

This is a Non-Standards Track Work Product. The patent provisions of the OASIS IPR Policy do not apply.

Key Management Interoperability Use Cases Version 1.2

Working Draft 6

6 September 2012

**Specification URIs**

**This version:**

N/A

**Previous version:**

N/A

**Latest version:**

N/A

Technical Committee:

[OASIS Key Management Interoperability Protocol (KMIP) TC](http://www.oasis-open.org/committees/kmip/)

Chairs:

Robert Griffin ([robert.griffin@rsa.com](mailto:robert.griffin@rsa.com)), [EMC Corporation](http://www.emc.com/)

Subhash Sankuratripati ([Subhash.Sankuratripati@netapp.com](mailto:Subhash.Sankuratripati@netapp.com)), [NetApp](http://www.netapp.com/)

Editors:

Michael Allen ([michael.allen@symantec.com](mailto:michael.allen@symantec.com)), Symantec

Denis Pochuev ([denis.pochuev@safenet.com](mailto:denis.pochuev@safenet.com)), SafeNet

Related work:

This is a new document.

This document is related to:

* Key Management Interoperability Protocol Specification Version 1.2
* Key Management Interoperability Protocol Profiles Version 1.2
* Key Management Interoperability Protocol Test Cases Version 1.2
* Key Management Interoperability Protocol Usage Guide Version 1.2

Abstract:

This document is intended to complement the Key Management Interoperability Protocol Specification by describing the use cases that KMIP is intended to address.

KMIP V1.2 enhances the V1.0 (October 2010) and V1.1 (Committee Spec July 2012) of the KMIP standard by defining new functionality in the protocol to support use cases related to PGP keys, server-to-server interactions related to key management, registration of clients, key management for cloud deployments, Quantum Key Distribution, Hardware Security Modules and other areas. The Key Management Interoperability Protocol (KMIP) is a single, comprehensive protocol for communication between clients that request any of a wide range of cryptographic obects and servers that store and manage those objects. By replacing redundant, incompatible key management protocols, KMIP provides better data security while at the same time reducing expenditures on multiple products.

Status:

This is a new document

Technical Committee members should send comments on this document to the Technical Committee’s email list. Others should send comments to the Technical Committee by using the “[Send A Comment](http://www.oasis-open.org/committees/comments/index.php?wg_abbrev=kmip)” button on the Technical Committee’s web page at <http://www.oasis-open.org/committees/kmip/>.

Citation format:

When referencing this document the following citation format should be used:

[KMIP-UC]

*Key Management Interoperability Protocol Use Cases Version 1.2*. 28 June 2012. OASIS Committee Working Draft 2.

Copyright © OASIS Open 2012. All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full [Policy](http://www.oasis-open.org/who/intellectualproperty.php) may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Table of Contents

[1 Introduction 15](#_Toc333601636)

[1.1 Terminology 15](#_Toc333601637)

[1.2 Normative References 19](#_Toc333601638)

[1.3 Non-normative References 20](#_Toc333601639)

[2 Use Cases for Key Management Administration 21](#_Toc333601640)

[2.1 Use Case KAUC-1a: End-User Administration 22](#_Toc333601641)

[2.1.1 Description / User Story 22](#_Toc333601642)

[2.1.2 Goal or Desired Outcome 22](#_Toc333601643)

[2.1.3 Notable Categorizations and Aspects 22](#_Toc333601644)

[2.1.4 Process Flow 23](#_Toc333601645)

[2.2 Use Case KAUC-1b: End-User Self-Administration 23](#_Toc333601646)

[2.2.1 Goal or Desired Outcome 23](#_Toc333601647)

[2.2.2 Notable Categorizations and Aspects 23](#_Toc333601648)

[2.2.3 Process Flow 24](#_Toc333601649)

[2.2.4 Extensions 24](#_Toc333601650)

[2.3 Use Case KAUC-1c: Multiple Administrators, Multiple Servers 25](#_Toc333601651)

[2.3.1 Notable Categorizations and Aspects 25](#_Toc333601652)

[2.3.2 Process Flow 26](#_Toc333601653)

[2.3.3 Extensions 26](#_Toc333601654)

[3 Use Cases for Key Management in the Cloud 27](#_Toc333601655)

[3.1 Use Cases for Deployment Model 1: KMS in Enterprise, KMC in CSP 31](#_Toc333601656)

[3.1.1 Use Case KCUC-1a: End-user requests key (KMC in CSP requests managed object from KMS in enterprise) 32](#_Toc333601657)

[3.1.1.1 User Story and Component Interaction 32](#_Toc333601658)

[3.1.1.2 Goal or Desired Outcome 32](#_Toc333601659)

[3.1.1.3 Notable Categorizations and Aspects 33](#_Toc333601660)

[3.1.1.4 Process Flow 33](#_Toc333601661)

[3.1.1.5 Extensions 34](#_Toc333601662)

[3.2 Use Cases for Deployment Model 2: KMS in Dedicated CSP Infrastructure; KMC in Enterprise 34](#_Toc333601663)

[3.2.1 Use case KCUC-2a: CSP Administrator manages resources for tenant KMS at CSP 35](#_Toc333601664)

[3.2.1.1 Description / User Story 35](#_Toc333601665)

[3.2.1.2 Goal or Desired Outcome 36](#_Toc333601666)

[3.2.1.3 Notable Categorizations and Aspects 36](#_Toc333601667)

[3.2.1.4 Process Flow 36](#_Toc333601668)

[3.2.2 Use case KCUC-2b: Tenant admin manages KMS at CSP (admin\_client-to-server) 36](#_Toc333601669)

[3.2.2.1 Description / User Story 37](#_Toc333601670)

[3.2.2.2 Goal or Desired Outcome 37](#_Toc333601671)

[3.2.2.3 Notable Categorizations and Aspects 37](#_Toc333601672)

[3.2.2.4 Process Flow 38](#_Toc333601673)

[3.2.2.5 Extensions 38](#_Toc333601674)

[3.2.3 Use case KCUC-2c: CSP Administrator provisions resources for a new tenant and sets up credentials for tenant administrator 39](#_Toc333601675)

[3.2.3.1 Description / User Story 39](#_Toc333601676)

[3.2.3.2 Goal or Desired Outcome 39](#_Toc333601677)

[3.2.3.3 Notable Categorizations and Aspects 39](#_Toc333601678)

[3.2.3.4 Process Flow 39](#_Toc333601679)

[3.2.3.5 Extensions 39](#_Toc333601680)

[3.2.4 Use case KCUC-2d: Tenant admin provisions clients and generates credentials for clients to access KMS at CSP (admin\_client-to-server) 40](#_Toc333601681)

[3.2.4.1 Description / User Story 40](#_Toc333601682)

[3.2.4.2 Goal or Desired Outcome 40](#_Toc333601683)

[3.2.4.3 Notable Categorizations and Aspects 40](#_Toc333601684)

[3.2.4.4 Process Flow 41](#_Toc333601685)

[3.2.4.5 Extensions 41](#_Toc333601686)

[3.2.5 Use case KCUC-2e: End-user requests key (KMC in enterprise requests managed object from KMS in CSP) 41](#_Toc333601687)

[3.2.5.1 User Story and Component Interaction 41](#_Toc333601688)

[3.2.5.2 Goal or Desired Outcome 41](#_Toc333601689)

[3.2.5.3 Notable Categorizations and Aspects 42](#_Toc333601690)

[3.2.5.4 Process Flow 42](#_Toc333601691)

[3.2.5.5 Extensions 43](#_Toc333601692)

[3.2.6 Use case KCUC-2f: End-user requests key (KMC in CSP requests managed object from KMS in CSP) 43](#_Toc333601693)

[3.2.6.1 User Story and Component Interaction 43](#_Toc333601694)

[3.2.6.2 Goal or Desired Outcome 43](#_Toc333601695)

[3.2.6.3 Notable Categorizations and Aspects 43](#_Toc333601696)

[3.2.6.4 Process Flow 43](#_Toc333601697)

[3.2.6.5 Extensions 44](#_Toc333601698)

[3.2.7 Use case KCUC-2g: Tenant admin requests enterprise KMS to export tenant keys to KMS at CSP (server-to-server) 44](#_Toc333601699)

[3.2.7.1 Description / User Story 44](#_Toc333601700)

[3.2.7.2 Goal or Desired Outcome 44](#_Toc333601701)

[3.2.7.3 Notable Categorizations and Aspects 44](#_Toc333601702)

[3.2.7.4 Process Flow 44](#_Toc333601703)

[3.2.7.5 Extensions 45](#_Toc333601704)

[3.2.8 Use case KCUC-2h: Tenant Admin requests Enterprise KMS to export policy to KMS at CSP 45](#_Toc333601705)

[3.2.8.1 Description / User Story 45](#_Toc333601706)

[3.2.8.2 Goal or Desired Outcome 45](#_Toc333601707)

[3.2.8.3 Notable Categorizations and Aspects 46](#_Toc333601708)

[3.2.8.4 Process Flow 46](#_Toc333601709)

[3.2.8.5 Extensions 46](#_Toc333601710)

[3.2.9 Use case KCUC-2i: Enterprise KMS sending key-specific attribute changes to KMS at CSP (server-to-server) 46](#_Toc333601711)

[3.2.9.1 Description / User Story 46](#_Toc333601712)

[3.2.9.2 Goal or Desired Outcome 47](#_Toc333601713)

[3.2.9.3 Notable Categorizations and Aspects 47](#_Toc333601714)

[3.2.9.4 Process Flow 47](#_Toc333601715)

[3.2.9.5 Extensions 47](#_Toc333601716)

[3.2.10 Use case KCUC-2j: CSP KMS exporting keys to Enterprise KMS (server-to-server) 47](#_Toc333601717)

[3.2.10.1 Description / User Story 47](#_Toc333601718)

[3.2.10.2 Goal or Desired Outcome 48](#_Toc333601719)

[3.2.10.3 Notable Categorizations and Aspects 48](#_Toc333601720)

[3.2.10.4 Process Flow 48](#_Toc333601721)

[3.2.10.5 Extensions 48](#_Toc333601722)

[3.2.11 tbdUse case KCUC-2k: Use case KCUC-6: Tenant Admin requests CSP KMS to export policy to KMS at Enterprise 48](#_Toc333601723)

[3.2.11.1 Description / User Story 48](#_Toc333601724)

[3.2.11.2 Goal or Desired Outcome 49](#_Toc333601725)

[3.2.11.3 Notable Categorizations and Aspects 49](#_Toc333601726)

[3.2.11.4 Process Flow 49](#_Toc333601727)

[3.2.11.5 Extensions 49](#_Toc333601728)

[3.2.12 Use case KCUC-2l: CSP KMS sending key-specific attribute changes to KMS at Enterprise (server-to-server) 49](#_Toc333601729)

[3.2.12.1 Description / User Story 49](#_Toc333601730)

[3.2.12.2 Goal or Desired Outcome 50](#_Toc333601731)

[3.2.12.3 Notable Categorizations and Aspects 50](#_Toc333601732)

[3.2.12.4 Process Flow 50](#_Toc333601733)

[3.2.12.5 Extensions 50](#_Toc333601734)

[3.3 Use Cases for Deployment Model 3: KMS as CSP Shared Service; KMC in Enterprise or CSP 50](#_Toc333601735)

[3.3.1 Use case KCUC-3a: CSP Administrator manages resources for tenant using shared KMS at CSP 51](#_Toc333601736)

[3.3.1.1 Description / User Story 52](#_Toc333601737)

[3.3.1.2 Goal or Desired Outcome 52](#_Toc333601738)

[3.3.1.3 Notable Categorizations and Aspects 53](#_Toc333601739)

[3.3.1.4 Process Flow 53](#_Toc333601740)

[3.3.1.5 Extensions 53](#_Toc333601741)

