e
html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token

- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- route53profiles(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

associate_profile	Associates a Route 53 Profiles profile with a VPC
associate_resource_to_profile	Associates a DNS reource configuration to a Route 53 Profile
create_profile	Creates an empty Route 53 Profile
delete_profile	Deletes the specified Route 53 Profile
disassociate_profile	Dissociates a specified Route 53 Profile from the specified VPC
disassociate_resource_from_profile	Dissoaciated a specified resource, from the Route 53 Profile
get_profile	Returns information about a specified Route 53 Profile, such as whether whether the Pr
get_profile_association	Retrieves a Route 53 Profile association for a VPC
get_profile_resource_association	Returns information about a specified Route 53 Profile resource association
list_profile_associations	Lists all the VPCs that the specified Route 53 Profile is associated with
list_profile_resource_associations	Lists all the resource associations for the specified Route 53 Profile
list_profiles	Lists all the Route 53 Profiles associated with your Amazon Web Services account
list_tags_for_resource	Lists the tags that you associated with the specified resource
tag_resource	Adds one or more tags to a specified resource
untag_resource	Removes one or more tags from a specified resource
update_profile_resource_association	Updates the specified Route 53 Profile resourse association

Examples

```
## Not run:
svc <- route53profiles()
svc$associate_profile(
  Foo = 123
)

## End(Not run)
```

`route53recoverycluster`

Route53 Recovery Cluster

Description

Welcome to the Routing Control (Recovery Cluster) API Reference Guide for Amazon Route 53 Application Recovery Controller.

With Route 53 ARC, you can use routing control with extreme reliability to recover applications by rerouting traffic across Availability Zones or Amazon Web Services Regions. Routing controls are simple on/off switches hosted on a highly available cluster in Route 53 ARC. A cluster provides a set of five redundant Regional endpoints against which you can run API calls to get or update the state of routing controls. To implement failover, you set one routing control to ON and another one to OFF, to reroute traffic from one Availability Zone or Amazon Web Services Region to another.

Be aware that you must specify a Regional endpoint for a cluster when you work with API cluster operations to get or update routing control states in Route 53 ARC. In addition, you must specify the US West (Oregon) Region for Route 53 ARC API calls. For example, use the parameter

`--region us-west-2` with AWS CLI commands. For more information, see Get and update routing control states using the API in the Amazon Route 53 Application Recovery Controller Developer Guide.

This API guide includes information about the API operations for how to get and update routing control states in Route 53 ARC. To work with routing control in Route 53 ARC, you must first create the required components (clusters, control panels, and routing controls) using the recovery cluster configuration API.

For more information about working with routing control in Route 53 ARC, see the following:

- Create clusters, control panels, and routing controls by using API operations. For more information, see the Recovery Control Configuration API Reference Guide for Amazon Route 53 Application Recovery Controller.
- Learn about the components in recovery control, including clusters, routing controls, and control panels, and how to work with Route 53 ARC in the Amazon Web Services console. For more information, see Recovery control components in the Amazon Route 53 Application Recovery Controller Developer Guide.
- Route 53 ARC also provides readiness checks that continually audit resources to help make sure that your applications are scaled and ready to handle failover traffic. For more information about the related API operations, see the Recovery Readiness API Reference Guide for Amazon Route 53 Application Recovery Controller.
- For more information about creating resilient applications and preparing for recovery readiness with Route 53 ARC, see the Amazon Route 53 Application Recovery Controller Developer Guide.

Usage

```
route53recoverycluster(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.

- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- route53recoverycluster(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
```

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

Operations

get_routing_control_state	Get the state for a routing control
list_routing_controls	List routing control names and Amazon Resource Names (ARNs), as well as the routing cont
update_routing_control_state	Set the state of the routing control to reroute traffic
update_routing_control_states	Set multiple routing control states

Examples

```
## Not run:
svc <- route53recoverycluster()
svc$get_routing_control_state(
  Foo = 123
)

## End(Not run)
```

`route53recoverycontrolconfig`

AWS Route53 Recovery Control Config

Description

Recovery Control Configuration API Reference for Amazon Route 53 Application Recovery Controller

