aws

API Reference

AWS Payment Cryptography Control Plane

API Version 2021-09-14

AWS Payment Cryptography Control Plane: API Reference

Table of Contents

Welcome

AWS Payment Cryptography Control Plane APIs manage encryption keys for use during payment-related cryptographic operations. You can create, import, export, share, manage, and delete keys. You can also manage AWS Identity and Access Management (IAM) policies for keys. For more information, see Identity and access management in the *AWS Payment Cryptography User Guide.*

To use encryption keys for payment-related transaction processing and associated cryptographic operations, you use the AWS Payment Cryptography Data Plane. You can perform actions like encrypt, decrypt, generate, and verify payment-related data.

All AWS Payment Cryptography API calls must be signed and transmitted using Transport Layer Security (TLS). We recommend you always use the latest supported TLS version for logging API requests.

AWS Payment Cryptography supports AWS CloudTrail for control plane operations, a service that logs AWS API calls and related events for your AWS account and delivers them to an Amazon S3 bucket you specify. By using the information collected by CloudTrail, you can determine what requests were made to AWS Payment Cryptography, who made the request, when it was made, and so on. If you don't configure a trail, you can still view the most recent events in the CloudTrail console. For more information, see the AWS CloudTrail User Guide.

This document was last published on July 25, 2025.

Actions

The following actions are supported:

- CreateAlias
- CreateKey
- DeleteAlias
- DeleteKey
- ExportKey
- GetAlias
- GetKey
- GetParametersForExport
- GetParametersForImport
- GetPublicKeyCertificate
- ImportKey
- ListAliases
- ListKeys
- ListTagsForResource
- RestoreKey
- StartKeyUsage
- StopKeyUsage
- TagResource
- UntagResource
- UpdateAlias

CreateAlias

Creates an *alias*, or a friendly name, for an AWS Payment Cryptography key. You can use an alias to identify a key in the console and when you call cryptographic operations such as EncryptData or DecryptData.

You can associate the alias with any key in the same AWS Region. Each alias is associated with only one key at a time, but a key can have multiple aliases. You can't create an alias without a key. The alias must be unique in the account and AWS Region, but you can create another alias with the same name in a different AWS Region.

To change the key that's associated with the alias, call UpdateAlias. To delete the alias, call DeleteAlias. These operations don't affect the underlying key. To get the alias that you created, call ListAliases.

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- DeleteAlias
- GetAlias
- ListAliases
- UpdateAlias

Request Syntax

```
{
   "AliasName": "string",
   "KeyArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

AliasName

A friendly name that you can use to refer to a key. An alias must begin with `alias/` followed by a name, for example `alias/ExampleAlias`. It can contain only alphanumeric characters, forward slashes (/), underscores (_), and dashes (-).

> **Important**
>
> Don't include personal, confidential or sensitive information in this field. This field may be displayed in plaintext in AWS CloudTrail logs and other output.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 256.

Pattern: `alias/[a-zA-Z0-9/_-]+`

Required: Yes

KeyArn

The KeyARN of the key to associate with the alias.

Type: String

Length Constraints: Minimum length of 70. Maximum length of 150.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:key/[0-9a-zA-Z]{16,64}`

Required: No

Response Syntax

```
{
   "Alias": {
      "AliasName": "string",
      "KeyArn": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Alias

The alias for the key.

Type: Alias object

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

ConflictException

This request can cause an inconsistent state for the resource.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceQuotaExceededException

This request would cause a service quota to be exceeded.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateKey

Creates an AWS Payment Cryptography key, a logical representation of a cryptographic key, that is unique in your account and AWS Region. You use keys for cryptographic functions such as encryption and decryption.

In addition to the key material used in cryptographic operations, an AWS Payment Cryptography key includes metadata such as the key ARN, key usage, key origin, creation date, description, and key state.

When you create a key, you specify both immutable and mutable data about the key. The immutable data contains key attributes that define the scope and cryptographic operations that you can perform using the key, for example key class (example: `SYMMETRIC_KEY`), key algorithm (example: `TDES_2KEY`), key usage (example: `TR31_P0_PIN_ENCRYPTION_KEY`) and key modes of use (example: `Encrypt`). AWS Payment Cryptography binds key attributes to keys using key blocks when you store or export them. AWS Payment Cryptography stores the key contents wrapped and never stores or transmits them in the clear.

For information about valid combinations of key attributes, see Understanding key attributes in the *AWS Payment Cryptography User Guide*. The mutable data contained within a key includes usage timestamp and key deletion timestamp and can be modified after creation.

You can use the `CreateKey` operation to generate an ECC (Elliptic Curve Cryptography) key pair used for establishing an ECDH (Elliptic Curve Diffie-Hellman) key agreement between two parties. In the ECDH key agreement process, both parties generate their own ECC key pair with key usage K3 and exchange the public keys. Each party then use their private key, the received public key from the other party, and the key derivation parameters including key derivation function, hash algorithm, derivation data, and key algorithm to derive a shared key.

To maintain the single-use principle of cryptographic keys in payments, ECDH derived keys should not be used for multiple purposes, such as a `TR31_P0_PIN_ENCRYPTION_KEY` and `TR31_K1_KEY_BLOCK_PROTECTION_KEY`. When creating ECC key pairs in AWS Payment Cryptography you can optionally set the `DeriveKeyUsage` parameter, which defines the key usage bound to the symmetric key that will be derived using the ECC key pair.

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- DeleteKey

- GetKey
- ListKeys

Request Syntax

```
{
   "DeriveKeyUsage": "string",
   "Enabled": boolean,
   "Exportable": boolean,
   "KeyAttributes": {
      "KeyAlgorithm": "string",
      "KeyClass": "string",
      "KeyModesOfUse": {
         "Decrypt": boolean,
         "DeriveKey": boolean,
         "Encrypt": boolean,
         "Generate": boolean,
         "NoRestrictions": boolean,
         "Sign": boolean,
         "Unwrap": boolean,
         "Verify": boolean,
         "Wrap": boolean
      },
      "KeyUsage": "string"
   },
   "KeyCheckValueAlgorithm": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Request Parameters

The request accepts the following data in JSON format.

DeriveKeyUsage

The intended cryptographic usage of keys derived from the ECC key pair to be created.

After creating an ECC key pair, you cannot change the intended cryptographic usage of keys derived from it using ECDH.

Type: String

Valid Values: `TR31_B0_BASE_DERIVATION_KEY | TR31_C0_CARD_VERIFICATION_KEY | TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY | TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS | TR31_E1_EMV_MKEY_CONFIDENTIALITY | TR31_E2_EMV_MKEY_INTEGRITY | TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS | TR31_E5_EMV_MKEY_CARD_PERSONALIZATION | TR31_E6_EMV_MKEY_OTHER | TR31_K0_KEY_ENCRYPTION_KEY | TR31_K1_KEY_BLOCK_PROTECTION_KEY | TR31_M3_ISO_9797_3_MAC_KEY | TR31_M1_ISO_9797_1_MAC_KEY | TR31_M6_ISO_9797_5_CMAC_KEY | TR31_M7_HMAC_KEY | TR31_P0_PIN_ENCRYPTION_KEY | TR31_P1_PIN_GENERATION_KEY | TR31_V1_IBM3624_PIN_VERIFICATION_KEY | TR31_V2_VISA_PIN_VERIFICATION_KEY`

Required: No

Enabled

Specifies whether to enable the key. If the key is enabled, it is activated for use within the service. If the key is not enabled, then it is created but not activated. The default value is enabled.

Type: Boolean

Required: No

Exportable

Specifies whether the key is exportable from the service.

Type: Boolean

Required: Yes

KeyAttributes

The role of the key, the algorithm it supports, and the cryptographic operations allowed with the key. This data is immutable after the key is created.

Type: KeyAttributes object

Required: Yes

KeyCheckValueAlgorithm

The algorithm that AWS Payment Cryptography uses to calculate the key check value (KCV). It is used to validate the key integrity.

For TDES keys, the KCV is computed by encrypting 8 bytes, each with value of zero, with the key to be checked and retaining the 3 highest order bytes of the encrypted result. For AES keys, the KCV is computed using a CMAC algorithm where the input data is 16 bytes of zero and retaining the 3 highest order bytes of the encrypted result.

Type: String

Valid Values: `CMAC | ANSI_X9_24`

Required: No

Tags

Assigns one or more tags to the AWS Payment Cryptography key. Use this parameter to tag a key when it is created. To tag an existing AWS Payment Cryptography key, use the TagResource operation.

Each tag consists of a tag key and a tag value. Both the tag key and the tag value are required, but the tag value can be an empty (null) string. You can't have more than one tag on an AWS Payment Cryptography key with the same tag key.

> **Important**
>
> Don't include personal, confidential or sensitive information in this field. This field may be displayed in plaintext in AWS CloudTrail logs and other output.

> **Note**
>
> Tagging or untagging an AWS Payment Cryptography key can allow or deny permission to the key.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: No

Response Syntax

```
{
   "Key": {
      "CreateTimestamp": number,
      "DeletePendingTimestamp": number,
      "DeleteTimestamp": number,
      "DeriveKeyUsage": "string",
      "Enabled": boolean,
      "Exportable": boolean,
      "KeyArn": "string",
      "KeyAttributes": {
         "KeyAlgorithm": "string",
         "KeyClass": "string",
         "KeyModesOfUse": {
            "Decrypt": boolean,
            "DeriveKey": boolean,
            "Encrypt": boolean,
            "Generate": boolean,
            "NoRestrictions": boolean,
            "Sign": boolean,
            "Unwrap": boolean,
            "Verify": boolean,
            "Wrap": boolean
         },
         "KeyUsage": "string"
      },
      "KeyCheckValue": "string",
      "KeyCheckValueAlgorithm": "string",
      "KeyOrigin": "string",
      "KeyState": "string",
      "UsageStartTimestamp": number,
      "UsageStopTimestamp": number
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Key

The key material that contains all the key attributes.

Type: Key object

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

ConflictException

This request can cause an inconsistent state for the resource.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceQuotaExceededException

This request would cause a service quota to be exceeded.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteAlias

Deletes the alias, but doesn't affect the underlying key.

Each key can have multiple aliases. To get the aliases of all keys, use the UpdateAlias operation. To change the alias of a key, first use DeleteAlias to delete the current alias and then use CreateAlias to create a new alias. To associate an existing alias with a different key, call UpdateAlias.

