Package: paws.migration (via r-universe)

September 4, 2024

Title 'Amazon Web Services' Migration & Transfer Services

Version 0.7.0

Description Interface to 'Amazon Web Services' migration & transfer services, including file and database migration to 'Amazon Web Services' <https://aws.amazon.com/>.

License Apache License (>= 2.0)

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

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

Imports paws.common (>= 0.5.4)

Suggests testthat

Encoding UTF-8

Roxygen list(markdown = TRUE, roclets = c(``rd'', ``namespace'', ``collate''))

RoxygenNote 7.1.1

Collate 'applicationdiscoveryservice_service.R' 'applicationdiscoveryservice_interfaces.R' 'applicationdiscoveryservice_operations.R' 'databasemigrationservice_service.R' 'databasemigrationservice_interfaces.R' 'databasemigrationservice_operations.R' 'datasync_service.R' 'datasync_interfaces.R' 'datasync_operations.R' 'importexport_service.R' 'importexport_interfaces.R' 'importexport_operations.R' 'migrationhub_service.R' 'migrationhub_interfaces.R' 'migrationhub_operations.R' 'sms_service.R' 'sms_interfaces.R' 'sms_operations.R' 'snowball_service.R' 'snowball_interfaces.R' 'snowball_operations.R' 'transfer_service.R' 'transfer_interfaces.R' 'transfer_operations.R'

Repository https://paws-r.r-universe.dev

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

RemoteRef HEAD

RemoteSha 8ab20b498262e0b343c0153b4f244483aff4987f

Contents

`applicationdiscoveryservice`

AWS Application Discovery Service

Description

AWS Application Discovery Service helps you plan application migration projects. It automatically identifies servers, virtual machines (VMs), and network dependencies in your on-premises data centers. For more information, see the AWS Application Discovery Service FAQ. Application Discovery Service offers three ways of performing discovery and collecting data about your on-premises servers:

- **Agentless discovery** is recommended for environments that use VMware vCenter Server. This mode doesn't require you to install an agent on each host. It does not work in non-VMware environments.
  - Agentless discovery gathers server information regardless of the operating systems, which minimizes the time required for initial on-premises infrastructure assessment.
  - Agentless discovery doesn't collect information about network dependencies, only agent-based discovery collects that information.
- **Agent-based discovery** collects a richer set of data than agentless discovery by using the AWS Application Discovery Agent, which you install on one or more hosts in your data center.
  - The agent captures infrastructure and application information, including an inventory of running processes, system performance information, resource utilization, and network dependencies.
  - The information collected by agents is secured at rest and in transit to the Application Discovery Service database in the cloud.
- **AWS Partner Network (APN) solutions** integrate with Application Discovery Service, enabling you to import details of your on-premises environment directly into Migration Hub without using the discovery connector or discovery agent.
  - Third-party application discovery tools can query AWS Application Discovery Service, and they can write to the Application Discovery Service database using the public API.
  - In this way, you can import data into Migration Hub and view it, so that you can associate applications with servers and track migrations.

Recommendations

We recommend that you use agent-based discovery for non-VMware environments, and whenever you want to collect information about network dependencies. You can run agent-based and agentless discovery simultaneously. Use agentless discovery to complete the initial infrastructure assessment quickly, and then install agents on select hosts to collect additional information.

Working With This Guide

This API reference provides descriptions, syntax, and usage examples for each of the actions and data types for Application Discovery Service. The topic for each action shows the API request parameters and the response. Alternatively, you can use one of the AWS SDKs to access an API that is tailored to the programming language or platform that you're using. For more information, see AWS SDKs.

- Remember that you must set your Migration Hub home region before you call any of these APIs.
- You must make API calls for write actions (create, notify, associate, disassociate, import, or put) while in your home region, or a `HomeRegionNotSetException` error is returned.
- API calls for read actions (list, describe, stop, and delete) are permitted outside of your home region.
- Although it is unlikely, the Migration Hub home region could change. If you call APIs outside the home region, an `InvalidInputException` is returned.
- You must call `GetHomeRegion` to obtain the latest Migration Hub home region.

This guide is intended for use with the AWS Application Discovery Service User Guide.

