Package 'aws.kms'

April 14, 2020

Title 'AWS Key Management Service' Client Package

Version 0.1.4

Description Client package for the 'AWS Key Management Service' <https://aws.amazon.com/kms/>, a cloud service for managing encryption keys.

License GPL (>= 2)

URL https://github.com/cloudyr/aws.kms

BugReports https://github.com/cloudyr/aws.kms/issues

Imports httr, jsonlite, base64enc, aws.signature (>= 0.4.0)

Encoding UTF-8

RoxygenNote 7.1.0

NeedsCompilation no

Author Thomas J. Leeper [aut] (<https://orcid.org/0000-0003-4097-6326>), Simon Urbanek [cre, ctb]

Maintainer Simon Urbanek <simon.urbanek@R-project.org>

Repository CRAN

Date/Publication 2020-04-14 08:40:03 UTC

R topics documented:

aws.kms-package *aws.kms*

Description

AWS Key Management Service (KMS) Client.

Details

This is a client for the AWS Key Management Service (KMS), which can be used to create and manage encryption keys used by AWS services or to setup a secure HTTP-based encryption service using `encrypt` and `decrypt`. KMS is also used natively by other AWS services.

Author(s)

Thomas J. Leeper <thosjleeper@gmail.com>

References

https://docs.aws.amazon.com/kms/latest/developerguide/overview.html https://docs.aws.amazon.com/kms/latest/APIReference/Welcome.html

See Also

`create_kms_key`, `list_kms_keys`, `generate_blob`, `encrypt`

create_kms_alias *Create/Delete KMS Key Alias*

Description

Manage KMS key aliases.

Usage

```
create_kms_alias(key, alias, ...)

delete_kms_alias(alias, ...)

update_kms_alias(key, alias, ...)

list_kms_aliases(n, marker, ...)
```

Arguments

key	A character string specifying a key ID, Amazon Resource Name (ARN), alias name, or alias ARN. When using an alias name, prefix it with "alias/".
alias	A character string specifying an alias name.
...	Additional arguments passed to kmsHTTP.
n	For list_kms_alises, an integer specifying a number of keys to return (for pagination).
marker	For list_kms_alises, a pagination marker.

Details

create_kms_alias creates an alias for KMS key, which can be used in place of the KeyId or ARN. A given key can have multiple aliases. delete_kms_alias deletes an named alias. update_kms_alias reassigns an alias to a new key.

See Also

create_kms_key, delete_kms_key, encrypt

---

create_kms_key *Create/Update/Retrieve/Delete Encryption Key*

Description

Create/update/retrieve/delete a KMS encryption key

Usage

```
create_kms_key(
  description = NULL,
  origin = c("AWS_KMS", "EXTERNAL"),
  usage = "ENCRYPT_DECRYPT",
  ...
)

update_kms_key(key, description, ...)

get_kms_key(key, ...)

delete_kms_key(key, delay = 7, ...)

undelete_kms_key(key, ...)
```

Arguments

description	Optionally, a character string describing the key. This can be updated later using update_kms_key. An alias for the key, which can be used in lieu of the KeyId in subsequent calls can be set with create_kms_alias.
origin	A character string specifying the origin. Default is "AWS_KMS". If "EXTERNAL", use put_kms_material to add a key created using other infrastructure. See https://docs.aws.amazon.com/kms/latest/developerguide/importing-keys.html for details.
usage	Ignored.
...	Additional arguments passed to kmsHTTP.
key	A character string specifying a key ID, Amazon Resource Name (ARN), alias name, or alias ARN. When using an alias name, prefix it with "alias/".
delay	An integer specifying a number of delays to wait before deleting key. Minimum 7 and maximum 30.

Value

create_kms_key and get_kms_key return a list of class "aws_kms_key". delete_kms_key and undelete_kms_key return a logical.

See Also

list_kms_keys, create_kms_alias, disable_kms_key, encrypt

Examples

```
## Not run:
  # create key
  k <- create_kms_key(description = "example")

  # get key
  get_kms_key(k)

  # delete in 30 days
  delete_kms_key(k, delay = 30)

## End(Not run)
```

---

enable_kms_key *Enable/Disable Encryption Key*

Description

Enable or disable a KMS encryption key

Usage

```
enable_kms_key(key, ...)

disable_kms_key(key, ...)
```

Arguments

key	A character string specifying a key ID, Amazon Resource Name (ARN), alias name, or alias ARN. When using an alias name, prefix it with "alias/".
...	Additional arguments passed to kmsHTTP.