[3.3.2 Use case KCUC-3b: Tenant admin manages tenant-specific capabilities in shared KMS at CSP (admin\_client-to-server) 53](#_Toc333601742)

[3.3.2.1 Description / User Story 53](#_Toc333601743)

[3.3.2.2 Goal or Desired Outcome 54](#_Toc333601744)

[3.3.2.3 Notable Categorizations and Aspects 54](#_Toc333601745)

[3.3.2.4 Process Flow 54](#_Toc333601746)

[3.3.2.5 Extensions 54](#_Toc333601747)

[3.3.3 Use case KCUC-3c: CSP Administrator provisions resources for a new tenant and sets up credentials for tenant administrator 54](#_Toc333601748)

[3.3.3.1 Description / User Story 55](#_Toc333601749)

[3.3.3.2 Goal or Desired Outcome 55](#_Toc333601750)

[3.3.3.3 Notable Categorizations and Aspects 55](#_Toc333601751)

[3.3.3.4 Process Flow 55](#_Toc333601752)

[3.3.3.5 Extensions 55](#_Toc333601753)

[3.3.4 Use case KCUC-3d: Tenant admin provisions clients and generates credentials for clients to access KMS at CSP (admin\_client-to-server) 55](#_Toc333601754)

[3.3.4.1 Description / User Story 56](#_Toc333601755)

[3.3.4.2 Goal or Desired Outcome 56](#_Toc333601756)

[3.3.4.3 Notable Categorizations and Aspects 56](#_Toc333601757)

[3.3.4.4 Process Flow 56](#_Toc333601758)

[3.3.4.5 Extensions 57](#_Toc333601759)

[3.3.5 Use case KCUC-3e: End-user requests key (KMC in enterprise requests managed object from KMS in CSP) 57](#_Toc333601760)

[3.3.5.1 User Story and Component Interaction 57](#_Toc333601761)

[3.3.5.2 Goal or Desired Outcome 57](#_Toc333601762)

[3.3.5.3 Notable Categorizations and Aspects 57](#_Toc333601763)

[3.3.5.4 Process Flow 58](#_Toc333601764)

[3.3.5.5 Extensions 58](#_Toc333601765)

[3.3.6 Use case KCUC-3f: End-user requests key (KMC in CSP requests managed object from KMS in CSP) 58](#_Toc333601766)

[3.3.6.1 User Story and Component Interaction 58](#_Toc333601767)

[3.3.6.2 Goal or Desired Outcome 58](#_Toc333601768)

[3.3.6.3 Notable Categorizations and Aspects 59](#_Toc333601769)

[3.3.6.4 Process Flow 59](#_Toc333601770)

[3.3.6.5 Extensions 59](#_Toc333601771)

[3.3.7 Use case KCUC-3g: Tenant admin requests enterprise KMS to export tenant keys to KMS at CSP (server-to-server) 59](#_Toc333601772)

[3.3.7.1 Description / User Story 59](#_Toc333601773)

[3.3.7.2 Goal or Desired Outcome 59](#_Toc333601774)

[3.3.7.3 Notable Categorizations and Aspects 60](#_Toc333601775)

[3.3.7.4 Process Flow 60](#_Toc333601776)

[3.3.7.5 Extensions 60](#_Toc333601777)

[3.3.8 Use case KCUC-3h: Tenant Admin requests Enterprise KMS to export policy to KMS at CSP 60](#_Toc333601778)

[3.3.8.1 Description / User Story 60](#_Toc333601779)

[3.3.8.2 Goal or Desired Outcome 61](#_Toc333601780)

[3.3.8.3 Notable Categorizations and Aspects 61](#_Toc333601781)

[3.3.8.4 Process Flow 61](#_Toc333601782)

[3.3.8.5 Extensions 61](#_Toc333601783)

[3.3.9 Use case KCUC-3i: Enterprise KMS sending key-specific attribute changes to KMS at CSP (server-to-server) 61](#_Toc333601784)

[3.3.9.1 Description / User Story 61](#_Toc333601785)

[3.3.9.2 Goal or Desired Outcome 62](#_Toc333601786)

[3.3.9.3 Notable Categorizations and Aspects 62](#_Toc333601787)

[3.3.9.4 Process Flow 62](#_Toc333601788)

[3.3.9.5 Extensions 62](#_Toc333601789)

[3.3.10 Use case KCUC-3j: CSP KMS exporting keys to Enterprise KMS (server-to-server) 62](#_Toc333601790)

[3.3.10.1 Description / User Story 63](#_Toc333601791)

[3.3.10.2 Goal or Desired Outcome 63](#_Toc333601792)

[3.3.10.3 Notable Categorizations and Aspects 63](#_Toc333601793)

[3.3.10.4 Process Flow 63](#_Toc333601794)

[3.3.10.5 Extensions 63](#_Toc333601795)

[3.3.11 Use case KCUC-3k: Tenant Admin requests CSP KMS to export policy to KMS at Enterprise 63](#_Toc333601796)

[3.3.11.1 Description / User Story 64](#_Toc333601797)

[3.3.11.2 Goal or Desired Outcome 64](#_Toc333601798)

[3.3.11.3 Notable Categorizations and Aspects 64](#_Toc333601799)

[3.3.11.4 Process Flow 64](#_Toc333601800)

[3.3.11.5 Extensions 64](#_Toc333601801)

[3.3.12 Use case KCUC-3l: CSP Admin requests CSP KMS to send key-specific attribute changes to KMS at Enterprise (server-to-server) 65](#_Toc333601802)

[3.3.12.1 Description / User Story 65](#_Toc333601803)

[3.3.12.2 Goal or Desired Outcome 65](#_Toc333601804)

[3.3.12.3 Notable Categorizations and Aspects 65](#_Toc333601805)

[Same as KCUC-3b. 65](#_Toc333601806)

[3.3.12.4 Process Flow 65](#_Toc333601807)

[3.3.12.5 Extensions 66](#_Toc333601808)

[3.3.13 Use case KCUC-3m: Tenant Administrator requests registration for tenant KMC using shared KMS at CSP 66](#_Toc333601809)

[3.3.13.1 Description / User Story 66](#_Toc333601810)

[3.3.13.2 Goal or Desired Outcome 66](#_Toc333601811)

[3.3.13.3 Notable Categorizations and Aspects 66](#_Toc333601812)

[3.3.13.4 Process Flow 66](#_Toc333601813)

[3.3.13.5 Extensions 67](#_Toc333601814)

[3.3.14 Use case KCUC-3n: Tenant Administrator requests revocation of registration for tenant KMC using shared KMS at CSP 67](#_Toc333601815)

[3.3.14.1 Description / User Story 67](#_Toc333601816)

[3.3.14.2 Goal or Desired Outcome 67](#_Toc333601817)

[3.3.14.3 Notable Categorizations and Aspects 67](#_Toc333601818)

[3.3.14.4 Process Flow 67](#_Toc333601819)

[3.3.14.5 Extensions 68](#_Toc333601820)

[3.3.15 Use case KCUC-3o: Tenant Administrator requests assurance that tenant KMC using shared KMS at CSP is acting within specified policies 68](#_Toc333601821)

[3.3.15.1 Description / User Story 68](#_Toc333601822)

[3.3.15.2 Goal or Desired Outcome 68](#_Toc333601823)

[3.3.15.3 Notable Categorizations and Aspects 68](#_Toc333601824)

[3.3.15.4 Process Flow 68](#_Toc333601825)

[3.3.15.5 Extensions 69](#_Toc333601826)

[3.3.16 Use case KCUC-3p: Shared KMS responds to increase in client resource needs (scaling) 69](#_Toc333601827)

[3.3.16.1 Description / User Story 69](#_Toc333601828)

[3.3.16.2 Goal or Desired Outcome 69](#_Toc333601829)

[3.3.16.3 Notable Categorizations and Aspects 69](#_Toc333601830)

[3.3.16.4 Process Flow 69](#_Toc333601831)

[3.3.16.5 Extensions 69](#_Toc333601832)

[3.3.17 Use case KCUC-3q: Tenant KMC manages cached keys from shared KMS at CSP 70](#_Toc333601833)

[3.3.17.1 Description / User Story 70](#_Toc333601834)

[3.3.17.2 Goal or Desired Outcome 70](#_Toc333601835)

[3.3.17.3 Notable Categorizations and Aspects 70](#_Toc333601836)

[3.3.17.4 Process Flow 70](#_Toc333601837)

[3.3.17.5 Extensions 70](#_Toc333601838)

[4 Use Cases for Enterprise Server-to-Server Key Management 71](#_Toc333601839)

[4.1 Use case KSUC-1: Admin Requests Transfer of Keys from one KMS to Another 71](#_Toc333601840)

[4.1.1 Description / User Story 71](#_Toc333601841)

[4.1.2 Goal or Desired Outcome 71](#_Toc333601842)

[4.1.3 Notable Categorizations and Aspects 72](#_Toc333601843)

[4.1.4 Process Flow 72](#_Toc333601844)

[4.1.5 Extensions 73](#_Toc333601845)

[4.2 Use Case KSUC-2: Admin Requests Transfer of Key Policy from one KMS to Another 73](#_Toc333601846)

[4.2.1 Description / User Story 73](#_Toc333601847)

[4.2.2 Goal or Desired Outcome 73](#_Toc333601848)

[4.2.3 Notable Categorizations and Aspects 74](#_Toc333601849)

[4.2.4 Process Flow 74](#_Toc333601850)

[4.2.5 Extensions 75](#_Toc333601851)

[4.3 Use Case KSUC-3: Admin Requests Transfer of Key Attributes from one KMS to Another 75](#_Toc333601852)

[4.3.1 Description / User Story 75](#_Toc333601853)

[4.3.2 Goal or Desired Outcome 76](#_Toc333601854)

[4.3.3 Notable Categorizations and Aspects 76](#_Toc333601855)

[4.3.4 Process Flow 77](#_Toc333601856)

[4.3.5 Extensions 77](#_Toc333601857)

[4.4 Use Case KSUC-4: Admin Sets Up Automated Transfer of Individual Keys from one KMS to Another 77](#_Toc333601858)

[4.4.1 Description / User Story 77](#_Toc333601859)

[4.4.2 Goal or Desired Outcome 78](#_Toc333601860)

[4.4.3 Notable Categorizations and Aspects 78](#_Toc333601861)

[4.4.4 Process Flow 79](#_Toc333601862)

[4.4.5 Extensions 79](#_Toc333601863)

[5 Use Cases for Hardware Security Module Key Management 80](#_Toc333601864)

[6 Use Cases for PGP Key Management 81](#_Toc333601865)

[6.1 Registration of a new Managed User 81](#_Toc333601866)

[6.1.1 Description / User Story 81](#_Toc333601867)

[6.1.2 Goal or Desired Outcome 81](#_Toc333601868)

[6.1.3 Environment 81](#_Toc333601869)

[6.1.4 Process Flow 82](#_Toc333601870)

[6.1.5 Extensions 82](#_Toc333601871)

[6.2 Registration of a Managed User with Pre-Existing Key Material 82](#_Toc333601872)

[6.2.1 Description / User Story 83](#_Toc333601873)

[6.2.2 Goal or Desired Outcome 83](#_Toc333601874)

[6.2.3 Environment 83](#_Toc333601875)

[6.2.4 Process Flow 83](#_Toc333601876)

[6.3 Registration of an Unmanaged User 83](#_Toc333601877)

[6.3.1 Description / User Story 83](#_Toc333601878)

[6.3.2 Goal or Desired Outcome 83](#_Toc333601879)

[6.3.3 Environment 83](#_Toc333601880)

[6.3.4 Process Flow 83](#_Toc333601881)

[6.4 Key Lookup by Email Address 83](#_Toc333601882)

