CipherTrust Application Data Protection for C PKCS#11

API REFERENCE GUIDE

Document Information

Document Information

Product Version	8.17.0
Release Date	09 January, 2025

Trademarks and Copyrights

Disclaimer

All information herein is either public information or is the property of and owned solely by Thales and/or its subsidiaries who shall have and keep the sole right to file patent applications or any other kind of intellectual property protection in connection with such information.

Nothing herein shall be construed as implying or granting to you any rights, by license, grant or otherwise, under any intellectual and/or industrial property rights of or concerning any of Thales DIS France S.A. and any of its subsidiaries and affiliates (collectively referred to herein after as “Thales”) information.

This document can be used for informational, non-commercial, internal, and personal use only provided that:

> The copyright notice, the confidentiality and proprietary legend and this full warning notice appear in all copies.

> This document shall not be posted on any publicly accessible network computer or broadcast in any media, and no modification of any part of this document shall be made.

Use for any other purpose is expressly prohibited and may result in severe civil and criminal liabilities.

The information contained in this document is provided “AS IS” without any warranty of any kind. Unless otherwise expressly agreed in writing, Thales makes no warranty as to the value or accuracy of information contained herein.

The document could include technical inaccuracies or typographical errors. Changes are periodically added to the information herein. Furthermore, Thales reserves the right to make any change or improvement in the specifications data, information, and the like described herein, at any time.

Thales hereby disclaims all warranties and conditions with regard to the information contained herein, including all implied warranties of merchantability, fitness for a particular purpose, title and non-infringement. In no event shall Thales be liable, whether in contract, tort or otherwise, for any indirect, special or consequential damages or any damages whatsoever including but not limited to damages resulting from loss of use, data, profits, revenues, or customers, arising out of or in connection with the use or performance of information contained in this document.

Thales does not and shall not warrant that this product will be resistant to all possible attacks and shall not incur, and disclaims, any liability in this respect. Even if each product is compliant with current security standards in force on the date of their design, security mechanisms' resistance necessarily evolves according to the state of the art in security and notably under the emergence of new attacks. Under no circumstances, shall Thales be held liable for any third party actions and in particular in case of any successful attack against systems or equipment incorporating Thales products. Thales disclaims any liability with respect to security for direct, indirect, incidental or consequential damages that result from any use of its products. It is further stressed that independent testing and verification by the person using the product is particularly encouraged, especially in any application in which defective, incorrect or insecure functioning could result in damage to persons or property, denial of service, or loss of privacy.

CONTENTS

PREFACE: About this Document

This introductory section identifies the audience, explains the importance of the release notes, explains how to best use the written material, and discusses the documentation conventions used. It contains the following sections:

Release Notes

The release notes provide important information about this release that is not included in other customer documentation. It is strongly recommended that you read the release notes to fully understand the capabilities, limitations, and known issues for this release. You can view or download the latest version of the release notes for this release at the following location:

https://supportportal.thalesgroup.com

Audience

This document is intended for CipherTrust Application Data Protection for C PKCS#11 (CADP for C PKCS#11) administrators responsible for deploying security policies on client machines. Administrators must have root permissions for the systems on which this product is installed.

All products manufactured and distributed by Thales Group are designed to be installed, operated, and maintained by personnel who have the knowledge, training, and qualifications required to safely perform the tasks assigned to them. The information, processes, and procedures contained in this document are intended for use by trained and qualified personnel only.

It is assumed that the users of this document are proficient with data security concepts. Readers must also possess knowledge of C and PKCS #11.

Document Conventions

This section describes the conventions used in this document.

Command Syntax and Typeface Conventions

This document uses the following conventions for command syntax descriptions, and to highlight elements of the user interface.

Format	Convention
bold	The bold attribute is used to indicate the following: > Command-line commands and options that you enter verbatim (Type **dir /p**.) > Button names (Click **Save As**.) > Check box and radio button names (Select the **Print Duplex** check box.) > Dialog box titles (On the **Protect Document** dialog box, click **Yes**.) > Field names (**User Name**: Enter the name of the user.) > Menu names (On the **File** menu, click **Save**.) (Click **Menu** > **Go To** > **Folders**.) > User input (In the **Date** box, type **April 1**.)
italics	In type, the italic attribute is used for emphasis or to indicate a related document. (See the *Installation Guide* for more information.)
<variable>	In command descriptions, angle brackets represent variables. You must substitute a value for command line arguments that are enclosed in angle brackets.
[**optional**] [<optional>]	Represent optional **keywords** or <variables> in a command line description. Optionally enter the keyword or <variable> that is enclosed in square brackets, if it is necessary or desirable to complete the task.
{**a**|**b**|**c**} {<a>|<b>|<c>}	Represent required alternate **keywords** or <variables> in a command line description. You must choose one command line argument enclosed within the braces. Choices are separated by vertical (OR) bars.
[**a**|**b**|**c**] [<a>|<b>|<c>]	Represent optional alternate keywords or variables in a command line description. Choose one command line argument enclosed within the braces, if desired. Choices are separated by vertical (OR) bars.

Notifications and Alerts

Notifications and alerts are used to highlight important information or alert you to the potential for data loss or personal injury.

Tips

Tips are used to highlight information that helps to complete a task more efficiently.

> **TIP** This is some information that will allow you to complete your task more efficiently.

Notes

Notes are used to highlight important or helpful information.

> **NOTE** Take note. Contains important or helpful information.

Cautions

Cautions are used to alert you to important information that may help prevent unexpected results or data loss.

> **CAUTION!** Exercise caution. Contains important information that may help prevent unexpected results or data loss.

Warnings

Warnings are used to alert you to the potential for catastrophic data loss or personal injury.

> ****WARNING** Be extremely careful and obey all safety and security measures. In this situation you might do something that could result in catastrophic data loss or personal injury.**

Related Documents

Refer to the online CADP for C documentation on the Thales documentation portal for related and additional information.

Support Contacts

If you encounter a problem while installing, registering, or operating this product, please refer to the documentation before contacting support. If you cannot resolve the issue, contact your supplier or Thales Customer Support.

Thales Customer Support operates 24 hours a day, 7 days a week. Your level of access to this service is governed by the support plan arrangements made between Thales and your organization. Please consult this support plan for further information about your entitlements, including the hours when telephone support is available to you.

Customer Support Portal

The Customer Support Portal, at https://supportportal.thalesgroup.com, is where you can find solutions for most common problems. The Customer Support Portal is a comprehensive, fully searchable database of support resources, including software and firmware downloads, release notes listing known problems and workarounds, a knowledge base, FAQs, product documentation, technical notes, and more. You can also use the portal to create and manage support cases.

> **NOTE** You require an account to access the Customer Support Portal. To create a new account, go to the portal and click on the **REGISTER** link.

Telephone

The support portal also lists telephone numbers for voice contact (Contact Us).

CHAPTER 1: Overview

CADP for C PKCS#11 enables data encryption at the application level. Applications can use CADP for C PKCS#11 to encrypt a column in a database or a field, such as credit card numbers or Social Security numbers. CADP for C PKCS#11 can also be used to encrypt an entire data file or directory.

CADP for C PKCS#11 APIs support real-time I/O encryption and decryption of “at rest” data and log files. Data is encrypted by adding CADP for C PKCS#11 API calls to existing applications.

The APIs in the CADP for C PKCS#11 library are a subset of the PKCS #11 specification version 2.20 with a focus on session management, key management, and cryptographic functions. These APIs are platform independent to cryptographic tokens, and are traditionally used for HSMs (hardware security modules) and smartcards.

The CADP for C PKCS#11 library acts as a bridge between an application written in the PKCS#11 protocol and the CipherTrust Manager (Key Manager). Your application makes calls with the industry standard PKCS11 APIs and the calls are translated to NAE-XML to communicate with the CipherTrust Manager. In addition, the CADP for C PKCS#11 library also offers an in-memory key cache for enhanced cryptographic performance.

CADP for C PKCS#11 supports single encryption and decryption using RSA 1024, RSA 2048, RSA 3072, RSA4096, AES 128, AES 192, and AES 256. It also supports multi-part encryption and decryption using AES 128, AES 192, and AES 256.

CADP for C PKCS#11 provides API support for the following:

- Create a key
- Find a key
- Destroy a key
- Export a key
- Import a key
- Versioned key support
- Encrypt data
- Decrypt data
- Sign data
- Verify data signature
- Compute the digest for data
- Compute the digest for data with a key (HMAC)
- Random number generation

> **NOTE** For Cryptographic operations supported in PKCS#11, the maximum data limit defined is 3500 bytes. For more details, refer to https://thalesdocs.com/ctp/cm/2.6/reference/xml/crypto-ops/index.html#the-data-elements

CADP for C PKCS#11 code samples for C, C#, and Java are available at the following Github site:

https://github.com/thalescpl-io/CipherTrust_Application_Protection/tree/master/pkcs11

For information about latest version of the PKCS #11 specification, refer to the following OASIS website:

http://docs.oasis-open.org/pkcs11/pkcs11-base/v2.40/os/pkcs11-base-v2.40-os.html

This document describes the CADP for C PKCS#11 APIs available within the CADP for C PKCS#11. Application developers can use these APIs to integrate the CADP for C PKCS#11 functionality into their applications.

The individual APIs are organized by chapter into following related functional groups:

Appendix A ("Mapping of VAE Compatibility Mode and CipherTrust Mode APIs" on page 112) summarizes the differences between the PKCS#11 APIs of the Vormetric Application Encryption (VAE) (in binary compatibility mode) and the CipherTrust Manager.

CHAPTER 2: General Purpose Functions

This chapter describes the following general-purpose functions:

C_Initialize

Initializes the CADP for C PKCS#11 library.

```
CK_DEFINE_FUNCTION(CK_RV, C_Initialize)(
CK_VOID_PTR pInitArgs
);
```

> **NOTE** Only the applications written in C will call `C_Initialize`.

> **NOTE** Application level or OS level mutexes are currently not supported.

> **NOTE** For information about the APIs, see the CADP for C PKCS#11 code samples.

Input Parameters	Description
`pInitArgs`	Optional parameters for `C_Initialize`. Currently, these parameters are not supported. `pInitArgs` must be set to `NULL`. Previous settings used in VAE PKCS11 are currently ignored.

Output Parameters	Description
`CKR_ARGUMENTS_BAD`	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
`CKR_CRYPTOKI_ALREADY_INITIALIZED`	The function could not be executed because the Cryptoki library is already initialized by a call to `C_Initialize`.
`CKR_FUNCTION_FAILED`	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
`CKR_GENERAL_ERROR`	An unrecoverable error occurred.
`CKR_HOST_MEMORY`	There is insufficient memory to perform the requested function.
`CKR_OK`	The function was executed successfully.

C_Finalize

Called to indicate that an application is finished with using the `pkcs11` library.

```
CK_DEFINE_FUNCTION(CK_RV, C_Finalize)(
CK_VOID_PTR pReserved
);
```

Input Parameters	Description
`CK_VOID_PTR pReserved`	This value must be set to `NULL`.

Return Values	Description
`CKR_ARGUMENTS_BAD`	Generic error code indicates that the arguments supplied to the Cryptoki function are invalid.
`CKR_CRYPTOKI_NOT_INITIALIZED`	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
`CKR_FUNCTION_FAILED`	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
`CKR_GENERAL_ERROR`	An unrecoverable error occurred.
`CKR_OK`	The function was executed successfully.

C_GetInfo

Provides manufacturer and version information about the Cryptoki library.

```
CK_DEFINE_FUNCTION(CK_RV, C_GetInfo)(CK_INFO_PTR pInfo);
```

Input Parameters	Description
CK_INFO_PTR pInfo	Pointer to a CK_INFO structure to receive the information.

Output Parameters	Description
CK_INFO	Provides general information about Cryptoki. Definitions are outlined in the next table.

Cryptoki Responses	Description
cryptokiVersion	Cryptoki interface version number (for compatibility with future revisions of this interface).
manufacturerID	ID of the Cryptoki library manufacturer. Must be padded with the blank character (' ') and must not be terminated with a null.
flags	Bit flags reserved for future versions. This value must be set to zero (0) for this version.
libraryDescription	Character-string description of the library. Must be padded with the blank character (' ') and must not be terminated with a null.
libraryVersion	Cryptoki library version number.

Return Values	Description
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_GENERAL_ERROR	An unrecoverable error occurred.

Return Values	Description
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_OK	The function was executed successfully.

C_GetFunctionList

Allows the application (caller of the CADP for C PKCS#11 library) to find the list of the supported APIs and their addresses.

```
CK_DEFINE_FUNCTION(CK_RV, C_GetFunctionList)(
CK_FUNCTION_LIST_PTR_PTR ppFunctionList
);
```

> **NOTE** Only the applications written in C will call `C_GetFunctionList`.

Input Parameters	Description
`CK_FUNCTION_LIST_PTR_PTR ppFunctionList;`	Pointer to a value that will receive a pointer to `CK_FUNCTION_LIST` structure. This structure contains function pointers for all the API routines in the library.

Output Parameters	Description
`ppFunctionList`	Filled in with the address of the list of function pointers from the CADP for C PKCS#11 library.

Return Values	Description
`CKR_ARGUMENTS_BAD`	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
`CKR_FUNCTION_FAILED`	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
`CKR_GENERAL_ERROR`	An unrecoverable error occurred.
`CKR_HOST_MEMORY`	There is insufficient memory to perform the requested function.
`CKR_OK`	The function was executed successfully.

CHAPTER 3: Slot and Token Management Functions

This chapter describes the following functions for slot and token management:

C_GetSlotList

Provides list of available slots. In C, call this function twice to get the actual slot list. The first time, it returns the number of available slots. Allocate memory to the available slots, and then the second call returns the actual slot list.

```
CK_DEFINE_FUNCTION(CK_RV, C_GetSlotList)(
CK_BBOOL tokenPresent,
CK_SLOT_ID_PTR pSlotList,
CK_ULONG_PTR pulCount);
);
```

> **NOTE** Only the applications written in C will call `C_GetSlotList`.

First Call to Retrieve Number of Slots

First call to `C_GetSlotList(CK_FALSE)` to retrieve the number of slots.

Input Parameters	Description
`CK_BBOOL tokenPresent`	Indicates whether the list obtained includes only those slots with a token present (`CK_TRUE`), or all slots (`CK_FALSE`). This parameter must set to `CK_FALSE`.
`CK_SLOT_ID_PTR pSlotList`	Pointer to the space allocated to hold the slot. This parameter must be set to `NULL`.
`CK_ULONG_PTR pulCount`	Pointer to an unsigned long to hold slot count.

Second Call to Retrieve Slot List

Second call to `C_GetSlotList` to retrieve the actual slot list.

Input Parameters	Description
`CK_BBOOL tokenPresent`	Indicates whether the list obtained includes only those slots with a token present (`CK_TRUE`), or all slots (`CK_FALSE`). This parameter must set to `CK_TRUE`.
`CK_SLOT_ID_PTR pSlotList`	Pointer to the space allocated to hold the slot based on the count retrieved from the first call.
`CK_ULONG_PTR pulCount`	Pointer to a `CK_ULONG` holding the number of slots allocated from the first call.