All data is handled according to the AWS Privacy Policy. You can operate Application Discovery Service offline to inspect collected data before it is shared with the service.

Usage

```
applicationdiscoveryservice(config = list())
```

Arguments

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

Value

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

Service syntax

```
svc <- applicationdiscoveryservice(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
```

```
      session_token = "string"
    ),
    profile = "string"
  ),
  endpoint = "string",
  region = "string"
 )
)
```

Operations

associate_configuration_items_to_application	Associates one or more configuration items with an application
batch_delete_import_data	Deletes one or more import tasks, each identified by their import ID
create_application	Creates an application with the given name and description
create_tags	Creates one or more tags for configuration items
delete_applications	Deletes a list of applications and their associations with configuration it
delete_tags	Deletes the association between configuration items and one or more tag
describe_agents	Lists agents or connectors as specified by ID or other filters
describe_configurations	Retrieves attributes for a list of configuration item IDs
describe_continuous_exports	Lists exports as specified by ID
describe_export_configurations	DescribeExportConfigurations is deprecated
describe_export_tasks	Retrieve status of one or more export tasks
describe_import_tasks	Returns an array of import tasks for your account, including status infor
describe_tags	Retrieves a list of configuration items that have tags as specified by the
disassociate_configuration_items_from_application	Disassociates one or more configuration items from an application
export_configurations	Deprecated
get_discovery_summary	Retrieves a short summary of discovered assets
list_configurations	Retrieves a list of configuration items as specified by the value passed to
list_server_neighbors	Retrieves a list of servers that are one network hop away from a specifie
start_continuous_export	Start the continuous flow of agent's discovered data into Amazon Athen
start_data_collection_by_agent_ids	Instructs the specified agents or connectors to start collecting data
start_export_task	Begins the export of discovered data to an S3 bucket
start_import_task	Starts an import task, which allows you to import details of your on-pre
stop_continuous_export	Stop the continuous flow of agent's discovered data into Amazon Athen
stop_data_collection_by_agent_ids	Instructs the specified agents or connectors to stop collecting data
update_application	Updates metadata about an application

Examples

```
## Not run:
svc <- applicationdiscoveryservice()
svc$associate_configuration_items_to_application(
  Foo = 123
)

## End(Not run)
```

databasemigrationservice *AWS Database Migration Service*

Description

AWS Database Migration Service (AWS DMS) can migrate your data to and from the most widely used commercial and open-source databases such as Oracle, PostgreSQL, Microsoft SQL Server, Amazon Redshift, MariaDB, Amazon Aurora, MySQL, and SAP Adaptive Server Enterprise (ASE). The service supports homogeneous migrations such as Oracle to Oracle, as well as heterogeneous migrations between different database platforms, such as Oracle to MySQL or SQL Server to PostgreSQL.

For more information about AWS DMS, see What Is AWS Database Migration Service? in the *AWS Database Migration User Guide.*