[6.4.1 Description / User Story 83](#_Toc333601883)

[6.4.2 Goal or Desired Outcome 83](#_Toc333601884)

[6.4.3 Environment 83](#_Toc333601885)

[6.4.4 Process Flow 83](#_Toc333601886)

[6.5 Key Lookup via Key ID 83](#_Toc333601887)

[6.5.1 Description / User Story 83](#_Toc333601888)

[6.5.2 Goal or Desired Outcome 83](#_Toc333601889)

[6.5.3 Environment 83](#_Toc333601890)

[6.5.4 Process Flow 83](#_Toc333601891)

[6.6 Arbitrary Key Signing 84](#_Toc333601892)

[6.6.1 Description / User Story 84](#_Toc333601893)

[6.6.2 Goal or Desired Outcome 84](#_Toc333601894)

[6.6.3 Environment 84](#_Toc333601895)

[6.6.4 Process Flow 84](#_Toc333601896)

[6.7 Decipher a Data Encryption Key 84](#_Toc333601897)

[6.7.1 Description / User Story 84](#_Toc333601898)

[6.7.2 Goal or Desired Outcome 84](#_Toc333601899)

[6.7.3 Environment 84](#_Toc333601900)

[6.7.4 Process Flow 85](#_Toc333601901)

[6.8 Trust Validation 86](#_Toc333601902)

[6.8.1 Description / User Story 86](#_Toc333601903)

[6.8.2 Goal or Desired Outcome 86](#_Toc333601904)

[6.8.3 Environment 86](#_Toc333601905)

[6.8.4 Process Flow 86](#_Toc333601906)

[7 Use Cases for Registration of Key Management Clients 87](#_Toc333601907)

[8 Use Cases for Continuous Stream and Interval Delivery of Keys 88](#_Toc333601908)

[8.1 Deliver Key Stream Use Cases 89](#_Toc333601909)

[8.1.1 Configure Key Stream Service 89](#_Toc333601910)

[8.1.1.1 Pre-Condition 89](#_Toc333601911)

[8.1.1.2 Guarantee 89](#_Toc333601912)

[8.1.1.3 Trigger 89](#_Toc333601913)

[8.1.1.4 Main Success Scenario 89](#_Toc333601914)

[8.1.1.5 Extensions 90](#_Toc333601915)

[8.1.2 Request Key Stream 90](#_Toc333601916)

[8.1.2.1 Pre-Condition 90](#_Toc333601917)

[8.1.2.2 Guarantee 90](#_Toc333601918)

[8.1.2.3 Trigger 90](#_Toc333601919)

[8.1.2.4 Main Success Scenario 90](#_Toc333601920)

[8.1.2.5 Extensions 90](#_Toc333601921)

[8.1.3 Deliver Key Stream 90](#_Toc333601922)

[8.1.3.1 Pre-Condition 90](#_Toc333601923)

[8.1.3.2 Guarantee 90](#_Toc333601924)

[8.1.3.3 Trigger 91](#_Toc333601925)

[8.1.3.4 Main Success Scenario 91](#_Toc333601926)

[8.1.3.5 Extensions 91](#_Toc333601927)

[8.1.4 Cancel Key Stream 91](#_Toc333601928)

[8.1.4.1 Pre-Condition 91](#_Toc333601929)

[8.1.4.2 Guarantee 91](#_Toc333601930)

[8.1.4.3 Trigger 91](#_Toc333601931)

[8.1.4.4 Main Success Scenario 91](#_Toc333601932)

[8.1.4.5 Extensions 92](#_Toc333601933)

[8.2 Deliver Keys Asynchronously Use Cases 93](#_Toc333601934)

[8.2.1 Configure Asynchronous Delivery Service 93](#_Toc333601935)

[8.2.1.1 Pre-Condition 93](#_Toc333601936)

[8.2.1.2 Guarantee 93](#_Toc333601937)

[8.2.1.3 Trigger 93](#_Toc333601938)

[8.2.1.4 Main Success Scenario 93](#_Toc333601939)

[8.2.1.5 Extensions 94](#_Toc333601940)

[8.2.2 Request Asynchronous Delivery 94](#_Toc333601941)

[8.2.2.1 Pre-Condition 94](#_Toc333601942)

[8.2.2.2 Guarantee 94](#_Toc333601943)

[8.2.2.3 Trigger 94](#_Toc333601944)

[8.2.2.4 Main Success Scenario 94](#_Toc333601945)

[8.2.2.5 Extensions 94](#_Toc333601946)

[8.2.3 Deliver Keys Asynchronously 94](#_Toc333601947)

[8.2.3.1 Pre-Condition 94](#_Toc333601948)

[8.2.3.2 Guarantee 94](#_Toc333601949)

[8.2.3.3 Trigger 94](#_Toc333601950)

[8.2.3.4 Main Success Scenario 95](#_Toc333601951)

[8.2.3.5 Extensions 95](#_Toc333601952)

[8.2.4 Cancel Asynchronous Delivery 95](#_Toc333601953)

[8.2.4.1 Pre-Condition 95](#_Toc333601954)

[8.2.4.2 Guarantee 95](#_Toc333601955)

[8.2.4.3 Trigger 95](#_Toc333601956)

[8.2.4.4 Main Success Scenario 95](#_Toc333601957)

[8.2.4.5 Extensions 95](#_Toc333601958)

[9 Use Cases for Key Management for Storage Environments 96](#_Toc333601959)

[9.1 Use case KSTUC-1: Storage Device Requests Key from KMS 96](#_Toc333601960)

[9.1.1 Description / User Story 96](#_Toc333601961)

[9.1.2 Goal or Desired Outcome 96](#_Toc333601962)

[9.1.3 Notable Categorizations and Aspects 96](#_Toc333601963)

[9.1.4 Process Flow 97](#_Toc333601964)

[9.1.5 Extensions 97](#_Toc333601965)

[9.2 Use case KSTUC-2: Storage Device Registers Key with KMS 97](#_Toc333601966)

[9.2.1 Description / User Story 98](#_Toc333601967)

[9.2.2 Goal or Desired Outcome 98](#_Toc333601968)

[9.2.3 Notable Categorizations and Aspects 98](#_Toc333601969)

[9.2.4 Process Flow 99](#_Toc333601970)

[9.2.5 Extensions 99](#_Toc333601971)

[9.3 Use case KSTUC-3: Storage Device Sends Key Attributes KMS 99](#_Toc333601972)

[9.3.1 Description / User Story 99](#_Toc333601973)

[9.3.2 Goal or Desired Outcome 99](#_Toc333601974)

[9.3.3 Notable Categorizations and Aspects 99](#_Toc333601975)

[9.3.4 Process Flow 100](#_Toc333601976)

[9.3.5 Extensions 101](#_Toc333601977)

[9.4 Use case KSTUC-4: Storage Device Requests Encryption or Authentication Key from KMS 101](#_Toc333601978)

[9.4.1 Description / User Story 101](#_Toc333601979)

[9.4.2 Goal or Desired Outcome 101](#_Toc333601980)

[9.4.3 Notable Categorizations and Aspects 101](#_Toc333601981)

[9.4.4 Process Flow 102](#_Toc333601982)

[9.4.5 Extensions 102](#_Toc333601983)

[9.5 Use case KSTUC-5: Admin Deletes Key from KMS 102](#_Toc333601984)

[9.5.1 Description / User Story 102](#_Toc333601985)

[9.5.2 Goal or Desired Outcome 103](#_Toc333601986)

[9.5.3 Notable Categorizations and Aspects 103](#_Toc333601987)

[9.5.4 Process Flow 103](#_Toc333601988)

[9.5.5 Extensions 104](#_Toc333601989)

[10 Implementation Conformance 105](#_Toc333601991)

[Appendix A. Acknowledgements 106](#_Toc333601992)

[Original Authors of the initial contribution: 106](#_Toc333601993)

[Appendix B. Acronyms 107](#_Toc333601994)

[Appendix C. Revision History 108](#_Toc333601995)

Introduction

This Key Management Interoperability Protocol Use Cases Version 1.2 document is intended to complement the Key Management Interoperability Protocol Specification **[KMIP-Spec]** by describing the use cases that this version and prior versions of KMIP are intended to address. For each use case, the following information is provided:

* Introductory description of the use case
* User story to illustrate the interaction between user(s) and system for that use case
* Important information regarding the use case, such as actors
* Process flow for the use case, including significant variants to the primary (normal) process flow

Further assistance for implementing KMIP is provided by the KMIP Test Cases V1.2 document [KMIP-TC] that describes a set of recommended test cases and provides the TTLV (Tag/Type/Length/Value) format for the message exchanges defined by those test cases.

## Terminology

Use cases are identified in this document using the following K***x***UC-***n***, where:

* ***x*** represents the category of use case (Administration, Cloud, HSM, PGP, Registration of client, Stream, Storage)
* ***n*** represents a sequential number for the use case within that category.

For example, KAUC-1 is the first use case in the Administrative group of use cases.

For a list of terminology applicable to all KMIP documents refer to **[KMIP-Spec].**

Table 1 below provides additional terminology used in this document. Some definitions are copied directly from external sources. Other terms have been modified to fit the KMIP framework, while the rest were created anew. Please refer to the list of external sources for definitions of terms not explicitly defined in this document.

Table 1: Definition of Terms

|  |  |
| --- | --- |
| Term | Definition |
| Actor | An actor is a role that a user plays with respect to the system. Actors carry out use cases. An actor doesn’t have to be human. |
| Administrator | * A person who installs or maintains a system (for example, a Key Management system) or who uses it to manage system entities, users, and/or content (as opposed to application purposes; see also End User). An administrator is typically affiliated with a particular administrative domain and may be affiliated with more than one administrative domain. [SAML-GL, with modifications] |
| Application | A component that interacts with a KMC on behalf of a user. |
| Credential | Data that is transferred to establish a claimed principal identity. [SAML-Gl, x.800] |
| End User | * A natural person who makes use of resources for application purposes (as opposed to system management purposes; see Administrator, User). [SAML-GL] |
| Entity | * An active element of a computer/network system. For example, an automated process or set of processes, a subsystem, a person or group of persons that incorporates a distinct set of functionality. [SAML-GL] |
| Extension | Other scenarios related to the main success scenario. |
| Guarantee | A set of conditions the system will fulfill as a result of the use case execution. |
| Identifier | A data object (for example, a string) mapped to a system entity that uniquely refers to the system entity. A system entity may have multiple distinct identifiers referring to it. An identifier is essentially a "distinguished attribute" of an entity. [SAML-Gl] |
| Key Management Client (KMC) | That component of the system that issues requests to a Key Management Server on behalf of an application (client-to-server) or that receives notifications from a Key Management Server (server-to-client). |
| Key Management Server (KMS) | That component of the system that receives and processes requests from a Key Management Client (client-to-server) or that sends notifications to a Key Management Client (server-to-server). |
| Main Success Scenario (MSS) | Sequence of steps describing the main set of scenarios for a use case. |
| Pre-condition | A pre-condition describes what the system should ensure is true before the system allows the use case to begin. |
| Principal | A system entity whose identity can be authenticated. [SAML-Gl, X.811] |
| System | The set of components with which an actor interacts. |
| Subject | * A principal in the context of a security domain (i.e. a representation of a principal by a Key Management server). [SAML-GL, modified) |
| System | * An abstraction of all components and functionality that support a use case, representing the set of components with which an actor interacts. |
| System Entity | * See “entity” above. |
| Trigger | A trigger specifies the event that gets the use case started |
| User | * A natural person who makes use of a system and its resources for any purpose. [SAML-GL] |

Table 2 shows the names used for various types of users in the use cases described in this document.