Usage

```
route53recoverycontrolconfig(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- route53recoverycontrolconfig(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
```

```
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string",
  close_connection = "logical",
  timeout = "numeric",
  s3_force_path_style = "logical",
  sts_regional_endpoint = "string"
),
credentials = list(
  creds = list(
    access_key_id = "string",
    secret_access_key = "string",
    session_token = "string"
  ),
  profile = "string",
  anonymous = "logical"
),
endpoint = "string",
region = "string"
)
```

Operations

create_cluster	Create a new cluster
create_control_panel	Creates a new control panel
create_routing_control	Creates a new routing control
create_safety_rule	Creates a safety rule in a control panel
delete_cluster	Delete a cluster
delete_control_panel	Deletes a control panel
delete_routing_control	Deletes a routing control
delete_safety_rule	Deletes a safety rule
describe_cluster	Display the details about a cluster
describe_control_panel	Displays details about a control panel
describe_routing_control	Displays details about a routing control
describe_safety_rule	Returns information about a safety rule
get_resource_policy	Get information about the resource policy for a cluster
list_associated_route_53_health_checks	Returns an array of all Amazon Route 53 health checks associated with a specific r
list_clusters	Returns an array of all the clusters in an account
list_control_panels	Returns an array of control panels in an account or in a cluster
list_routing_controls	Returns an array of routing controls for a control panel
list_safety_rules	List the safety rules (the assertion rules and gating rules) that you've defined for the
list_tags_for_resource	Lists the tags for a resource
tag_resource	Adds a tag to a resource

untag_resource	Removes a tag from a resource
update_control_panel	Updates a control panel
update_routing_control	Updates a routing control
update_safety_rule	Update a safety rule (an assertion rule or gating rule)

Examples

```
## Not run:
svc <- route53recoverycontrolconfig()
svc$create_cluster(
  Foo = 123
)

## End(Not run)
```

`route53recoveryreadiness`

AWS Route53 Recovery Readiness

Description

Recovery readiness

Usage

```
route53recoveryreadiness(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.

- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- route53recoveryreadiness(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
```

```
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

create_cell	Creates a cell in an account
create_cross_account_authorization	Creates a cross-account readiness authorization
create_readiness_check	Creates a readiness check in an account
create_recovery_group	Creates a recovery group in an account
create_resource_set	Creates a resource set
delete_cell	Delete a cell
delete_cross_account_authorization	Deletes cross account readiness authorization
delete_readiness_check	Deletes a readiness check
delete_recovery_group	Deletes a recovery group
delete_resource_set	Deletes a resource set
get_architecture_recommendations	Gets recommendations about architecture designs for improving resiliency for an a
get_cell	Gets information about a cell including cell name, cell Amazon Resource Name (A
get_cell_readiness_summary	Gets readiness for a cell
get_readiness_check	Gets details about a readiness check
get_readiness_check_resource_status	Gets individual readiness status for a readiness check
get_readiness_check_status	Gets the readiness status for an individual readiness check
get_recovery_group	Gets details about a recovery group, including a list of the cells that are included i
get_recovery_group_readiness_summary	Displays a summary of information about a recovery group's readiness status
get_resource_set	Displays the details about a resource set, including a list of the resources in the set
list_cells	Lists the cells for an account
list_cross_account_authorizations	Lists the cross-account readiness authorizations that are in place for an account
list_readiness_checks	Lists the readiness checks for an account
list_recovery_groups	Lists the recovery groups in an account
list_resource_sets	Lists the resource sets in an account
list_rules	Lists all readiness rules, or lists the readiness rules for a specific resource type
list_tags_for_resources	Lists the tags for a resource
tag_resource	Adds a tag to a resource
untag_resource	Removes a tag from a resource
update_cell	Updates a cell to replace the list of nested cells with a new list of nested cells
update_readiness_check	Updates a readiness check
update_recovery_group	Updates a recovery group
update_resource_set	Updates a resource set

Examples

```
## Not run:
svc <- route53recoveryreadiness()
svc$create_cell(
  Foo = 123
)