Usage

```
databasemigrationservice(config = list())
```

Arguments

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

Value

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

Service syntax

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

Operations

add_tags_to_resource	Adds metadata tags to an AWS DMS resource, including replication insta
apply_pending_maintenance_action	Applies a pending maintenance action to a resource (for example, to a rep
cancel_replication_task_assessment_run	Cancels a single premigration assessment run
create_endpoint	Creates an endpoint using the provided settings
create_event_subscription	Creates an AWS DMS event notification subscription
create_replication_instance	Creates the replication instance using the specified parameters
create_replication_subnet_group	Creates a replication subnet group given a list of the subnet IDs in a VPC
create_replication_task	Creates a replication task using the specified parameters
delete_certificate	Deletes the specified certificate
delete_connection	Deletes the connection between a replication instance and an endpoint
delete_endpoint	Deletes the specified endpoint
delete_event_subscription	Deletes an AWS DMS event subscription
delete_replication_instance	Deletes the specified replication instance
delete_replication_subnet_group	Deletes a subnet group
delete_replication_task	Deletes the specified replication task
delete_replication_task_assessment_run	Deletes the record of a single premigration assessment run
describe_account_attributes	Lists all of the AWS DMS attributes for a customer account
describe_applicable_individual_assessments	Provides a list of individual assessments that you can specify for a new pr
describe_certificates	Provides a description of the certificate
describe_connections	Describes the status of the connections that have been made between the
describe_endpoints	Returns information about the endpoints for your account in the current r
describe_endpoint_types	Returns information about the type of endpoints available
describe_event_categories	Lists categories for all event source types, or, if specified, for a specified s
describe_events	Lists events for a given source identifier and source type
describe_event_subscriptions	Lists all the event subscriptions for a customer account
describe_orderable_replication_instances	Returns information about the replication instance types that can be creat
describe_pending_maintenance_actions	For internal use only
describe_refresh_schemas_status	Returns the status of the RefreshSchemas operation
describe_replication_instances	Returns information about replication instances for your account in the cu
describe_replication_instance_task_logs	Returns information about the task logs for the specified task
describe_replication_subnet_groups	Returns information about the replication subnet groups
describe_replication_task_assessment_results	Returns the task assessment results from Amazon S3
describe_replication_task_assessment_runs	Returns a paginated list of premigration assessment runs based on filter s
describe_replication_task_individual_assessments	Returns a paginated list of individual assessments based on filter settings
describe_replication_tasks	Returns information about replication tasks for your account in the curren
describe_schemas	Returns information about the schema for the specified endpoint
describe_table_statistics	Returns table statistics on the database migration task, including table nam
import_certificate	Uploads the specified certificate
list_tags_for_resource	Lists all metadata tags attached to an AWS DMS resource, including repl
modify_endpoint	Modifies the specified endpoint
modify_event_subscription	Modifies an existing AWS DMS event notification subscription
modify_replication_instance	Modifies the replication instance to apply new settings
modify_replication_subnet_group	Modifies the settings for the specified replication subnet group
modify_replication_task	Modifies the specified replication task
move_replication_task	Moves a replication task from its current replication instance to a differen
reboot_replication_instance	Reboots a replication instance

refresh_schemas	Populates the schema for the specified endpoint
reload_tables	Reloads the target database table with the source data
remove_tags_from_resource	Removes metadata tags from an AWS DMS resource, including replicatio
start_replication_task	Starts the replication task
start_replication_task_assessment	Starts the replication task assessment for unsupported data types in the so
start_replication_task_assessment_run	Starts a new premigration assessment run for one or more individual asse
stop_replication_task	Stops the replication task
test_connection	Tests the connection between the replication instance and the endpoint

Examples

```
## Not run:
svc <- databasemigrationservice()
# Adds metadata tags to an AWS DMS resource, including replication
# instance, endpoint, security group, and migration task. These tags can
# also be used with cost allocation reporting to track cost associated
# with AWS DMS resources, or used in a Condition statement in an IAM
# policy for AWS DMS.
svc$add_tags_to_resource(
 ResourceArn = "arn:aws:dms:us-east-1:123456789012:endpoint:ASXWXJZLNWNT5HTWCGV2BUJQ7E",
  Tags = list(
    list(
      Key = "Acount",
      Value = "1633456"
    )
  )
)

## End(Not run)
```

`datasync` *AWS DataSync*

Description

AWS DataSync is a managed data transfer service that makes it simpler for you to automate moving data between on-premises storage and Amazon Simple Storage Service (Amazon S3) or Amazon Elastic File System (Amazon EFS).

This API interface reference for AWS DataSync contains documentation for a programming interface that you can use to manage AWS DataSync.

Usage

```
datasync(config = list())
```

Arguments

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

Value

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

Service syntax

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

Operations

cancel_task_execution	Cancels execution of a task
create_agent	Activates an AWS DataSync agent that you have deployed on your host
create_location_efs	Creates an endpoint for an Amazon EFS file system
create_location_fsx_windows	Creates an endpoint for an Amazon FSx for Windows file system
create_location_nfs	Defines a file system on a Network File System (NFS) server that can be read from or writ
create_location_object_storage	Creates an endpoint for a self-managed object storage bucket
create_location_s3	Creates an endpoint for an Amazon S3 bucket
create_location_smb	Defines a file system on a Server Message Block (SMB) server that can be read from or w
create_task	Creates a task
delete_agent	Deletes an agent
delete_location	Deletes the configuration of a location used by AWS DataSync
delete_task	Deletes a task
describe_agent	Returns metadata such as the name, the network interfaces, and the status (that is, whether
describe_location_efs	Returns metadata, such as the path information about an Amazon EFS location
describe_location_fsx_windows	Returns metadata, such as the path information about an Amazon FSx for Windows locati
describe_location_nfs	Returns metadata, such as the path information, about an NFS location
describe_location_object_storage	Returns metadata about a self-managed object storage server location
describe_location_s3	Returns metadata, such as bucket name, about an Amazon S3 bucket location
describe_location_smb	Returns metadata, such as the path and user information about an SMB location
describe_task	Returns metadata about a task