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- CreateAlias
- GetAlias
- ListAliases
- UpdateAlias

Request Syntax

```
{
   "AliasName": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

AliasName

A friendly name that you can use to refer AWS Payment Cryptography key. This value must begin with `alias/` followed by a name, such as `alias/ExampleAlias`.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 256.

Pattern: `alias/[a-zA-Z0-9/_-]+`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

ConflictException

This request can cause an inconsistent state for the resource.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteKey

Deletes the key material and metadata associated with AWS Payment Cryptography key.

Key deletion is irreversible. After a key is deleted, you can't perform cryptographic operations using the key. For example, you can't decrypt data that was encrypted by a deleted AWS Payment Cryptography key, and the data may become unrecoverable. Because key deletion is destructive, AWS Payment Cryptography has a safety mechanism to prevent accidental deletion of a key. When you call this operation, AWS Payment Cryptography disables the specified key but doesn't delete it until after a waiting period set using `DeleteKeyInDays`. The default waiting period is 7 days. During the waiting period, the `KeyState` is `DELETE_PENDING`. After the key is deleted, the `KeyState` is `DELETE_COMPLETE`.

You should delete a key only when you are sure that you don't need to use it anymore and no other parties are utilizing this key. If you aren't sure, consider deactivating it instead by calling StopKeyUsage.

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- RestoreKey
- StartKeyUsage
- StopKeyUsage

Request Syntax

```
{
   "DeleteKeyInDays": number,
   "KeyIdentifier": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

DeleteKeyInDays

The waiting period for key deletion. The default value is seven days.

Type: Integer

Valid Range: Minimum value of 3. Maximum value of 180.

Required: No

KeyIdentifier

The KeyARN of the key that is scheduled for deletion.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 322.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:(key/[0-9a-zA-Z]{16,64}|alias/[a-zA-Z0-9/_-]+)$|^alias/[a-zA-Z0-9/_-]+`

Required: Yes

Response Syntax

```
{
   "Key": {
      "CreateTimestamp": number,
      "DeletePendingTimestamp": number,
      "DeleteTimestamp": number,
      "DeriveKeyUsage": "string",
      "Enabled": boolean,
      "Exportable": boolean,
      "KeyArn": "string",
      "KeyAttributes": {
         "KeyAlgorithm": "string",
         "KeyClass": "string",
         "KeyModesOfUse": {
            "Decrypt": boolean,
            "DeriveKey": boolean,
            "Encrypt": boolean,
            "Generate": boolean,
            "NoRestrictions": boolean,
            "Sign": boolean,
            "Unwrap": boolean,
            "Verify": boolean,
```

```
            "Wrap": boolean
         },
         "KeyUsage": "string"
      },
      "KeyCheckValue": "string",
      "KeyCheckValueAlgorithm": "string",
      "KeyOrigin": "string",
      "KeyState": "string",
      "UsageStartTimestamp": number,
      "UsageStopTimestamp": number
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Key

The KeyARN of the key that is scheduled for deletion.

Type: Key object

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

ConflictException

This request can cause an inconsistent state for the resource.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ExportKey

Exports a key from AWS Payment Cryptography.

AWS Payment Cryptography simplifies key exchange by replacing the existing paper-based approach with a modern electronic approach. With `ExportKey` you can export symmetric keys using either symmetric and asymmetric key exchange mechanisms. Using this operation, you can share your AWS Payment Cryptography generated keys with other service partners to perform cryptographic operations outside of AWS Payment Cryptography

For symmetric key exchange, AWS Payment Cryptography uses the ANSI X9 TR-31 norm in accordance with PCI PIN guidelines. And for asymmetric key exchange, AWS Payment Cryptography supports ANSI X9 TR-34 norm, RSA unwrap, and ECDH (Elliptic Curve Diffie-Hellman) key exchange mechanisms. Asymmetric key exchange methods are typically used to establish bi-directional trust between the two parties exhanging keys and are used for initial key exchange such as Key Encryption Key (KEK). After which you can export working keys using symmetric method to perform various cryptographic operations within AWS Payment Cryptography.

PCI requires specific minimum key strength of wrapping keys used to protect the keys being exchanged electronically. These requirements can change when PCI standards are revised. The rules specify that wrapping keys used for transport must be at least as strong as the key being protected. For more information on recommended key strength of wrapping keys and key exchange mechanism, see Importing and exporting keys in the *AWS Payment Cryptography User Guide*.

You can also use `ExportKey` functionality to generate and export an IPEK (Initial Pin Encryption Key) from AWS Payment Cryptography using either TR-31 or TR-34 export key exchange. IPEK is generated from BDK (Base Derivation Key) and `ExportDukptInitialKey` attribute KSN (`KeySerialNumber`). The generated IPEK does not persist within AWS Payment Cryptography and has to be re-generated each time during export.

For key exchange using TR-31 or TR-34 key blocks, you can also export optional blocks within the key block header which contain additional attribute information about the key. The `KeyVersion` within `KeyBlockHeaders` indicates the version of the key within the key block. Furthermore, `KeyExportability` within `KeyBlockHeaders` can be used to further restrict exportability of the key after export from AWS Payment Cryptography.

The `OptionalBlocks` contain the additional data related to the key. For information on data type that can be included within optional blocks, refer to ASC X9.143-2022.

> **Note**
>
> Data included in key block headers is signed but transmitted in clear text. Sensitive or confidential information should not be included in optional blocks. Refer to ASC X9.143-2022 standard for information on allowed data type.

To export initial keys (KEK) or IPEK using TR-34

Using this operation, you can export initial key using TR-34 asymmetric key exchange. You can only export KEK generated within AWS Payment Cryptography. In TR-34 terminology, the sending party of the key is called Key Distribution Host (KDH) and the receiving party of the key is called Key Receiving Device (KRD). During key export process, KDH is AWS Payment Cryptography which initiates key export and KRD is the user receiving the key.

To initiate TR-34 key export, the KRD must obtain an export token by calling GetParametersForExport. This operation also generates a key pair for the purpose of key export, signs the key and returns back the signing public key certificate (also known as KDH signing certificate) and root certificate chain. The KDH uses the private key to sign the the export payload and the signing public key certificate is provided to KRD to verify the signature. The KRD can import the root certificate into its Hardware Security Module (HSM), as required. The export token and the associated KDH signing certificate expires after 30 days.

Next the KRD generates a key pair for the the purpose of encrypting the KDH key and provides the public key cerificate (also known as KRD wrapping certificate) back to KDH. The KRD will also import the root cerificate chain into AWS Payment Cryptography by calling ImportKey for `RootCertificatePublicKey`. The KDH, AWS Payment Cryptography, will use the KRD wrapping cerificate to encrypt (wrap) the key under export and signs it with signing private key to generate a TR-34 WrappedKeyBlock. For more information on TR-34 key export, see section Exporting symmetric keys in the *AWS Payment Cryptography User Guide*.

Set the following parameters:

- `ExportAttributes`: Specify export attributes in case of IPEK export. This parameter is optional for KEK export.
- `ExportKeyIdentifier`: The `KeyARN` of the KEK or BDK (in case of IPEK) under export.
- `KeyMaterial`: Use `Tr34KeyBlock` parameters.

- `CertificateAuthorityPublicKeyIdentifier`: The `KeyARN` of the certificate chain that signed the KRD wrapping key certificate.
- `ExportToken`: Obtained from KDH by calling GetParametersForImport.
- `WrappingKeyCertificate`: The public key certificate in PEM format (base64 encoded) of the KRD wrapping key AWS Payment Cryptography uses for encryption of the TR-34 export payload. This certificate must be signed by the root certificate (CertificateAuthorityPublicKeyIdentifier) imported into AWS Payment Cryptography.

When this operation is successful, AWS Payment Cryptography returns the KEK or IPEK as a TR-34 WrappedKeyBlock.

To export initial keys (KEK) or IPEK using RSA Wrap and Unwrap

Using this operation, you can export initial key using asymmetric RSA wrap and unwrap key exchange method. To initiate export, generate an asymmetric key pair on the receiving HSM and obtain the public key certificate in PEM format (base64 encoded) for the purpose of wrapping and the root certifiate chain. Import the root certificate into AWS Payment Cryptography by calling ImportKey for `RootCertificatePublicKey`.

Next call `ExportKey` and set the following parameters:

- `CertificateAuthorityPublicKeyIdentifier`: The `KeyARN` of the certificate chain that signed wrapping key certificate.
- `KeyMaterial`: Set to `KeyCryptogram`.
- `WrappingKeyCertificate`: The public key certificate in PEM format (base64 encoded) obtained by the receiving HSM and signed by the root certificate (CertificateAuthorityPublicKeyIdentifier) imported into AWS Payment Cryptography. The receiving HSM uses its private key component to unwrap the WrappedKeyCryptogram.

When this operation is successful, AWS Payment Cryptography returns the WrappedKeyCryptogram.

To export working keys or IPEK using TR-31

Using this operation, you can export working keys or IPEK using TR-31 symmetric key exchange. In TR-31, you must use an initial key such as KEK to encrypt or wrap the key under export. To establish a KEK, you can use CreateKey or ImportKey.

Set the following parameters:

- `ExportAttributes`: Specify export attributes in case of IPEK export. This parameter is optional for KEK export.
- `ExportKeyIdentifier`: The `KeyARN` of the KEK or BDK (in case of IPEK) under export.
- `KeyMaterial`: Use `Tr31KeyBlock` parameters.

To export working keys using ECDH

You can also use ECDH key agreement to export working keys in a TR-31 keyblock, where the wrapping key is an ECDH derived key.

To initiate a TR-31 key export using ECDH, both sides must create an ECC key pair with key usage K3 and exchange public key certificates. In AWS Payment Cryptography, you can do this by calling `CreateKey`. If you have not already done so, you must import the CA chain that issued the receiving public key certificate by calling `ImportKey` with input `RootCertificatePublicKey` for root CA or `TrustedPublicKey` for intermediate CA. You can then complete a TR-31 key export by deriving a shared wrapping key using the service ECC key pair, public certificate of your ECC key pair outside of AWS Payment Cryptography, and the key derivation parameters including key derivation function, hash algorithm, derivation data, key algorithm.

- `KeyMaterial`: Use `DiffieHellmanTr31KeyBlock` parameters.
- `PrivateKeyIdentifier`: The `KeyArn` of the ECC key pair created within AWS Payment Cryptography to derive a shared KEK.
- `PublicKeyCertificate`: The public key certificate of the receiving ECC key pair in PEM format (base64 encoded) to derive a shared KEK.
- `CertificateAuthorityPublicKeyIdentifier`: The `keyARN` of the CA that signed the public key certificate of the receiving ECC key pair.

When this operation is successful, AWS Payment Cryptography returns the working key as a TR-31 WrappedKeyBlock, where the wrapping key is the ECDH derived key.

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- GetParametersForExport

- ImportKey

Request Syntax

```
{
   "ExportAttributes": {
      "ExportDukptInitialKey": {
         "KeySerialNumber": "string"
      },
      "KeyCheckValueAlgorithm": "string"
   },
   "ExportKeyIdentifier": "string",
   "KeyMaterial": { ... }
}
```

Request Parameters

The request accepts the following data in JSON format.

ExportAttributes

The attributes for IPEK generation during export.

Type: ExportAttributes object

Required: No

ExportKeyIdentifier

The KeyARN of the key under export from AWS Payment Cryptography.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 322.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:(key/[0-9a-zA-Z]{16,64}|alias/[a-zA-Z0-9/_-]+)$|^alias/[a-zA-Z0-9/_-]+`

Required: Yes

KeyMaterial

The key block format type, for example, TR-34 or TR-31, to use during key material export.

Type: ExportKeyMaterial object

Note: This object is a Union. Only one member of this object can be specified or returned.

Required: Yes

Response Syntax

```
{
   "WrappedKey": {
      "KeyCheckValue": "string",
      "KeyCheckValueAlgorithm": "string",
      "KeyMaterial": "string",
      "WrappedKeyMaterialFormat": "string",
      "WrappingKeyArn": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

WrappedKey

The key material under export as a TR-34 WrappedKeyBlock or a TR-31 WrappedKeyBlock. or a RSA WrappedKeyCryptogram.