Output Parameters	Description
`pulCount`	Points to the number of slots available. It is always 1 for the CADP for C PKCS#11 library.

Return Values	Description
CKR_BUFFER_TOO_SMALL	The output of the function is too large to fit in the supplied buffer.
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_OK	The function was executed successfully.
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.

C_GetSlotInfo

Provides information about a particular slot in the system.

```
CK_DEFINE_FUNCTION(CK_RV, C_GetSlotInfo)(
CK_SLOT_ID slotID,
CK_SLOT_INFO_PTR pInfo
);
```

Input Parameters	Description
CK_SLOT_ID slotID	Indicates the ID of the slot.
CK_SLOT_INFO_PTR pInfo	Pointer to a CK_SLOT_INFO object.

Output Parameters	Description
CK_SLOT_INFO_PTR pInfo	The structure where CK_SLOT_INFO_PTR pointer will be filled in with relevant information about the slot.

Return Values	Description
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_OK	The function was executed successfully.
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_SLOT_ID_INVALID	The specified slot ID is invalid.

C_GetTokenInfo

Provides information about a specific token.

```
CK_DEFINE_FUNCTION(CK_RV, C_GetTokenInfo)(
   CK_SLOT_ID slotID,
   CK_SLOT_INFO_PTR pInfo );
```

Input Parameters	Description
CK_SLOT_ID slotID	Indicates the ID of the slot. This value must be between 0 to 127.
CK_TOKEN_INFO_PTR pInfo	Pointer to a CK_TOKEN_INFO object.

Output Parameters	Description
CK_TOKEN_INFO_PTR	The structure where CK_TOKEN_INFO_PTR pointer will be filled in with relevant information about the token.

Return Values	Description
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_OK	The function was executed successfully.
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_SLOT_ID_INVALID	The specified slot ID is invalid.
CKR_TOKEN_NOT_PRESENT	The token was not present in its slot at the time that the function was invoked.
CKR_TOKEN_NOT_RECOGNIZED	The Cryptoki library and/or slot does not recognize the token in the slot.

C_GetMechanismList

Provides a list of supported mechanisms.

```
CK_DEFINE_FUNCTION(CK_RV,C_GetMechanismList)(
CK_SLOT_ID slotID, CK_MECHANISM_TYPE_PTR pMechanismList,
CK_ULONG_PTR pulCount
);
```

> **NOTE** For the list of mechanisms returned by `C_GetMechanismList`, refer "Supported Mechanisms" on page 17.

Input Parameters	Description
`CK_SLOT_ID slotID`	Indicates the ID of the slot. This value must be between 0 to 127.
`CK_MECHANISM_TYPE_PTR pMechanismList`	Pointer to memory to hold a list of `CK_MECHANISM_TYPE`.
`CK_ULONG_PTR pulCount`	Number of `CK_MECHANISM_TYPE` objects the memory above can hold.

Output Parameters	Description
`CK_MECHANISM`	The `CK_MECHANISM` structure where `CK_MECHANISM_TYPE_PTR` pointer will be filled in with the list of supported mechanism types.

Return Values	Description
`CKR_ARGUMENTS_BAD`	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
`CKR_FUNCTION_FAILED`	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
`CKR_GENERAL_ERROR`	An unrecoverable error occurred.
`CKR_HOST_MEMORY`	There is insufficient memory to perform the requested function.
`CKR_OK`	The function was executed successfully.
`CKR_CRYPTOKI_NOT_INITIALIZED`	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
`CKR_SLOT_ID_INVALID`	The specified slot ID is invalid.

Return Values	Description
CKR_TOKEN_NOT_PRESENT	The token was not present in its slot at the time that the function was invoked.
CKR_TOKEN_NOT_RECOGNIZED	The Cryptoki library and/or slot does not recognize the token in the slot.
CKR_BUFFER_TOO_SMALL	The output of the function is too large to fit in the supplied buffer.

C_GetMechanismInfo

Provides information about a particular mechanism.

```
CK_DEFINE_FUNCTION(CK_RV, C_GetMechanismInfo)(
CK_SLOT_ID slotID, CK_MECHANISM_TYPE type,
CK_MECHANISM_INFO_PTR pInfo
);
```

Input Parameters	Description
CK_SLOT_ID slotID	Indicates the ID of the slot. This value must be between 0 to 127.
CK_MECHANISM_TYPE type	Mechanism type for which to retrieve the information.
CK_MECHANISM_INFO_PTR pInfo	Pointer to a CK_MECHANISM_INFO structure.

Output Parameters	Description
CK_MECHANISM_INFO	The CK_MECHANISM_INFO structure where pInfo points is filled in with information about a particular mechanism.

Return Values	Description
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_OK	The function was executed successfully.
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_SLOT_ID_INVALID	The specified slot ID is invalid.
CKR_TOKEN_NOT_PRESENT	The token was not present in its slot at the time that the function was invoked.
CKR_TOKEN_NOT_RECOGNIZED	The Cryptoki library and/or slot does not recognize the token in the slot.

Return Values	Description
CKR_BUFFER_TOO_SMALL	The output of the function is too large to fit in the supplied buffer.
CKR_MECHANISM_INVALID	An invalid mechanism was specified in the cryptographic operation.

Supported Mechanisms

List of supported mechanisms	
CKM_AES_CBC	CKM_AES_GCM
CKM_AES_CBC_PAD	CKM_AES_CTR
CKM_AES_ECB	CKM_AES_KEY_GEN
CKM_RSA_PKCS	CKM_RSA_PKCS_KEY_PAIR_GEN
CKM_SHA256_RSA_PKCS	CKM_SHA1_RSA_PKCS
CKM_SHA512_RSA_PKCS	CKM_SHA384_RSA_PKCS
CKM_SHA1_RSA_PKCS	CKM_SHA_1_HMAC
CKM_SHA256_HMAC	CKM_SHA384_HMAC
CKM_SHA512_HMAC	CKM_THALES_FF1
CKM_THALES_FPE	CKM_THALES_FF3_1

CHAPTER 4: Session Management Functions

This chapter describes the following session management functions:

C_OpenSession

Starts a cryptographic session with a specific slot.

```
CK_DEFINE_FUNCTION(CK_RV, C_OpenSession)(
CK_SLOT_ID slotID,
CK_FLAGS flags, CK_VOID_PTR pApplication,
CK_NOTIFY Notify,
CK_SESSION_HANDLE_PTR phSession
);
```

Supported Functionality

- Only read/write sessions are supported. Maximum number of read/write sessions is 65536.
- No read-only sessions are supported.
- Each session is bound to one distinct thread.

Input Parameters	Description
`CK_SLOT_ID slotID`	Indicates the ID of the slot. This value must be between 0 to 127.
`CK_FLAGS flags`	Indicates the type of session. Consists of the logical OR of zero or more bit flags defined in the `CK_SESSION_INFO` data type. Thales only supports 0.
`CK_VOID_PTR pApplication`	Indicates an application-defined pointer to be passed to the notification callback. Thales does **not** support this parameter. This value must be set to `NULL`.
`CK_NOTIFY Notify`	Indicates the address of the notification callback function. This value must be set to `NULL`.
`CK_SESSION_HANDLE_PTR phSession`	Pointer to the location that receives the handle for the new session.

Output Parameters	Description
`CK_SESSION_HANDLE_PTR phSession`	The session ID pointer (`CK_SESSION_HANDLE_PTR phSession`) will be filled in with the session handle.

Return Values	Description
`CKR_ARGUMENTS_BAD`	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.

Return Values	Description
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_OK	The function was executed successfully.
CKR_SLOT_ID_INVALID	The specified slot ID is invalid.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.

C_CloseSession

Closes a cryptographic session.

```
CK_DEFINE_FUNCTION(CK_RV, C_CloseSession)(
   CK_SESSION_HANDLE hSession
);
```

Input Parameters	Description
CK_SESSION_HANDLE hSession	Session to close, identified by the session handle.

Return Values	Description
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_OK	The function was executed successfully.
CKR_SLOT_ID_INVALID	The specified slot ID is invalid.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.

C_CloseAllSessions

Closes all sessions on a slot. This function is only applicable to the applications written in C.

```
CK_DEFINE_FUNCTION(CK_RV, C_CloseAllSessions)(
   CK_SLOT_ID slotID
);
```

Input Parameters	Description
CK_SLOT_ID slotID	Indicates the ID of the slot. This value must be set to 0.

Return Values	Description
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_OK	The function was executed successfully.

C_GetSessionInfo

Provides information about a session.

```
CK_DEFINE_FUNCTION(CK_RV, C_GetSessionInfo)(
   CK_SESSION_HANDLE hSession,
   CK_SESSION_INFO_PTR pInfo
);
```

Input Parameters	Description
CK_SESSION_HANDLE hSession	Session handle.
CK_SESSION_INFO_PTR pInfo	Pointer to a CK_SESSION_INFO object.

Output Parameters	Description
CK_SESSION_INFO_PTR	The structure where CK_SESSION_INFO_PTR pointer will be filled in with relevant information about the session.

Return Values	Description
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_OK	The function was executed successfully.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.

C_Login

Passes user credentials to log into a slot. The credentials can be PIN only for global user access. Include a user name and password to secure the session access based on key group permissions. The format of the credential depends on the compatibility mode.

```
CK_DEFINE_FUNCTION(CK_RV, C_Login)(
CK_SESSION_HANDLE hSession,
CK_USER_TYPE userType,
CK_UTF8CHAR_PTR pPin,
CK_ULONG ulPinLen
);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle into which to log the user.
`CK_USER_TYPE userType`	Indicates the user type. CADP for C only supports `CKU_USER`.
`CK_CHAR_PTR pPin`	Pointer to the credential used to authenticate the user against the CipherTrust Manager. The format of this argument depends on one of the following compatibility modes: > LegacyVAE mode: PIN:CTuser:password > CipherTrust mode: CTuser:password where the following applies: > PIN: PIN entered during connector registration > CTUser: CipherTrust Manager user > password: Password of CipherTrust Manager user **NOTE** If you want to use domain keys: > **CTUser**: DomainName||CipherTrust Manager user
`CK_ULONG ulPinLen`	Indicates the length of the PIN in bytes.

Return Values	Description
`CKR_ARGUMENTS_BAD`	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
`CKR_CRYPTOKI_NOT_INITIALIZED`	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
`CKR_FUNCTION_FAILED`	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
`CKR_GENERAL_ERROR`	An unrecoverable error occurred.

Return Values	Description
CKR_OK	The function was executed successfully.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_FUNCTION_REJECTED	This error is returned when the identity check fails.
CKR_PIN_INVALID	The specified PIN has invalid characters in it. This is the default error when a check for `loginforIdentity` fails.
CKR_PIN_INCORRECT	The specified PIN is incorrect.
CKR_USER_PIN_NOT_INITIALIZED	Indicates that the normal user's PIN has not yet been initialized.
CKR_USER_TYPE_INVALID	An invalid value was specified for the user type.

C_Logout

Logs a user out from a token. You can call `C_logout()` multiple times in a row.

```
CK_DEFINE_FUNCTION(CK_RV, C_Logout)(
   CK_SESSION_HANDLE hSession
);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle that you want to log out of.

Return Values	Description
`CKR_ARGUMENTS_BAD`	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
`CKR_CRYPTOKI_NOT_INITIALIZED`	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
`CKR_FUNCTION_FAILED`	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
`CKR_GENERAL_ERROR`	An unrecoverable error occurred.
`CKR_OK`	The function was executed successfully.
`CKR_SESSION_HANDLE_INVALID`	The specified session handle was invalid at the time that the function was invoked.
`CKR_SESSION_CLOSED`	The session was closed during the execution of the function.
`CKR_HOST_MEMORY`	There is insufficient memory to perform the requested function.

CHAPTER 5: callObject Management Functions

This chapter describes the following functions for managing objects:

C_WrapKey

Supports wrapping of a key with another key. It supports the following cases:

Wrap cases:

- Wrap a symmetric key with another symmetric key
- Wrap a symmetric key with an asymmetric key (public)
- Wrap an asymmetric public key with a symmetric key
- Wrap an asymmetric private key with a symmetric key

Wrap with no wrapping key:

- Asymmetric public key with no wrapping key
- Asymmetric private key with no wrapping key
- Symmetric key with no wrapping key

```
CK_DEFINE_FUNCTION(CK_RV, C_WrapKey)(
CK_SESSION_HANDLE hSession,
CK_MECHANISM_PTR pMechanism,
CK_OBJECT_HANDLE hWrappingKey,
CK_OBJECT_HANDLE hKey,
CK_BYTE_PTR pWrappedKey,
CK_ULONG_PTR pulWrappedKeyLen
);
```

The `C_WrapKey` only succeeds when `CKA_EXTRACTABLE` is true for the wrapping key and the key to be wrapped.

> **NOTE** For asymmetric keys, `C_WrapKey` is supported in local mode only.

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.

`CK_MECHANISM_PTR pMechanism`	If the wrapping key is:
	- **Symmetric key**: The mechanism should be `CKM_AES_CBC_PAD`. It requires a 16-byte Initialization Vector (IV). The structure looks like this:
	`{CKM_AES_CBC_PAD,iv,16};`
	- **Asymmetric key**: The mechanism should be CKM_RSA_PKCS. The structure looks like this:
	`{CKM_RSA_PKCS, NULL, 0};`

`CK_OBJECT_HANDLE hWrappingKey`	Key used to wrap another key.
`CK_OBJECT_HANDLE hKey`	Key or Opaque Object to export.
`CK_BYTE_PTR pWrappedKey`	Points to an array to put the wrapped key bytes.

Input Parameters	Description
CK_ULONG_PTR pulWrappedKeyLen	Points to the location that receives the length of the wrapped key.

Constants for CK_MECHANISM	Description
CKM_AES_CBC_PAD	> Exports a symmetric key wrapped with a symmetric key. > Exports an asymmetric key wrapped with a symmetric key.
CKM_RSA_PKCS	Exports a symmetric key wrapped with an asymmetric key.

Output Parameters	Description
CK_BYTE_PTR pWrappedKey	The array where CK_BYTE_PTR pWrappedKey pointer will be filled in with the wrapped key bytes.

Return Values	Description
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_FUNCTION_CANCELLED	The function was canceled in mid-execution.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_OK	The function was executed successfully.
CKR_KEY_SIZE_RANGE	The supplied key‘s size is outside the range of key sizes.
CKR_MECHANISM_INVALID	An invalid mechanism was specified in the cryptographic operation.
CKR_MECHANISM_PARAM_INVALID	Invalid parameters were supplied to the mechanism specified in the cryptographic operation.