describe_task_execution	Returns detailed metadata about a task that is being executed
list_agents	Returns a list of agents owned by an AWS account in the AWS Region specified in the req
list_locations	Returns a list of source and destination locations
list_tags_for_resource	Returns all the tags associated with a specified resource
list_task_executions	Returns a list of executed tasks
list_tasks	Returns a list of all the tasks
start_task_execution	Starts a specific invocation of a task
tag_resource	Applies a key-value pair to an AWS resource
untag_resource	Removes a tag from an AWS resource
update_agent	Updates the name of an agent
update_task	Updates the metadata associated with a task
update_task_execution	Updates execution of a task

Examples

```
## Not run:
svc <- datasync()
svc$cancel_task_execution(
  Foo = 123
)

## End(Not run)
```

`importexport` *AWS Import/Export*

Description

AWS Import/Export Service AWS Import/Export accelerates transferring large amounts of data between the AWS cloud and portable storage devices that you mail to us. AWS Import/Export transfers data directly onto and off of your storage devices using Amazon's high-speed internal network and bypassing the Internet. For large data sets, AWS Import/Export is often faster than Internet transfer and more cost effective than upgrading your connectivity.

Usage

```
importexport(config = list())
```

Arguments

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

Value

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

Service syntax

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

Operations

cancel_job	This operation cancels a specified job
create_job	This operation initiates the process of scheduling an upload or download of your data
get_shipping_label	This operation generates a pre-paid UPS shipping label that you will use to ship your device to AWS for
get_status	This operation returns information about a job, including where the job is in the processing pipeline, the
list_jobs	This operation returns the jobs associated with the requester
update_job	You use this operation to change the parameters specified in the original manifest file by supplying a nev

Examples

```
## Not run:
svc <- importexport()
svc$cancel_job(
  Foo = 123
)

## End(Not run)
```

migrationhub *AWS Migration Hub*

Description

The AWS Migration Hub API methods help to obtain server and application migration status and integrate your resource-specific migration tool by providing a programmatic interface to Migration Hub.

Remember that you must set your AWS Migration Hub home region before you call any of these APIs, or a `HomeRegionNotSetException` error will be returned. Also, you must make the API calls while in your home region.

Usage

```
migrationhub(config = list())
```

Arguments

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

Value

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

Service syntax

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

Operations

associate_created_artifact	Associates a created artifact of an AWS cloud resource, the target receiving the migration
associate_discovered_resource	Associates a discovered resource ID from Application Discovery Service with a migration
create_progress_update_stream	Creates a progress update stream which is an AWS resource used for access control as we
delete_progress_update_stream	Deletes a progress update stream, including all of its tasks, which was previously created
describe_application_state	Gets the migration status of an application
describe_migration_task	Retrieves a list of all attributes associated with a specific migration task
disassociate_created_artifact	Disassociates a created artifact of an AWS resource with a migration task performed by a
disassociate_discovered_resource	Disassociate an Application Discovery Service discovered resource from a migration task
import_migration_task	Registers a new migration task which represents a server, database, etc
list_application_states	Lists all the migration statuses for your applications
list_created_artifacts	Lists the created artifacts attached to a given migration task in an update stream
list_discovered_resources	Lists discovered resources associated with the given MigrationTask
list_migration_tasks	Lists all, or filtered by resource name, migration tasks associated with the user account ma
list_progress_update_streams	Lists progress update streams associated with the user account making this call
notify_application_state	Sets the migration state of an application
notify_migration_task_state	Notifies Migration Hub of the current status, progress, or other detail regarding a migratio
put_resource_attributes	Provides identifying details of the resource being migrated so that it can be associated in t