Type: WrappedKey object

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

ConflictException

This request can cause an inconsistent state for the resource.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3

- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

GetAlias

Gets the AWS Payment Cryptography key associated with the alias.

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- CreateAlias
- DeleteAlias
- ListAliases
- UpdateAlias

Request Syntax

```
{
   "AliasName": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

AliasName

The alias of the AWS Payment Cryptography key.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 256.

Pattern: `alias/[a-zA-Z0-9/_-]+`

Required: Yes

Response Syntax

```
{
```

```
   "Alias": {
      "AliasName": "string",
      "KeyArn": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Alias

The alias of the AWS Payment Cryptography key.

Type: Alias object

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

GetKey

Gets the key metadata for an AWS Payment Cryptography key, including the immutable and mutable attributes specified when the key was created. Returns key metadata including attributes, state, and timestamps, but does not return the actual cryptographic key material.

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- CreateKey
- DeleteKey
- ListKeys

Request Syntax

```
{
   "KeyIdentifier": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

KeyIdentifier

The KeyARN of the AWS Payment Cryptography key.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 322.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:(key/[0-9a-zA-Z]{16,64}|alias/[a-zA-Z0-9/_-]+)$|^alias/[a-zA-Z0-9/_-]+`

Required: Yes

Response Syntax

```
{
   "Key": {
      "CreateTimestamp": number,
      "DeletePendingTimestamp": number,
      "DeleteTimestamp": number,
      "DeriveKeyUsage": "string",
      "Enabled": boolean,
      "Exportable": boolean,
      "KeyArn": "string",
      "KeyAttributes": {
         "KeyAlgorithm": "string",
         "KeyClass": "string",
         "KeyModesOfUse": {
            "Decrypt": boolean,
            "DeriveKey": boolean,
            "Encrypt": boolean,
            "Generate": boolean,
            "NoRestrictions": boolean,
            "Sign": boolean,
            "Unwrap": boolean,
            "Verify": boolean,
            "Wrap": boolean
         },
         "KeyUsage": "string"
      },
      "KeyCheckValue": "string",
      "KeyCheckValueAlgorithm": "string",
      "KeyOrigin": "string",
      "KeyState": "string",
      "UsageStartTimestamp": number,
      "UsageStopTimestamp": number
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Key

Contains the key metadata, including both immutable and mutable attributes for the key, but does not include actual cryptographic key material.

Type: Key object

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

GetParametersForExport

Gets the export token and the signing key certificate to initiate a TR-34 key export from AWS Payment Cryptography.

The signing key certificate signs the wrapped key under export within the TR-34 key payload. The export token and signing key certificate must be in place and operational before calling ExportKey. The export token expires in 30 days. You can use the same export token to export multiple keys from your service account.

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- ExportKey
- GetParametersForImport

Request Syntax

```
{
   "KeyMaterialType": "string",
   "SigningKeyAlgorithm": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

KeyMaterialType

The key block format type (for example, TR-34 or TR-31) to use during key material export. Export token is only required for a TR-34 key export, TR34_KEY_BLOCK. Export token is not required for TR-31 key export.

Type: String

Valid Values: TR34_KEY_BLOCK | TR31_KEY_BLOCK | ROOT_PUBLIC_KEY_CERTIFICATE | TRUSTED_PUBLIC_KEY_CERTIFICATE | KEY_CRYPTOGRAM

Required: Yes

SigningKeyAlgorithm

The signing key algorithm to generate a signing key certificate. This certificate signs the wrapped key under export within the TR-34 key block. RSA_2048 is the only signing key algorithm allowed.

Type: String

Valid Values: TDES_2KEY | TDES_3KEY | AES_128 | AES_192 | AES_256 | RSA_2048 | RSA_3072 | RSA_4096 | ECC_NIST_P256 | ECC_NIST_P384 | ECC_NIST_P521

Required: Yes

Response Syntax

```
{
   "ExportToken": "string",
   "ParametersValidUntilTimestamp": number,
   "SigningKeyAlgorithm": "string",
   "SigningKeyCertificate": "string",
   "SigningKeyCertificateChain": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

ExportToken

The export token to initiate key export from AWS Payment Cryptography. The export token expires after 30 days. You can use the same export token to export multiple keys from the same service account.

Type: String

Pattern: export-token-[0-9a-zA-Z]{16,64}

ParametersValidUntilTimestamp

The validity period of the export token.

Type: Timestamp

SigningKeyAlgorithm

The algorithm of the signing key certificate for use in TR-34 key block generation. RSA_2048 is the only signing key algorithm allowed.

Type: String

Valid Values: TDES_2KEY | TDES_3KEY | AES_128 | AES_192 | AES_256 | RSA_2048 | RSA_3072 | RSA_4096 | ECC_NIST_P256 | ECC_NIST_P384 | ECC_NIST_P521

SigningKeyCertificate

The signing key certificate in PEM format (base64 encoded) of the public key for signature within the TR-34 key block. The certificate expires after 30 days.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 32768.

Pattern: [^\[;\]<>]+

SigningKeyCertificateChain

The root certificate authority (CA) that signed the signing key certificate in PEM format (base64 encoded).

Type: String

Length Constraints: Minimum length of 1. Maximum length of 32768.

Pattern: [^\[;\]<>]+

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

ConflictException

This request can cause an inconsistent state for the resource.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceQuotaExceededException

This request would cause a service quota to be exceeded.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET

- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

GetParametersForImport

Gets the import token and the wrapping key certificate in PEM format (base64 encoded) to initiate a TR-34 WrappedKeyBlock or a RSA WrappedKeyCryptogram import into AWS Payment Cryptography.

The wrapping key certificate wraps the key under import. The import token and wrapping key certificate must be in place and operational before calling ImportKey. The import token expires in 30 days. You can use the same import token to import multiple keys into your service account.

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- GetParametersForExport
- ImportKey

Request Syntax

```
{
   "KeyMaterialType": "string",
   "WrappingKeyAlgorithm": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

KeyMaterialType

The method to use for key material import. Import token is only required for TR-34 WrappedKeyBlock (TR34_KEY_BLOCK) and RSA WrappedKeyCryptogram (KEY_CRYPTOGRAM).

Import token is not required for TR-31, root public key cerificate or trusted public key certificate.

Type: String

Valid Values: TR34_KEY_BLOCK | TR31_KEY_BLOCK | ROOT_PUBLIC_KEY_CERTIFICATE | TRUSTED_PUBLIC_KEY_CERTIFICATE | KEY_CRYPTOGRAM

Required: Yes

WrappingKeyAlgorithm

The wrapping key algorithm to generate a wrapping key certificate. This certificate wraps the key under import.

At this time, RSA_2048 is the allowed algorithm for TR-34 WrappedKeyBlock import. Additionally, RSA_2048, RSA_3072, RSA_4096 are the allowed algorithms for RSA WrappedKeyCryptogram import.

Type: String

Valid Values: TDES_2KEY | TDES_3KEY | AES_128 | AES_192 | AES_256 | RSA_2048 | RSA_3072 | RSA_4096 | ECC_NIST_P256 | ECC_NIST_P384 | ECC_NIST_P521

Required: Yes

Response Syntax

```
{
   "ImportToken": "string",
   "ParametersValidUntilTimestamp": number,
   "WrappingKeyAlgorithm": "string",
   "WrappingKeyCertificate": "string",
   "WrappingKeyCertificateChain": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

ImportToken

The import token to initiate key import into AWS Payment Cryptography. The import token expires after 30 days. You can use the same import token to import multiple keys to the same service account.

Type: String

Pattern: `import-token-[0-9a-zA-Z]{16,64}`

ParametersValidUntilTimestamp

The validity period of the import token.

Type: Timestamp

WrappingKeyAlgorithm

The algorithm of the wrapping key for use within TR-34 WrappedKeyBlock or RSA WrappedKeyCryptogram.

Type: String

Valid Values: `TDES_2KEY | TDES_3KEY | AES_128 | AES_192 | AES_256 | RSA_2048 | RSA_3072 | RSA_4096 | ECC_NIST_P256 | ECC_NIST_P384 | ECC_NIST_P521`

WrappingKeyCertificate

The wrapping key certificate in PEM format (base64 encoded) of the wrapping key for use within the TR-34 key block. The certificate expires in 30 days.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 32768.

Pattern: `[^\[;\]<>]+`

WrappingKeyCertificateChain

The AWS Payment Cryptography root certificate authority (CA) that signed the wrapping key certificate in PEM format (base64 encoded).

Type: String

Length Constraints: Minimum length of 1. Maximum length of 32768.

Pattern: `[^\[;\]<>]+`

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

ConflictException

This request can cause an inconsistent state for the resource.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceQuotaExceededException

This request would cause a service quota to be exceeded.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

GetPublicKeyCertificate

Gets the public key certificate of the asymmetric key pair that exists within AWS Payment Cryptography.

Unlike the private key of an asymmetric key, which never leaves AWS Payment Cryptography unencrypted, callers with `GetPublicKeyCertificate` permission can download the public key certificate of the asymmetric key. You can share the public key certificate to allow others to encrypt messages and verify signatures outside of AWS Payment Cryptography

Cross-account use: This operation can't be used across different AWS accounts.

Request Syntax

```
{
   "KeyIdentifier": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

KeyIdentifier

The KeyARN of the asymmetric key pair.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 322.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:(key/[0-9a-zA-Z]{16,64}|alias/[a-zA-Z0-9/_-]+)$|^alias/[a-zA-Z0-9/_-]+`

Required: Yes

Response Syntax

```
{
```

```
    "KeyCertificate": "string",
    "KeyCertificateChain": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

KeyCertificate

The public key component of the asymmetric key pair in a certificate PEM format (base64 encoded). It is signed by the root certificate authority (CA). The certificate expires in 90 days.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 32768.

Pattern: `[^\[;\]<>]+`

KeyCertificateChain

The root certificate authority (CA) that signed the public key certificate in PEM format (base64 encoded) of the asymmetric key pair.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 32768.