Return Values	Description
CKR_OPERATION_ACTIVE	There is already an active operation (or combination of active operations) which prevents Cryptoki from activating the specified operation.
CKR_PIN_EXPIRED	The specified PIN has expired, and the requested operation cannot be carried out.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_USER_NOT_LOGGED_IN	The desired action cannot be performed because the appropriate user (or an appropriate user) is not logged in.
CKR_WRAPPING_KEY_HANDLE_ INVALID	Indicates that the key handle specified to be used to wrap another key is not valid.
CKR_WRAPPING_KEY_SIZE_RANGE	Indicates that although the requested wrapping operation could in principle be carried out, the token is unable to actually do it because the supplied wrapping key's size is outside the range of key sizes that it can handle.
CKR_WRAPPING_KEY_TYPE_ INCONSISTENT	Indicates that the type of the key specified to wrap another key is not consistent with the mechanism specified for wrapping.

C_UnwrapKey

Used in conjunction with `C_WrapKey`, where a key is wrapped with another key and then exported. `C_UnwrapKey` completes the `C_WrapKey` feature, allowing the exported key to be imported to a different CipherTrust Manager (or the same CipherTrust Manager with a different name/label) and ready for future use from the destination.

When exporting and importing the key into a new CipherTrust Manager, the key attributes are not preserved. Only the key material is imported. The new template specifies the key attributes.

`C_UnwrapKey` supports the following cases:

- Unwrap wrapped symmetric key bytes with an asymmetric private key
- Unwrap wrapped symmetric key bytes with a symmetric key
- Unwrap wrapped asymmetric public key bytes with a symmetric key
- Unwrap wrapped asymmetric private key bytes with a symmetric key

Refer to the code sample `vpkcs11_sample_import_key.c` to use `C_UnwrapKey`.

```
(CK_RV, C_UnwrapKey)
( CK_SESSION_HANDLE hSession,
CK_MECHANISM_PTR pMechanism,
CK_OBJECT_HANDLE hUnwrappingKey,
CK_BYTE_PTR pWrappedKey,
CK_ULONG ulWrappedKeyLen,
CK_ATTRIBUTE_PTR pTemplate,
CK_ULONG ulAttributeCount,
CK_OBJECT_HANDLE_PTR phKey
);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.

`CK_MECHANISM_PTR pMechanism`	If the unwrapping key is:
	- **Symmeteric key**: The mechanism should be `CKM_AES_CBC_PAD`. It requires a 16-byte Initialization Vector (IV). The structure looks like this:
	`{CKM_AES_CBC_PAD,iv,16};`
	- **Asymmetric key**: The mechanism should be `CKM_RSA_PKCS`. The structure looks like this:
	`{CKM_RSA_PKCS, NULL, 0};`

`CK_OBJECT_HANDLE hUnwrappingKey`	Key used to wrap another key.
`CK_BYTE_PTR pWrappedKey`	Points to an array to hold the wrapped key bytes as input.
`CK_ULONG_PTR ulWrappedKeyLen`	Length of the above array.

Input Parameters	Description
CK_ATTRIBUTE_PTR pTemplate	Pointer to the new key template such as the ImportKeyTemplate portion of pkcs11_sample_import_key.c.
CK_ULONG ulAttributeCount	Template length (number of attributes).
CK_OBJECT_HANDLE_PTR phKey	Returns pointer to a new handle.

Template Attribute Name	Type	Value
CKA_LABEL	CK_UTF8CHAR	Required. Key name; also displayed on the CipherTrust Manager.
CKA_APPLICATION	CK_UTF8CHAR	Description of the application generating the key name. This value can be NULL.
CKA_CLASS	CK_OBJECT_CLASS	Required. This value must be set to CKO_SECRET_KEY, CKO_PUBLIC_KEY, or CKO_PRIVATE_KEY.
CKA_KEY_TYPE	CK_KEY_TYPE	Required. This value must be set to CKK_AES or CKK_RSA.
CKA_VALUE_LEN	CK_ULONG	Length of the key imported, passed in bytes.
CKA_TOKEN	CK_BBOOL	Required. This value must be set to true.
CKA_ENCRYPT	CK_BBOOL	The default value is set to true.
CKA_DECRYPT	CK_BBOOL	The default value is set to true.
CKA_SIGN	CK_BBOOL	The default value is set to true.
CKA_VERIFY	CK_BBOOL	The default value is set to true.
CKA_WRAP	CK_BBOOL	The default value is set to true.
CKA_UNWRAP	CK_BBOOL	The default value is set to true.
CKA_EXTRACTABLE	CK_BBOOL	In LegacyVAE mode (formerly called VAE mode), the default value is set to true. Otherwise, this value is set to false.
CKA_ALWAYS_SENSITIVE	CK_BBOOL	The default value is set to true.

Template Attribute Name	Type	Value
CKA_NEVER_EXTRACTABLE	CK_BBOOL	The default value is set to true.
CKA_SENSITIVE	CK_BBOOL	The default value is set to true.
CKA_MODIFIABLE	CKK_BBOOL	In LegacyVAE mode (formerly called VAE mode), the default value is set to true. Otherwise, this value is set to false.

Output Parameters	Description
CK_OBJECT_HANDLE_PTR phKey	Returns pointer to a new handle that was imported into either the new CipherTrust Manager or the old CipherTrust Manager with a new key name.

Return Values	Description
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_TEMPLATE_INCOMPLETE	The template specified for creating an object is incomplete and lacks some necessary attributes.
CKR_ATTRIBUTE_VALUE_INVALID	An invalid value was specified for a particular attribute in a template.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_OPERATION_NOT_INITIALIZED	There is no active operation of an appropriate type in the specified session.
CKR_DATA_LEN_RANGE	The plaintext input data to a cryptographic operation has a bad length.
CKR_OBJECT_HANDLE_INVALID	The specified object handle is not valid.
CKR_KEY_FUNCTION_NOT_PERMITTED	An attempt has been made to use a key for a cryptographic purpose that the key's attributes are not set to allow it to do.

Return Values	Description
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_WRAPPED_KEY_LEN_RANGE	Indicates that the provided wrapped key can be seen to be invalid solely on the basis of its length.
CKR_USER_NOT_LOGGED_IN	The desired action cannot be performed because the appropriate user (or an appropriate user) is not logged in.

C_CreateObject

Imports key bytes into the CipherTrust Manager and creates a symmetric key or an asymmetric key on the CipherTrust Manager. Maximum length of the key name is 256 characters. Minimum length is of 1 character.

```
CK_DEFINE_FUNCTION(CK_RV, C_CreateObject)(
CK_SESSION_HANDLE hSession,
CK_ATTRIBUTE_PTR pTemplate,
CK_ULONG ulCount,
CK_OBJECT_HANDLE_PTR phObject
);
```

Input Parameters	Description
CK_SESSION_HANDLE hSession	Session handle.
CK_ATTRIBUTE pTemplate	Same template as C_GenerateKey, with the addition of CKA_VALUE and CKA_VALUE_LEN, which hold the key bytes and the length of the key bytes, respectively. > While importing an opaque object, if the CKA_THALES_OBJECT_IKID in the input template pTemplate is missing, a default value will not be generated by the Ciphertrust Manager. > While importing an opaque object, if the CKA_THALES_OBJECT_MUID in the input template pTemplate is missing, a default value will not be generated by the Ciphertrust Manager.
CK_ULONG ulCount	Number of attributes in pTemplate.
CK_OBJECT_HANDLE phObject	Generated key handle.

Template Attribute Name	Type	Value
CKA_LABEL	CK_UTF8CHAR	Required. Key name; also displayed on the CipherTrust Manager.
CKA_APPLICATION	CK_UTF8CHAR	Description of the application generating the key name. This value can be NULL.
CKA_CLASS	CK_OBJECT_CLASS	This value must be set to: > CKO_SECRET_KEY for importing symmetric keys > CKO_THALES_OPAQUE_OBJECT for importing an opaque object > CKO_PUBLIC_KEY or CKO_PRIVATE_KEY for importing asymmetric keys

Template Attribute Name	Type	Value
CKA_KEY_TYPE	CK_KEY_TYPE	Required. This value can be set to any of these keytypes: CKK_AES, CKK_SHA256_HMAC, CKK_RSA
CKA_VALUE_LEN	CK_ULONG	Length of key bytes. Pass a long value to the CK_ATTRIBUTE constructor when creating the CKA_VALUE_LEN attribute. For example: new CK_ATTRIUBTE(CKA_VALUE_LEN, 32L) or: long len = 32; new CK_ATTRIBUTE(CKA_VALUE_LEN, len)
CKA_TOKEN	CK_BBOOL	Required. This value must be set to true.
CKA_ENCRYPT	CK_BBOOL	The default value is set to true. Set to false for an opaque object.
CKA_DECRYPT	CK_BBOOL	The default value is set to true. Set to false for an opaque object.
CKA_SIGN	CK_BBOOL	The default value is set to true.
CKA_VERIFY	CK_BBOOL	The default value is set to true.
CKA_WRAP	CK_BBOOL	The default value is set to true.
CKA_UNWRAP	CK_BBOOL	The default value is set to true.
CKA_EXTRACTABLE	CK_BBOOL	In LegacyVAE mode (formerly called VAE mode), the default value is set to true. Otherwise, this value is set to false.
CKA_ALWAYS_SENSITIVE	CK_BBOOL	The default value is set to false.
CKA_NEVER_EXTRACTABLE	CK_BBOOL	In LegacyVAE mode (formerly called VAE mode), the default value is set to false. Otherwise, this value is set to true.
CKA_SENSITIVE	CK_BBOOL	In LegacyVAE mode (formerly called VAE mode), the default value is set to false. Otherwise, this value is set to true.
CKA_MODIFIABLE	CK_BBOOL	In LegacyVAE mode (formerly called VAE mode), the default value is set to true. Otherwise, this value is set to false.
CKA_ID	UTF8CHAR	A searchable non-unique ID.

Template Attribute Name	Type	Value

CKA_THALES_OBJECT_ALIAS	CK_UTF8CHAR	Alias of the key. There can be multiple aliases. For example, alias1, alias 2, and so on.
		`{CKA_THALES_OBJECT_ALIAS, "alias1", strlen("alias1")},`
		`{CKA_THALES_OBJECT_ALIAS, "alias2", strlen("alias2")},`
		`{CKA_THALES_OBJECT_ALIAS, "alias3", strlen("alias3")},`
		NOTE For asymmetric keys, alias is not supported.

Output Parameters	Description
CK_OBJECT_HANDLE hGenKey	This parameter is filled in with the handle of the newly generated key.

Return Values	Description
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_TEMPLATE_INCOMPLETE	The template specified for creating an object is incomplete and lacks some necessary attributes.
CKR_ATTRIBUTE_VALUE_INVALID	An invalid value was specified for a particular attribute in a template.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_OBJECT_HANDLE_INVALID	The specified object handle is not valid.
CKR_USER_NOT_LOGGED_IN	The desired action cannot be performed because the appropriate user (or an appropriate user) is not logged in.

C_DestroyObject

Deletes a key.

```
CK_DEFINE_FUNCTION(CK_RV, C_DestroyObject)(
CK_SESSION_HANDLE hSession,
CK_OBJECT_HANDLE hObject
);
```

****WARNING** Use this function VERY CAREFULLY. Deleted keys cannot be recovered and all data encrypted by that key will be lost.**

NOTE

- To delete an entire chain of a versioned key, you must set the state of each key to `Restricted` (`Deactivated`) using the `C_SetAttributeValue` function and then call the `C_DestroyObject` function. This function will delete the entire chain of versioned keys.
- If you wish to destroy a specific version key, destroy it from the CipherTrust Manager UI. Refer to the CipherTrust Manager product documentation for instructions.
- If an asymmetric key is generated through `C_GenerateKeyPair`, deletion of one key handle (public or private) will delete another key handle as well as delete the whole key from the CipherTrust Manager.

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_OBJECT_HANDLE hObject`	Handle of the key to be deleted.

Return Values	Description
`CKR_CRYPTOKI_NOT_INITIALIZED`	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
`CKR_GENERAL_ERROR`	An unrecoverable error occurred.
`CKR_HOST_MEMORY`	There is insufficient memory to perform the requested function.
`CKR_OBJECT_HANDLE_INVALID`	The specified object handle is not valid.
`CKR_OK`	The function was executed successfully.
`CKR_PIN_EXPIRED`	The specified PIN has expired, and the requested operation cannot be carried out.

Return Values	Description
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_SESSION_READ_ONLY	The specified session was unable to accomplish the desired action because it is a read-only session.
CKR_TOKEN_WRITE_PROTECTED	The requested action could not be performed because the token is write-protected.

C_FindObjectsInit

Initializes a search for a token and session objects that match a template.

```
CK_DEFINE_FUNCTION(CK_RV, C_FindObjectsInit)(
CK_SESSION_HANDLE hSession,
CK_ATTRIBUTE_PTR pTemplate,
CK_ULONG ulCount
);
```

> **NOTE** This function has limited scope and can only perform exact matches.

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_ATTRIBUTE_PTR pTemplate`	Attribute(s) for which to search. The following attributes are supported: > `CKA_LABEL` > `CKA_THALES_CACHED_CLEAR` > `CKA_THALES_OBJECT_UUID` > `CKA_THALES_OBJECT_MUID` > `CKA_THALES_OBJECT_IKID` > `CKA_ID` > `CKA_THALES_KEY_VERSION`: This attribute is used in conjunction with `CKA_LABEL` to find a specific version of a key. > `CKA_KEY_TYPE` > `CKA_THALES_OBJECT_ALIAS`: You can search a key by an alias by specifying the alias in the object template. For example, `{CKA_THALES_OBJECT_ALIAS, "alias1", strlen("alias1")}` **NOTE** > Finding the symmetric key with `CKA_THALES_OBJECT_ALIAS` is partially supported as it returns the most active version of the key. > Finding the asymmetric key with `CKA_THALES_OBJECT_ALIAS` is not supported.
`CK_ULONG ulCount`	Attribute template count.

NOTE

> **Asymmetric Keys**: The Asymmetric Keys includes:
>
> - **LegacyVAE (formerly called VAE mode)** : The CM honors the public key name. To query for a public key, pass in the public key name and class as `CKO_PUBLIC_KEY`. To query for the private key, pass in the public key name and class as `CKO_PRIVATE_KEY`.
> - **CipherTrust Keys**: The CM honors the private key name. To query for a public key, pass in the private key name and class as `CKO_PUBLIC_KEY`. To query for the private key, pass in the private key name and class as `CKO_PRIVATE_KEY`.

The following code shows various attributes for which you can search:

```
CK_ATTRIBUTE findKeyTemplatePass[] =
{
{CKA_LABEL, ksid, ksid_len},
{CKA_CLASS, &keyType, sizeof(keyType)}
};
switch(keyidType)
{
case keyIdLabel:
findKeyTemplatePass[0].type = CKA_LABEL;
 break;

case keyIdUuid:
findKeyTemplatePass[0].type = CKA_THALES_OBJECT_UUID;
 break;

case keyIdMuid:
findKeyTemplatePass[0].type = CKA_THALES_OBJECT_MUID;
 break;

case keyIdImport:
findKeyTemplatePass[0].type = CKA_THALES_OBJECT_IKID;
 break;

case keyIdAlias:
findKeyTemplatePass[0].type = CKA_THALES_OBJECT_ALIAS;
 break;

case keyId:
findKeyTemplatePass[0].type = CKA_ID;
 break;

case keyIdKeyType:
findKeyTemplatePass[0].type = CKA_KEY_TYPE;
 break;

}
```

Template Attribute Name	Type	Value
`CKA_LABEL`	`CK_UTF8CHAR`	Key name; also displayed on the CipherTrust Manager.

Template Attribute Name	Type	Value
CKA_THALES_OBJECT_UUID	CK_UTF8CHAR	The universally unique identifier (UUID) of a key.
CKA_THALES_OBJECT_MUID	CK_UTF8CHAR	The Message Unit Identifier (MUID) of a key.
CKA_THALES_OBJECT_IKID	CK_UTF8CHAR	Key ID of the key.
CKA_THALES_OBJECT_ALIAS	CK_UTF8CHAR	Alias of the key.
CKA_ID	CK_UTF8CHAR	Non-unique ID associated to a key. Using this attribute will return all the objects that have **CKA_ID** set to the given value. **NOTE** For keys migrated from DSM, CipherTrust Manager stores the CKA_ID value in Base64 encoding instead of ASCII format. In the C_GetAttributeValue API, this value will be returned in Base64 encoding and not in ASCII format. Therefore, while using the **C_FindObjectInit** function, you must provide the CKA_ID in Base64 encoding to search for keys based on the CKA_ID.
CKA_KEY_TYPE	CK_KEY_TYPE	This value could be set to CKK_AES, CKK_RSA or CKK_EC, CKK_SHA_1_HMAC, CKK_SHA256_HMAC, CKK_SHA384_HMAC, and CKK_SHA512_HMAC.

Return Values	Description
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_ATTRIBUTE_TYPE_INVALID	An invalid attribute type was specified in a template.
CKR_ATTRIBUTE_VALUE_INVALID	An invalid value was specified for a particular attribute in a template.
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_GENERAL_ERROR	An unrecoverable error occurred.

Return Values	Description
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_OK	The function was executed successfully.
CKR_OPERATION_ACTIVE	There is already an active operation (or combination of active operations) which prevents Cryptoki from activating the specified operation.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_KEY_FUNCTION_NOT_PERMITTED	The identity check failed.

C_FindObjects

Finds an object on the CipherTrust Manager using the following attributes:

- CKA_LABEL
- CKA_THALES_OBJECT_UUID
- CKA_THALES_OBJECT_MUID
- CKA_THALES_OBJECT_IKID
- CKA_THALES_OBJECT_ALIAS
- CKA_ID
- CKA_KEY_TYPE

For more information about these attributes, see C_FindObjectsInit.

```
CK_DEFINE_FUNCTION(CK_RV, C_FindObjects)(
CK_SESSION_HANDLE hSession,
CK_OBJECT_HANDLE_PTR phObject,
CK_ULONG ulMaxObjectCount,
CK_ULONG_PTR pulObjectCount
);
```

Input Parameters	Description
CK_SESSION_HANDLE hSession	Session handle.
CK_OBJECT_HANDLE_PTR phObject	Pointer to the buffer to put the object handles.
CK_ULONG ulMaxObjectCount	The maximum number of objects to put into the buffer.
CK_ULONG_PTR pulObjectCount	Pointer to CK_ULONG where the number of objects found is returned.

Output Parameters	Description
CK_OBJECT_HANDLE_PTR phObject	Returns a single key that matches the CKA_LABEL.
CK_ULONG_PTR pulObjectCount	Pointer to the number of objects C_FindObjects found.

Return Values	Description
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.

Return Values	Description
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_OK	The function was executed successfully.
CKR_OPERATION_NOT_INITIALIZED	There is no active operation of an appropriate type in the specified session.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_KEY_FUNCTION_NOT_PERMITTED	The identity check failed.
CKR_KEY_UNEXTRACTABLE	The key is not exportable.

C_FindObjectsFinal

Terminates a search for a token and session objects.

```
CK_DEFINE_FUNCTION(CK_RV, C_FindObjectsFinal)(
CK_SESSION_HANDLE hSession
);
```

Input Parameters	Description
CK_SESSION_HANDLE hSession	Session handle.

Return Values	Description
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_OK	The function was executed successfully.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_OPERATION_NOT_INITIALIZED	There is no active operation of an appropriate type in the specified session.
CKR_KEY_FUNCTION_NOT_PERMITTED	The identity check failed.

C_GetAttributeValue

Gets attribute value of a specific Security Object. Argument specifies the Security Object ID.

```
K_RV C_GetAttributeValue ( CK_SESSION_HANDLE hSession,
CK_OBJECT_HANDLE hObject,
CK_ATTRIBUTE_PTR pTemplate,
CK_ULONG ulCount
) ;
```

Input Parameters	Description
CK_SESSION_HANDLE hSession	Session handle.
CK_OBJECT_HANDLE hObject	Handle of the object for which to get attribute.
CK_ATTRIBUTE_PTR pTemplate	Template containing the attributes for which to search. Points to the CK_ATTRIBUTE structure.
CK_ULONG ulCount	Number of attributes in the template.

Output Parameters	Description
CK_ATTRIBUTE_PTR pTemplate	Any attribute that is supported by the key type or was set during key creation (with the exception of CKA_VALUE).

Return Values	Description
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_ATTRIBUTE_SENSITIVE	An attempt was made to obtain the value of an attribute of an object which cannot be satisfied because the object is either sensitive or un-extractable.
CKR_ATTRIBUTE_TYPE_INVALID	An invalid attribute type was specified in a template.
CKR_BUFFER_TOO_SMALL	The output of the function is too large to fit in the supplied buffer.
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_GENERAL_ERROR	An unrecoverable error occurred.

Return Values	Description
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_OBJECT_HANDLE_INVALID	The specified object handle is not valid.
CKR_OK	The function was executed successfully.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.

NOTE Key state cannot be changed for asymmetric keys in LEGACYVAE mode (formerly called VAE mode).

NOTE If hObject belongs to Asymmetric public key (found using class as CKO_PUBLIC_KEY in C_FindObjects) then in pTemplate you can get CKA_MODULUS and CKA_PUBLIC_EXPONENT for RSA keys and CKA_EC_PARAMS for EC keys.
If hObject belongs to Asymmetric private key (found using class as CKO_PRIVATE_KEY in C_FindObjects) then in pTemplate you can get CKA_MODULUS, CKA_PUBLIC_EXPONENT and CKA_PRIVATE_EXPONENT for RSA keys and CKA_EC_PARAMS for EC keys.

C_SetAttributeValue

Sets the attribute value of a specific Security Object. The argument specifies the Security Object ID.

```
CK_DEFINE_FUNCTION(CK_RV, C_SetAttributeValue)(
CK_SESSION_HANDLE hSession,
CK_OBJECT_HANDLE hObject,
CK_ATTRIBUTE_PTR pTemplate,
CK_ULONG ulCount)
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_OBJECT_HANDLE hObject`	Handle of the object for which to get attribute.

Input Parameters	Description

CK_ ATTRIBUT E_PTR pTemplate	Attributes that can be set for backwards compatibility with VAE (version 5.2.5), and are applicable to CADP for C PKCS#11:
	> CKA_START_DATE
	> CKA_END_DATE
	> CKA_THALES_OBJECT_CREATE_DATE
	> CKA_APPLICATION
	> CKA_OBJECT_ID
	> CKA_THALES_KEY_STATE
	The following attributes do not have any impact on the keys:
	> CKA_THALES_CACHED_ON_HOST
	> CKA_THALES_CACHED_TIME
	
	Valid key states are:
	1 - active
	2 - suspended (not supported)
	3 - deactivated
	4 - compromised (not supported)
	5 - destroyed (not supported)
	
	Supported VAE custom attributes:
	> CKA_THALES_CUSTOM_1
	> CKA_THALES_CUSTOM_2
	> CKA_THALES_CUSTOM_3
	> CKA_THALES_CUSTOM_4
	> CKA_THALES_CUSTOM_5
	
	Modifiable Attributes
	> CKA_MODIFIABLE
	> CKA_EXTRACTABLE
	> CKA_ID
	> CKA_THALES_OBJECT_ALIAS
	Sets an alias for a key. There can be multiple aliases. For example, alias1, alias 2, and so on.
	`{CKA_THALES_OBJECT_ALIAS, "alias1", strlen("alias1")},`
	`{CKA_THALES_OBJECT_ALIAS, "alias2", strlen("alias2")},`
	`{CKA_THALES_OBJECT_ALIAS, "alias3", strlen("alias3")}`
	> CKA_THALES_OBJECT_ALIAS | CKA_THALES_KEY_ACTION_DELETE
	Deletes the specified key alias. To Delete key aliases:
	`{CKA_THALES_OBJECT_ALIAS | CKA_THALES_KEY_ACTION_DELETE , "alias1", strlen ("alias1")},`

Input Parameters	Description
	`{CKA_THALES_OBJECT_ALIAS | CKA_THALES_KEY_ACTION_DELETE, "alias2", strlen ("alias2")},` `{CKA_THALES_OBJECT_ALIAS | CKA_THALES_KEY_ACTION_DELETE, "alias3", strlen ("alias3")}`
`CK_ULONG ulCount`	Number of attributes in the template.

Return Values	Description
`CKR_ARGUMENTS_BAD`	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
`CKR_ATTRIBUTE_READ_ONLY`	An attempt was made to set a value for an attribute which may not be set or modified.
`CKR_ATTRIBUTE_TYPE_INVALID`	An invalid attribute type was specified in a template.
`CKR_ATTRIBUTE_VALUE_INVALID`	An invalid value was specified for a particular attribute in a template.
`CKR_CRYPTOKI_NOT_INITIALIZED`	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
`CKR_GENERAL_ERROR`	An unrecoverable error occurred.
`CKR_HOST_MEMORY`	There is insufficient memory to perform the requested function.
`CKR_OK`	The function was executed successfully.
`CKR_SESSION_HANDLE_INVALID`	The specified session handle was invalid at the time that the function was invoked.
`CKR_OBJECT_HANDLE_INVALID`	The specified object handle is not valid.
`CKR_SESSION_READ_ONLY`	The specified session was unable to accomplish the desired action because it is a read-only session.
`CKR_SESSION_CLOSED`	The session was closed during the execution of the function.
`CKR_TOKEN_WRITE_PROTECTED`	The requested action could not be performed because the token is write-protected.
`CKR_TEMPLATE_INCONSISTENT`	The template specified for creating an object has conflicting attributes.

Return Values	Description
CKR_USER_NOT_LOGGED_IN	The desired action cannot be performed because the appropriate user (or an appropriate user) is not logged in.

NOTE The state of the Key cannot be changed for asymmetrical keys in LegacyVAE mode (formerly called VAE mode).

C_GenerateKey

Generates a symmetric key.

```
CK_DEFINE_FUNCTION(CK_RV, C_GenerateKey)(
CK_SESSION_HANDLE hSession,
CK_MECHANISM_PTR pMechanism,
CK_ATTRIBUTE_PTR pTemplate,
CK_ULONG ulCount,
CK_OBJECT_HANDLE_PTR phKey
);
```

> **NOTE** After you generate a versioned key using the `C_GenerateKey` function and then rotate the key, the previous version of the key is not deactivated.

> **NOTE** To generate the keys with KEY ID, set the `Gen_Key_Id` property to `YES`.

Supported Functionality

AES-128, AES-192, and AES-256 symmetric encryption keys are supported.

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_MECHANISM_PTR pMechanism`	Key generation mechanism; must have the following values: `{CKM_AES_KEY_GEN, NULL_PTR, 0};`
`CK_ULONG ulCount`	The number of attributes in the template.
`CK_OBJECT_HANDLE_PTR phKey`	Pointer to a key handle.
`CK_ATTRIBUTE_PTR pTemplate`	Template for the key to be generated. See the pTemplate Attribute table below.

Template Attribute Name	Type	Value
`CKA_LABEL`	`CK_UTF8CHAR`	Required. Key name; also displayed on the CipherTrust Manager.
`CKA_APPLICATION`	`CK_UTF8CHAR`	Description of the application generating the key name. This value can be `NULL`.

Template Attribute Name	Type	Value
CKA_CLASS	CK_OBJECT_CLASS	Required. This value must be set to CKO_SECRET_KEY.
CKA_KEY_TYPE	CK_KEY_TYPE	Required. This value must be set to CKK_AES.
CKA_VALUE_LEN	CK_ULONG	Key size (in Bytes).
CKA_TOKEN	CK_BBOOL	Optional.
CKA_ENCRYPT	CK_BBOOL	The default value is set to true.
CKA_DECRYPT	CK_BBOOL	The default value is set to true.
CKA_SIGN	CK_BBOOL	The default value is set to true.
CKA_VERIFY	CK_BBOOL	The default value is set to true.
CKA_WRAP	CK_BBOOL	The default value is set to true.
CKA_UNWRAP	CK_BBOOL	The default value is set to true.
CKA_EXTRACTABLE	CK_BBOOL	The default value is set to false.

CKA_THALES_OBJECT_ALIAS	CK_UTF8CHAR	Alias of the key. There can be multiple aliases. For example, alias1, alias 2, and so on.
		`{CKA_THALES_OBJECT_ALIAS, "alias1", strlen("alias1")},{CKA_THALES_OBJECT_ALIAS, "alias2", strlen("alias2")},`
		`{CKA_THALES_OBJECT_ALIAS, "alias3", strlen("alias3")},`

Template Attribute Name	Type	Value
CKA_ID	CK_UTF8CHAR	A searchable non-unique ID.

Output Parameters	Description
CK_OBJECT_HANDLE_PTR phKey	The structure where CK_OBJECT_HANDLE_PTR phKey points is filled in with the newly created key handle.

Return Values	Description
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_ATTRIBUTE_READ_ONLY	An attempt was made to set a value for an attribute which may not be set or modified.
CKR_ATTRIBUTE_TYPE_INVALID	An invalid attribute type was specified in a template.
CKR_ATTRIBUTE_VALUE_INVALID	An invalid value was specified for a particular attribute in a template.
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_OK	The function was executed successfully.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_FUNCTION_CANCELED	The function was canceled in mid-execution.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_MECHANISM_INVALID	An invalid mechanism was specified in the cryptographic operation.