## End(Not run)
```

`route53resolver` *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 Elastic Load Balancing 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 Amazon Web Services 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(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e
html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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",
      anonymous = "logical"
    ),
    endpoint = "string",
    region = "string",
    close_connection = "logical",
    timeout = "numeric",
    s3_force_path_style = "logical",
    sts_regional_endpoint = "string"
  ),
  credentials = list(
    creds = list(
      access_key_id = "string",
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string"
)
```

Operations

associate_firewall_rule_group	Associates a FirewallRuleGroup with a VPC, to provide DNS filtering for the V
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_firewall_domain_list	Creates an empty firewall domain list for use in DNS Firewall rules
create_firewall_rule	Creates a single DNS Firewall rule in the specified rule group, using the specifi
create_firewall_rule_group	Creates an empty DNS Firewall rule group for filtering DNS network traffic in
create_outpost_resolver	Creates a Route 53 Resolver on an Outpost
create_resolver_endpoint	Creates a Resolver endpoint
create_resolver_query_log_config	Creates a Resolver query logging configuration, which defines where you want
create_resolver_rule	For DNS queries that originate in your VPCs, specifies which Resolver endpoi
delete_firewall_domain_list	Deletes the specified domain list
delete_firewall_rule	Deletes the specified firewall rule

delete_firewall_rule_group	Deletes the specified firewall rule group
delete_outpost_resolver	Deletes a Resolver on the Outpost
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_firewall_rule_group	Disassociates a FirewallRuleGroup from a VPC, to remove DNS filtering from
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 specified VPC
get_firewall_config	Retrieves the configuration of the firewall behavior provided by DNS Firewall
get_firewall_domain_list	Retrieves the specified firewall domain list
get_firewall_rule_group	Retrieves the specified firewall rule group
get_firewall_rule_group_association	Retrieves a firewall rule group association, which enables DNS filtering for a V
get_firewall_rule_group_policy	Returns the Identity and Access Management (Amazon Web Services IAM) po
get_outpost_resolver	Gets information about a specified Resolver on the Outpost, such as its instanc
get_resolver_config	Retrieves the behavior configuration of Route 53 Resolver behavior for a single
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 i
get_resolver_query_log_config	Gets information about a specified Resolver query logging configuration, such
get_resolver_query_log_config_association	Gets information about a specified association between a Resolver query loggi
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
get_resolver_rule_association	Gets information about an association between a specified Resolver rule and a
get_resolver_rule_policy	Gets information about the Resolver rule policy for a specified rule
import_firewall_domains	Imports domain names from a file into a domain list, for use in a DNS firewall
list_firewall_configs	Retrieves the firewall configurations that you have defined
list_firewall_domain_lists	Retrieves the firewall domain lists that you have defined
list_firewall_domains	Retrieves the domains that you have defined for the specified firewall domain li
list_firewall_rule_group_associations	Retrieves the firewall rule group associations that you have defined
list_firewall_rule_groups	Retrieves the minimal high-level information for the rule groups that you have
list_firewall_rules	Retrieves the firewall rules that you have defined for the specified firewall rule
list_outpost_resolvers	Lists all the Resolvers on Outposts that were created using the current Amazon
list_resolver_configs	Retrieves the Resolver configurations that you have defined
list_resolver_dnssec_configs	Lists the configurations for DNSSEC validation that are associated with the cu
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 Amazon We
list_resolver_query_log_config_associations	Lists information about associations between Amazon VPCs and query logging
list_resolver_query_log_configs	Lists information about the specified query logging configurations
list_resolver_rule_associations	Lists the associations that were created between Resolver rules and VPCs using
list_resolver_rules	Lists the Resolver rules that were created using the current Amazon Web Servi
list_tags_for_resource	Lists the tags that you associated with the specified resource
put_firewall_rule_group_policy	Attaches an Identity and Access Management (Amazon Web Services IAM) po
put_resolver_query_log_config_policy	Specifies an Amazon Web Services account that you want to share a query logg
put_resolver_rule_policy	Specifies an Amazon Web Services rule that you want to share with another ac
tag_resource	Adds one or more tags to a specified resource
untag_resource	Removes one or more tags from a specified resource
update_firewall_config	Updates the configuration of the firewall behavior provided by DNS Firewall fo
update_firewall_domains	Updates the firewall domain list from an array of domain specifications