Examples

```
## Not run:
svc <- migrationhub()
svc$associate_created_artifact(
  Foo = 123
)

## End(Not run)
```

`sms` *AWS Server Migration Service*

Description

AWS Server Migration Service (AWS SMS) makes it easier and faster for you to migrate your on-premises workloads to AWS. To learn more about AWS SMS, see the following resources:

- AWS Server Migration Service product page
- AWS Server Migration Service User Guide

Usage

```
sms(config = list())
```

Arguments

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

Value

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

Service syntax

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

Operations

create_app	Creates an application
create_replication_job	Creates a replication job
delete_app	Deletes the specified application
delete_app_launch_configuration	Deletes the launch configuration for the specified application
delete_app_replication_configuration	Deletes the replication configuration for the specified application
delete_app_validation_configuration	Deletes the validation configuration for the specified application
delete_replication_job	Deletes the specified replication job
delete_server_catalog	Deletes all servers from your server catalog
disassociate_connector	Disassociates the specified connector from AWS SMS
generate_change_set	Generates a target change set for a currently launched stack and writes it to an Amazo
generate_template	Generates an AWS CloudFormation template based on the current launch configuratio
get_app	Retrieve information about the specified application
get_app_launch_configuration	Retrieves the application launch configuration associated with the specified applicatio
get_app_replication_configuration	Retrieves the application replication configuration associated with the specified applic
get_app_validation_configuration	Retrieves information about a configuration for validating an application
get_app_validation_output	Retrieves output from validating an application
get_connectors	Describes the connectors registered with the AWS SMS
get_replication_jobs	Describes the specified replication job or all of your replication jobs
get_replication_runs	Describes the replication runs for the specified replication job
get_servers	Describes the servers in your server catalog

import_app_catalog	Allows application import from AWS Migration Hub
import_server_catalog	Gathers a complete list of on-premises servers
launch_app	Launches the specified application as a stack in AWS CloudFormation
list_apps	Retrieves summaries for all applications
notify_app_validation_output	Provides information to AWS SMS about whether application validation is successful
put_app_launch_configuration	Creates or updates the launch configuration for the specified application
put_app_replication_configuration	Creates or updates the replication configuration for the specified application
put_app_validation_configuration	Creates or updates a validation configuration for the specified application
start_app_replication	Starts replicating the specified application by creating replication jobs for each server
start_on_demand_app_replication	Starts an on-demand replication run for the specified application
start_on_demand_replication_run	Starts an on-demand replication run for the specified replication job
stop_app_replication	Stops replicating the specified application by deleting the replication job for each serv
terminate_app	Terminates the stack for the specified application
update_app	Updates the specified application
update_replication_job	Updates the specified settings for the specified replication job

Examples

```
## Not run:
svc <- sms()
svc$create_app(
  Foo = 123
)

## End(Not run)
```

`snowball` *Amazon Import/Export Snowball*

Description

AWS Snow Family is a petabyte-scale data transport solution that uses secure devices to transfer large amounts of data between your on-premises data centers and Amazon Simple Storage Service (Amazon S3). The Snow commands described here provide access to the same functionality that is available in the AWS Snow Family Management Console, which enables you to create and manage jobs for a Snow device. To transfer data locally with a Snow device, you'll need to use the Snowball Edge client or the Amazon S3 API Interface for Snowball or AWS OpsHub for Snow Family. For more information, see the User Guide.