Pattern: `[^\[;\]<>]+`

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ImportKey

Imports symmetric keys and public key certificates in PEM format (base64 encoded) into AWS Payment Cryptography.

AWS Payment Cryptography simplifies key exchange by replacing the existing paper-based approach with a modern electronic approach. With `ImportKey` you can import symmetric keys using either symmetric and asymmetric key exchange mechanisms.

For symmetric key exchange, AWS Payment Cryptography uses the ANSI X9 TR-31 norm in accordance with PCI PIN guidelines. And for asymmetric key exchange, AWS Payment Cryptography supports ANSI X9 TR-34 norm, RSA unwrap, and ECDH (Elliptic Curve Diffie-Hellman) key exchange mechanisms. Asymmetric key exchange methods are typically used to establish bi-directional trust between the two parties exhanging keys and are used for initial key exchange such as Key Encryption Key (KEK) or Zone Master Key (ZMK). After which you can import working keys using symmetric method to perform various cryptographic operations within AWS Payment Cryptography.

PCI requires specific minimum key strength of wrapping keys used to protect the keys being exchanged electronically. These requirements can change when PCI standards are revised. The rules specify that wrapping keys used for transport must be at least as strong as the key being protected. For more information on recommended key strength of wrapping keys and key exchange mechanism, see Importing and exporting keys in the *AWS Payment Cryptography User Guide*.

You can also import a *root public key certificate*, used to sign other public key certificates, or a *trusted public key certificate* under an already established root public key certificate.

To import a public root key certificate

Using this operation, you can import the public component (in PEM cerificate format) of your private root key. You can use the imported public root key certificate for digital signatures, for example signing wrapping key or signing key in TR-34, within your AWS Payment Cryptography account.

Set the following parameters:

- `KeyMaterial`: `RootCertificatePublicKey`
- `KeyClass`: `PUBLIC_KEY`

- `KeyModesOfUse`: `Verify`
- `KeyUsage`: `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`
- `PublicKeyCertificate`: The public key certificate in PEM format (base64 encoded) of the private root key under import.

To import a trusted public key certificate

The root public key certificate must be in place and operational before you import a trusted public key certificate. Set the following parameters:

- `KeyMaterial`: `TrustedCertificatePublicKey`
- `CertificateAuthorityPublicKeyIdentifier`: `KeyArn` of the `RootCertificatePublicKey`.
- `KeyModesOfUse` and `KeyUsage`: Corresponding to the cryptographic operations such as wrap, sign, or encrypt that you will allow the trusted public key certificate to perform.
- `PublicKeyCertificate`: The trusted public key certificate in PEM format (base64 encoded) under import.

To import initial keys (KEK or ZMK or similar) using TR-34

Using this operation, you can import initial key using TR-34 asymmetric key exchange. In TR-34 terminology, the sending party of the key is called Key Distribution Host (KDH) and the receiving party of the key is called Key Receiving Device (KRD). During the key import process, KDH is the user who initiates the key import and KRD is AWS Payment Cryptography who receives the key.

To initiate TR-34 key import, the KDH must obtain an import token by calling GetParametersForImport. This operation generates an encryption keypair for the purpose of key import, signs the key and returns back the wrapping key certificate (also known as KRD wrapping certificate) and the root certificate chain. The KDH must trust and install the KRD wrapping certificate on its HSM and use it to encrypt (wrap) the KDH key during TR-34 WrappedKeyBlock generation. The import token and associated KRD wrapping certificate expires after 30 days.

Next the KDH generates a key pair for the purpose of signing the encrypted KDH key and provides the public certificate of the signing key to AWS Payment Cryptography. The KDH will also need to import the root certificate chain of the KDH signing certificate by calling `ImportKey` for `RootCertificatePublicKey`. For more information on TR-34 key import, see section Importing symmetric keys in the *AWS Payment Cryptography User Guide*.

Set the following parameters:

- `KeyMaterial`: Use `Tr34KeyBlock` parameters.
- `CertificateAuthorityPublicKeyIdentifier`: The `KeyARN` of the certificate chain that signed the KDH signing key certificate.
- `ImportToken`: Obtained from KRD by calling GetParametersForImport.
- `WrappedKeyBlock`: The TR-34 wrapped key material from KDH. It contains the KDH key under import, wrapped with KRD wrapping certificate and signed by KDH signing private key. This TR-34 key block is typically generated by the KDH Hardware Security Module (HSM) outside of AWS Payment Cryptography.
- `SigningKeyCertificate`: The public key certificate in PEM format (base64 encoded) of the KDH signing key generated under the root certificate (CertificateAuthorityPublicKeyIdentifier) imported in AWS Payment Cryptography.

To import initial keys (KEK or ZMK or similar) using RSA Wrap and Unwrap

Using this operation, you can import initial key using asymmetric RSA wrap and unwrap key exchange method. To initiate import, call GetParametersForImport with `KeyMaterial` set to `KEY_CRYPTOGRAM` to generate an import token. This operation also generates an encryption keypair for the purpose of key import, signs the key and returns back the wrapping key certificate in PEM format (base64 encoded) and its root certificate chain. The import token and associated KRD wrapping certificate expires after 30 days.

You must trust and install the wrapping certificate and its certificate chain on the sending HSM and use it to wrap the key under export for WrappedKeyCryptogram generation. Next call `ImportKey` with `KeyMaterial` set to `KEY_CRYPTOGRAM` and provide the `ImportToken` and `KeyAttributes` for the key under import.

To import working keys using TR-31

AWS Payment Cryptography uses TR-31 symmetric key exchange norm to import working keys. A KEK must be established within AWS Payment Cryptography by using TR-34 key import or by using CreateKey. To initiate a TR-31 key import, set the following parameters:

- `KeyMaterial`: Use `Tr31KeyBlock` parameters.
- `WrappedKeyBlock`: The TR-31 wrapped key material. It contains the key under import, encrypted using KEK. The TR-31 key block is typically generated by a HSM outside of AWS Payment Cryptography.

- `WrappingKeyIdentifier`: The `KeyArn` of the KEK that AWS Payment Cryptography uses to decrypt or unwrap the key under import.

To import working keys using ECDH

You can also use ECDH key agreement to import working keys as a TR-31 keyblock, where the wrapping key is an ECDH derived key.

To initiate a TR-31 key import using ECDH, both sides must create an ECC key pair with key usage K3 and exchange public key certificates. In AWS Payment Cryptography, you can do this by calling `CreateKey` and then `GetPublicKeyCertificate` to retrieve its public key certificate. Next, you can then generate a TR-31 WrappedKeyBlock using your own ECC key pair, the public certificate of the service's ECC key pair, and the key derivation parameters including key derivation function, hash algorithm, derivation data, and key algorithm. If you have not already done so, you must import the CA chain that issued the receiving public key certificate by calling `ImportKey` with input `RootCertificatePublicKey` for root CA or `TrustedPublicKey` for intermediate CA. To complete the TR-31 key import, you can use the following parameters. It is important that the ECDH key derivation parameters you use should match those used during import to derive the same shared wrapping key within AWS Payment Cryptography.

- `KeyMaterial`: Use `DiffieHellmanTr31KeyBlock` parameters.
- `PrivateKeyIdentifier`: The `KeyArn` of the ECC key pair created within AWS Payment Cryptography to derive a shared KEK.
- `PublicKeyCertificate`: The public key certificate of the receiving ECC key pair in PEM format (base64 encoded) to derive a shared KEK.
- `CertificateAuthorityPublicKeyIdentifier`: The `keyARN` of the CA that signed the public key certificate of the receiving ECC key pair.

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- ExportKey
- GetParametersForImport

Request Syntax

```
{
   "Enabled": boolean,
   "KeyCheckValueAlgorithm": "string",
   "KeyMaterial": { ... },
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Request Parameters

The request accepts the following data in JSON format.

Enabled

Specifies whether import key is enabled.

Type: Boolean

Required: No

KeyCheckValueAlgorithm

The algorithm that AWS Payment Cryptography uses to calculate the key check value (KCV). It is used to validate the key integrity.

For TDES keys, the KCV is computed by encrypting 8 bytes, each with value of zero, with the key to be checked and retaining the 3 highest order bytes of the encrypted result. For AES keys, the KCV is computed using a CMAC algorithm where the input data is 16 bytes of zero and retaining the 3 highest order bytes of the encrypted result.

Type: String

Valid Values: `CMAC | ANSI_X9_24`

Required: No

KeyMaterial

The key or public key certificate type to use during key material import, for example TR-34 or RootCertificatePublicKey.

Type: ImportKeyMaterial object

Note: This object is a Union. Only one member of this object can be specified or returned.

Required: Yes

Tags

Assigns one or more tags to the AWS Payment Cryptography key. Use this parameter to tag a key when it is imported. To tag an existing AWS Payment Cryptography key, use the TagResource operation.

Each tag consists of a tag key and a tag value. Both the tag key and the tag value are required, but the tag value can be an empty (null) string. You can't have more than one tag on an AWS Payment Cryptography key with the same tag key. If you specify an existing tag key with a different tag value, AWS Payment Cryptography replaces the current tag value with the specified one.

> **Important**
>
> Don't include personal, confidential or sensitive information in this field. This field may be displayed in plaintext in AWS CloudTrail logs and other output.

> **Note**
>
> Tagging or untagging an AWS Payment Cryptography key can allow or deny permission to the key.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: No

Response Syntax

```
{
   "Key": {
      "CreateTimestamp": number,
      "DeletePendingTimestamp": number,
      "DeleteTimestamp": number,
      "DeriveKeyUsage": "string",
      "Enabled": boolean,
      "Exportable": boolean,
      "KeyArn": "string",
      "KeyAttributes": {
         "KeyAlgorithm": "string",
         "KeyClass": "string",
         "KeyModesOfUse": {
            "Decrypt": boolean,
            "DeriveKey": boolean,
            "Encrypt": boolean,
            "Generate": boolean,
            "NoRestrictions": boolean,
            "Sign": boolean,
            "Unwrap": boolean,
            "Verify": boolean,
            "Wrap": boolean
         },
         "KeyUsage": "string"
      },
      "KeyCheckValue": "string",
      "KeyCheckValueAlgorithm": "string",
      "KeyOrigin": "string",
      "KeyState": "string",
      "UsageStartTimestamp": number,
      "UsageStopTimestamp": number
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Key

The KeyARN of the key material imported within AWS Payment Cryptography.

Type: Key object

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

ConflictException

This request can cause an inconsistent state for the resource.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceQuotaExceededException

This request would cause a service quota to be exceeded.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListAliases

Lists the aliases for all keys in the caller's AWS account and AWS Region. You can filter the aliases by keyARN. For more information, see Using aliases in the *AWS Payment Cryptography User Guide*.