update_firewall_rule	Updates the specified firewall rule
update_firewall_rule_group_association	Changes the association of a FirewallRuleGroup with a VPC
update_outpost_resolver	You can use UpdateOutpostResolver to update the instance count, type, or nam
update_resolver_config	Updates the behavior configuration of Route 53 Resolver behavior for a single
update_resolver_dnssec_config	Updates an existing DNSSEC validation configuration
update_resolver_endpoint	Updates the name, or endpoint type for an inbound or an outbound Resolver en
update_resolver_rule	Updates settings for a specified Resolver rule

Examples

```
## Not run:
svc <- route53resolver()
svc$associate_firewall_rule_group(
  Foo = 123
)

## End(Not run)
```

`servicediscovery` *AWS Cloud Map*

Description

Cloud Map

With Cloud Map, you can configure public DNS, private DNS, or HTTP namespaces that your microservice applications run in. When an instance becomes available, you can call the Cloud Map API to register the instance with Cloud Map. For public or private DNS namespaces, 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(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- servicediscovery(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
```

```
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string",
  close_connection = "logical",
  timeout = "numeric",
  s3_force_path_style = "logical",
  sts_regional_endpoint = "string"
),
credentials = list(
  creds = list(
    access_key_id = "string",
    secret_access_key = "string",
    session_token = "string"
  ),
  profile = "string",
  anonymous = "logical"
),
endpoint = "string",
region = "string"
)
```

Operations

create_http_namespace	Creates an HTTP namespace
create_private_dns_namespace	Creates a private namespace based on DNS, which is visible only inside a specified A
create_public_dns_namespace	Creates a public namespace based on DNS, which is visible on the internet
create_service	Creates a service
delete_namespace	Deletes a namespace from the current account
delete_service	Deletes a specified service and all associated service attributes
delete_service_attributes	Deletes specific attributes associated with a service
deregister_instance	Deletes the Amazon Route 53 DNS records and health check, if any, that Cloud Map
discover_instances	Discovers registered instances for a specified namespace and service
discover_instances_revision	Discovers the increasing revision associated with an instance
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 insta
get_namespace	Gets information about a namespace
get_operation	Gets information about any operation that returns an operation ID in the response, su
get_service	Gets the settings for a specified service
get_service_attributes	Returns the attributes associated with a specified service
list_instances	Lists summary information about the instances that you registered by using a specifi
list_namespaces	Lists summary information about the namespaces that were created by the current A
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 n

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
tag_resource	Adds one or more tags to the specified resource
untag_resource	Removes one or more tags from the specified resource
update_http_namespace	Updates an HTTP namespace
update_instance_custom_health_status	Submits a request to change the health status of a custom health check to healthy or
update_private_dns_namespace	Updates a private DNS namespace
update_public_dns_namespace	Updates a public DNS namespace
update_service	Submits a request to perform the following operations:
update_service_attributes	Submits a request to update a specified service to add service-level attributes

Examples

```
## 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)
```

---

telconetworkbuilder *AWS Telco Network Builder*

Description

Amazon Web Services Telco Network Builder (TNB) is a network automation service that helps you deploy and manage telecom networks. AWS TNB helps you with the lifecycle management of your telecommunication network functions throughout planning, deployment, and post-deployment activities.

Usage

```
telconetworkbuilder(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- telconetworkbuilder(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
```