Table 2: User Types

|  |  |
| --- | --- |
| User Type | Corresponding Users |
| Administrative Users | Xerxes, Yvonne, Zander |
| Agents | Alpha, Beta, Gamma, Delta |
| End Users | Alice, Bob, Carol, Dave |

Figure 1 below shows the relationship between entity and related terms as used in this document.



Figure 1: Entity Terminology.

Figure 2 below illustrates the relationship between entity, principal and subject.



Figure 2: Entity Terminology.

Figure 3 below illustrates the use of the term “system” in this document, particularly in relationship to components typically included within the system in this document. The interaction between components for which KMIP is used is enclosed in a dotted line within the larger system.



Figure 3: Illustration of System

## Normative References

[KMIP-Spec]

Key Management Interoperability Protocol Specification Version 1.2. [tbd].

[KMIP-Prof]

Key Management Interoperability Protocol Profiles Version 1.1. [tbd]

[RFC2119]

S. Bradner, Key words for use in RFCs to Indicate Requirement Levels, <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2119, March 1997.

[SAML-GL]

Glossary for the OASIS Security Assertion Markup Language (SAML). Version 2.0, 15 March 2005,  <http://docs.oasis-open.org/security/saml/v2.0/saml-glossary-2.0-os.pdf>

[SP800-57-1]

E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid, Recommendations for Key Management - Part 1: General (Revised), NIST Special Publication 800-57 part 1, March 2007, <http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57-Part1-revised2_Mar08-2007.pdf>

[SP800-57-2]

E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid, Recommendations for Key Management - Part 1: General (Revised), NIST Special Publication 800-57 part 2, March 2007, <http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57-Part2-revised2_Mar08-2007.pdf>

[SP800-57-3]

E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid, Recommendations for Key Management - Part 1: General (Revised), NIST Special Publication 800-57 part 3, March 2007, <http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57-Part3-revised2_Mar08-2007.pdf>

[SP800-130]

E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid, A Framework for Defining Cryptographic Key Management Systems, NIST Special Publication 800-130, Revised 15 June 2010, <http://csrc.nist.gov/publications/drafts/800-130/draft-sp800-130_june2010.pdf>

## Non-normative References

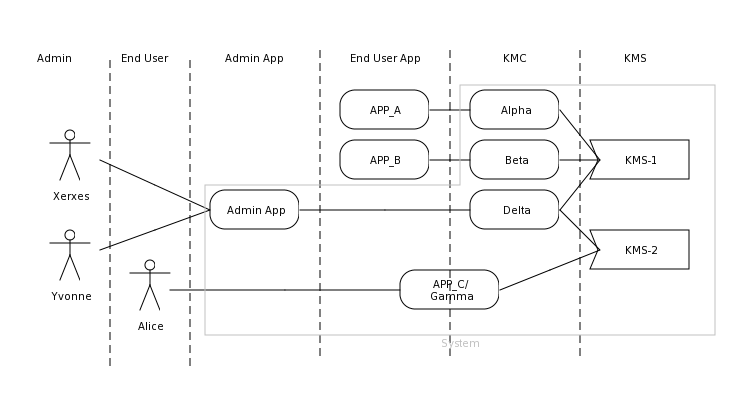
[KMIP**-TC]**

Key Management Interoperability Protocol Test Cases Version 1.2. [tbd]

[KMIP-UG]

Key Management Interoperability Protocol Usage Guide Version 1.2. [tbd]

Use Cases for Key Management Administration

The section describes assumptions that underlie the KMIP protocol and the implementation of clients and servers utilizing the protocol. The following diagram illustrates the use-cases in this chapter.



## Use Case KAUC-1a: End-User Administration

This section describes administration use cases covering management of end users.

### Description / User Story

Xerxes, an enterprise security administrator, is responsible for securing his company’s sensitive data and managing devices and users that access confidential data. Among other things Xerxes oversees database applications, called APP\_A and APP\_B, which use a key management server to help secure database contents. APP\_A and APP\_B are designed to use an external server to store the keys they use for cryptographic operations and request their keys only when needed. They use distinct access credentials to request keys from the key server. These credentials are configured both on the server and on the applications by Xerxes. In order to reflect the fact that APP\_A is an older application, which supports only a limited set of keys, Xerxes attaches an attribute (e.g. x-version=legacy) to APP\_A’s representation on the server.

### Goal or Desired Outcome

Administrator is able to manage End User or End User Application representations on a Key Management Server. This includes creation, deletion, editing of User representations (User Subjects).

### Notable Categorizations and Aspects

|  |  |
| --- | --- |
| Categories Covered:   * User management by an Admin * User attributes | Applicable Deployment and Service Models:   * N/A |
| Actors:   * Xerxes, a KMS administrator * Yvonne, a KMS administrator * APP\_A. A database application * APP\_B: A database application | System includes:   * Key Management Sever (KMS-1) * Administration Software Application (Admin App) * Key Management Client Applications Alpha, Beta and Delta, associated with APP\_A, APP\_B and Admin App respectively |
| Notable Services:   * None | |
| Dependencies:   * Servers provide a mechanism for Users and Admins to have credentials that can be used for authentication * Admin configures own initial access credentials on the server during server installation out-of-band * Admin uses initial access credentials to authenticate self | |
| Assumptions:   * User stories are limited to provisioning of users and their characteristics, not sharing of objects and access control * User credentials contain unique User Identifiers * Admin credentials contain unique Administrator Identifiers | |

### Process Flow

1. Xerxes logs into the KMS using an Administration Application.
2. Xerxes lists User Identifiers registered with server KMS-1 (APP\_A is listed).
3. Xerxes resets User credential for APP\_A.
4. Xerxes attaches attribute x-version=legacy to APP\_A’s entry on server KMS-1.
5. Xerxes creates APP\_B’s representation on the server with its access credentials.
6. Xerxes removes APP\_B’s access to the keys residing on KMS-1 by deleting User list entry for APP\_B, including associated credentials.
7. Xerxes lists User identifiers and attributes present on server KMS-1.

## Use Case KAUC-1b: End-User Self-Administration

Alice is a software developer in the same enterprise. She regularly needs to digitally sign a version of the software she creates when it is ready to be released. She uses a key management server to store the signing key and the code-signing application APP\_C, which also acts as a Key-Management Client (KMC). Keys are requested by APP\_C only when needed. The server, KMS-2, used by APP\_C is different from the one used by APP\_A and APP\_B, yet it is managed by Xerxes and Yvonne.

Alice’s credentials need to reflect her own identity and should not be known to Xerxes or Yvonne on a permanent basis. When generating Alice’s credentials, Yvonne creates a temporary password for Alice. When the code-signing application APP\_C is ran for the first time, Alice is asked to authenticate using the temporary password and also provide her own new password. This new password is subsequently used to access the software signing key by APP\_C.

### Goal or Desired Outcome

The goal of the functionality supporting this self-administration use-case is to provide a mechanism to securely change user’s own credentials as well as access or edit other meta-data associated with a user.

### Notable Categorizations and Aspects

|  |  |
| --- | --- |
| Categories Covered:   * User self-administration | Applicable Deployment and Service Models:   * N/A |
| Actors:   * Yvonne, a KMS administrator * Alice, a system End-User | System includes:   * Key Management Sever (KMS-2) * Administration Software Application (Admin App) * APP\_C: code-signing application **combined** with associated agent GAMMA, acting as a Key Management Client (KMC) |
| Notable Services:   * None | |
| Dependencies:   * Servers provide a mechanism for Users and Admins to have credentials that can be used for authentication * Admin configures own initial access credentials on the server during server installation out-of-band * Admin uses initial access credentials to authenticate self | |
| Assumptions:   * User stories are limited to provisioning of users and their characteristics, not sharing of objects and access control * User credentials contain unique User Identifiers * Admin credentials contain unique Administrator Identifiers | |

### Process Flow

1. Yvonne creates an End User Alice on KMS-2 with a set of temporary credentials, which are shared with Alice out-of-band.
2. Alice initiates the interaction with APP\_C, which prompts Alice for temporary credentials and requests permanent credentials for future use.
3. Every time Alice builds a new software image, APP\_C transparently requests a code-signing key associated with Alice from KMS-2, uses it to sign the image and erases the key from memory after use.
4. Alice, through APP\_C and agent Gamma, checks the validity period of her credentials on server KMS-2.
5. Alice, through APP\_C and agent Gamma, updates her own credentials on server KMS-2 before they expire.

### Extensions

[tbd]

## Use Case KAUC-1c: Multiple Administrators, Multiple Servers

Yvonne is also an enterprise security administrator working at Xerxes’s company. She too can access, configure or re-configure APP\_A’s credentials on the key management server KMS-1. KMS-2 is a Key Management Server in the same enterprise. It is also managed by Xerxes and Yvonne. Yvonne and Xerxes also use a distinct set of credentials when they access the key management servers.

Xerxes and Yvonne are not happy because they have to learn a new way of using a key management server when they start using KMS-2. They would like a consistent, or better yet, uniform way to perform their tasks of providing access to key management servers. They would also like to automate it. Ideally, they would like to use a single management application, which would allow them to perform their tasks regardless of which user on which server they are trying to manage.

### Goal or Desired Outcome

The goal is to provide a mechanism for managing administrator access to Key Management Servers.

Another goal is the ability to access different servers in a uniform fashion. This includes not only the ability of the servers to support the same message exchange protocol, but also have a common paradigm of client representations.

### Notable Categorizations and Aspects

|  |  |
| --- | --- |
| Categories Covered:   * Administrator management (by another Admin) | Applicable Deployment and Service Models:   * N/A |
| Actors:   * Xerxes, a KMS administrator * Yvonne, a KMS administrator * Zander, a KMS administrator | System includes:   * Key Management Severs (KMS-1, KMS-2) * Administration Software Application (Admin App) * Key Management Client Application (Delta) |
| Notable Services:   * None | |
| Dependencies:   * Servers provide a mechanism for Users and Admins to have credentials that can be used for authentication * Admin configures own initial access credentials on the server during server installation out-of-band * Admin uses initial access credentials to authenticate self | |
| Assumptions:   * User stories are limited to provisioning of users and their characteristics, not sharing of objects and access control * User credentials contain unique User Identifiers * Admin credentials contain unique Administrator Identifiers * Different servers in an enterprise may or may not provide single sign-on capabilities for server administrators. | |

### Process Flow

1. Xerxes lists all Admin identifiers registered with server KMS-1.
2. Xerxes lists all Admin identifiers registered with server KMS-2.
3. Xerxes creates new entry Yvonne in the list of administrators along with a new set of Admin credentials on KMS-1.
4. Xerxes deletes Zander’s entry in the list of administrators along with his credentials on KMS-2.

### Extensions

[tbd]

# Use Cases for Key Management in the Cloud

The KMIP use cases describe the key-management-related interactions between users and systems. The systems defined in these cloud-related use cases include at least two components: a Key Management Client (KMC) and a Key Management Server (KMS). The KMC is the client component that can be the application that performs encryption or an intermediate agent that fetches the key from the KMS to the application that performs encryption. The KMC should fetch key from the KMS on each request. If the KMC were to perform caching actions and policy enforcement in a certain deployment, it needs to have KMS functionality and acts like another KMS. Please refer to Server-to-Server scenarios for the functionality.

These use cases start from the assumption that one or both of these components reside outside the enterprise in a separate Cloud Service Provider (CSP) environment. They reflect three deployment models, differing primarily in terms of where the KMS component resides:

1. The KMS resides within an enterprise. The KMS communicates with KMCs in a CSP infrastructure (or within the enterprise, not discussed in these use cases).
2. The KMS resides as a tenant within a CSP infrastructure, but is specific to a given enterprise and is in a sub-environment within the CSP infrastructure dedicated to that tenant. The KMS communicates with KMCs that are tenants in the same CSP infrastructure, tenants in another CSP infrastructure or resident within the enterprise.
3. The KMS resides as a common resource shared across tenants within the CSP infrastructure. The KMS communicates with KMCs that are tenants in the same CSP infrastructure, tenants in another CSP infrastructures, or resident within the enterprise.