See Also

create_kms_key, list_kms_keys

Examples

```
## Not run:
  # create key
  k <- create_kms_key(description = "example")

  # disable key
  disable_kms_key(k)

  # enable key
  enable_kms_key(k)

  # delete in 7 days
  delete_kms_key(k)

## End(Not run)
```

---

enable_kms_rotation *Enable/Disable Key Rotation*

Description

Enable or disable a encryption key rotation

Usage

```
enable_kms_rotation(key, ...)

disable_kms_rotation(key, ...)

get_kms_rotation(key, ...)
```

Arguments

key	A character string specifying a key ID, Amazon Resource Name (ARN), alias name, or alias ARN. When using an alias name, prefix it with "alias/".
...	Additional arguments passed to kmsHTTP.

See Also

create_kms_key, list_kms_keys

Examples

```
## Not run:
  # create key
  k <- create_kms_key(description = "example")

  # enable rotation
  enable_kms_rotation(k)

  # disable rotation
  disable_kms_rotation(k)

  # confirm rotation is disabled
  get_kms_rotation(k)

  # delete in 7 days
  delete_kms_key(k)

## End(Not run)
```

---

encrypt *Perform encryption/decryption*

Description

Encrypt plain text into ciphertext, or the reverse

Usage

```
encrypt(text, key, encode = TRUE, ...)

decrypt(text, key, encode = TRUE, ...)

reencrypt(text, key, encode = TRUE, ...)
```

Arguments

text	For encrypt, a character string specifying up to 4 kilobytes of data to be encrypted using the specified key. For decrypt, ciphertext of maximum 6144 bytes.
key	A character string specifying a key ID, Amazon Resource Name (ARN), alias name, or alias ARN. When using an alias name, prefix it with "alias/".
encode	A logical specifying whether to base 64 encode text.
...	Additional arguments passed to kmsHTTP.

Details

encrypt encrypts source text using a KMS key. decrypt reverses this process using the same key. reencrypt reencrypts an (encrypted) ciphertext using a new key. The purpose of these functions, according to AWS, to is encrypt and decrypt data keys (of the source created with generate_data_key) rather than general purpose encryption given the relatively low upper limit on the size of text.

Value

encrypt returns a base64-encoded binary object as a character string.

See Also

create_kms_key, generate_data_key, generate_blob

Examples

```
## Not run:
  # create a key
  k <- create_kms_key()

  # encrypt
  tmp <- tempfile()
  cat("example test", file = tmp)
  (etext <- encrypt(tmp, k))

  # decrypt
  (dtext <- decrypt(etext, k, encode = FALSE))
  if (require("base64enc")) {
    rawToChar(base64enc::base64decode(dtext))
  }

  # cleanup
  delete_kms_key(k)

## End(Not run)
```

generate_blob *Generate Random Blob*

Description

Generate a random byte string

Usage

```
generate_blob(bytes = 1, ...)
```

Arguments

bytes	An integer specifying a number of bytes between 1 and 1024.
...	Additional arguments passed to kmsHTTP.

Details

create_kms_alias creates an alias for KMS key, which can be used in place of the KeyId or ARN. A given key can have multiple aliases. delete_kms_alias deletes an named alias. update_kms_alias reassigns an alias to a new key.

Value

A base64-encoded character string.

See Also

create_kms_key, encrypt

Examples

```
## Not run:
  b <- generate_blob()
  if (require("base64enc")) {
     base64enc::base64decode(b)
  }

## End(Not run)
```

`generate_data_key` *Generate data keys*

Description

Generate data keys for local encryption

Usage

```
generate_data_key(key, spec = c("AES_256", "AES_128"), plaintext = TRUE, ...)
```

Arguments

`key`	A character string specifying a key ID, Amazon Resource Name (ARN), alias name, or alias ARN. When using an alias name, prefix it with "alias/".
`spec`	A character string specifying the length of the data encryption key, either "AES_256" or "AES_128".
`plaintext`	A logical indicating whether to return the data key in plain text, as well as in encrypted form.
`...`	Additional arguments passed to `kmsHTTP`.