This is a paginated operation, which means that each response might contain only a subset of all the aliases. When the response contains only a subset of aliases, it includes a `NextToken` value. Use this value in a subsequent `ListAliases` request to get more aliases. When you receive a response with no NextToken (or an empty or null value), that means there are no more aliases to get.

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- CreateAlias
- DeleteAlias
- GetAlias
- UpdateAlias

Request Syntax

```
{
   "KeyArn": "string",
   "MaxResults": number,
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

KeyArn

The keyARN for which you want to list all aliases.

Type: String

Length Constraints: Minimum length of 70. Maximum length of 150.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:key/[0-9a-zA-Z]{16,64}`

Required: No

MaxResults

Use this parameter to specify the maximum number of items to return. When this value is present, AWS Payment Cryptography does not return more than the specified number of items, but it might return fewer.

This value is optional. If you include a value, it must be between 1 and 100, inclusive. If you do not include a value, it defaults to 50.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken

Use this parameter in a subsequent request after you receive a response with truncated results. Set it to the value of `NextToken` from the truncated response you just received.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 8192.

Required: No

Response Syntax

```
{
   "Aliases": [
      {
         "AliasName": "string",
         "KeyArn": "string"
      }
   ],
   "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Aliases

The list of aliases. Each alias describes the `KeyArn` contained within.

Type: Array of Alias objects

NextToken

The token for the next set of results, or an empty or null value if there are no more results.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 8192.

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListKeys

Lists the keys in the caller's AWS account and AWS Region. You can filter the list of keys.

This is a paginated operation, which means that each response might contain only a subset of all the keys. When the response contains only a subset of keys, it includes a `NextToken` value. Use this value in a subsequent `ListKeys` request to get more keys. When you receive a response with no NextToken (or an empty or null value), that means there are no more keys to get.

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- CreateKey
- DeleteKey
- GetKey

Request Syntax

```
{
   "KeyState": "string",
   "MaxResults": number,
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

KeyState

The key state of the keys you want to list.

Type: String

Valid Values: `CREATE_IN_PROGRESS | CREATE_COMPLETE | DELETE_PENDING | DELETE_COMPLETE`

Required: No

MaxResults

Use this parameter to specify the maximum number of items to return. When this value is present, AWS Payment Cryptography does not return more than the specified number of items, but it might return fewer.

This value is optional. If you include a value, it must be between 1 and 100, inclusive. If you do not include a value, it defaults to 50.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken

Use this parameter in a subsequent request after you receive a response with truncated results. Set it to the value of `NextToken` from the truncated response you just received.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 8192.

Required: No

Response Syntax

```
{
   "Keys": [
      {
         "Enabled": boolean,
         "Exportable": boolean,
         "KeyArn": "string",
         "KeyAttributes": {
            "KeyAlgorithm": "string",
            "KeyClass": "string",
            "KeyModesOfUse": {
               "Decrypt": boolean,
               "DeriveKey": boolean,
               "Encrypt": boolean,
               "Generate": boolean,
```

```
                "NoRestrictions": boolean,
                "Sign": boolean,
                "Unwrap": boolean,
                "Verify": boolean,
                "Wrap": boolean
             },
             "KeyUsage": "string"
          },
          "KeyCheckValue": "string",
          "KeyState": "string"
       }
   ],
   "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Keys

The list of keys created within the caller's AWS account and AWS Region.

Type: Array of KeySummary objects

NextToken

The token for the next set of results, or an empty or null value if there are no more results.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 8192.

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin

- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListTagsForResource

Lists the tags for an AWS resource.

This is a paginated operation, which means that each response might contain only a subset of all the tags. When the response contains only a subset of tags, it includes a `NextToken` value. Use this value in a subsequent `ListTagsForResource` request to get more tags. When you receive a response with no NextToken (or an empty or null value), that means there are no more tags to get.

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- TagResource
- UntagResource

Request Syntax

```
{
   "MaxResults": number,
   "NextToken": "string",
   "ResourceArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

MaxResults

Use this parameter to specify the maximum number of items to return. When this value is present, AWS Payment Cryptography does not return more than the specified number of items, but it might return fewer.

This value is optional. If you include a value, it must be between 1 and 100, inclusive. If you do not include a value, it defaults to 50.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken

Use this parameter in a subsequent request after you receive a response with truncated results. Set it to the value of `NextToken` from the truncated response you just received.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 8192.

Required: No

ResourceArn

The KeyARN of the key whose tags you are getting.

Type: String

Length Constraints: Minimum length of 70. Maximum length of 150.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:key/[0-9a-zA-Z]{16,64}`

Required: Yes

Response Syntax

```
{
   "NextToken": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

NextToken

The token for the next set of results, or an empty or null value if there are no more results.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 8192.

Tags

The list of tags associated with a `ResourceArn`. Each tag will list the key-value pair contained within that tag.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

RestoreKey

Cancels a scheduled key deletion during the waiting period. Use this operation to restore a Key that is scheduled for deletion.

During the waiting period, the KeyState is DELETE_PENDING and deletePendingTimestamp contains the date and time after which the Key will be deleted. After Key is restored, the KeyState is CREATE_COMPLETE, and the value for deletePendingTimestamp is removed.

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- DeleteKey
- StartKeyUsage
- StopKeyUsage

Request Syntax

```
{
   "KeyIdentifier": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

KeyIdentifier

The KeyARN of the key to be restored within AWS Payment Cryptography.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 322.

Pattern: arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:(key/[0-9a-zA-Z]{16,64}|alias/[a-zA-Z0-9/_-]+)$|^alias/[a-zA-Z0-9/_-]+

Required: Yes

Response Syntax

```
{
   "Key": {
      "CreateTimestamp": number,
      "DeletePendingTimestamp": number,
      "DeleteTimestamp": number,
      "DeriveKeyUsage": "string",
      "Enabled": boolean,
      "Exportable": boolean,
      "KeyArn": "string",
      "KeyAttributes": {
         "KeyAlgorithm": "string",
         "KeyClass": "string",
         "KeyModesOfUse": {
            "Decrypt": boolean,
            "DeriveKey": boolean,
            "Encrypt": boolean,
            "Generate": boolean,
            "NoRestrictions": boolean,
            "Sign": boolean,
            "Unwrap": boolean,
            "Verify": boolean,
            "Wrap": boolean
         },
         "KeyUsage": "string"
      },
      "KeyCheckValue": "string",
      "KeyCheckValueAlgorithm": "string",
      "KeyOrigin": "string",
      "KeyState": "string",
      "UsageStartTimestamp": number,
      "UsageStopTimestamp": number
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Key

The key material of the restored key. The `KeyState` will change to `CREATE_COMPLETE` and value for `DeletePendingTimestamp` gets removed.

Type: Key object

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

ConflictException

This request can cause an inconsistent state for the resource.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceQuotaExceededException

This request would cause a service quota to be exceeded.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

StartKeyUsage

Enables an AWS Payment Cryptography key, which makes it active for cryptographic operations within AWS Payment Cryptography

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- StopKeyUsage

Request Syntax

```
{
   "KeyIdentifier": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

KeyIdentifier

The KeyArn of the key.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 322.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:(key/[0-9a-zA-Z]{16,64}|alias/[a-zA-Z0-9/_-]+)$|^alias/[a-zA-Z0-9/_-]+`

Required: Yes

Response Syntax

```
{
   "Key": {
```

```
      "CreateTimestamp": number,
      "DeletePendingTimestamp": number,
      "DeleteTimestamp": number,
      "DeriveKeyUsage": "string",
      "Enabled": boolean,
      "Exportable": boolean,
      "KeyArn": "string",
      "KeyAttributes": {
         "KeyAlgorithm": "string",
         "KeyClass": "string",
         "KeyModesOfUse": {
            "Decrypt": boolean,
            "DeriveKey": boolean,
            "Encrypt": boolean,
            "Generate": boolean,
            "NoRestrictions": boolean,
            "Sign": boolean,
            "Unwrap": boolean,
            "Verify": boolean,
            "Wrap": boolean
         },
         "KeyUsage": "string"
      },
      "KeyCheckValue": "string",
      "KeyCheckValueAlgorithm": "string",
      "KeyOrigin": "string",
      "KeyState": "string",
      "UsageStartTimestamp": number,
      "UsageStopTimestamp": number
   }
}
```


Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Key

The KeyARN of the AWS Payment Cryptography key activated for use.

Type: Key object

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

ConflictException

This request can cause an inconsistent state for the resource.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceQuotaExceededException

This request would cause a service quota to be exceeded.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

StopKeyUsage

Disables an AWS Payment Cryptography key, which makes it inactive within AWS Payment Cryptography.

You can use this operation instead of DeleteKey to deactivate a key. You can enable the key in the future by calling StartKeyUsage.

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- DeleteKey
- StartKeyUsage

Request Syntax

```
{
   "KeyIdentifier": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

KeyIdentifier

The KeyArn of the key.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 322.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:(key/[0-9a-zA-Z]{16,64}|alias/[a-zA-Z0-9/_-]+)$|^alias/[a-zA-Z0-9/_-]+`

Required: Yes

Response Syntax

```
{
   "Key": {
      "CreateTimestamp": number,
      "DeletePendingTimestamp": number,
      "DeleteTimestamp": number,
      "DeriveKeyUsage": "string",
      "Enabled": boolean,
      "Exportable": boolean,
      "KeyArn": "string",
      "KeyAttributes": {
         "KeyAlgorithm": "string",
         "KeyClass": "string",
         "KeyModesOfUse": {
            "Decrypt": boolean,
            "DeriveKey": boolean,
            "Encrypt": boolean,
            "Generate": boolean,
            "NoRestrictions": boolean,
            "Sign": boolean,
            "Unwrap": boolean,
            "Verify": boolean,
            "Wrap": boolean
         },
         "KeyUsage": "string"
      },
      "KeyCheckValue": "string",
      "KeyCheckValueAlgorithm": "string",
      "KeyOrigin": "string",
      "KeyState": "string",
      "UsageStartTimestamp": number,
      "UsageStopTimestamp": number
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Key

The KeyARN of the key.

Type: Key object

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

ConflictException

This request can cause an inconsistent state for the resource.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceQuotaExceededException

This request would cause a service quota to be exceeded.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

TagResource

Adds or edits tags on an AWS Payment Cryptography key.

> **Note**
>
> Tagging or untagging an AWS Payment Cryptography key can allow or deny permission to the key.

Each tag consists of a tag key and a tag value, both of which are case-sensitive strings. The tag value can be an empty (null) string. To add a tag, specify a new tag key and a tag value. To edit a tag, specify an existing tag key and a new tag value. You can also add tags to an AWS Payment Cryptography key when you create it with CreateKey.

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- ListTagsForResource
- UntagResource

Request Syntax

```
{
   "ResourceArn": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Request Parameters

The request accepts the following data in JSON format.

ResourceArn

The KeyARN of the key whose tags are being updated.