The first of these deployment models, in which the KMS is in the enterprise, is shown in Figure 4.

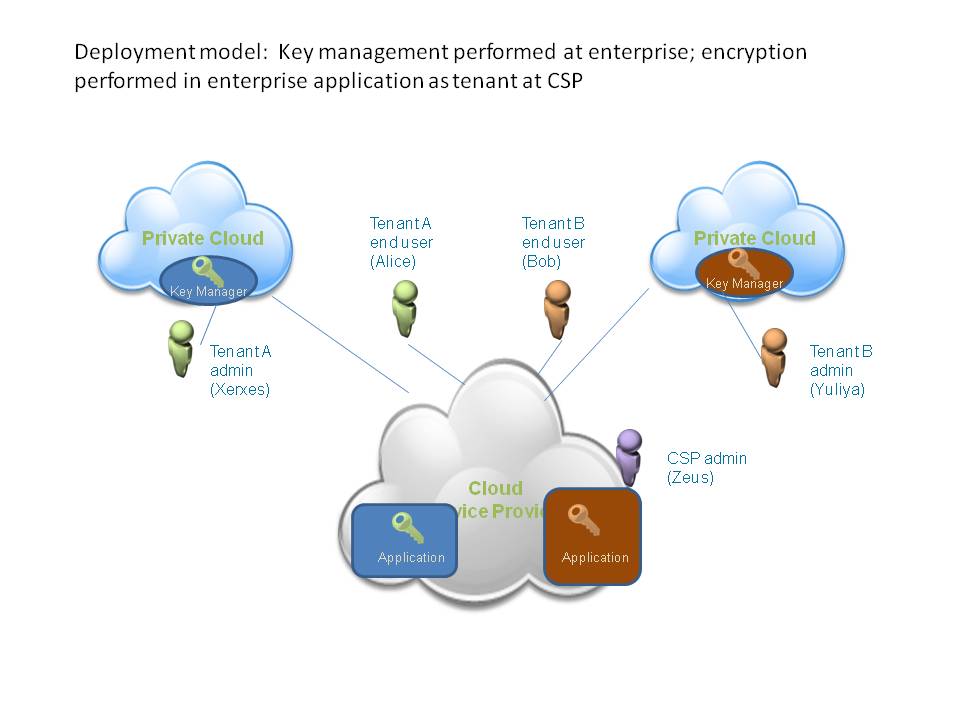


Figure 4: Deployment Model 1: Key Manager Server within Enterprise; Client in CSP

In this deployment model, a key management administrator within the enterprise is responsible for such activities as authorizing KMCs, defining key policy, monitoring the KMS and so on. A system administrator within the CSP is responsible for managing resource allocation for the application and so on. Use cases related to these two users in this deployment model are not defined in this document, as they do not have KMIP-related implications specific to cloud deployments. For example, see KAUC-1 for use cases related to interaction between an administrator within an enterprise and a KMS within the enterprise (including in a private cloud in that enterprise).

Use cases applicable to use of KMIP in this deployment model include:

* End user requesting cryptographic operation resulting in KMC in CSP requesting key-management related operations from KMS in enterprise [**KCUC-1a**]

The second deployment model, in which the KMS for an enterprise is a tenant in a portion of the CSP infrastructure dedicated to that tenant, is shown in Figure 5. In this model there may optionally be a key manager in the enterprise that acts as a master or a slave for the tenant-dedicated key manager in the CSP.

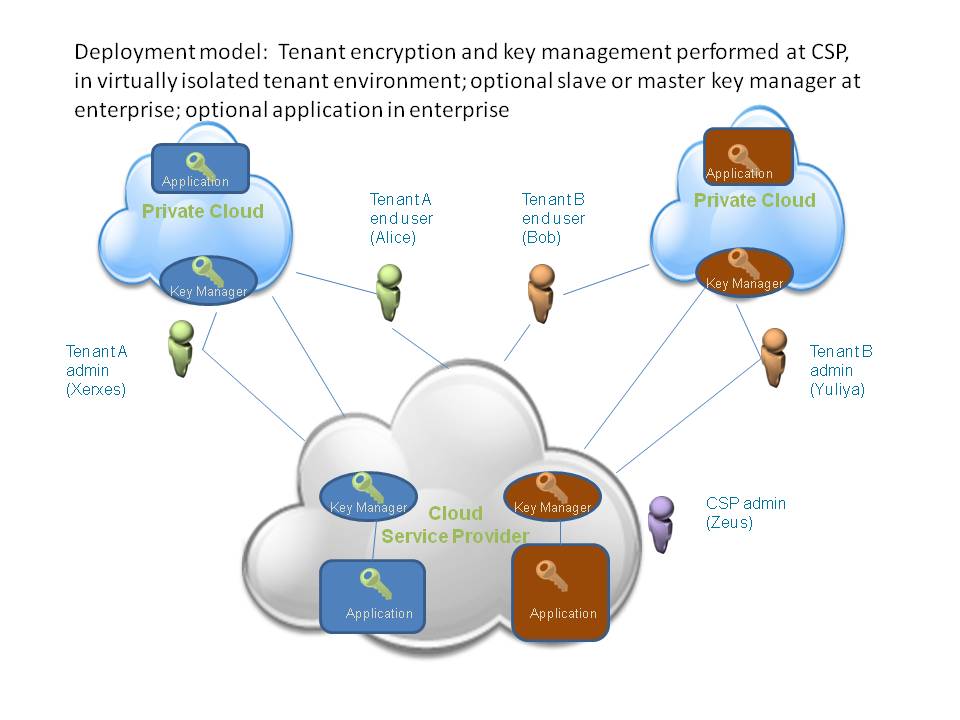


Figure 5: Deployment Model 2: Dedicated KMS in CSP Infrastructure

In this deployment model, a key management administrator within the enterprise is responsible for such activities as authorizing KMCs, defining key policy, monitoring the KMS and so on for the KMS in the CSP (and optionally in the enterprise). A system administrator within the CSP is responsible for managing resource allocation for the KMS, but is not allowed to perform key-management-related administrative activities. Use cases related to the CSP administrator in this deployment model are not defined in this document, as they do not have KMIP-related implications specific to cloud deployments.

Use cases applicable to use of KMIP in this deployment model include:

* Admin use cases applicable to all variants of this deployment model
  + CSP admin manages resources for tenant KMS at CSP **[KCUC-2a]**
  + Tenant admin requesting administrative operation from KMS at CSP (client-to-server) **[KCUC-2b]**
  + CSP admin provisions resources for a new tenant, sets up credentials for tenant admin **[KCUC-2c]**
  + Tenant admin provisions clients and generates credentials for clients to access KMS **[KCUC-2d]**
* End user use cases for variants of this deployment model
  + End user requesting cryptographic operation resulting in KMC in enterprise requesting one or more keys from tenant KMS in CSP (client-to-server) **[KCUC-2e]**
  + End user requesting cryptographic operation resulting in tenant KMC in CSP requesting one or more keys from tenant KMS in CSP (client-to-server) **[KCUC-2f]**
* For deployment models with CSP KMS as slave to enterprise KMS
  + Tenant admin requests enterprise KMS to export one or more tenant keys to KMS at CSP (server-to-server) **[KCUC-2g]**
  + Tenant admin requests enterprise KMS to export policy to KMS at CSP (server-to-server) **[KCUC-2h]**
  + Tenant admin requests enterprise KMS to send key-specific attribute changes to KMS at CSP (server-to-server) **[KCUC-2i]**
* For deployment model with enterprise KMS as slave to CSP KMS
  + Tenant admin requests CSP KMS to export one or more tenant keys to enterprise KMS (server-to-server) **[KCUC-2j]**
  + Tenant admin requests CSP KMS to export policy to enterprise KMS (server-to-server) **[KCUC-2k]**
  + Tenant admin requests CSP KMS to send key-specific attribute changes to enterprise KMS (server-to-server) **[KCUC-2l]**

The third deployment model, in which the enterprise uses a shared KMS provided by the CSP, is shown in Figure 6.

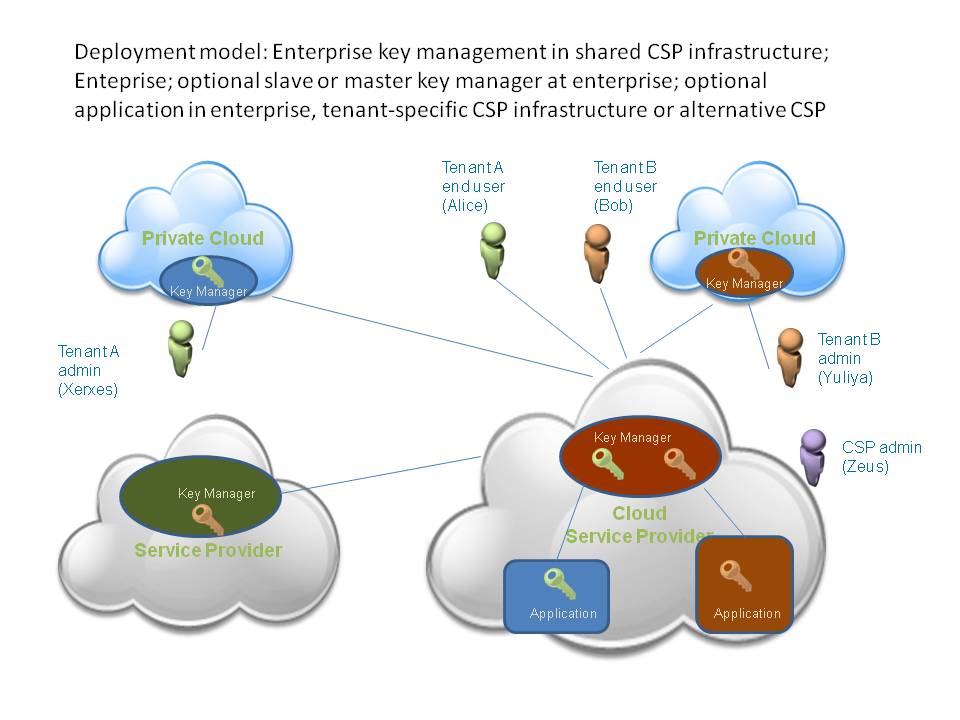


Figure 6: Deployment Model 3: Shared Key Server in CSP Infrastructure

Use cases applicable to use of KMIP in this deployment model include:

* Admin use cases applicable to all variants of this deployment model
  + CSP admin manages resources for KMS at CSP (KMIP interaction related to segmentation of tenants in CSP KMS) **[KCUC-3a]**
  + Tenant admin requests tenant-specific operations in KMS at CSP (client-to-server) **[KCUC-3b]**
* For deployment models with KMS only at CSP
  + End user agent requests cryptographic operation resulting in tenant KMC at CSP accessing CSP KM (client-to-server) **[KCUC-3c]**
* For deployment models with CSP KMS as slave to enterprise KMS
  + Tenant admin requests enterprise KMS to move/export keys to CSP KMS (server-to-server) **[KCUC-3d]**
  + Tenant admin requests enterprise KMS to export policy to KMS at CSP (server-to-server) **[KCUC-2e]**
  + Tenant admin requests enterprise KMS to send/notify key-specific attribute changes to KMS at CSP (server-to-server) **[KCUC-2f]**
* For deployment model with enterprise KMS as slave to CSP KMS
  + Tenant admin creates tenant keys at CSP KMS and moves/exports them to enterprise KMS (server-to-server) **[KCUC-3g]**
  + Tenant admin requests CSP KMS to export policy to tenant KMS (server-to-server) **[KCUC-2h]**
  + Tenant admin requests CSP KMS to send key-specific attribute changes to enterprise KMS (server-to-server) **[KCUC-2i]**
* For deployment models with KMS in alternative CSP
  + Tenant admin migrates keys from CSP KMS to alternative CSP KMS (server-to-server) **[KCUC-3j]**
  + Tenant admin requests CSP KMS to export policy to alternative CSP KMS (server-to-server) **[KCUC-3k]**
  + Tenant admin requests CSP KMS to send key-specific attribute changes to alternative CSP KMS (server-to-server) **[KCUC-3l]**
* For all deployment models, use cases related to management of CSP KMS
  + Registration of KMC **[KCUC-3m]**
  + Revocation of KMC registration **[KCUC-3n]**
  + KMC assurance of CSP KMS acting within policy **[KCUC-3o]**
  + KMS scaling in response to increase in KMC demand level **[KCUC-3p]**
  + KMC caching of keys or other material from KMS **[KCUC-3q]**

Use cases are described below for each of these three deployment models. The applicable model for each use case is defined in the “Applicable Deployment and Service Models” for each use case.