Usage

```
snowball(config = list())
```

Arguments

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

Value

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

Service syntax

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

Operations

cancel_cluster	Cancels a cluster job
cancel_job	Cancels the specified job
create_address	Creates an address for a Snow device to be shipped to
create_cluster	Creates an empty cluster
create_job	Creates a job to import or export data between Amazon S3 and your on-premises data cente
create_return_shipping_label	Creates a shipping label that will be used to return the Snow device to AWS
describe_address	Takes an AddressId and returns specific details about that address in the form of an Address
describe_addresses	Returns a specified number of ADDRESS objects
describe_cluster	Returns information about a specific cluster including shipping information, cluster status, a
describe_job	Returns information about a specific job including shipping information, job status, and othe
describe_return_shipping_label	Information on the shipping label of a Snow device that is being returned to AWS
get_job_manifest	Returns a link to an Amazon S3 presigned URL for the manifest file associated with the spe
get_job_unlock_code	Returns the UnlockCode code value for the specified job
get_snowball_usage	Returns information about the Snow Family service limit for your account, and also the num
get_software_updates	Returns an Amazon S3 presigned URL for an update file associated with a specified JobId
list_cluster_jobs	Returns an array of JobListEntry objects of the specified length
list_clusters	Returns an array of ClusterListEntry objects of the specified length
list_compatible_images	This action returns a list of the different Amazon EC2 Amazon Machine Images (AMIs) tha
list_jobs	Returns an array of JobListEntry objects of the specified length
update_cluster	While a cluster's ClusterState value is in the AwaitingQuorum state, you can update some o
update_job	While a job's JobState value is New, you can update some of the information associated wit
update_job_shipment_state	Updates the state when a the shipment states changes to a different state

Examples

```
## Not run:
svc <- snowball()
# This operation cancels a cluster job. You can only cancel a cluster job
# while it's in the AwaitingQuorum status.
svc$cancel_cluster(
  ClusterId = "CID123e4567-e89b-12d3-a456-426655440000"
)

## End(Not run)
```

---

transfer *AWS Transfer Family*

Description

AWS Transfer Family is a fully managed service that enables the transfer of files over the File Transfer Protocol (FTP), File Transfer Protocol over SSL (FTPS), or Secure Shell (SSH) File Transfer Protocol (SFTP) directly into and out of Amazon Simple Storage Service (Amazon S3). AWS helps you seamlessly migrate your file transfer workflows to AWS Transfer Family by integrating with existing authentication systems, and providing DNS routing with Amazon Route 53 so nothing changes for your customers and partners, or their applications. With your data in Amazon S3, you can use it with AWS services for processing, analytics, machine learning, and archiving. Getting started with AWS Transfer Family is easy since there is no infrastructure to buy and set up.

Usage

```
transfer(config = list())
```

Arguments

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

Value

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

Service syntax

```
svc <- transfer(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
```

```
      session_token = "string"
    ),
    profile = "string"
  ),
  endpoint = "string",
  region = "string"
 )
)
```

Operations

create_server	Instantiates an autoscaling virtual server based on the selected file transfer protocol in AWS
create_user	Creates a user and associates them with an existing file transfer protocol-enabled server
delete_server	Deletes the file transfer protocol-enabled server that you specify
delete_ssh_public_key	Deletes a user's Secure Shell (SSH) public key
delete_user	Deletes the user belonging to a file transfer protocol-enabled server you specify
describe_security_policy	Describes the security policy that is attached to your file transfer protocol-enabled server
describe_server	Describes a file transfer protocol-enabled server that you specify by passing the ServerId paramete
describe_user	Describes the user assigned to the specific file transfer protocol-enabled server, as identified by its
import_ssh_public_key	Adds a Secure Shell (SSH) public key to a user account identified by a UserName value assigned t
list_security_policies	Lists the security policies that are attached to your file transfer protocol-enabled servers
list_servers	Lists the file transfer protocol-enabled servers that are associated with your AWS account
list_tags_for_resource	Lists all of the tags associated with the Amazon Resource Number (ARN) you specify
list_users	Lists the users for a file transfer protocol-enabled server that you specify by passing the ServerId p
start_server	Changes the state of a file transfer protocol-enabled server from OFFLINE to ONLINE
stop_server	Changes the state of a file transfer protocol-enabled server from ONLINE to OFFLINE
tag_resource	Attaches a key-value pair to a resource, as identified by its Amazon Resource Name (ARN)
test_identity_provider	If the IdentityProviderType of a file transfer protocol-enabled server is API_Gateway, tests whethe
untag_resource	Detaches a key-value pair from a resource, as identified by its Amazon Resource Name (ARN)
update_server	Updates the file transfer protocol-enabled server's properties after that server has been created
update_user	Assigns new properties to a user

Examples

```
## Not run:
svc <- transfer()
svc$create_server(
  Foo = 123
)

## End(Not run)
```

Index