Type: String

Length Constraints: Minimum length of 70. Maximum length of 150.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:key/[0-9a-zA-Z]{16,64}`

Required: Yes

Tags

One or more tags. Each tag consists of a tag key and a tag value. The tag value can be an empty (null) string. You can't have more than one tag on an AWS Payment Cryptography key with the same tag key. If you specify an existing tag key with a different tag value, AWS Payment Cryptography replaces the current tag value with the new one.

> **Important**
>
> Don't include personal, confidential or sensitive information in this field. This field may be displayed in plaintext in AWS CloudTrail logs and other output.

To use this parameter, you must have TagResource permission in an IAM policy.

> **Important**
>
> Don't include personal, confidential or sensitive information in this field. This field may be displayed in plaintext in AWS CloudTrail logs and other output.

Type: Array of Tag objects

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

ConflictException

This request can cause an inconsistent state for the resource.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceQuotaExceededException

This request would cause a service quota to be exceeded.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

UntagResource

Deletes a tag from an AWS Payment Cryptography key.

> **Note**
>
> Tagging or untagging an AWS Payment Cryptography key can allow or deny permission to the key.

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- ListTagsForResource
- TagResource

Request Syntax

```
{
   "ResourceArn": "string",
   "TagKeys": [ "string" ]
}
```

Request Parameters

The request accepts the following data in JSON format.

ResourceArn

The KeyARN of the key whose tags are being removed.

Type: String

Length Constraints: Minimum length of 70. Maximum length of 150.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:key/[0-9a-zA-Z]{16,64}`

Required: Yes

TagKeys

One or more tag keys. Don't include the tag values.

If the AWS Payment Cryptography key doesn't have the specified tag key, AWS Payment Cryptography doesn't throw an exception or return a response. To confirm that the operation succeeded, use the ListTagsForResource operation.

Type: Array of strings

Array Members: Minimum number of 0 items. Maximum number of 200 items.

Length Constraints: Minimum length of 1. Maximum length of 128.

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

ConflictException

This request can cause an inconsistent state for the resource.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

UpdateAlias

Associates an existing AWS Payment Cryptography alias with a different key. Each alias is associated with only one AWS Payment Cryptography key at a time, although a key can have multiple aliases. The alias and the AWS Payment Cryptography key must be in the same AWS account and AWS Region

Cross-account use: This operation can't be used across different AWS accounts.

Related operations:

- CreateAlias
- DeleteAlias
- GetAlias
- ListAliases

Request Syntax

```
{
   "AliasName": "string",
   "KeyArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

AliasName

The alias whose associated key is changing.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 256.

Pattern: `alias/[a-zA-Z0-9/_-]+`

Required: Yes

KeyArn

The KeyARN for the key that you are updating or removing from the alias.

Type: String

Length Constraints: Minimum length of 70. Maximum length of 150.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:key/[0-9a-zA-Z]{16,64}`

Required: No

Response Syntax

```
{
   "Alias": {
      "AliasName": "string",
      "KeyArn": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Alias

The alias name.

Type: Alias object

Errors

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

ConflictException

This request can cause an inconsistent state for the resource.

HTTP Status Code: 400

InternalServerException

The request processing has failed because of an unknown error, exception, or failure.

HTTP Status Code: 500

ResourceNotFoundException

The request was denied due to an invalid resource error.

HTTP Status Code: 400

ServiceUnavailableException

The service cannot complete the request.

HTTP Status Code: 500

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationException

The request was denied due to an invalid request error.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++

- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for Kotlin
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

Data Types

The Payment Cryptography Control Plane API contains several data types that various actions use. This section describes each data type in detail.

> **Note**
>
> The order of each element in a data type structure is not guaranteed. Applications should not assume a particular order.

The following data types are supported:

- Alias
- DiffieHellmanDerivationData
- ExportAttributes
- ExportDiffieHellmanTr31KeyBlock
- ExportDukptInitialKey
- ExportKeyCryptogram
- ExportKeyMaterial
- ExportTr31KeyBlock
- ExportTr34KeyBlock
- ImportDiffieHellmanTr31KeyBlock
- ImportKeyCryptogram
- ImportKeyMaterial
- ImportTr31KeyBlock
- ImportTr34KeyBlock
- Key
- KeyAttributes
- KeyBlockHeaders
- KeyModesOfUse
- KeySummary
- RootCertificatePublicKey

- Tag
- TrustedCertificatePublicKey
- WrappedKey

Alias

Contains information about an alias.

Contents

AliasName

A friendly name that you can use to refer to a key. The value must begin with `alias/`.

> **Important**
>
> Do not include confidential or sensitive information in this field. This field may be displayed in plaintext in AWS CloudTrail logs and other output.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 256.

Pattern: `alias/[a-zA-Z0-9/_-]+`

Required: Yes

KeyArn

The KeyARN of the key associated with the alias.

Type: String

Length Constraints: Minimum length of 70. Maximum length of 150.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:key/[0-9a-zA-Z]{16,64}`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

DiffieHellmanDerivationData

The shared information used when deriving a key using ECDH.

Contents

> **Important**
>
> This data type is a UNION, so only one of the following members can be specified when used or returned.

SharedInformation

A string containing information that binds the ECDH derived key to the two parties involved or to the context of the key.

It may include details like identities of the two parties deriving the key, context of the operation, session IDs, and optionally a nonce. It must not contain zero bytes. It is not recommended to reuse shared information for multiple ECDH key derivations, as it could result in derived key material being the same across different derivations.

Type: String

Length Constraints: Minimum length of 2. Maximum length of 2048.

Pattern: `(?:[0-9a-fA-F][0-9a-fA-F])+`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ExportAttributes

The attributes for IPEK generation during export.

Contents

ExportDukptInitialKey

Parameter information for IPEK export.

Type: ExportDukptInitialKey object

Required: No

KeyCheckValueAlgorithm

The algorithm that AWS Payment Cryptography uses to calculate the key check value (KCV). It is used to validate the key integrity. Specify KCV for IPEK export only.

For TDES keys, the KCV is computed by encrypting 8 bytes, each with value of zero, with the key to be checked and retaining the 3 highest order bytes of the encrypted result. For AES keys, the KCV is computed using a CMAC algorithm where the input data is 16 bytes of zero and retaining the 3 highest order bytes of the encrypted result.

Type: String

Valid Values: `CMAC | ANSI_X9_24`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ExportDiffieHellmanTr31KeyBlock

Key derivation parameter information for key material export using asymmetric ECDH key exchange method.

Contents

CertificateAuthorityPublicKeyIdentifier

The `keyARN` of the CA that signed the `PublicKeyCertificate` for the client's receiving ECC key pair.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 322.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:(key/[0-9a-zA-Z]{16,64}|alias/[a-zA-Z0-9/_-]+)$|^alias/[a-zA-Z0-9/_-]+`

Required: Yes

DerivationData

The shared information used when deriving a key using ECDH.

Type: DiffieHellmanDerivationData object

Note: This object is a Union. Only one member of this object can be specified or returned.

Required: Yes

DeriveKeyAlgorithm

The key algorithm of the shared derived ECDH key.

Type: String

Valid Values: `TDES_2KEY | TDES_3KEY | AES_128 | AES_192 | AES_256`

Required: Yes

KeyDerivationFunction

The key derivation function to use when deriving a key using ECDH.

Type: String

Valid Values: `NIST_SP800 | ANSI_X963`

Required: Yes

KeyDerivationHashAlgorithm

The hash type to use when deriving a key using ECDH.

Type: String

Valid Values: `SHA_256 | SHA_384 | SHA_512`

Required: Yes

PrivateKeyIdentifier

The `keyARN` of the asymmetric ECC key created within AWS Payment Cryptography.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 322.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:(key/[0-9a-zA-Z]{16,64}|alias/[a-zA-Z0-9/_-]+)$|^alias/[a-zA-Z0-9/_-]+`

Required: Yes

PublicKeyCertificate

The public key certificate of the client's receiving ECC key pair, in PEM format (base64 encoded), to use for ECDH key derivation.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 32768.

Pattern: `[^\[;\]<>]+`

Required: Yes

KeyBlockHeaders

Optional metadata for export associated with the key material. This data is signed but transmitted in clear text.

Type: KeyBlockHeaders object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ExportDukptInitialKey

Parameter information for IPEK generation during export.

Contents

KeySerialNumber

The KSN for IPEK generation using DUKPT.

KSN must be padded before sending to AWS Payment Cryptography. KSN hex length should be 20 for a TDES_2KEY key or 24 for an AES key.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 24.

Pattern: `[0-9A-F]{20}$|^[0-9A-F]{24}`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ExportKeyCryptogram

Parameter information for key material export using asymmetric RSA wrap and unwrap key exchange method.

Contents

CertificateAuthorityPublicKeyIdentifier

The KeyARN of the certificate chain that signs the wrapping key certificate during RSA wrap and unwrap key export.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 322.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:(key/[0-9a-zA-Z]{16,64}|alias/[a-zA-Z0-9/_-]+)$|^alias/[a-zA-Z0-9/_-]+`

Required: Yes

WrappingKeyCertificate

The wrapping key certificate in PEM format (base64 encoded). AWS Payment Cryptography uses this certificate to wrap the key under export.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 32768.

Pattern: `[^\[;\]<>]+`

Required: Yes

WrappingSpec

The wrapping spec for the key under export.

Type: String

Valid Values: `RSA_OAEP_SHA_256 | RSA_OAEP_SHA_512`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ExportKeyMaterial

Parameter information for key material export from AWS Payment Cryptography using TR-31 or TR-34 or RSA wrap and unwrap key exchange method.

Contents

> **Important**
>
> This data type is a UNION, so only one of the following members can be specified when used or returned.

DiffieHellmanTr31KeyBlock

Key derivation parameter information for key material export using asymmetric ECDH key exchange method.

Type: ExportDiffieHellmanTr31KeyBlock object

Required: No

KeyCryptogram

Parameter information for key material export using asymmetric RSA wrap and unwrap key exchange method

Type: ExportKeyCryptogram object

Required: No

Tr31KeyBlock

Parameter information for key material export using symmetric TR-31 key exchange method.

Type: ExportTr31KeyBlock object

Required: No

Tr34KeyBlock

Parameter information for key material export using the asymmetric TR-34 key exchange method.

Type: ExportTr34KeyBlock object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ExportTr31KeyBlock

Parameter information for key material export using symmetric TR-31 key exchange method.

Contents

WrappingKeyIdentifier

The KeyARN of the the wrapping key. This key encrypts or wraps the key under export for TR-31 key block generation.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 322.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:(key/[0-9a-zA-Z]{16,64}|alias/[a-zA-Z0-9/_-]+)$|^alias/[a-zA-Z0-9/_-]+`

Required: Yes

KeyBlockHeaders

Optional metadata for export associated with the key material. This data is signed but transmitted in clear text.

Type: KeyBlockHeaders object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ExportTr34KeyBlock

Parameter information for key material export using the asymmetric TR-34 key exchange method.