## Use Cases for Deployment Model 1: KMS in Enterprise, KMC in CSP

There is one use case defined for this deployment model:

* End user requesting cryptographic operation resulting in KMC in CSP requesting key-management related operations from KMS in enterprise [**KCUC-1a**]

Note that use cases relevant to this deployment model, but which do not have significant implications for the KMIP protocol, are not included in this document.

### Use Case KCUC-1a: End-user requests key (KMC in CSP requests managed object from KMS in enterprise)

As described in the KMIP Specification, KMIP exchanges between a client and server support a number of operations on a number of objects with a number of attributes. This use case is similar to any KMIP client/server exchange: any KMIP-defined object, operation and attribute can be included in an exchange between the KMS and KMC in this deployment model.

However, the request for KMIP operations by an KMC residing in a CSP to a KMS residing outside that CSP has important implications in terms of the trust establishment model (for example, TLS channel may be proxied by CSP), protection of keys (for example, a possible requirement for key wrapping) and the additional context that may need to be exchanged (for example, finer-grained application-related information used in authenticating and authorizing the request).

#### User Story and Component Interaction

Alice is an internet user who wants to order a new laptop through the AllPC order processing application, which is hosted by BestCloud CSP. At its highest level, this entails Alice submitting the order to the system (comprising the order processing application and various key-management-related components) and receiving back confirmation that the order has been accepted.

Within the system with which Alice interacts, the ALLPC order processing application component encrypts the order information once Alice has entered it, before storing the order in a database component, also resident in the CSP infrastructure. In order to encrypt the information, the ALLPC order processing application uses a symmetric encryption key that is changed on a periodic basis. The key has not yet been requested from the KMS component for that period. Therefore the ALLPC application requests the KMC component to request the key securely from the KMS component. The KMS securely returns the key appropriate for that period to the application, which encrypts the order, stores the order and then completes the transaction with Alice by performing payment processing, submission of the order to a fulfillment application and so on.

Other user stories corresponding to this use case could entail other operations (register, revoke, modify attributes, etc), other objects (asymmetric keys, certificates, PGP keys, etc) and other attributes (operation policy, lease, state, etc).

#### Goal or Desired Outcome

The goal of this use case is to have a valid key-related request, initiated by an end user via a CSP be processed a KMS component in the enterprise. This includes not only the ability of the KMS to support the message exchange protocol, but also to perform the operations, understand the objects and interpret the attributes it receives from the KMC.

#### Notable Categorizations and Aspects

|  |  |
| --- | --- |
| Categories Covered:   * Trust establishment * KMC request * KMS response | Applicable Deployment and Service Models:   * Deployment Model 1 (KMS in enterprise, KMC in CSP) |
| Actors:   * End user (Alice or Bob) | System:  System includes the following components   * End-user Application (at CSP) * KMC (at CSP) * KMS (in enterprise) |
| Notable Services:   * Request/receive cryptographic object | |
| Dependencies:   * Servers and clients have credentials that can be used to establish a mutually-authenticated communication between them. * Client has been configured to identify target server and to accept server credential * Server has been configured to accept client credential and request. * Client and server systems are active and able to initiate (client) and receive (server) messages. | |
| Assumptions:   * Client and server credentials contain unique identifiers that can be used to authenticate and authorize communication between them. | |

#### Process Flow

1. Alice requests new laptop from system
   1. Alice enters laptop request in end-user application
   2. Application component sends key request to KMC component.
   3. KMC component securely sends request to KMS component.
2. Alice receives response from system
   1. KMS component processes request, creates response message and securely sends message to KMC component.
   2. KMC component sends response to application component
   3. Application component displays response to Alice

#### Extensions

[tbd]

## Use Cases for Deployment Model 2: KMS in Dedicated CSP Infrastructure; KMC in Enterprise

There are ten use cases defined for this deployment model:

* Use cases applicable to all variants of this deployment model
  + CSP admin manages resources for tenant KMS at CSP **[KCUC-2a]**
  + Tenant admin requesting administrative operation from KMS at CSP (client-to-server) **[KCUC-2b]**
  + CSP admin provisions resources for a new tenant, sets up credentials for tenant admin **[KCUC-2c]**
  + Tenant admin provisions clients and generates credentials for clients to access KMS **[KCUC-2d]**
* End user use cases for variants of this deployment model
  + End user requesting cryptographic operation resulting in KMC in enterprise requesting one or more keys from tenant KMS in CSP (client-to-server) **[KCUC-2e]**
  + End user requesting cryptographic operation resulting in tenant KMC in CSP requesting one or more keys from tenant KMS in CSP (client-to-server) **[KCUC-2f]**
* For deployment models with CSP KMS as slave to enterprise KMS
  + Tenant admin requests enterprise KMS to export one or more tenant keys to KMS at CSP (server-to-server) **[KCUC-2g]**
  + Tenant admin requests enterprise KMS to export policy to KMS at CSP (server-to-server) **[KCUC-2h]**
  + Tenant admin requests enterprise KMS to send key-specific attribute changes to KMS at CSP (server-to-server) **[KCUC-2i]**
* For deployment model with enterprise KMS as slave to CSP KMS
  + Tenant admin requests CSP KMS to export one or more tenant keys to enterprise KMS (server-to-server) **[KCUC-2j]**
  + Tenant admin requests CSP KMS to export policy to enterprise KMS (server-to-server) **[KCUC-2k]**
  + Tenant admin requests CSP KMS to send key-specific attribute changes to enterprise KMS (server-to-server) **[KCUC-2l]**

Note that use cases relevant to this deployment model, but which either are covered elsewhere in this document or do not have significant implications for the KMIP protocol, are not included in this document.

### Use case KCUC-2a: CSP Administrator manages resources for tenant KMS at CSP

Although the management of CSP resources for a tenant-specific KMS does not have implications for KMIP, the use case is important as it details the restrictions on interaction between the CSP admin and the key management capabilities that are instantiated in the CSP infrastructure. All administrative tasks related to the tenant KMS capabilities (defining key policy, authorizing key clients, delegating administrative responsibility, etc) are performed by the tenant KMS administrator, as defined in Use Case KCUC-2b below.

The CSP admin is responsible for administrative tasks such as allocating compute and storage resources to the tenant KMS, tracking and resolving tenant KMS performance issues, and setting up and maintaining security measures related to the tenant KMS (such as firewall rules, anti-virus, intrusion detection, tenant isolation, log inspection etc). These responsibilities can give the CSP admin visibility into the keys stored in the tenant KMS, such through memory dumps in which cached keys may be visible, depending on what security controls are in place. The implications of CSP admin access are not defined in this use case, however, as they have no impact on KMIP.

In order to perform these administrative tasks, however, the CSP admin must have visibility into the physical and virtual networks over which KMIP messages travel. This use case, therefore, implies that KMIP messages should be protected against this potential for access by the CSP admin in any of a number of ways (for example, message encryption at the protocol level, channel encryption, and so on).

#### Description / User Story

Zander is a CSP administrator at BestCloud CSP who is responsible for managing physical and virtual resources in the CSP virtual environment (CPU, storage, network). BestCloud has been selected by the AllPC company to host both its AllPC application and the KMS supporting that application (and potentially other applications) within a portion of the BestCloud virtual infrastructure dedicated to AllPC. Zander uses tools provided by the BestCloud physical and virtual environment to perform such activities as monitoring performance of the physical and virtual resources, inspecting memory dumps and other information to analyze system failures and monitoring security information to detect potential malware infection of the BestCloud environment.

Zander should not be able to interact with the tenant-specific administrative tools for the AllPC application and KMS. However, interactions such as those described in use case KCUC-1a above take place across the CSP infrastructure and within CPU resources allocated in the CSP infrastructure. One example of this, the routine monitoring of security-related events in network event logs, is shown in this use case.

Other user stories corresponding to this use case could entail other operations (monitor cpu usage, allocate or de-allocate resources, etc) and other objects or data (server event logs, firewall activity, etc.).

#### Goal or Desired Outcome

The goal of this use case is to have the CSP administrator perform authorized monitoring activities on network information related to the interactions between the tenant KMC and tenant KMS. This includes not only the ability of the CSP administrator to detect anomalous volume of network activity that might indicate a malware-infected tenant application.

#### Notable Categorizations and Aspects

|  |  |
| --- | --- |
| Categories Covered:   * Physical infrastructure monitoring * Virtual infrastructure monitoring | Applicable Deployment and Service Models:   * Deployment Model 2 (KMS in tenant-specific CSP infrastructure) |
| Actors:   * CSP administrator (Zander) | System:  System includes the following components   * Physical/virtual network monitor * End user Application (at CSP) * KMS administrative application (at CSP) * KMC (at CSP) * KMS (at CSP) |
| Notable Services:   * None | |
| Dependencies:   * CSP monitoring of network traffic within dedicated tenant domain is allowed | |
| Assumptions:   * One or more instances of other use cases for this deployment model, such as KCUC-2c, have been performed prior to this use case. | |

#### Process Flow

1. Zander reviews activity recorded in network monitor for the preceding interaction.
   1. Zander requests display of network information from network monitor component.
   2. Network monitor component prepares and displays interaction information to Zander.
   3. Use case KCUC-2b: Tenant admin manages KMS at CSP (admin\_client-to-server)

In the deployment model for this use case, in which the enterprise KMS is deployed within a dedicated part of the CSP infrastructure, the tenant administrator is responsible for the same kinds of tasks as when the KMS is deployed within the enterprise. This includes such activities as defining key policy, authorizing KMCs, delegating administrative responsibility and so on. In this model, however, those activities are performed by means of a KMS management application residing either within the enterprise or within the tenant-dedicated portion of the CSP infrastructure.

These administrative activities are not currently instrumented through objects and operations in KMIP. However, such objects and operations are being considered for V1.2 and, if defined, the particular considerations for the security of such objects and operations in this deployment model would need to be addressed. These considerations include potential vulnerability of administrative operations to capture, corruption or subversion by adversaries.

Note that tenant administrative capabilities or activities not expected to be relevant to the KMIP V1.2 protocol, such as administrator login to the admin app, administrator authorization by the admin app or logging of activity by KMC and KMS are not included in this use case.

#### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise specific KMS that is hosted by BestCloud CSP. This entails Xerxes submitting administrative requests to the system (comprising the administrative application and various key-management-related components) and receiving back confirmation that the request has been received and processed. Such a request might be to have the KMS set a particular key rotation policy as the default for all keys created in the KMS.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

Other user stories corresponding to this use case could entail other operations (modify key policy, etc), other objects (asymmetric keys, certificates, PGP keys, etc) and other attributes (operation policy, permissions, etc).