Return Values	Description
CKR_MECHANISM_PARAM_INVALID	Invalid parameters were supplied to the mechanism specified in the cryptographic operation.
CKR_OPERATION_ACTIVE	There is already an active operation (or combination of active operations) which prevents Cryptoki from activating the specified operation.
CKR_PIN_EXPIRED	The specified PIN has expired, and the requested operation cannot be carried out.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_SESSION_READ_ONLY	The specified session was unable to accomplish the desired action because it is a read-only session.
CKR_TEMPLATE_INCOMPLETE	The template specified for creating an object is incomplete and lacks some necessary attributes.
CKR_TEMPLATE_INCONSISTENT	The template specified for creating an object has conflicting attributes.
CKR_TOKEN_WRITE_PROTECTED	The requested action could not be performed because the token is write-protected.
CKR_USER_NOT_LOGGED_IN	The desired action cannot be performed because the appropriate user (or an appropriate user) is not logged in.

C_GenerateKeyPair

Generates an asymmetric key pair. Arguments specify:

> length of the key pair for RSA key

> CurveOid for EC key

```
CK_DEFINE_FUNCTION(CK_RV, C_GenerateKeyPair)(
CK_SESSION_HANDLE hSession,
CK_MECHANISM_PTR pMechanism,
CK_ATTRIBUTE_PTR pPublicKeyTemplate,
CK_ULONG ulPublicKeyAttributeCount,
CK_ATTRIBUTE_PTR pPrivateKeyTemplate,
CK_ULONG ulPrivateKeyAttributeCount,
CK_OBJECT_HANDLE_PTR phPublicKey,
CK_OBJECT_HANDLE_PTR phPrivateKey
);
```

Supported Functionality

> RSA-1024, RSA-2048, RSA-3072, and RSA-4096 asymmetric key pairs are supported.

> Supported CurveOid with respect to their Curve id:

- secp224k1-06052b81040020
- secp224r1-06052b81040021
- secp256k1-06052b8104000a
- secp384r1-06052b81040022
- secp521r1-06052b81040023
- prime256v1-06082a8648ce3d030107
- brainpoolP224r1-06092b2403030208010105
- brainpoolP224t1-06092b2403030208010106
- brainpoolP256r1-06092b2403030208010107
- brainpoolP256t1-06092b2403030208010108
- brainpoolP384r1-06092b240303020801010b
- brainpoolP384t1-06092b240303020801010c
- brainpoolP512r1-06092b240303020801010d
- brainpoolP512t1-06092b240303020801010e

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.

Input Parameters	Description
CK_MECHANISM_PTR pMechanism	Key generation mechanism; must have the following values: > mechanism: CKM_RSA_PKCS_KEY_PAIR_GEN for RSA keys and CKM_EC_KEY_PAIR_GEN for EC keys > pParameter: set to "NULL_PTR" > ulParameterLen: set to "0"
CK_ATTRIBUTE_PTR pPublicKeyTemplate	Supported attributes are: > CKA_CLASS > CKA_DECRYPT > CKA_ENCRYPT > CKA_EXTRACTABLE > CKA_LABEL > CKA_MODULUS_BITS > CKA_SENSITIVE > CKA_SIGN > CKA_TOKEN > CKA_VERIFY > CKA_WRAP > CKA_ID > CKA_KEY_TYPE > CKA_EC_PARAMS
CK_ULONG ulPublicKeyAttributeCount	Size of the key template publicKeyTemplate.
CK_ATTRIBUTE_PTR privateKeyTemplate	Supported attributes are: > CKA_CLASS > CKA_DECRYPT > CKA_ENCRYPT > CKA_EXTRACTABLE > CKA_LABEL > CKA_MODULUS_BITS > CKA_SENSITIVE > CKA_SIGN > CKA_TOKEN > CKA_VERIFY > CKA_WRAP > CKA_ID > CKA_KEY_TYPE > CKA_EC_PARAMS

Input Parameters	Description
`CK_ULONG ulPrivateKeyAttributeCount`	Size of the key template privateKeyTemplate.
`CK_OBJECT_HANDLE_PTR phPublicKey`	Filled in with the newly created public key handle.
`CK_OBJECT_HANDLE_PTR phPrivateKey`	Filled in with the newly created private key handle.

> **NOTE** For **LegacyVAE** mode, CKA_ID is obtained from public template and for **CipherTrust** mode, CKA_ID is obtained from private template.

Output Parameters	Description
`CK_OBJECT_HANDLE_PTR phPublicKey`	Filled in with the newly created public key handle.
`CK_OBJECT_HANDLE_PTR phPrivateKey`	Filled in with the newly created private key handle.

Return Values	Description
`CKR_ARGUMENTS_BAD`	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
`CKR_ATTRIBUTE_READ_ONLY`	An attempt was made to set a value for an attribute which may not be set or modified.
`CKR_ATTRIBUTE_TYPE_INVALID`	An invalid attribute type was specified in a template.
`CKR_ATTRIBUTE_VALUE_INVALID`	An invalid value was specified for a particular attribute in a template.
`CKR_CRYPTOKI_NOT_INITIALIZED`	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
`CKR_GENERAL_ERROR`	An unrecoverable error occurred.
`CKR_HOST_MEMORY`	There is insufficient memory to perform the requested function.

Return Values	Description
CKR_OK	The function was executed successfully.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_FUNCTION_CANCELED	The function was canceled in mid-execution.
CKR_FUNCTION_FAILED	The requested function could not be performed. However, detailed information about the function failure is not available in this return error.
CKR_MECHANISM_INVALID	An invalid mechanism was specified in the cryptographic operation.
CKR_MECHANISM_PARAM_INVALID	Invalid parameters were supplied to the mechanism specified in the cryptographic operation.
CKR_OPERATION_ACTIVE	There is already an active operation (or combination of active operations) which prevents Cryptoki from activating the specified operation.
CKR_PIN_EXPIRED	The specified PIN has expired, and the requested operation cannot be carried out.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_SESSION_READ_ONLY	The specified session was unable to accomplish the desired action because it is a read-only session.
CKR_TEMPLATE_INCOMPLETE	The template specified for creating an object is incomplete, and lacks some necessary attributes.
CKR_TEMPLATE_INCONSISTENT	The template specified for creating an object has conflicting attributes.
CKR_TOKEN_WRITE_PROTECTED	The requested action could not be performed because the token is write-protected.
CKR_USER_NOT_LOGGED_IN	The desired action cannot be performed because the appropriate user (or an appropriate user) is not logged in.
CKR_DOMAIN_PARAMS_INVALID	Invalid or unsupported domain parameters were supplied to the function.

Below table depicts the value of different attributes in different mode (LegacyVAE and CipherTrust). The attributes highlighted are the mandatory attributes. If you provide values for both public and/or private templates, the highlighted value gets picked during key creation.

VAE

Attribute	Default	VAE	
		Public Template	Private Template
CKA_LABEL *	-	**Pu_K**	Pr_K
CKA_LABEL *	-	**Pu_K**	
CKA_LABEL *	-	-	**Pr_K**
CKA_THALES_CUSTOM_1	-	**C_1**	C_2
CKA_THALES_CUSTOM_2	-	**C_2**	
CKA_THALES_CUSTOM_3	-	-	C_3
CKA_MODULUS_BITS *	-	**1024/2048/3072/4096**	-
CKA_THALES_OBJECT_ALIAS	-	Ignored	Ignored
CKA_EXTRACTABLE	TRUE	**TRUE/FALSE**	TRUE/FALSE
CKA_EXTRACTABLE	TRUE	**TRUE/FALSE**	
CKA_EXTRACTABLE	TRUE	-	**TRUE/FALSE**
CKA_MODIFIABLE	TRUE	**TRUE/FALSE**	TRUE/FALSE
CKA_MODIFIABLE	TRUE	**TRUE/FALSE**	
CKA_MODIFIABLE	TRUE		**TRUE/FALSE**
CKA_EC_PARAMS	-	**CURVE Oid**	-
CKA_KEY_TYPE	CKK_RSA	**CKK_EC/CKK_RSA**	-

CipherTrust

Attribute	Default	VAE	
		Public Template	Private Template
CKA_LABEL *	-	-	**Pr_K**
CKA_THALES_CUSTOM_1	-	-	**C_1**
CKA_EXTRACTABLE	-	-	**TRUE/FALSE**

Attribute	Default	VAE	
		Public Template	Private Template
CKA_MODIFIABLE	-	-	**TRUE/FALSE**
CKA_MODULUS_BITS	1024	-	**1024/2048/3072/4096**
CKA_THALES_OBJECT_ALIAS	-	-	**ALIAS_1**
CKA_EC_PARAMS	-	-	**CURVE Oid**
CKA_KEY_TYPE	CKK_RSA	-	**CKK_EC**/**CKK_RSA**

NOTE **CKA_KEY_TYPE** and **CKA_EC_PARAMS** template attributes are mandatory for EC key generation.

CHAPTER 6: Digest and MAC Functions

This chapter describes the following digest and MAC functions:

C_DigestInit

Initializes a digest operation.

```
CK_DEFINE_FUNCTION(CK_RV, C_DigestInit)(
CK_SESSION_HANDLE hSession,
CK_MECHANISM_PTR pMechanism,
);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.

`CK_MECHANISM_PTR pMechanism`	Indicates the cryptographic hash function to use. Can be one of following values:
	`{ CKM_SHA256, NULL_PTR, 0 },{ CKM_SHA384, NULL_PTR, 0 },`
	`{ CKM_SHA512, NULL_PTR, 0 },`
	OR
	`{ CKM_SHA256_HMAC, NULL_PTR, 0 },{ CKM_SHA384_HMAC, NULL_PTR, 0 },`
	`{ CKM_SHA512_HMAC, NULL_PTR, 0 }`

Return Values	Description
`CKR_CRYPTOKI_NOT_INITIALIZED`	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
`CKR_FUNCTION_FAILED`	The requested function could not be performed.
`CKR_GENERAL_ERROR`	An unrecoverable error occurred.
`CKR_HOST_MEMORY`	There is insufficient memory to perform the requested function.
`CKR_MECHANISM_INVALID`	An invalid mechanism was specified in the cryptographic operation.
`CKR_MECHANISM_PARAM_INVALID`	Invalid parameters were supplied to the mechanism specified in the cryptographic operation.
`CKR_OK`	The function was executed successfully.
`CKR_SESSION_HANDLE_INVALID`	The specified session handle was invalid at the time that the function was invoked

C_Digest

Compute data digest in a single part. Must call `C_DigestInit` before calling `C_Digest`.

```
CK_DEFINE_FUNCTION(CK_RV, C_DigestInit)(
CK_SESSION_HANDLE hSession,
CK_BYTE_PTR pData,
CK_ULONG ulDataLen,
CK_BYTE_PTR pDigest,
CK_ULONG_PTR pulDigestLen
);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_BYTE_PTR pData`	Pointer to the digest input data.
`CK_ULONG ulDataLen`	Length of the data.
`CK_BYTE_PTR pDigest`	Results in the message digest.
`CK_ULONG_PTR pulDigestLen`	The byte count of the digest.

Output Parameters	Description
`CK_BYTE_PTR pDigest`	Results in the message digest.
`CK_ULONG_PTR pulDigestLen`	The byte count of the digest.

Return Values	Description
`CKR_CRYPTOKI_NOT_INITIALIZED`	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
`CKR_ARGUMENTS_BAD`	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
`CKR_FUNCTION_FAILED`	The requested function could not be performed.
`CKR_GENERAL_ERROR`	An unrecoverable error occurred.
`CKR_HOST_MEMORY`	There is insufficient memory to perform the requested function.
`CKR_MECHANISM_INVALID`	An invalid mechanism was specified in the cryptographic operation.

Return Values	Description
CKR_MECHANISM_PARAM_INVALID	Invalid parameters were supplied to the mechanism specified in the cryptographic operation.
CKR_OK	The function was executed successfully.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked
CKR_BUFFER_TOO_SMALL	The output of the function is too large to fit in the supplied buffer.
CKR_FUNCTION_CANCELED	The function was canceled in mid-execution.
CKR_OPERATION_NOT_INITIALIZED	There is no active operation of an appropriate type in the specified session.

C_DigestKey

Specifies an HMAC key for digest operation. You must call `C_DigestInit` before calling `C_DigestKey`.

```
CK_DEFINE_FUNCTION(CK_RV, C_DigestKey)(CK_SESSION_HANDLE hSession, CK_OBJECT_HANDLE hKey
);
```

> **NOTE** The key must be exportable from CipherTrust Manager. CADP for C only supports the AES key for the HMAC key for digest operation.

> **NOTE** You can call `C_DigestKey` only once, and only directly after calling `C_DigestInit`.

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_OBJECT_HANDLE hKey`	Handle of the secret key to be digested.

Return Values	Description
`CKR_CRYPTOKI_NOT_INITIALIZED`	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
`CKR_FUNCTION_FAILED`	The requested function could not be performed.
`CKR_GENERAL_ERROR`	An unrecoverable error occurred.
`CKR_HOST_MEMORY`	There is insufficient memory to perform the requested function.
`CKR_KEY_SIZE_RANGE`	The supplied key‘s size is outside the range of key sizes.
`CKR_MECHANISM_PARAM_INVALID`	Invalid parameters were supplied to the mechanism specified in the cryptographic operation.
`CKR_OK`	The function was executed successfully.
`CKR_SESSION_HANDLE_INVALID`	The specified session handle was invalid at the time that the function was invoked.
`CKR_ARGUMENTS_BAD`	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
`CKR_KEY_INDIGESTIBLE`	The supplied key‘s size is outside the range of key sizes.

Return Values	Description
CKR_KEY_FUNCTION_NOT_PERMITTED	The identity check failed.

C_DigestUpdate

Continues a multi-part digest operation. Must call `C_DigestInit` before calling `C_DigestUpdate`. Can optionally call `C_DigestKey` before calling `C_DigestUpdate`.

```
CK_DEFINE_FUNCTION(CK_RV, C_DigestUpdate)(
CK_SESSION_HANDLE hSession,
CK_BYTE_PTR pPart,
CK_ULONG ulPartLen
);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_BYTE_PTR pPart`	Pointer to the digest input data.
`CK_ULONG ulPartLen`	Length of the data.

Return Values	Description
`CKR_CRYPTOKI_NOT_INITIALIZED`	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
`CKR_FUNCTION_FAILED`	The requested function could not be performed.
`CKR_GENERAL_ERROR`	An unrecoverable error occurred.
`CKR_HOST_MEMORY`	There is insufficient memory to perform the requested function.
`CKR_OK`	The function was executed successfully.
`CKR_SESSION_HANDLE_INVALID`	The specified session handle was invalid at the time that the function was invoked.
`CKR_KEY_INDIGESTIBLE`	The supplied key‘s size is outside the range of key sizes.
`CKR_ARGUMENTS_BAD`	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
`CKR_KEY_TYPE_INCONSISTENT`	The specified key is not the correct type of key to use with the specified mechanism.
`CKR_OPERATION_NOT_INITIALIZED`	There is no active operation of an appropriate type in the specified session.

C_DigestFinal

Finishes a multi-part digest operation. This works only with the local key cache. Not supported if crypto is done remotely on the CipherTrust Manager.