Contents

CertificateAuthorityPublicKeyIdentifier

The KeyARN of the certificate chain that signs the wrapping key certificate during TR-34 key export.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 322.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:(key/[0-9a-zA-Z]{16,64}|alias/[a-zA-Z0-9/_-]+)$|^alias/[a-zA-Z0-9/_-]+`

Required: Yes

ExportToken

The export token to initiate key export from AWS Payment Cryptography. It also contains the signing key certificate that will sign the wrapped key during TR-34 key block generation. Call GetParametersForExport to receive an export token. It expires after 30 days. You can use the same export token to export multiple keys from the same service account.

Type: String

Pattern: `export-token-[0-9a-zA-Z]{16,64}`

Required: Yes

KeyBlockFormat

The format of key block that AWS Payment Cryptography will use during key export.

Type: String

Valid Values: X9_TR34_2012

Required: Yes

WrappingKeyCertificate

The KeyARN of the wrapping key certificate. AWS Payment Cryptography uses this certificate to wrap the key under export.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 32768.

Pattern: `[^\[;\]<>]+`

Required: Yes

KeyBlockHeaders

Optional metadata for export associated with the key material. This data is signed but transmitted in clear text.

Type: KeyBlockHeaders object

Required: No

RandomNonce

A random number value that is unique to the TR-34 key block generated using 2 pass. The operation will fail, if a random nonce value is not provided for a TR-34 key block generated using 2 pass.

Type: String

Length Constraints: Minimum length of 16. Maximum length of 32.

Pattern: `(?:[0-9a-fA-F][0-9a-fA-F])+`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2

- AWS SDK for Ruby V3

ImportDiffieHellmanTr31KeyBlock

Key derivation parameter information for key material import using asymmetric ECDH key exchange method.

Contents

CertificateAuthorityPublicKeyIdentifier

The `keyARN` of the CA that signed the `PublicKeyCertificate` for the client's receiving ECC key pair.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 322.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:(key/[0-9a-zA-Z]{16,64}|alias/[a-zA-Z0-9/_-]+)$|^alias/[a-zA-Z0-9/_-]+`

Required: Yes

DerivationData

The shared information used when deriving a key using ECDH.

Type: DiffieHellmanDerivationData object

Note: This object is a Union. Only one member of this object can be specified or returned.

Required: Yes

DeriveKeyAlgorithm

The key algorithm of the shared derived ECDH key.

Type: String

Valid Values: `TDES_2KEY | TDES_3KEY | AES_128 | AES_192 | AES_256`

Required: Yes

KeyDerivationFunction

The key derivation function to use when deriving a key using ECDH.

Type: String

Valid Values: `NIST_SP800 | ANSI_X963`

Required: Yes

KeyDerivationHashAlgorithm

The hash type to use when deriving a key using ECDH.

Type: String

Valid Values: `SHA_256 | SHA_384 | SHA_512`

Required: Yes

PrivateKeyIdentifier

The `keyARN` of the asymmetric ECC key created within AWS Payment Cryptography.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 322.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:(key/[0-9a-zA-Z]{16,64}|alias/[a-zA-Z0-9/_-]+)$|^alias/[a-zA-Z0-9/_-]+`

Required: Yes

PublicKeyCertificate

The public key certificate of the client's receiving ECC key pair, in PEM format (base64 encoded), to use for ECDH key derivation.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 32768.

Pattern: `[^\[;\]<>]+`

Required: Yes

WrappedKeyBlock

The ECDH wrapped key block to import.

Type: String

Length Constraints: Minimum length of 56. Maximum length of 9984.

Pattern: `[0-9A-Z]+`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ImportKeyCryptogram

Parameter information for key material import using asymmetric RSA wrap and unwrap key exchange method.

Contents

Exportable

Specifies whether the key is exportable from the service.

Type: Boolean

Required: Yes

ImportToken

The import token that initiates key import using the asymmetric RSA wrap and unwrap key exchange method into AWS Payment Cryptography. It expires after 30 days. You can use the same import token to import multiple keys to the same service account.

Type: String

Pattern: `import-token-[0-9a-zA-Z]{16,64}`

Required: Yes

KeyAttributes

The role of the key, the algorithm it supports, and the cryptographic operations allowed with the key. This data is immutable after the key is created.

Type: KeyAttributes object

Required: Yes

WrappedKeyCryptogram

The RSA wrapped key cryptogram under import.

Type: String

Length Constraints: Minimum length of 16. Maximum length of 4096.

Pattern: `[0-9A-F]+`

Required: Yes

WrappingSpec

The wrapping spec for the wrapped key cryptogram.

Type: String

Valid Values: `RSA_OAEP_SHA_256 | RSA_OAEP_SHA_512`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ImportKeyMaterial

Parameter information for key material import into AWS Payment Cryptography using TR-31 or TR-34 or RSA wrap and unwrap key exchange method.

Contents

> **Important**
>
> This data type is a UNION, so only one of the following members can be specified when used or returned.

DiffieHellmanTr31KeyBlock

Key derivation parameter information for key material import using asymmetric ECDH key exchange method.

Type: ImportDiffieHellmanTr31KeyBlock object

Required: No

KeyCryptogram

Parameter information for key material import using asymmetric RSA wrap and unwrap key exchange method.

Type: ImportKeyCryptogram object

Required: No

RootCertificatePublicKey

Parameter information for root public key certificate import.

Type: RootCertificatePublicKey object

Required: No

Tr31KeyBlock

Parameter information for key material import using symmetric TR-31 key exchange method.

Type: ImportTr31KeyBlock object

Required: No

Tr34KeyBlock

Parameter information for key material import using the asymmetric TR-34 key exchange method.

Type: ImportTr34KeyBlock object

Required: No

TrustedCertificatePublicKey

Parameter information for trusted public key certificate import.

Type: TrustedCertificatePublicKey object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ImportTr31KeyBlock

Parameter information for key material import using symmetric TR-31 key exchange method.

Contents

WrappedKeyBlock

The TR-31 wrapped key block to import.

Type: String

Length Constraints: Minimum length of 56. Maximum length of 9984.

Pattern: `[0-9A-Z]+`

Required: Yes

WrappingKeyIdentifier

The KeyARN of the key that will decrypt or unwrap a TR-31 key block during import.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 322.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:(key/[0-9a-zA-Z]{16,64}|alias/[a-zA-Z0-9/_-]+)$|^alias/[a-zA-Z0-9/_-]+`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ImportTr34KeyBlock

Parameter information for key material import using the asymmetric TR-34 key exchange method.

Contents

CertificateAuthorityPublicKeyIdentifier

The KeyARN of the certificate chain that signs the signing key certificate during TR-34 key import.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 322.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:(key/[0-9a-zA-Z]{16,64}|alias/[a-zA-Z0-9/_-]+)$|^alias/[a-zA-Z0-9/_-]+`

Required: Yes

ImportToken

The import token that initiates key import using the asymmetric TR-34 key exchange method into AWS Payment Cryptography. It expires after 30 days. You can use the same import token to import multiple keys to the same service account.

Type: String

Pattern: `import-token-[0-9a-zA-Z]{16,64}`

Required: Yes

KeyBlockFormat

The key block format to use during key import. The only value allowed is X9_TR34_2012.

Type: String

Valid Values: X9_TR34_2012

Required: Yes

SigningKeyCertificate

The public key component in PEM certificate format of the private key that signs the KDH TR-34 WrappedKeyBlock.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 32768.

Pattern: `[^\[;\]<>]+`

Required: Yes

WrappedKeyBlock

The TR-34 wrapped key block to import.

Type: String

Length Constraints: Minimum length of 2. Maximum length of 4096.

Pattern: `[0-9A-F]+`

Required: Yes

RandomNonce

A random number value that is unique to the TR-34 key block generated using 2 pass. The operation will fail, if a random nonce value is not provided for a TR-34 key block generated using 2 pass.

Type: String

Length Constraints: Minimum length of 16. Maximum length of 32.

Pattern: `(?:[0-9a-fA-F][0-9a-fA-F])+`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Key

Metadata about an AWS Payment Cryptography key.

Contents

CreateTimestamp

The date and time when the key was created.

Type: Timestamp

Required: Yes

Enabled

Specifies whether the key is enabled.

Type: Boolean

Required: Yes

Exportable

Specifies whether the key is exportable. This data is immutable after the key is created.

Type: Boolean

Required: Yes

KeyArn

The Amazon Resource Name (ARN) of the key.

Type: String

Length Constraints: Minimum length of 70. Maximum length of 150.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:key/[0-9a-zA-Z]{16,64}`

Required: Yes

KeyAttributes

The role of the key, the algorithm it supports, and the cryptographic operations allowed with the key. This data is immutable after the key is created.

Type: KeyAttributes object

Required: Yes

KeyCheckValue

The key check value (KCV) is used to check if all parties holding a given key have the same key or to detect that a key has changed.

Type: String

Length Constraints: Minimum length of 4. Maximum length of 16.

Pattern: `[0-9a-fA-F]+`

Required: Yes

KeyCheckValueAlgorithm

The algorithm that AWS Payment Cryptography uses to calculate the key check value (KCV). It is used to validate the key integrity.

For TDES keys, the KCV is computed by encrypting 8 bytes, each with value of zero, with the key to be checked and retaining the 3 highest order bytes of the encrypted result. For AES keys, the KCV is computed using a CMAC algorithm where the input data is 16 bytes of zero and retaining the 3 highest order bytes of the encrypted result.

Type: String

Valid Values: `CMAC | ANSI_X9_24`

Required: Yes

KeyOrigin

The source of the key material. For keys created within AWS Payment Cryptography, the value is `AWS_PAYMENT_CRYPTOGRAPHY`. For keys imported into AWS Payment Cryptography, the value is `EXTERNAL`.

Type: String

Valid Values: `EXTERNAL | AWS_PAYMENT_CRYPTOGRAPHY`

Required: Yes

KeyState

The state of key that is being created or deleted.

Type: String

Valid Values: `CREATE_IN_PROGRESS | CREATE_COMPLETE | DELETE_PENDING | DELETE_COMPLETE`

Required: Yes

DeletePendingTimestamp

The date and time after which AWS Payment Cryptography will delete the key. This value is present only when `KeyState` is `DELETE_PENDING` and the key is scheduled for deletion.

Type: Timestamp

Required: No

DeleteTimestamp

The date and time after which AWS Payment Cryptography will delete the key. This value is present only when when the `KeyState` is `DELETE_COMPLETE` and the AWS Payment Cryptography key is deleted.

Type: Timestamp

Required: No

DeriveKeyUsage

The cryptographic usage of an ECDH derived key as defined in section A.5.2 of the TR-31 spec.