```
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string",
  close_connection = "logical",
  timeout = "numeric",
  s3_force_path_style = "logical",
  sts_regional_endpoint = "string"
),
credentials = list(
  creds = list(
    access_key_id = "string",
    secret_access_key = "string",
    session_token = "string"
  ),
  profile = "string",
  anonymous = "logical"
),
endpoint = "string",
region = "string"
)
```

Operations

cancel_sol_network_operation	Cancels a network operation
create_sol_function_package	Creates a function package
create_sol_network_instance	Creates a network instance
create_sol_network_package	Creates a network package
delete_sol_function_package	Deletes a function package
delete_sol_network_instance	Deletes a network instance
delete_sol_network_package	Deletes network package
get_sol_function_instance	Gets the details of a network function instance, including the instantiation state and
get_sol_function_package	Gets the details of an individual function package, such as the operational state and
get_sol_function_package_content	Gets the contents of a function package
get_sol_function_package_descriptor	Gets a function package descriptor in a function package
get_sol_network_instance	Gets the details of the network instance
get_sol_network_operation	Gets the details of a network operation, including the tasks involved in the network
get_sol_network_package	Gets the details of a network package
get_sol_network_package_content	Gets the contents of a network package
get_sol_network_package_descriptor	Gets the content of the network service descriptor
instantiate_sol_network_instance	Instantiates a network instance
list_sol_function_instances	Lists network function instances
list_sol_function_packages	Lists information about function packages
list_sol_network_instances	Lists your network instances

list_sol_network_operations	Lists details for a network operation, including when the operation started and the s
list_sol_network_packages	Lists network packages
list_tags_for_resource	Lists tags for AWS TNB resources
put_sol_function_package_content	Uploads the contents of a function package
put_sol_network_package_content	Uploads the contents of a network package
tag_resource	Tags an AWS TNB resource
terminate_sol_network_instance	Terminates a network instance
untag_resource	Untags an AWS TNB resource
update_sol_function_package	Updates the operational state of function package
update_sol_network_instance	Update a network instance
update_sol_network_package	Updates the operational state of a network package
validate_sol_function_package_content	Validates function package content
validate_sol_network_package_content	Validates network package content

Examples

```
## Not run:
svc <- telconetworkbuilder()
svc$cancel_sol_network_operation(
  Foo = 123
)

## End(Not run)
```

`vpclattice` *Amazon VPC Lattice*

Description

Amazon VPC Lattice is a fully managed application networking service that you use to connect, secure, and monitor all of your services across multiple accounts and virtual private clouds (VPCs). Amazon VPC Lattice interconnects your microservices and legacy services within a logical boundary, so that you can discover and manage them more efficiently. For more information, see the Amazon VPC Lattice User Guide

Usage

```
vpclattice(
  config = list(),
  credentials = list(),
  endpoint = NULL,
  region = NULL
)
```

Arguments

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

- **credentials**:
  - **creds**:
    - **access_key_id**: AWS access key ID
    - **secret_access_key**: AWS secret access key
    - **session_token**: AWS temporary session token
  - **profile**: The name of a profile to use. If not given, then the default profile is used.
  - **anonymous**: Set anonymous credentials.
- **endpoint**: The complete URL to use for the constructed client.
- **region**: The AWS Region used in instantiating the client.
- **close_connection**: Immediately close all HTTP connections.
- **timeout**: The time in seconds till a timeout exception is thrown when attempting to make a connection. The default is 60 seconds.
- **s3_force_path_style**: Set this to `true` to force the request to use path-style addressing, i.e. `http://s3.amazonaws.com/BUCKET/KEY`.
- **sts_regional_endpoint**: Set sts regional endpoint resolver to regional or legacy https://docs.aws.amazon.com/sdkref/latest/guide/feature-sts-regionalized-e html

`credentials` Optional credentials shorthand for the config parameter

- **creds**:
  - **access_key_id**: AWS access key ID
  - **secret_access_key**: AWS secret access key
  - **session_token**: AWS temporary session token
- **profile**: The name of a profile to use. If not given, then the default profile is used.
- **anonymous**: Set anonymous credentials.

`endpoint` Optional shorthand for complete URL to use for the constructed client.

`region` Optional shorthand for AWS Region used in instantiating the client.

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 <- vpclattice(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
```

```
      secret_access_key = "string",
      session_token = "string"
    ),
    profile = "string",
    anonymous = "logical"
  ),
  endpoint = "string",
  region = "string",
  close_connection = "logical",
  timeout = "numeric",
  s3_force_path_style = "logical",
  sts_regional_endpoint = "string"
),
credentials = list(
  creds = list(
    access_key_id = "string",
    secret_access_key = "string",
    session_token = "string"
  ),
  profile = "string",
  anonymous = "logical"
),
endpoint = "string",
region = "string"
)
```