#### Goal or Desired Outcome

The goal of this use case is to have a valid administrative request, initiated by a tenant administrative user for an enterprise, be processed by the KMS component in the CSP. This includes not only the ability of the KMS to support the message exchange protocol, but also to perform the operations, understand the objects and interpret the attributes received from the KMC on behalf of the tenant administrative user.

#### Notable Categorizations and Aspects

|  |  |
| --- | --- |
| Categories Covered:   * Trust establishment * KMC request * KMS response | Applicable Deployment and Service Models:   * Deployment Model 2 (KMS in CSP, KMC in CSP) |
| Actors:   * KMS tenant administrator (Xerxes) | System:  System includes the following components   * Administrative Application (in CSP) * KMC (in administrative app CSP) * KMS (in CSP) |
| Notable Services:   * Request/response for key management administration operation | |
| Dependencies:   * Servers and clients have credentials that can be used to establish a mutually-authenticated communication between them. * Client has been configured to identify target server and to accept server credential * Server has been configured to accept client credential and request. * Client and server systems are active and able to initiate (client) and receive (server) messages. | |
| Assumptions:   * Client and server credentials contain unique identifiers that can be used to authenticate and authorize communication between them. | |

#### Process Flow

1. Xerxes requests admin app to set key rotation policy for KMS
   1. Admin application component sends admin request to KMC component.
   2. KMC component sends admin request to KMS component via secure channel.
2. Xerxes receives response from system
   1. KMS component processes request, creates response message and sends message to KMC component via established secure channel.
   2. KMC component sends response to admin application component
   3. Admin application component (in system) displays response to Xerxes

#### Extensions

tbd

### Use case KCUC-2c: CSP Administrator provisions resources for a new tenant and sets up credentials for tenant administrator

As described in Use Case KCUC-2a, the CSP admin is responsible for administrative tasks related to tenant resources. This use case describes additional tasks related to tenant administrators.

In order to perform these administrative tasks, however, the CSP admin may have visibility into the credentials used by tenant administrators. This use case, therefore, implies that administrative messages sent via should be protected against this potential for access by the CSP admin or attackers in any of a number of ways (for example, message encryption at the protocol level, channel encryption, and so on).

#### Description / User Story

Zander is a CSP administrator at BestCloud CSP who is responsible for establishing and managing tenant administrator credentials and privileges in the CSP environment. BestCloud has been selected by the AllPC company to host both its AllPC application and the KMS supporting that application (and potentially other applications) within a portion of the BestCloud virtual infrastructure dedicated to AllPC. Zander uses tools provided by the BestCloud physical and virtual environment to perform such activities as loading credentials for tenant administrators into known locations, defining access privileges related to those credentials and so on.

Zander should not be able to interact with the tenant-specific administrative tools for the AllPC application and KMS. However, interactions such as those described in use case KCUC-1a above take place across the CSP infrastructure and within CPU resources allocated in the CSP infrastructure.

Other user stories corresponding to this use case could entail other operations (modify tenant administrator, monitor tenant administrator, remove tenant administrator).

#### Goal or Desired Outcome

Same as KCUC-2a

#### Notable Categorizations and Aspects

Same as KCUC-2a

#### Process Flow

1. Zander receives request to enable a tenant administrator access to dedicated parts of CSP infrastructure, such as access to directory locations for storing KMC entity credentials for application access to tenant-dedicated KMS.
   1. Zander verifies request.
   2. Zander performs requested action.

#### Extensions

tbd

### Use case KCUC-2d: Tenant admin provisions clients and generates credentials for clients to access KMS at CSP (admin\_client-to-server)

As described in Use Case KCUC-2b, in the deployment model for this use case, the tenant administrator is responsible for the same kinds of tasks as when the KMS is deployed within the enterprise. This includes such activities as establishing credentials for KMCs that need to access KMS located in the CSP. In this model, however, those activities are performed by means of a KMS management application residing either within the enterprise or within the tenant-dedicated portion of the CSP infrastructure.

These administrative activities are not currently instrumented through objects and operations in KMIP. However, such objects and operations are being considered for V1.2 and, if defined, the particular considerations for the security of such objects and operations in this deployment model would need to be addressed. These considerations include potential vulnerability of administrative operations to capture, corruption or subversion by adversaries.

Note that tenant administrative capabilities or activities not expected to be relevant to the KMIP V1.2 protocol, such as administrator login to the admin app, administrator authorization by the admin app or logging of activity by KMC and KMS are not included in this use case.

#### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise specific KMS that is hosted by BestCloud CSP. This entails Xerxes submitting administrative requests to the system (comprising the administrative application and various key-management-related components) and receiving back confirmation that the request has been received and processed. Such a request might be to have the KMS to generate and accept a particular X.509 certificate associated with a KMC as the credential for the KMS to use when authenticating and authorizing requests from that KMC.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

Other user stories corresponding to this use case could entail other operations (remove client credential).

#### Goal or Desired Outcome

Same as KCUC-2b.

#### Notable Categorizations and Aspects

Same as KCUC-2b.

#### Process Flow

1. Xerxes requests admin app to set X.509 certificate as credential for KMS
   1. Admin application component sends admin request to KMC component.
   2. KMC component sends admin request to KMS component via secure channel.
2. Xerxes receives response from system
   1. KMS component processes request, creates response message and sends message to KMC component via established secure channel.
   2. KMC component sends response to admin application component
   3. Admin application component (in system) displays response to Xerxes

#### Extensions

tbd

### Use case KCUC-2e: End-user requests key (KMC in enterprise requests managed object from KMS in CSP)

This use case is similar to KCUC-1a. In this case, however, the request is sent from a KMC in the enterprise to a tenant KMS in a dedicated area of the CSP infrastructure.

The request for KMIP operations by an KMC residing in an enterprise to a KMS residing in a dedicated part of a CSP infrastructure has important implications in terms of the trust establishment model (for example, TLS channel may be proxied by CSP), protection of keys (for example, a possible requirement for key wrapping) and the additional context that may need to be exchanged (for example, finer-grained application-related information used in authenticating and authorizing the request).

#### User Story and Component Interaction

The user story is the same as for KCUC-1a.

Within the system with which Alice interacts, the ALLPC order processing application component and the KMC reside within the enterprise. The ALLPC application requests the KMC component to request the key securely from the KMS component at the CSP. The KMS securely returns the key to the KMC.

Other user stories corresponding to this use case could entail other operations (register, revoke, modify attributes, etc), other objects (asymmetric keys, certificates, PGP keys, etc) and other attributes (operation policy, lease, state, etc).

#### Goal or Desired Outcome

The goal of this use case is to have a valid key-related request, initiated by an end user via an application within the enterprise be processed a KMS component in the CSP. This includes not only the ability of the KMS to support the message exchange protocol, but also to perform the operations, understand the objects and interpret the attributes it receives from the KMC.

#### Notable Categorizations and Aspects

|  |  |
| --- | --- |
| Categories Covered:   * Trust establishment * KMC request * KMS response | Applicable Deployment and Service Models:   * Deployment Model 2 (KMS in CSP, KMC in enterprise) |
| Actors:   * End user (Alice) | System:   * System includes the following components * End-user Application (in enterprise) * KMC (in enterprise) * KMS (at CSP) |
| Notable Services:   * Request/receive cryptographic object | |
| Dependencies:   * Servers and clients have credentials that can be used to establish a mutually-authenticated communication between them. * Client has been configured to identify target server and to accept server credential * Server has been configured to accept client credential and request. * Client and server systems are active and able to initiate (client) and receive (server) messages. | |
| Assumptions:   * Client and server credentials contain unique identifiers that can be used to authenticate and authorize communication between them. | |

#### Process Flow

1 Alice requests new laptop from system

* 1. Alice enters laptop request in end-user application
  2. Application component in enterprise sends key request to KMC component.
  3. KMC component securely sends request to KMS component at CSP.

1. Alice receives response from system
   1. KMS component processes request, creates response message and securely sends message to KMC component in enterprise.
   2. KMC component sends response to application component
   3. Application component displays response to Alice

#### Extensions

tbd

### Use case KCUC-2f: End-user requests key (KMC in CSP requests managed object from KMS in CSP)

This use case is similar to KCUC-2e. In this case, however, the request is sent from a tenant KMC in a CSP to a tenant KMS also in a CSP, with the KMS in a dedicated area of the CSP infrastructure.

The request for KMIP operations by an KMC residing in a CSP to a KMS also residing in that CSP has important implications in terms of the trust establishment model (for example, TLS channel may not be point-to-point), protection of keys (for example, a possible requirement for key wrapping) and the additional context that may need to be exchanged (for example, finer-grained application-related information used in authenticating and authorizing the request).

#### User Story and Component Interaction

User story is the same as for KCUC-2c.

Within the system with which Alice interacts, the ALLPC order processing application component and the KMC reside within the CSP. The ALLPC application requests the KMC component to request the key securely from the KMS component at the CSP. The KMS securely returns the key to the KMC.

Other user stories corresponding to this use case could entail other operations (register, revoke, modify attributes, etc), other objects (asymmetric keys, certificates, PGP keys, etc) and other attributes (operation policy, lease, state, etc).

#### Goal or Desired Outcome

Same as KCUC-2e.

#### Notable Categorizations and Aspects

Same as KCUC-2e.

#### Process Flow

1. Alice requests new laptop from system

1.1 Alice enters laptop request in end-user application at CSP.

1.2 Application component sends key request to KMC component.

* 1. KMC component securely sends request to KMS component also at CSP.

1. Alice receives response from system
   1. KMS component processes request, creates response message and securely sends message to KMC component.
   2. KMC component sends response to application component
   3. Application component displays response to Alice

#### Extensions

tbd

### Use case KCUC-2g: Tenant admin requests enterprise KMS to export tenant keys to KMS at CSP (server-to-server)

This use case is a special case of KCUC-2b, resulting in cryptographic objects being sent from an enterprise KMS to a KMS at a CSP.

This use case entails implications both for what is included in the exported material and how that material is protected.

#### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS and also the associated tenant-specific KMS that is hosted by BestCloud CSP. Xerxes submits an administrative request to the system (comprising the administrative application and various key-management-related components) to export keys from the enterprise KMS to the tenant-specific KMS at the CSP. He receives back confirmation that the request has been received and processed.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS performs the key export operation, then securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

#### Goal or Desired Outcome

Same as KCUC-2b

#### Notable Categorizations and Aspects

Same as KCUC-2b

#### Process Flow

1. Xerxes requests admin app to export keys to KMS at CSP
   1. Admin application component in enterprise sends admin request to KMC component.
   2. KMC component in enterprise securely sends admin request to KMS component in CSP.
2. Xerxes receives response from system
   1. KMS component prepares keys for export and sends them to KMS in enterprise.
   2. KMS in enterprise processes exported keys and returns response.
   3. KMS in CSP creates response message and securely sends message to KMC component in enterprise.
   4. KMC component sends response to admin application component
   5. Admin application component (in system) displays response to Xerxes

#### Extensions

[tbd]

### Use case KCUC-2h: Tenant Admin requests Enterprise KMS to export policy to KMS at CSP

This use case is a special case of KCUC-2b, resulting in cryptographic policy being sent from an enterprise KMS to a KMS at a CSP.

This use case entails implications both for what is included in the policy and how that policy is protected from tampering or other attacks.

#### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS and also the associated tenant-specific KMS that is hosted by BestCloud CSP. Xerxes submits an administrative request to the system (comprising the administrative application and various key-management-related components) to send cryptographic policy from the enterprise KMS to the tenant-specific KMS at the CSP. He receives back confirmation that the request has been received and processed.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS performs the policy import operation, then securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

#### Goal or Desired Outcome

Same as KCUC-2b

#### Notable Categorizations and Aspects

Same as KCUC-2b.