```
CK_DEFINE_FUNCTION(CK_RV, C_DigestFinal)(
CK_SESSION_HANDLE hSession,
CK_BYTE_PTR pDigest,
CK_ULONG_PTR pulDigestLen
);
```

Input Parameters	Description
CK_SESSION_HANDLE hSession	Session handle.
CK_BYTE_PTR pDigest	Results in the message digest.
CK_ULONG_PTR pulDigestLen	The byte count of the digest.

Output Parameters	Description
CK_BYTE_PTR pDigest	Results in the message digest.
CK_ULONG_PTR pulDigestLen	The byte count of the digest.

Return Values	Description
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_FUNCTION_FAILED	The requested function could not be performed.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_OK	The function was executed successfully.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_KEY_INDIGESTIBLE	The supplied key‘s size is outside the range of key sizes.

Return Values	Description
CKR_KEY_TYPE_INCONSISTENT	The specified key is not the correct type of key to use with the specified mechanism.
CKR_OPERATION_NOT_INITIALIZED	There is no active operation of an appropriate type in the specified session.
CKR_BUFFER_TOO_SMALL	The output of the function is too large to fit in the supplied buffer.

CHAPTER 7: Signing and Calculating MAC Functions

This chapter describes the following signing and calculating MAC functions:

NOTE These Signing and Calculating MAC functions do not work in Remote mode.

C_SignInit

Initializes a signature operation.

```
CK_DEFINE_FUNCTION(CK_RV, C_SignInit)(
   CK_SESSION_HANDLE hSession,
   CK_MECHANISM_PTR pMechanism,
   CK_OBJECT_HANDLE hKey
);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.

Input Parameters	Description
`CK_MECHANISM_PTR pMechanism`	The mechanism (algorithm) you wish to use for the signature operation. The mechanism has the following parameters: `;{CKM_RSA_PKCS, NULL_PTR, 0}` Valid mechanisms are: > `CKM_SHA224_HMAC` > `CKM_SHA256_HMAC` > `CKM_SHA384_HMAC` > `CKM_SHA512_HMAC` > `CKM_RSA_PKCS` > `CKM_SHA1_RSA_PKCS` > `CKM_SHA256_RSA_PKCS` > `CKM_SHA512_RSA_PKCS` > `CKM_SHA384_RSA_PKCS` > `CKM_ECDSA_SHA1` > `CKM_ECDSA_SHA256` > `CKM_ECDSA_SHA384` > `CKM_ECDSA_SHA512` To use RSA Data Protection Manager (DPM) compatible headers with AES keys for HMAC, you must OR the desired mechanism with the corresponding header version. For more information about the RSA DPM compatible headers, refer to Key Management, CADP for C PKCS11. > `CKM_THALES_V15HDR` - Specifies Version 1.5 of the DPM header. This header requires an external key ID to be enabled. For information about creating a key ID during key generation, refer to Gen_Key_Id in the Miscellaneous Parameters. > `CKM_THALES_BASE64` - Indicates to encode the resulting header in base 64 when this parameter is ORed with `CKM_THALES_V15HDR`. The use of `CKM_THALES_BASE64` only applies to a Version 1.5 of the DPM header. > `CKM_THALES_V21HDR` - Specifies Version 2.1 of the DPM header. This header uses the UUID of a key. > `CKM_THALES_V27HDR` - Specifies Version 2.7 of the DPM header. This header uses the MUID of a key.
`CK_OBJECT_HANDLE hKey`	Handle to the private key. The key that this handle references can be a symmetric key, an asymmetric key, or an opaque object. The RSA DPM headers are not supported for non-versioned keys, RSA keys, EC keys, and opaque objects.

Return Values	Description
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_FUNCTION_FAILED	The requested function could not be performed.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_MECHANISM_INVALID	An invalid mechanism was specified in the cryptographic operation.
CKR_MECHANISM_PARAM_INVALID	Invalid parameters were supplied to the mechanism specified in the cryptographic operation.
CKR_OK	The function was executed successfully.
CKR_FUNCTION_CANCELED	The function was canceled in mid-execution.
CKR_KEY_FUNCTION_NOT_PERMITTED	An attempt has been made to use a key for a cryptographic purpose that the key's attributes are not set to allow it to do.
CKR_KEY_HANDLE_INVALID	The specified key handle is invalid.
CKR_KEY_SIZE_RANGE	The supplied key's size is outside the range of key sizes.
CKR_KEY_TYPE_INCONSISTENT	The specified key is not the correct type of key to use with the specified mechanism.
CKR_OPERATION_ACTIVE	There is already an active operation (or combination of active operations) which prevents Cryptoki from activating the specified operation.
CKR_USER_NOT_LOGGED_IN	The desired action cannot be performed because the appropriate user (or an appropriate user) is not logged in.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.

C_Sign

Signs data in a single part.

```
CK_DEFINE_FUNCTION(CK_RV, C_Sign)(
   CK_SESSION_HANDLE hSession,
   CK_BYTE_PTR pData,
   CK_ULONG ulDataLen,
   CK_BYTE_PTR pSignature,
   CK_ULONG_PTR pulSignatureLen
);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_BYTE_PTR pData`	Pointer to data to be signed.
`CK_ULONG ulDataLen`	Length of data to be signed.
`CK_BYTE_PTR pSignature`	Pointer to buffer to receive the signed data. If `C_SignInit` is called with a specified header, the signature returns the corresponding header prepended to it.
`CK_ULONG_PTR pulSignatureLen`	Pointer to buffer length.

Return Values	Description
`CKR_CRYPTOKI_NOT_INITIALIZED`	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
`CKR_FUNCTION_FAILED`	The requested function could not be performed.
`CKR_GENERAL_ERROR`	An unrecoverable error occurred.
`CKR_HOST_MEMORY`	There is insufficient memory to perform the requested function.
`CKR_MECHANISM_INVALID`	An invalid mechanism was specified in the cryptographic operation.
`CKR_MECHANISM_PARAM_INVALID`	Invalid parameters were supplied to the mechanism specified in the cryptographic operation.
`CKR_OK`	The function was executed successfully.
`CKR_FUNCTION_CANCELED`	The function was canceled in mid-execution.

Return Values	Description
CKR_KEY_FUNCTION_NOT_PERMITTED	An attempt has been made to use a key for a cryptographic purpose that the key's attributes are not set to allow it to do.
CKR_KEY_HANDLE_INVALID	The specified key handle is invalid.
CKR_KEY_SIZE_RANGE	The supplied key's size is outside the range of key sizes.
CKR_KEY_TYPE_INCONSISTENT	The specified key is not the correct type of key to use with the specified mechanism.
CKR_OPERATION_ACTIVE	There is already an active operation (or combination of active operations) which prevents Cryptoki from activating the specified operation.
CKR_USER_NOT_LOGGED_IN	The desired action cannot be performed because the appropriate user (or an appropriate user) is not logged in.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_BUFFER_TOO_SMALL	The output of the function is too large to fit in the supplied buffer.
CKR_DATA_INVALID	The plaintext input data to a cryptographic operation is invalid.
CKR_DATA_LEN_RANGE	The provided signature/MAC can be seen to be invalid solely on the basis of its length.
CKR_FUNCTION_REJECTED	The signature request is rejected by the user.
CKR_OPERATION_NOT_INITIALIZED	There is no active operation of an appropriate type in the specified session.

C_SignUpdate

Continues a multiple-part signature operation.

> **NOTE** This function is not supported in this release.

```
CK_DEFINE_FUNCTION(CK_RV, C_SignUpdate)(
   CK_SESSION_HANDLE hSession,
   CK_BYTE_PTR pPart,
   CK_ULONG ulPartLen
);
```

Input Parameters	Description
CK_SESSION_HANDLE hSession	Session handle.
CK_BYTE_PTR pPart	Points to the data.
CK_ULONG ulPartLen	Length of data.

Return Values	Description
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_FUNCTION_FAILED	The requested function could not be performed.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_MECHANISM_INVALID	An invalid mechanism was specified in the cryptographic operation.
CKR_MECHANISM_PARAM_INVALID	Invalid parameters were supplied to the mechanism specified in the cryptographic operation.
CKR_OK	The function was executed successfully.
CKR_FUNCTION_CANCELED	The function was canceled in mid-execution.
CKR_KEY_FUNCTION_NOT_PERMITTED	An attempt has been made to use a key for a cryptographic purpose that the key's attributes are not set to allow it to do.
CKR_KEY_HANDLE_INVALID	The specified key handle is invalid.
CKR_KEY_SIZE_RANGE	The supplied key's size is outside the range of key sizes.

Return Values	Description
CKR_KEY_TYPE_INCONSISTENT	The specified key is not the correct type of key to use with the specified mechanism.
CKR_OPERATION_ACTIVE	There is already an active operation (or combination of active operations) which prevents Cryptoki from activating the specified operation.
CKR_USER_NOT_LOGGED_IN	The desired action cannot be performed because the appropriate user (or an appropriate user) is not logged in.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_BUFFER_TOO_SMALL	The output of the function is too large to fit in the supplied buffer.
CKR_DATA_INVALID	The plaintext input data to a cryptographic operation is invalid.
CKR_DATA_LEN_RANGE	The provided signature/MAC can be seen to be invalid solely on the basis of its length.
CKR_FUNCTION_REJECTED	The signature request is rejected by the user.
CKR_OPERATION_NOT_INITIALIZED	There is no active operation of an appropriate type in the specified session.

C_SignFinal

Finishes a multiple-part signature operation.

> **NOTE** This function is not supported in this release.

```
CK_DEFINE_FUNCTION(CK_RV, C_SignFinal)(
   CK_SESSION_HANDLE hSession,
   CK_BYTE_PTR pSignature,
   CK_ULONG_PTR pulSignatureLen
);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_BYTE_PTR pSignature`	Pointer to buffer to receive the signed data.
`CK_ULONG pulSignatureLen`	Pointer to buffer length.

Output Parameters	Description
`CK_BYTE_PTR pSignature`	Pointer to buffer to receive the signed data.
`CK_ULONG_PTR pulSignatureLen`	Pointer to buffer length.

Return Values	Description
`CKR_CRYPTOKI_NOT_INITIALIZED`	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
`CKR_FUNCTION_FAILED`	The requested function could not be performed.
`CKR_GENERAL_ERROR`	An unrecoverable error occurred.
`CKR_HOST_MEMORY`	There is insufficient memory to perform the requested function.
`CKR_MECHANISM_INVALID`	An invalid mechanism was specified in the cryptographic operation.
`CKR_MECHANISM_PARAM_INVALID`	Invalid parameters were supplied to the mechanism specified in the cryptographic operation.

Return Values	Description
CKR_OK	The function was executed successfully.
CKR_FUNCTION_CANCELED	The function was canceled in mid-execution.
CKR_KEY_FUNCTION_NOT_PERMITTED	An attempt has been made to use a key for a cryptographic purpose that the key's attributes are not set to allow it to do.
CKR_KEY_HANDLE_INVALID	The specified key handle is invalid.
CKR_KEY_SIZE_RANGE	The supplied key's size is outside the range of key sizes.
CKR_KEY_TYPE_INCONSISTENT	The specified key is not the correct type of key to use with the specified mechanism.
CKR_OPERATION_ACTIVE	There is already an active operation (or combination of active operations) which prevents Cryptoki from activating the specified operation.
CKR_PIN_EXPIRED	The specified PIN has expired, and the requested operation cannot be carried out.
CKR_USER_NOT_LOGGED_IN	The desired action cannot be performed because the appropriate user (or an appropriate user) is not logged in.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_BUFFER_TOO_SMALL	The output of the function is too large to fit in the supplied buffer.
CKR_DATA_INVALID	The plaintext input data to a cryptographic operation is invalid.
CKR_DATA_LEN_RANGE	The provided signature/MAC can be seen to be invalid solely on the basis of its length.
CKR_FUNCTION_REJECTED	The signature request is rejected by the user.
CKR_OPERATION_NOT_INITIALIZED	There is no active operation of an appropriate type in the specified session.

C_VerifyInit

Initializes a verification operation.

```
CK_DEFINE_FUNCTION(CK_RV, C_VerifyInit)
(CK_SESSION_HANDLE hSession,
CK_MECHANISM_PTR pMechanism,
CK_OBJECT_HANDLE hKey);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_MECHANISM_PTR pMechanism`	The mechanism should have the following parameters: `;{ CKM_RSA_PKCS, NULL_PTR, 0 }` Valid Mechanisms are: > `CKM_SHA224_HMACCKM_SHA256_HMAC` > `CKM_SHA384_HMAC` > `CKM_SHA512_HMAC` > `CKM_RSA_PKCS` > `CKM_SHA1_RSA_PKCS` > `CKM_SHA256_RSA_PKCS` > `CKM_SHA512_RSA_PKCS` > `CKM_SHA384_RSA_PKCS` > `CKM_ECDSA_SHA1` > `CKM_ECDSA_SHA256` > `CKM_ECDSA_SHA384` > `CKM_ECDSA_SHA512` When using RSA DPM compatible headers with AES symmetric keys for HMAC operations, you must specify that the ciphertext has a header. For more information about the RSA DPM compatible headers, refer to Key Management, CADP for C PKCS11. `CKM_THALES_ALLHDR` - Specifies that the signature contains a header.
`hKey`	Handle to the public key. The key that this handle references can be a symmetric key, an asymmetric key, or an opaque object. The RSA DPM headers are not supported for non-versioned keys, RSA keys, EC keys, and opaque objects.

Return Values	Description
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_FUNCTION_FAILED	The requested function could not be performed.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_MECHANISM_INVALID	An invalid mechanism was specified in the cryptographic operation.
CKR_KEY_FUNCTION_NOT_PERMITTED	An attempt has been made to use a key for a cryptographic purpose that the key's attributes are not set to allow it to do.
CKR_USER_NOT_LOGGED_IN	The desired action cannot be performed because the appropriate user (or an appropriate user) is not logged in.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_KEY_HANDLE_INVALID	The specified key handle is invalid.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.

C_Verify

Performs a single-part verification operation. To perform verification, the signature is verified with the key's handle provided in `C_Verify_Init`.

If `C_Verify_Init` is called specifying the use of a header, the signature contains the key which is prepended when signed.

```
CK_DEFINE_FUNCTION(CK_RV, C_Verify)
   (CK_SESSION_HANDLE hSession,
   CK_BYTE_PTR pData,
   CK_ULONG ulDataLen,
   CK_BYTE_PTR pSignature,
   CK_ULONG ulSignatureLen);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_BYTE_PTR pData`	Pointer to data to be verified.
`CK_ULONG ulDataLen`	Length of data.
`CK_BYTE_PTR pSignature`	Pointer to buffer to receive the signed data.
`CK_ULONG ulSignatureLen`	Pointer to buffer length.

Output Parameters	Description
`CK_BYTE_PTR pSignature`	Pointer to buffer to receive the signed data.
`CK_ULONG ulSignatureLen`	Pointer to buffer length.