Operations

batch_update_rule	Updates the listener rules in a batch
create_access_log_subscription	Enables access logs to be sent to Amazon CloudWatch, Amazon S3, and A
create_listener	Creates a listener for a service
create_resource_configuration	Creates a resource configuration
create_resource_gateway	Creates a resource gateway
create_rule	Creates a listener rule
create_service	Creates a service
create_service_network	Creates a service network
create_service_network_resource_association	Associates the specified service network with the specified resource config
create_service_network_service_association	Associates the specified service with the specified service network
create_service_network_vpc_association	Associates a VPC with a service network
create_target_group	Creates a target group
delete_access_log_subscription	Deletes the specified access log subscription
delete_auth_policy	Deletes the specified auth policy
delete_listener	Deletes the specified listener
delete_resource_configuration	Deletes the specified resource configuration
delete_resource_endpoint_association	Disassociates the resource configuration from the resource VPC endpoint
delete_resource_gateway	Deletes the specified resource gateway
delete_resource_policy	Deletes the specified resource policy
delete_rule	Deletes a listener rule

delete_service	Deletes a service
delete_service_network	Deletes a service network
delete_service_network_resource_association	Deletes the association between a service network and a resource configur
delete_service_network_service_association	Deletes the association between a service and a service network
delete_service_network_vpc_association	Disassociates the VPC from the service network
delete_target_group	Deletes a target group
deregister_targets	Deregisters the specified targets from the specified target group
get_access_log_subscription	Retrieves information about the specified access log subscription
get_auth_policy	Retrieves information about the auth policy for the specified service or ser
get_listener	Retrieves information about the specified listener for the specified service
get_resource_configuration	Retrieves information about the specified resource configuration
get_resource_gateway	Retrieves information about the specified resource gateway
get_resource_policy	Retrieves information about the specified resource policy
get_rule	Retrieves information about the specified listener rules
get_service	Retrieves information about the specified service
get_service_network	Retrieves information about the specified service network
get_service_network_resource_association	Retrieves information about the specified association between a service ne
get_service_network_service_association	Retrieves information about the specified association between a service ne
get_service_network_vpc_association	Retrieves information about the specified association between a service ne
get_target_group	Retrieves information about the specified target group
list_access_log_subscriptions	Lists the access log subscriptions for the specified service network or servi
list_listeners	Lists the listeners for the specified service
list_resource_configurations	Lists the resource configurations owned by or shared with this account
list_resource_endpoint_associations	Lists the associations for the specified VPC endpoint
list_resource_gateways	Lists the resource gateways that you own or that were shared with you
list_rules	Lists the rules for the specified listener
list_service_network_resource_associations	Lists the associations between a service network and a resource configurat
list_service_networks	Lists the service networks owned by or shared with this account
list_service_network_service_associations	Lists the associations between a service network and a service
list_service_network_vpc_associations	Lists the associations between a service network and a VPC
list_service_network_vpc_endpoint_associations	Lists the associations between a service network and a VPC endpoint
list_services	Lists the services owned by the caller account or shared with the caller acc
list_tags_for_resource	Lists the tags for the specified resource
list_target_groups	Lists your target groups
list_targets	Lists the targets for the target group
put_auth_policy	Creates or updates the auth policy
put_resource_policy	Attaches a resource-based permission policy to a service or service networ
register_targets	Registers the targets with the target group
tag_resource	Adds the specified tags to the specified resource
untag_resource	Removes the specified tags from the specified resource
update_access_log_subscription	Updates the specified access log subscription
update_listener	Updates the specified listener for the specified service
update_resource_configuration	Updates the specified resource configuration
update_resource_gateway	Updates the specified resource gateway
update_rule	Updates a specified rule for the listener
update_service	Updates the specified service
update_service_network	Updates the specified service network
update_service_network_vpc_association	Updates the service network and VPC association

update_target_group	Updates the specified target group

Examples

```
## Not run:
svc <- vpclattice()
svc$batch_update_rule(
  Foo = 123
)

## End(Not run)
```

Index