Type: String

Valid Values: `TR31_B0_BASE_DERIVATION_KEY | TR31_C0_CARD_VERIFICATION_KEY | TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY | TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS | TR31_E1_EMV_MKEY_CONFIDENTIALITY | TR31_E2_EMV_MKEY_INTEGRITY | TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS | TR31_E5_EMV_MKEY_CARD_PERSONALIZATION | TR31_E6_EMV_MKEY_OTHER | TR31_K0_KEY_ENCRYPTION_KEY | TR31_K1_KEY_BLOCK_PROTECTION_KEY | TR31_M3_ISO_9797_3_MAC_KEY | TR31_M1_ISO_9797_1_MAC_KEY | TR31_M6_ISO_9797_5_CMAC_KEY | TR31_M7_HMAC_KEY | TR31_P0_PIN_ENCRYPTION_KEY |`

`TR31_P1_PIN_GENERATION_KEY | TR31_V1_IBM3624_PIN_VERIFICATION_KEY | TR31_V2_VISA_PIN_VERIFICATION_KEY`

Required: No

UsageStartTimestamp

The date and time after which AWS Payment Cryptography will start using the key material for cryptographic operations.

Type: Timestamp

Required: No

UsageStopTimestamp

The date and time after which AWS Payment Cryptography will stop using the key material for cryptographic operations.

Type: Timestamp

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

KeyAttributes

The role of the key, the algorithm it supports, and the cryptographic operations allowed with the key. This data is immutable after the key is created.

Contents

KeyAlgorithm

The key algorithm to be use during creation of an AWS Payment Cryptography key.

For symmetric keys, AWS Payment Cryptography supports `AES` and `TDES` algorithms. For asymmetric keys, AWS Payment Cryptography supports `RSA` and `ECC_NIST` algorithms.

Type: String

Valid Values: `TDES_2KEY | TDES_3KEY | AES_128 | AES_192 | AES_256 | RSA_2048 | RSA_3072 | RSA_4096 | ECC_NIST_P256 | ECC_NIST_P384 | ECC_NIST_P521`

Required: Yes

KeyClass

The type of AWS Payment Cryptography key to create, which determines the classification of the cryptographic method and whether AWS Payment Cryptography key contains a symmetric key or an asymmetric key pair.

Type: String

Valid Values: `SYMMETRIC_KEY | ASYMMETRIC_KEY_PAIR | PRIVATE_KEY | PUBLIC_KEY`

Required: Yes

KeyModesOfUse

The list of cryptographic operations that you can perform using the key.

Type: KeyModesOfUse object

Required: Yes

KeyUsage

The cryptographic usage of an AWS Payment Cryptography key as defined in section A.5.2 of the TR-31 spec.

Type: String

Valid Values: TR31_B0_BASE_DERIVATION_KEY | TR31_C0_CARD_VERIFICATION_KEY | TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY | TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION | TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS | TR31_E1_EMV_MKEY_CONFIDENTIALITY | TR31_E2_EMV_MKEY_INTEGRITY | TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS | TR31_E5_EMV_MKEY_CARD_PERSONALIZATION | TR31_E6_EMV_MKEY_OTHER | TR31_K0_KEY_ENCRYPTION_KEY | TR31_K1_KEY_BLOCK_PROTECTION_KEY | TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT | TR31_M3_ISO_9797_3_MAC_KEY | TR31_M1_ISO_9797_1_MAC_KEY | TR31_M6_ISO_9797_5_CMAC_KEY | TR31_M7_HMAC_KEY | TR31_P0_PIN_ENCRYPTION_KEY | TR31_P1_PIN_GENERATION_KEY | TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE | TR31_V1_IBM3624_PIN_VERIFICATION_KEY | TR31_V2_VISA_PIN_VERIFICATION_KEY | TR31_K2_TR34_ASYMMETRIC_KEY

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

KeyBlockHeaders

Optional metadata for export associated with the key material. This data is signed but transmitted in clear text.

Contents

KeyExportability

Specifies subsequent exportability of the key within the key block after it is received by the receiving party. It can be used to further restrict exportability of the key after export from AWS Payment Cryptography.

When set to `EXPORTABLE`, the key can be subsequently exported by the receiver under a KEK using TR-31 or TR-34 key block export only. When set to `NON_EXPORTABLE`, the key cannot be subsequently exported by the receiver. When set to `SENSITIVE`, the key can be exported by the receiver under a KEK using TR-31, TR-34, RSA wrap and unwrap cryptogram or using a symmetric cryptogram key export method. For further information refer to ANSI X9.143-2022.

Type: String

Valid Values: `EXPORTABLE | NON_EXPORTABLE | SENSITIVE`

Required: No

KeyModesOfUse

The list of cryptographic operations that you can perform using the key. The modes of use are defined in section A.5.3 of the TR-31 spec.

Type: KeyModesOfUse object

Required: No

KeyVersion

Parameter used to indicate the version of the key carried in the key block or indicate the value carried in the key block is a component of a key.

Type: String

Length Constraints: Fixed length of 2.

Pattern: `[0-9A-Z]{2}+`

Required: No

OptionalBlocks

Parameter used to indicate the type of optional data in key block headers. Refer to ANSI X9.143-2022 for information on allowed data type for optional blocks.

Optional block character limit is 112 characters. For each optional block, 2 characters are reserved for optional block ID and 2 characters reserved for optional block length. More than one optional blocks can be included as long as the combined length does not increase 112 characters.

Type: String to string map

Key Length Constraints: Fixed length of 2.

Key Pattern: `[0-9A-Z]{2}+`

Value Length Constraints: Minimum length of 1. Maximum length of 108.

Value Pattern: `[0-9A-Z]+`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

KeyModesOfUse

The list of cryptographic operations that you can perform using the key. The modes of use are defined in section A.5.3 of the TR-31 spec.

Contents

Decrypt

Specifies whether an AWS Payment Cryptography key can be used to decrypt data.

Type: Boolean

Required: No

DeriveKey

Specifies whether an AWS Payment Cryptography key can be used to derive new keys.

Type: Boolean

Required: No

Encrypt

Specifies whether an AWS Payment Cryptography key can be used to encrypt data.

Type: Boolean

Required: No

Generate

Specifies whether an AWS Payment Cryptography key can be used to generate and verify other card and PIN verification keys.

Type: Boolean

Required: No

NoRestrictions

Specifies whether an AWS Payment Cryptography key has no special restrictions other than the restrictions implied by `KeyUsage`.

Type: Boolean

Required: No

Sign

Specifies whether an AWS Payment Cryptography key can be used for signing.

Type: Boolean

Required: No

Unwrap

Specifies whether an AWS Payment Cryptography key can be used to unwrap other keys.

Type: Boolean

Required: No

Verify

Specifies whether an AWS Payment Cryptography key can be used to verify signatures.

Type: Boolean

Required: No

Wrap

Specifies whether an AWS Payment Cryptography key can be used to wrap other keys.

Type: Boolean

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

KeySummary

Metadata about an AWS Payment Cryptography key.

Contents

Enabled

Specifies whether the key is enabled.

Type: Boolean

Required: Yes

Exportable

Specifies whether the key is exportable. This data is immutable after the key is created.

Type: Boolean

Required: Yes

KeyArn

The Amazon Resource Name (ARN) of the key.

Type: String

Length Constraints: Minimum length of 70. Maximum length of 150.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:key/[0-9a-zA-Z]{16,64}`

Required: Yes

KeyAttributes

The role of the key, the algorithm it supports, and the cryptographic operations allowed with the key. This data is immutable after the key is created.

Type: KeyAttributes object

Required: Yes

KeyCheckValue

The key check value (KCV) is used to check if all parties holding a given key have the same key or to detect that a key has changed.

Type: String

Length Constraints: Minimum length of 4. Maximum length of 16.

Pattern: `[0-9a-fA-F]+`

Required: Yes

KeyState

The state of an AWS Payment Cryptography that is being created or deleted.

Type: String

Valid Values: `CREATE_IN_PROGRESS | CREATE_COMPLETE | DELETE_PENDING | DELETE_COMPLETE`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

RootCertificatePublicKey

Parameter information for root public key certificate import.

Contents

KeyAttributes

The role of the key, the algorithm it supports, and the cryptographic operations allowed with the key. This data is immutable after the root public key is imported.

Type: KeyAttributes object

Required: Yes

PublicKeyCertificate

Parameter information for root public key certificate import.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 32768.

Pattern: `[^\[;\]<>]+`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Tag

A structure that contains information about a tag.

Contents

Key

The key of the tag.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 128.

Required: Yes

Value

The value of the tag.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

TrustedCertificatePublicKey

Parameter information for trusted public key certificate import.

Contents

CertificateAuthorityPublicKeyIdentifier

The KeyARN of the root public key certificate or certificate chain that signs the trusted public key certificate import.

Type: String

Length Constraints: Minimum length of 7. Maximum length of 322.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:(key/[0-9a-zA-Z]{16,64}|alias/[a-zA-Z0-9/_-]+)$|^alias/[a-zA-Z0-9/_-]+`

Required: Yes

KeyAttributes

The role of the key, the algorithm it supports, and the cryptographic operations allowed with the key. This data is immutable after a trusted public key is imported.

Type: KeyAttributes object

Required: Yes

PublicKeyCertificate

Parameter information for trusted public key certificate import.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 32768.

Pattern: `[^\[;\]<>]+`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

WrappedKey

Parameter information for generating a WrappedKeyBlock for key exchange.

Contents

KeyMaterial

Parameter information for generating a wrapped key using TR-31 or TR-34 skey exchange method.

Type: String

Length Constraints: Minimum length of 48. Maximum length of 16384.

Required: Yes

WrappedKeyMaterialFormat

The key block format of a wrapped key.

Type: String

Valid Values: `KEY_CRYPTOGRAM | TR31_KEY_BLOCK | TR34_KEY_BLOCK`

Required: Yes

WrappingKeyArn

The `KeyARN` of the wrapped key.

Type: String

Length Constraints: Minimum length of 70. Maximum length of 150.

Pattern: `arn:aws:payment-cryptography:[a-z]{2}-[a-z]{1,16}-[0-9]+:[0-9]{12}:key/[0-9a-zA-Z]{16,64}`

Required: Yes

KeyCheckValue

The key check value (KCV) is used to check if all parties holding a given key have the same key or to detect that a key has changed.

Type: String

Length Constraints: Minimum length of 4. Maximum length of 16.

Pattern: `[0-9a-fA-F]+`

Required: No

KeyCheckValueAlgorithm

The algorithm that AWS Payment Cryptography uses to calculate the key check value (KCV). It is used to validate the key integrity.

For TDES keys, the KCV is computed by encrypting 8 bytes, each with value of zero, with the key to be checked and retaining the 3 highest order bytes of the encrypted result. For AES keys, the KCV is computed using a CMAC algorithm where the input data is 16 bytes of zero and retaining the 3 highest order bytes of the encrypted result.

Type: String

Valid Values: `CMAC | ANSI_X9_24`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3