Return Values	Description
`CKR_OK`	The function was executed successfully.
`CKR_SIGNATURE_INVALID`	The provided signature/MAC is invalid.
`CKR_CRYPTOKI_NOT_INITIALIZED`	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
`CKR_SIGNATURE_LEN_RANGE`	The provided signature/MAC can be seen to be invalid solely on the basis of its length.

C_VerifyUpdate

Performs a multiple-part verification operation. To perform verification, the signature is verified with the key's handle provided in `C_Verify_Init`.

> **NOTE** This function is not supported in this release.

```
CK_DEFINE_FUNCTION(CK_RV, C_VerifyUp date)
   (CK_SESSION_HANDLE hSession,
   CK_BYTE_PTR pPart,
   CK_ULONG ulPartLen);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_BYTE_PTR pData`	Pointer to data to be verified.
`CK_ULONG ulPartLen`	Length of data.

Return Values	Description
`CKR_OK`	The function was executed successfully.
`CKR_SIGNATURE_INVALID`	The provided signature/MAC is invalid.
`CKR_CRYPTOKI_NOT_INITIALIZED`	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
`CKR_SIGNATURE_LEN_RANGE`	The provided signature/MAC can be seen to be invalid solely on the basis of its length.

C_VerifyFinal

Finishes a multi-part verification operation. To perform verification, the signature is verified with the key's handle provided in `C_Verify_Init` and compared with the data supplied by previous `C_VerifyUpdate` calls.

> **NOTE** This function is not supported in this release.

```
CK_DEFINE_FUNCTION(CK_RV, C_VerifyFinal)
   (CK_SESSION_HANDLE hSession,
   CK_BYTE_PTR pSignature,
   CK_ULONG ulSignatureLen);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_BYTE_PTR pSignature`	Pointer to buffer to receive the signed data.
`CK_ULONG ulSignatureLen`	Pointer to buffer length.

Output Parameters	Description
`CK_BYTE_PTR pSignature`	Pointer to buffer to receive the signed data.
`CK_ULONG ulSignatureLen`	Pointer to buffer length.

Return Values	Description
`CKR_OK`	The function was executed successfully.
`CKR_SIGNATURE_INVALID`	The provided signature/MAC is invalid.
`CKR_CRYPTOKI_NOT_INITIALIZED`	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
`CKR_SIGNATURE_LEN_RANGE`	The provided signature/MAC can be seen to be invalid solely on the basis of its length.

CHAPTER 8:

CHAPTER 8: Encryption Functions

This chapter describes the following encryption functions:

C_EncryptInit

Initializes an encryption operation. `C_EncryptInit` must be called before `C_Encrypt` or `C_EncryptUpdate` is called:

```
CK_DEFINE_FUNCTION(CK_RV, C_EncryptInit)(
   CK_SESSION_HANDLE hSession,
   CK_MECHANISM_PTR pMechanism,
   CK_OBJECT_HANDLE hKey
);
```

> **NOTE** When using `CKM_AES_GCM`, the `pParameter` field of the `CK_MECHANISM` structure must point to the address of a `CK_GCM_PARAMS` structure, and the `ulParameterLen` field must be equal to the length of the structure.

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.

Input Parameters	Description
`CK_MECHANISM_PTR pMechanism`	The pointer to the structure that specifies the encryption mechanism. The following mechanisms are supported: • `CKM_AES_ECB` • `CKM_AES_CBC` • `CKM_AES_CBC_PAD` • `CKM_AES_CTR` • `CKM_THALES_FPE (in Java, use: 0x80004001L)` • `CKM_THALES_FF1 (in Java, use: 0x80004002L)` • `CKM_THALES_FF3_1 (in Java, use: 0x80004003L)` • `CKM_RSA_PKCS` • `CKM_AES_GCM` For `CKM_AES_GCM`, MultiPart chunk size should always be greater than TagLen. In case of versionkey (Ciphertrust mode), chunk size should be the sum of TagLen and version header (3 bytes). For `CKM_THALES_FPE`, the IV contains the following field values: • tweak (8 bytes) • character set: – ASCII or Unicode characters. Supported Unicode encodings are: – UTF-8 – UTF-16 – UTF-32 – big-endian – little-endian byte order – radix (character set size) **NOTE** Ensure there are no duplicate characters in the character set. • radix (character set size) The size can be from 2 to 65535. For `CKM_THALES_FF3_1`, the `pParameter`contains the following field values: • tweakAlgo : – NONE (tweakAlgolen must be 7 bytes (14 characters HEX encoded string)) – SHA1 – SHA256 • tweak • mode: – CARD10

Input Parameters	Description
	– CARD26 – CARD62 – ASCII (charsetlen between 2-255) or Unicode characters. Supported Unicode encodings are: – UTF-8 – UTF16 – UTF-32 – big-endian – little-endian byte order – radix (character set size; the size can be from 2 to 65535) **NOTE** - Ensure there must be at least two characters to encrypt from the character set of FF3-1. - Ensure there are no duplicate characters in the character set. - For Java, `CKM_THALES_FF3_1` is supported with custom Java wrapper only. For more information, see the CADP for C PKCS#11 code samples. To use RSA DPM compatible headers with AES symmetric key ciphers (ECB, CBC, CBCPAD and CTR), you must OR the desired mechanism with the corresponding header version. For more information about the RSA DPM compatible headers, refer to the Key Management, CADP for C PKCS11. `CKM_THALES_V15HDR` - Specifies Version 1.5 of the DPM header. This header requires an external key ID to be enabled. For information about creating a key ID during key generation, refer to Gen_Key_Id in the Miscellaneous Parameters. > `CKM_THALES_BASE64` - Indicates to encode the resulting header in base 64 when this parameter is ORed with `CKM_THALES_V15HDR`. The use of `CKM_THALES_BASE64` only applies to Version 1.5 of the DPM header. > `CKM_THALES_V21HDR` - Specifies Version 2.1 of the DPM header. This header uses the UUID of a key. **NOTE** DPM headers are not supported for asymmetric keys. > `CKM_THALES_V27HDR` - Specifies Version 2.7 of the DPM header. This header uses the MUID of a key.
`CK_OBJECT_HANDLE hKey`	The handle to the encryption key to perform the encryption. For asymmetric keys, this handle should be an RSA public key handle.

Return Values	Description
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_FUNCTION_FAILED	The requested function could not be performed.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_KEY_FUNCTION_NOT_PERMITTED	An attempt has been made to use a key for a cryptographic purpose that the key's attributes are not set to allow it to do.
CKR_KEY_SIZE_RANGE	The supplied key's size is outside the range of key sizes.
CKR_KEY_HANDLE_INVALID	The specified key handle is invalid.
CKR_KEY_TYPE_INCONSISTENT	The specified key is not the correct type of key to use with the specified mechanism.
CKR_MECHANISM_INVALID	An invalid mechanism was specified in the cryptographic operation.
CKR_MECHANISM_PARAM_INVALID	Invalid parameters were supplied to the mechanism specified in the cryptographic operation.
CKR_OK	The function was executed successfully.
CKR_OPERATION_ACTIVE	There is already an active operation (or combination of active operations) which prevents Cryptoki from activating the specified operation.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_USER_NOT_LOGGED_IN	The desired action cannot be performed because the appropriate user (or an appropriate user) is not logged in.

C_Encrypt

Encrypts single-part data. Must call `C_EncryptInit` before calling `C_Encrypt`.

```
CK_DEFINE_FUNCTION(CK_RV, C_Encrypt)(
   CK_SESSION_HANDLE hSession,
   CK_BYTE_PTR pData,
   CK_ULONG ulDataLen,
   CK_BYTE_PTR pEncryptedData,
   CK_ULONG_PTR pulEncryptedDataLen
);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_BYTE_PTR pData`	Pointer to data to be encrypted.
`CK_ULONG ulDataLen`	Length of the data (in bytes).
`CK_BYTE_PTR pEncryptedData`	Pointer of the buffer to put the data. > **First call**: Pointer to the metadata to be passed and logged on the application's behalf. Set the buffer length to zero (0) to get the ciphertext size. > **Second call**: Pointer to the resulting encrypted data; output parameter.
`CK_ULONG_PTR pulEncryptedDataLen`	Pointer of the encrypted buffer length. > **First call**: Pointer to encrypted buffer length. Encrypted buffer length is zero (0) for metadata logging. > **Second call**: actual buffer length used for encrypted data, written encrypted data length. For GCM, this parameter contains the length of the tag in bytes, plus the length of the remaining ciphertext from encryption.

Output Parameters	Description
`CK_BYTE_PTR pEncryptedData`	Pointer to the encrypted data. For GCM, the tag of length specified in the `ulTagBits` field for `CK_GCM_PARAMS` in bytes is appended to the end of the buffer.
`CK_ULONG_PTR pulEncryptedDataLen`	Buffer length used for the encrypted data.

Return Values	Description
CKR_OK	The function was executed successfully.
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_BUFFER_TOO_SMALL	The output of the function is too large to fit in the supplied buffer.
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_DATA_INVALID	The plaintext input data to a cryptographic operation is invalid.
CKR_DATA_LEN_RANGE	The plaintext input data to a cryptographic operation has a bad length.
CKR_FUNCTION_FAILED	The requested function could not be performed.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_OPERATION_NOT_INITIALIZED	There is no active operation of an appropriate type in the specified session.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_PIN_INVALID	The specified PIN has invalid characters in it.
CKR_MECHANISM_INVALID	An invalid mechanism was specified in the cryptographic operation.

C_EncryptUpdate

Continues a multiple-part encryption operation by processing another data part.

```
CK_DEFINE_FUNCTION(CK_RV, C_EncryptUpdate)(
   CK_SESSION_HANDLE              hSession,
   CK_BYTE_PTR                    pPart,
   CK_ULONG                       ulPartLen,
   CK_BYTE_PTR                    pEncryptedPart,
   CK_ULONG_PTR                   pulEncryptedPartLen
);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_BYTE_PTR pData`	Pointer to data to be encrypted.
`CK_ULONG ulPartLen`	Length of the data.
`CK_BYTE_PTR pEncryptedPart`	Pointer of the buffer to put the data. > **First call**: Pointer to the metadata to be passed and logged on the application's behalf. > **Second call**: Pointer to the resulting encrypted data; output parameter.
`CK_ULONG_PTR pulEncryptedPartLen`	Pointer of the encrypted buffer length. > **First call**: Pointer to encrypted buffer length. Encrypted buffer length is zero (0) for metadata logging. > **Second call**: Actual buffer length used for encrypted data, written encrypted data length.

Output Parameters	Description
`CK_BYTE_PTR pEncryptedData`	Pointer to the resulting encrypted data.
`CK_ULONG_PTR pulEncryptedDataLen`	Buffer length used for the encrypted data.

Return Values	Description
`CKR_OK`	The function was executed successfully.
`CKR_ARGUMENTS_BAD`	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.

Return Values	Description
CKR_BUFFER_TOO_SMALL	The output of the function is too large to fit in the supplied buffer.
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_DATA_INVALID	The plaintext input data to a cryptographic operation is invalid.
CKR_DATA_LEN_RANGE	The plaintext input data to a cryptographic operation has a bad length.
CKR_FUNCTION_CANCELED	The function was canceled in mid-execution.

C_EncryptFinal

Finishes a multi-part encryption.

```
CK_DEFINE_FUNCTION(CK_RV, C_EncryptFinal)(
   CK_SESSION_HANDLE       hSession,
   CK_BYTE_PTR             pLastEncryptedPart,
   CK_ULONG_PTR            pulEncryptedPartLen
);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_BYTE_PTR pLastEncryptedPart`	Pointer to any remaining data to be encrypted. > **First call**: Pointer to the metadata to be passed and logged on the application's behalf. > **Second call**: Pointer to the resulting encrypted data; output parameter.
`CK_ULONG_PTR pulLastEncryptedPartLen`	Pointer of the encrypted buffer length. > **First call**: Pointer to encrypted buffer length. Encrypted buffer length is zero (0) for metadata logging. > **Second call**: Actual buffer length used for encrypted data, written encrypted data length. For GCM, this parameter contains the length of the tag in bytes plus the length of the remaining ciphertext from encryption.

Output Parameters	Description
`CK_BYTE_PTR pLastEncryptedPart`	Pointer to the encrypted data. For GCM, the tag of length is specified in the `ulTagBits` field for `CK_GCM_PARAMS` in bytes, at the end of the buffer.
`CK_ULONG_PTR pulEncryptedPartLen`	Pointer to the buffer used for the encrypted data.

Return Values	Description
`CKR_OK`	The function was executed successfully.
`CKR_ARGUMENTS_BAD`	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
`CKR_BUFFER_TOO_SMALL`	The output of the function is too large to fit in the supplied buffer.

Return Values	Description
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
CKR_DATA_INVALID	The plaintext input data to a cryptographic operation is invalid.
CKR_DATA_LEN_RANGE	The plaintext input data to a cryptographic operation has a bad length.
CKR_FUNCTION_CANCELED	The function was canceled in mid-execution.

CHAPTER 9: pullDecryption Functions

This chapter describes the following decryption functions:

C_DecryptInit

Initializes a decryption option. `C_DecryptInit` must be called before `C_Decrypt` or `C_DecryptUpdate`.