Details

This function generates and returns a "data key" for use in local encrption. The suggested workflow from AWS is to encrypt, do the following:

1. Use this operation (`generate_data_key`) to get a data encryption key.
2. Use the plaintext data encryption key (returned in the Plaintext field of the response) to encrypt data locally, then erase the plaintext data key from memory.
3. Store the encrypted data key (returned in the CiphertextBlob field of the response) alongside the locally encrypted data.

Then to decrypt locally:

1. Use `decrypt` to decrypt the encrypted data key into a plaintext copy of the data key.
2. Use the plaintext data key to decrypt data locally, then erase the plaintext data key from memory.

Value

`encrypt` returns a base64-encoded binary object as a character string.

References

https://docs.aws.amazon.com/kms/latest/APIReference/API_GenerateDataKey.html

See Also

create_kms_key, generate_blob

Examples

```
## Not run:
  # create a (CMK) key
  k <- create_kms_key()

  # generate a data key for local encryption
  datakey <- generate_data_key(key = k)

  ## encrypt something locally using datakey$Plaintext
  ## then delete the plaintext key
  datakey$Plaintext <- NULL

  # decrypt the encrypted data key
  datakey$Plaintext <- decrypt(datakey$CiphertextBlob, k, encode = FALSE)
  ## then use this to decrypt locally

  # cleanup
  delete_kms_key(k)

## End(Not run)
```

---

kmsHTTP *Execute AWS KMS API Request*

Description

This is the workhorse function to execute calls to the KMS API.

Usage

```
kmsHTTP(
  action,
  query = list(),
  headers = list(),
  body = NULL,
  verbose = getOption("verbose", FALSE),
  region = Sys.getenv("AWS_DEFAULT_REGION", "us-east-1"),
  key = NULL,
  secret = NULL,
  session_token = NULL,
  ...
)
```

Arguments

action	A character string specifying the API action to take
query	An optional named list containing query string parameters and their character values.
headers	A list of headers to pass to the HTTP request.
body	A request body
verbose	A logical indicating whether to be verbose. Default is given by options("verbose").
region	A character string specifying an AWS region. See locate_credentials.
key	A character string specifying an AWS Access Key. See locate_credentials.
secret	A character string specifying an AWS Secret Key. See locate_credentials.
session_token	Optionally, a character string specifying an AWS temporary Session Token to use in signing a request. See locate_credentials.
...	Additional arguments passed to GET.

Details

This function constructs and signs a KMS API request and returns the results thereof, or relevant debugging information in the case of error.

Value

If successful, a named list. Otherwise, a data structure of class "aws-error" containing any error message(s) from AWS and information about the request attempt.

Author(s)

Thomas J. Leeper

---

list_kms_keys *List Encryption Keys*

Description

List encryption keys in KMS

Usage

```
list_kms_keys(n = 100, marker = NULL, ...)
```

Arguments

n	An integer specifying a number of keys to return (for pagination).
marker	A pagination marker.
...	Additional arguments passed to kmsHTTP.

Value

A data frame

See Also

get_kms_key, create_kms_key, delete_kms_key

Examples

```
## Not run:
  list_kms_keys()

## End(Not run)
```

---

put_kms_material *Put/Delete KMS Key Material*

Description

Manage key material for "external" keys.

Usage

```
put_kms_material(key, material, token, expires = TRUE, valid_to = NULL, ...)

delete_kms_material(key, ...)

get_material_parameters(
  key,
  algorithm = c("RSAES_PKCS1_V1_5", "RSAES_OAEP_SHA_1", "RSAES_OAEP_SHA_256"),
  spec = "RSA_2048",
  ...
)
```

Arguments

key	A character string specifying a key ID, Amazon Resource Name (ARN), alias name, or alias ARN. When using an alias name, prefix it with "alias/".
material	A character string specifying the base64-encoded key material (encrypted according to parameters returned by get_material_parameters).
token	A character string returned in get_material_parameters()$ImportToken.
expires	Optionally, a logical indicating whether the key material expires. If TRUE (the default), valid_to is required.
valid_to	Optionally (if expires = TRUE), a number specifying when the key material expires.
...	Additional arguments passed to kmsHTTP.

algorithm	A character string specifying an encryption algorithm used to encrypt the key material.
spec	Ignored.

Details

put_kms_material adds key material to an "external" KMS key, which can be created using create_kms_key. The import requires delete_kms_material deletes the imported material (but not the key itself).

References

https://docs.aws.amazon.com/kms/latest/developerguide/importing-keys-encrypt-key-material.html

See Also

create_kms_key

Index