```
CK_DEFINE_FUNCTION(CK_RV, C_DecryptInit)(
   CK_SESSION_HANDLE              hSession,
   CK_MECHANISM_PTR               pMechanism,
   CK_OBJECT_HANDLE               hKey
);
```

> **NOTE** When using `CKM_AES_GCM`, the `pParameter` field of the `CK_MECHANISM` structure must point to the address of a `CK_GCM_PARAMS` structure, and the `ulParameterLen` field must be equal to the length of the structure.

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.

Input Parameters	Description
CK_MECHANISM_PTR pMechanism	Pointer to the mechanism for decryption. Must match the mechanism used to encrypt, including the correct IV, if using CKM_AES_CBC or CKM_AES_CBC_PAD, or the correct nonce and counter concatenated, if using CKM_AES_CTR. The following mechanisms are supported: • CKM_AES_ECB • CKM_AES_CBC • CKM_AES_CBC_PAD • CKM_AES_CTR • CKM_THALES_FPE (in Java, use: 0x80004001L) • CKM_THALES_FF1 (in Java, use: 0x80004002L) • CKM_THALES_FF3_1 (in Java, use: 0x80004003L) • CKM_RSA_PKCS • CKM_AES_GCM For CKM_THALES_FPE, the IV contains the following field values: • tweak (8 bytes) • character set. ASCII characters (0-127) • radix (character set size). For more information, see the CADP for C PKCS#11 code samples. For CKM_THALES_FF1, the IV contains the same field values, except that the tweak is not a fixed size: it can range from 0 to 32 bytes. **NOTE** Be sure there are no duplicate characters in the character set. For CKM_THALES_FF3_1, the pParametercontains the following field values: • tweakAlgo : – NONE (tweakAlgolen must be 7 bytes (14 characters HEX encoded string)) – SHA1 – SHA256 • tweak • mode: – CARD10 – CARD26 – CARD62 – ASCII (charsetlen between 2-255) or Unicode characters. Supported Unicode encodings are: – UTF-8 – UTF-16

Input Parameters	Description
	– UTF-32 – big-endian – little-endian byte order – radix (character set size) **NOTE** - Ensure there must be at least two characters to encrypt from the character set of FF3-1. - Ensure there are no duplicate characters in the character set. - For Java, `CKM_THALES_FF3_1` is supported with custom Java wrapper only.. For more information, see the CADP for C PKCS#11 code samples. When using RSA DPM compatible headers with AES symmetric key ciphers (ECB, CBC, CBCPAD, and CTR), you must specify that the ciphertext has a header. To determine the key and the IV, if applicable, the header must be parsed. For more information about the RSA DPM compatible headers, refer to the Key Management, CADP for C PKCS11. `CKM_THALES_ALLHDR` - Specifies that the signature contains a header. **NOTE** DPM headers are not supported for asymmetric keys.
`CK_OBJECT_HANDLE hKey`	Handle of the decryption key; (same as encryption key for symmetric encryption).For asymmetric keys, this handle should be an RSA private key handle.
`CK_GCM_PARAMS ulEncryptedPartLen`	For GCM, this parameter must include the length of the tag in bytes, in addition to the ciphertext length.

Return Values	Description
`CKR_OK`	The function was executed successfully.
`CKR_ARGUMENTS_BAD`	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
`CKR_BUFFER_TOO_SMALL`	The output of the function is too large to fit in the supplied buffer.
`CKR_CRYPTOKI_NOT_INITIALIZED`	The function could not be executed because the Cryptoki library has not yet been initialized by a call to `C_Initialize`.
`CKR_DATA_INVALID`	The plaintext input data to a cryptographic operation is invalid.

Return Values	Description
CKR_DATA_LEN_RANGE	The plaintext input data to a cryptographic operation has a bad length.
CKR_FUNCTION_CANCELED	The function was canceled in mid-execution.
CKR_FUNCTION_FAILED	The requested function could not be performed.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_KEY_FUNCTION_NOT_ PERMITTED	An attempt has been made to use a key for a cryptographic purpose that the key's attributes are not set to allow it to do.
CKR_KEY_HANDLE_INVALID	The specified key handle is invalid.
CKR_KEY_SIZE_RANGE	The supplied key's size is outside the range of key sizes.
CKR_KEY_TYPE_INCONSISTENT	The specified key is not the correct type of key to use with the specified mechanism.
CKR_MECHANISM_INVALID	An invalid mechanism was specified in the cryptographic operation.
CKR_MECHANISM_PARAM_INVALID	Invalid parameters were supplied to the mechanism specified in the cryptographic operation.
CKR_OPERATION_ACTIVE	There is already an active operation (or combination of active operations) which prevents Cryptoki from activating the specified operation.
CKR_PIN_EXPIRED	The specified PIN has expired, and the requested operation cannot be carried out.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_USER_NOT_LOGGED_IN	The desired action cannot be performed because the appropriate user (or an appropriate user) is not logged in.

C_Decrypt

Decrypts encrypted data in a single part. The operation must have been initialized by a prior call to `C_DecryptInit`.

```
CK_DEFINE_FUNCTION(CK_RV, C_Decrypt)(
   CK_SESSION_HANDLE       hSession,
   CK_BYTE_PTR             pEncryptedData,
   CK_ULONG                ulEncryptedDataLen,
   CK_BYTE_PTR             pData,
   CK_ULONG_PTR            pulDataLen
);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_BYTE_PTR pEncryptedData`	Pointer to the encrypted data to be decrypted.
`CK_ULONG ulEncryptedDataLen`	Length of the encrypted data to be decrypted. For GCM, this parameter must include the length of the tag in bytes, in addition to the ciphertext length.
`CK_BYTE_PTR pData`	Pointer of the buffer for the decrypted text. > **First call**: Pointer to the metadata to be passed and logged on the application's behalf. > **Second call**: Pointer to the resulting decrypted data; output parameter.
`CK_ULONG_PTR pulDataLen`	Pointer to the length of the buffer for the decrypted text. > **First call**: Pointer to decrypted buffer length. First call buffer length is zero (0) for metadata logging. > **Second call**: actual buffer length used for decrypted data, length of written plaintext.
`CK_GCM_PARAMS`	The first bytes of length are specified in the `ulTagBits` field and `pEncryptedData` must contain the tag. Also for GCM, if the decryption cannot be validated, the error `CKR_ENCRYPTED_DATA_INVALID` is returned.

Output Parameters	Description
`pData`	When input parameter `pData` is `NULL`, this function returns a calculated output buffer length value pointed to by `pulDataLen`.

Return Values	Description
CKR_OK	The function was executed successfully.
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_BUFFER_TOO_SMALL	The output of the function is too large to fit in the supplied buffer.
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_DATA_INVALID	The plaintext input data to a cryptographic operation is invalid.
CKR_DATA_LEN_RANGE	The plaintext input data to a cryptographic operation has a bad length.
CKR_FUNCTION_CANCELED	The function was canceled in mid-execution.
CKR_FUNCTION_FAILED	The requested function could not be performed.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_OPERATION_NOT_INITIALIZED	There is no active operation of an appropriate type in the specified session.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_USER_NOT_LOGGED_IN	The desired action cannot be performed because the appropriate user (or an appropriate user) is not logged in.
CKR_MECHANISM_INVALID	An invalid mechanism was specified in the cryptographic operation.

C_DecryptUpdate

Decrypts multi-part encrypted data. The operation must have been initialized by a prior call to `C_DecryptInit`.

```
CK_DEFINE_FUNCTION(CK_RV, C_DecryptUpdate)(
   CK_SESSION_HANDLE       hSession,
   CK_BYTE_PTR             pEncryptedPart,
   CK_ULONG                ulEncryptedPartLen,
   CK_BYTE_PTR             pPart,
   CK_ULONG_PTR            pulPartLen
);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_BYTE_PTR pEncryptedData`	Pointer to the data to be decrypted. For GCM, on the first call to `C_DecryptUpdate`, this parameter includes the first bytes of length specified in the `ulTagBits` field in bytes and `pEncryptedPart` must contain the tag.
`CK_ULONG ulEncryptedDataLen`	Length of the data to be decrypted. For GCM, this parameter must include the length of the tag in bytes, in addition to the ciphertext length.
`CK_BYTE_PTR pData`	Pointer of the buffer for the decrypted text. > **First call**: Null pointer to be passed. > **Second call**: Pointer to the resulting encrypted data; output parameter.
`CK_ULONG_PTR pulDataLen`	Pointer to the length of the buffer for the decrypted text. > **First call**: Pointer to buffer length. pulDataLen first call is set to zero (0) for calculating the output buffer size. > **Second call**: Actual buffer length used for encrypted data, length of written encrypted data length.

Output Parameters	Description
`pData`	When input parameter `pData` is `NULL`, this function returns a calculated output buffer length value pointed to by `pulDataLen`.

Return Values	Description
`CKR_OK`	The function was executed successfully.

Return Values	Description
CKR_ARGUMENTS_BAD	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
CKR_BUFFER_TOO_SMALL	The output of the function is too large to fit in the supplied buffer.
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_DATA_INVALID	The plaintext input data to a cryptographic operation is invalid.
CKR_DATA_LEN_RANGE	The plaintext input data to a cryptographic operation has a bad length.
CKR_FUNCTION_CANCELED	The function was canceled in mid-execution.
CKR_FUNCTION_FAILED	The requested function could not be performed.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_OPERATION_NOT_INITIALIZED	There is no active operation of an appropriate type in the specified session.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_USER_NOT_LOGGED_IN	The desired action cannot be performed because the appropriate user (or an appropriate user) is not logged in.
CKR_MECHANISM_INVALID	An invalid mechanism was specified in the cryptographic operation.

C_DecryptFinal

Finishes a multi-part decryption. This works only with the local key cache.

```
CK_DEFINE_FUNCTION(CK_RV, C_DecryptFinal)(
   CK_SESSION_HANDLE       hSession,
   CK_BYTE_PTR             pLastPart,
   CK_ULONG_PTR            pulLastPartLen
);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_BYTE_PTR pLastPart`	Pointer to a buffer to write the decrypted data. > **First call**: Pointer to the metadata to be passed and logged on the application's behalf. > **Second call**: Pointer to the resulting decrypted data; output parameter.
`CK_ULONG_PTR pulLastPartLen`	Buffer length pointer. > **First call**: Pointer to decrypted buffer length. `pulLLastPartLen` first call is set to zero (0) for calculating the output buffer size. > **Second call**: Actual buffer length used for decrypted data, length of written encrypted data length.

Output Parameters	Description
`pLastPart`	To be filled in with the decrypted text.
`pulLastPartLen`	To be filled in with the length of the decrypted text.
`pLastPart`	When input parameter `pLastPart` is `NULL`, this function returns a calculated output buffer length value pointed to by `pulLastPartLen`.

Return Values	Description
`CKR_OK`	The function was executed successfully.
`CKR_ARGUMENTS_BAD`	Generic error code which indicates that the arguments supplied to the Cryptoki function are invalid.
`CKR_BUFFER_TOO_SMALL`	The output of the function is too large to fit in the supplied buffer.

Return Values	Description
CKR_CRYPTOKI_NOT_INITIALIZED	The function could not be executed because the Cryptoki library has not yet been initialized by a call to C_Initialize.
CKR_DATA_INVALID	The plaintext input data to a cryptographic operation is invalid.
CKR_DATA_LEN_RANGE	The plaintext input data to a cryptographic operation has a bad length.
CKR_FUNCTION_CANCELED	The function was canceled in mid-execution.
CKR_FUNCTION_FAILED	The requested function could not be performed.
CKR_GENERAL_ERROR	An unrecoverable error occurred.
CKR_HOST_MEMORY	There is insufficient memory to perform the requested function.
CKR_OPERATION_NOT_INITIALIZED	There is no active operation of an appropriate type in the specified session.
CKR_SESSION_CLOSED	The session was closed during the execution of the function.
CKR_SESSION_HANDLE_INVALID	The specified session handle was invalid at the time that the function was invoked.
CKR_ENCRYPTED_DATA_INVALID	For GCM, if the decryption cannot be validated.

CHAPTER 10: Random Data Generation

This chapter describes the following random data generation function:

C_GenerateRandom

Use this function to generate and return random data.

> **NOTE** `C_GenerateRandom` is supported only in remote cryptographic mode.

```
CK_RV  C_GenerateRandom  (
CK_SESSION_HANDLE hSession,
CK_BYTE_PTR pRandomData,
CK_ULONG ulRandomLen
);
```

Input Parameters	Description
`CK_SESSION_HANDLE hSession`	Session handle.
`CK_BYTE_PTR pRandomData`	Pointer to empty buffer for receiving the random data.
`CK_ULONG ulRandomLen`	The number of bytes to generate.

Output Parameters	Description
`RandomResult`	A randomized token is returned.

APPENDIX A: Mapping of VAE Compatibility Mode and CipherTrust Mode APIs

This appendix covers the differences between the PKCS11 APIs of the VAE Compatibility Mode and the CipherTrust Mode.

The VAE binary compatibility mode enables backward compatibility with the VAE library. The newer PKCS11 functionality is available from CipherTrust Mode.

This appendix covers the following sections:

APIs with Differences

Function Name	VAE Compatibility Mode	CipherTrust Mode
C_Login	Login with PIN is configured in VAE Compatibility Mode (formerly called VAE mode). Ability to log in using the following: >PIN >PIN:Username:Password	Ability to log in using the following: > Username/password > Certificate > anonymous (global)

Function Name	VAE Compatibility Mode	CipherTrust Mode
C_CreateObject	CKA_EXTRACTABLE and CKA_MODIFIABLE are set to true by default.	> **Symmetric Keys**: CKA_EXTRACTABLE and CKA_MODIFIABLE are set to true by default in local mode (Symmetric Cache is enabled). > **Asymmetric Keys**: CKA_EXTRACTABLE and CKA_MODIFIABLE are set to false by default. The CipherTrust Mode supports the following Custom Template Attributes: > CKA_THALES_CUSTOM_1 > CKA_THALES_CUSTOM_2 > CKA_THALES_CUSTOM_3 > CKA_THALES_CUSTOM_4 > CKA_THALES_CUSTOM_5
C_EncryptInit	Support for RSA DPM compatible headers. Versions 1.5, 2.1 and 2.7 are supported for AES keys with the following ciphers: ECB, CBC, CBC-PAD, and CTR.	Added CKM_AES_CTR. From CipherTrust Mode, the Ingrian 3-byte headers are the default for cryptographic operations with the versioned key.
C_DecryptInit	Support for RSA DPM compatible headers. Parsing of versions 1.5, 2.1, and 2.7 are supported for AES keys with the following ciphers: ECB, CBC, CBC-PAD, and CTR.	Added CKM_AES_CTR. From CipherTrust Mode, the Ingrian 3-byte headers are the default for cryptographic operations with the versioned key.
C_GenerateKey	CKA_EXTRACTABLE and CKA_MODIFIABLE are set to true by default.	CKA_EXTRACTABLE is set to true by default in the local mode (that is, when symmetric cache is enabled). CKA_MODIFIABLE is set to false by default.
C_GenerateKeyPair	CKA_EXTRACTABLE and CKA_MODIFIABLE are set to true by default.	CKA_EXTRACTABLE and CKA_MODIFIABLE are set to false by default.

Function Name	VAE Compatibility Mode	CipherTrust Mode
C_WrapKey	Available using CKM_AES_CBC and CKM_AES_CBC_PAD. Includes the ability to get key material with wrapping key handle set to 0.	Available using CKM_AES_CBC, CKM_RSA_PKCS, and CKM_AES_CBC_PAD. The function does not provide the ability to get key material with wrapping key handle set to 0.
C_SeedRandom	Not Implemented. However, this function returns SUCCESS.	Not Implemented. This function returns CKR_FUNCTION_NOT_SUPPORTED.
C_SetAttributeValue	Changing key states is not allowed for asymmetric keys.	Changing key states is allowed for asymmetric keys.

APIs with No Differences

The following list of PKCS11 APIs and their functions behave similarly for both VAE Compatibility Mode and CipherTrust Mode:

- C_Initialize
- C_Finalize
- C_GetInfo
- C_GetFunctionList
- C_GetSlotInfo
- C_GetTokenInfo
- C_GetMechanismList
- C_GetMechanismInfo
- C_OpenSession
- C_CloseSession
- C_CloseAllSessions
- C_Logout
- C_DestroyObject
- C_FindObjectsInit
- C_FindObjects
- C_FindObjectsFinal
- C_Encrypt
- C_EncryptUpdate
- C_EncryptFinal

- C_Decrypt
- C_DecryptUpdate
- C_DecryptFinal
- C_DigestInit
- C_Digest
- C_DigestUpdate
- C_DigestFinal
- C_SignInit
- C_Sign
- C_VerifyInit
- C_Verify
- C_GenerateRandom