#### Process Flow

1. Xerxes requests admin app to send policy to KMS at CSP
   1. Admin application component in enterprise sends admin request to KMC component.
   2. KMC component in enterprise securely sends admin request to KMS component in CSP.
2. Xerxes receives response from system
   1. KMS in CSP processes policy, creates response message and securely sends message to KMC component in enterprise.
   2. KMC component sends response to admin application component.
   3. Admin application component (in system) displays response to Xerxes.

#### Extensions

tbd

### Use case KCUC-2i: Enterprise KMS sending key-specific attribute changes to KMS at CSP (server-to-server)

This use case is a special case of KCUC-2b, resulting in cryptographic object attributes being sent from an enterprise KMS to a KMS at a CSP.

This use case entails implications both for what is included in the exported material and how that material is protected.

#### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS and also the associated tenant-specific KMS that is hosted by BestCloud CSP. Xerxes submits an administrative request to the system (comprising the administrative application and various key-management-related components) to send attributes from the enterprise KMS to the tenant-specific KMS at the CSP. He receives back confirmation that the request has been received and processed.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS performs the requested operation, then securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

#### Goal or Desired Outcome

Same as KCUC-2b

#### Notable Categorizations and Aspects

Same as KCUC-2b

#### Process Flow

1. Xerxes requests admin app to send attributes to KMS at CSP
   1. Admin application component in enterprise sends admin request to KMC component.
   2. KMC component in enterprise securely sends admin request to KMS component in CSP.
2. Xerxes receives response from system
   1. KMS in CSP processes attributes, creates response message and securely sends message to KMC component in enterprise.
   2. KMC component sends response to admin application component
   3. Admin application component (in system) displays response to Xerxes

#### Extensions

tbd

### Use case KCUC-2j: CSP KMS exporting keys to Enterprise KMS (server-to-server)

This use case is a special case of KCUC-2b, resulting in cryptographic object attributes being sent from KMS at a CSP to enterprise KMS. It is the inverse of use case KCUC-2e.

This use case entails implications both for what is included in the exported material and how that material is protected.

#### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS and also the associated tenant-specific KMS that is hosted by BestCloud CSP. Xerxes submits an administrative request to the system (comprising the administrative application and various key-management-related components) to export keys from the tenant-specific KMS at the CSP and have them sent to the enterprise KMS. He receives back confirmation that the request has been received and processed.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS performs the requested operation, then securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

#### Goal or Desired Outcome

Same as KCUC-2b

#### Notable Categorizations and Aspects

Same as KCUC-2b

#### Process Flow

1. Xerxes requests admin app to send key export request to KMS at CSP
   1. Admin application component in enterprise sends admin request to KMC component.
   2. KMC component in enterprise securely sends admin request to KMS component in CSP.
2. Xerxes receives response from system
   1. KMS in CSP processes export request, creates response message and securely sends keys to KMS in enterprise.
   2. KMS in enterprise sends message to KMC component in enterprise.
   3. KMC component sends response to admin application component.
   4. Admin application component (in system) displays response to Xerxes.

#### Extensions

tbd

### Use case KCUC-2k: Tenant Admin requests CSP KMS to export policy to KMS at Enterprise

This use case is a special case of KCUC-2b, resulting in cryptographic object attributes being sent from an KMS at a CSP to enterprise KMS . It is the inverse of use case KCUC-2f.

This use case entails implications both for what is included in the exported material and how that material is protected.

#### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS and also the associated tenant-specific KMS that is hosted by BestCloud CSP. Xerxes submits an administrative request to the system (comprising the administrative application and various key-management-related components) to export keys from the tenant-specific KMS at the CSP and have them sent to the enterprise KMS. He receives back confirmation that the request has been received and processed.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS performs the requested operation, then securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

#### Goal or Desired Outcome

Same as KCUC-2b

#### Notable Categorizations and Aspects

Same as KCUC-2b

#### Process Flow

1. Xerxes requests admin app to send policy export request to KMS at CSP
   1. Admin application component in enterprise sends admin request to KMC component.
   2. KMC component in enterprise securely sends admin request to KMS component in CSP.
2. Xerxes receives response from system
   1. KMS in CSP processes export request, creates response message and securely sends keys to KMS in enterprise.
   2. KMS in enterprise sends message to KMC component in enterprise.
   3. KMC component sends response to admin application component
   4. Admin application component (in system) displays response to Xerxes

#### Extensions

tbd

### Use case KCUC-2l: CSP KMS sending key-specific attribute changes to KMS at Enterprise (server-to-server)

This use case is a special case of KCUC-2b, resulting in cryptographic object attributes being sent from an KMS at a CSP to enterprise KMS. It is the inverse of use case KCUC-2f.

This use case entails implications both for what is included in the exported material and how that material is protected.

#### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS and also the associated tenant-specific KMS that is hosted by BestCloud CSP. Xerxes submits an administrative request to the system (comprising the administrative application and various key-management-related components) to export key attributes from the tenant-specific KMS at the CSP and have them sent to the enterprise KMS. He receives back confirmation that the request has been received and processed.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS performs the requested operation, then securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

#### Goal or Desired Outcome

Same as KCUC-2b

#### Notable Categorizations and Aspects

Same as KCUC-2b

#### Process Flow

1. Xerxes requests admin app to send key attribute export request to KMS at CSP
   1. Admin application component in enterprise sends admin request to KMC component.
   2. KMC component in enterprise securely sends admin request to KMS component in CSP.
2. Xerxes receives response from system
   1. KMS in CSP processes export request, creates response message and securely sends keys to KMS in enterprise.
   2. KMS in enterprise sends message to KMC component in enterprise.
   3. KMC component sends response to admin application component.
   4. Admin application component (in system) displays response to Xerxes.

#### Extensions

tbd

## Use Cases for Deployment Model 3: KMS as CSP Shared Service; KMC in Enterprise or CSP

There are sixteen cases defined for this deployment model:

* Admin use cases applicable to all variants of this deployment model
  + CSP admin manages resources for KMS at CSP (KMIP interaction related to segmentation of tenants in CSP KMS) **[KCUC-3a]**
  + Tenant admin requests tenant-specific operations in KMS at CSP (client-to-server) **[KCUC-3b]**
* For deployment models with KMS only at CSP
  + End user requests cryptographic operation resulting in tenant KMC at CSP accessing CSP KM (client-to-server) **[KCUC-3c]**
* For deployment models with CSP KMS as slave to enterprise KMS
  + Tenant admin moves enterprise keys to CSP KMS (server-to-server) **[KCUC-3d]**
  + Tenant admin requests enterprise KMS to export policy to KMS at CSP (server-to-server) **[KCUC-2e]**
  + Tenant admin requests enterprise KMS to send key-specific attribute changes to KMS at CSP (server-to-server) **[KCUC-2f]**
* For deployment model with enterprise KMS as slave to CSP KMS
  + Tenant admin creates tenant keys at CSP KMS and moves them to enterprise KMS (server-to-server) **[KCUC-3g]**
  + Tenant admin requests CSP KMS to export policy to tenant KMS (server-to-server) **[KCUC-2h]**
  + Tenant admin requests CSP KMS to send key-specific attribute changes to enterprise KMS (server-to-server) **[KCUC-2i]**
* For deployment models with KMS in alternative CSP
  + Tenant admin migrates keys from CSP KMS to alternative CSP KMS (server-to-server) **[KCUC-3j]**
  + Tenant admin requests CSP KMS to export policy to alternative CSP KMS (server-to-server) **[KCUC-3k]**
  + Tenant admin requests CSP KMS to send key-specific attribute changes to alternative CSP KMS (server-to-server) **[KCUC-3l]**
* For all deployment models, use cases related to management of CSP KMS
  + Registration of KMC **[KCUC-3m]**
  + Revocation of KMC registration **[KCUC-3n]**
  + KMC assurance of CSP KMS acting within policy **[KCUC-3o]**
  + KMS scaling in response to increase in KMC demand level **[KCUC-3p]**
  + KMC caching of keys or other material from KMS **[KCUC-3q]**

Note that use cases relevant to this deployment model, but that are described elsewhere in this document or that do not have significant implications for the KMIP protocol, are not included in this document.

### Use case KCUC-3a: CSP Administrator manages resources for tenant using shared KMS at CSP

Although the management of CSP resources for a tenant-specific KMS does not have implications for KMIP, the use case is important as it details the restrictions on interaction between the CSP admin and the key management capabilities that are instantiated in the CSP infrastructure. All administrative tasks related to the tenant KMS capabilities (defining key policy, authorizing key clients, delegating administrative responsibility, etc) are performed by the tenant KMS administrator, as defined in Use Case KCUC-3b below.

The CSP admin is responsible for administrative tasks such as allocating KMS resources to the tenant KMS, tracking and resolving tenant KMS performance issues, and setting up and maintaining security measures related to the shared KMS (such as firewall rules, anti-virus, intrusion detection, tenant isolation, log inspection etc). These responsibilities can give the CSP admin visibility into the keys stored in the tenant KMS, such through memory dumps in which cached keys may be visible, depending on what security controls are in place. The implications of CSP admin access are not defined in this use case, however, as they have no impact on KMIP.

In order to perform these administrative tasks, however, the CSP admin must have visibility into the physical and virtual networks over which KMIP messages travel. This use case, therefore, implies that KMIP messages should be protected against this potential for access by the CSP admin in any of a number of ways (for example, message encryption at the protocol level, channel encryption, and so on).

#### Description / User Story

Zander is a CSP administrator at BestCloud CSP who is responsible for managing physical and virtual resources in the CSP virtual environment (CPU, storage, network). BestCloud has been selected by the AllPC company to host both its AllPC application and to provide key management capabilities supporting that application (and potentially other applications) by means of a shared KMS in the BestCloud virtual infrastructure. Zander uses tools provided by the BestCloud physical and virtual environment to perform such activities as monitoring performance of the physical and virtual resources, inspecting memory dumps and other information to analyze system failures and monitoring security information to detect potential malware infection of the BestCloud environment.

Zander should not be able to interact with the tenant-specific administrative tools for the AllPC application and KMS. However, interactions such as those described in use case KCUC-1a above take place across the CSP infrastructure and within CPU resources allocated in the CSP infrastructure. One example of this, the routine monitoring of security-related events in network event logs, is shown in this use case.

Other user stories corresponding to this use case could entail other operations (monitor cpu usage, allocate or de-allocate resources, etc) and other objects or data (server event logs, firewall activity, etc.).

#### Goal or Desired Outcome

The goal of this use case is to have the CSP administrator perform authorized monitoring activities on network information related to the interactions between the tenant KMC and tenant KMC. This includes not only the ability of the CSP administrator to detect anomalous volume of network activity that might indicate a malware-infected tenant application.

#### Notable Categorizations and Aspects

Same as KCUC-2a, except using Deployment Model 3.

#### Process Flow

1. Zander reviews activity recorded in network monitor for the preceding interaction.
   1. Zander requests display of network information from network monitor component.
   2. Network monitor component prepares and displays interaction information to Zander.

#### Extensions

Tbd

### Use case KCUC-3b: Tenant admin manages tenant-specific capabilities in shared KMS at CSP (admin\_client-to-server)

In the deployment model for this use case, in which the enterprise uses a shared KMS in the CSP infrastructure, the tenant administrator is responsible for the same kinds of tasks as when the KMS is deployed within the enterprise. This includes such activities as defining key policy, authorizing KMCs, delegating administrative responsibility and so on. In this model, however, those activities are performed by means of a KMS management application residing either within the enterprise or within the tenant-dedicated portion of the CSP infrastructure.

These administrative activities are not currently instrumented through objects and operations in KMIP. However, such objects and operations are being considered for V1.2 and, if defined, the particular considerations for the security of such objects and operations in this deployment model would need to be addressed. These considerations include potential vulnerability of administrative operations to capture, corruption or subversion by adversaries.

Note that tenant administrative capabilities or activities not expected to be relevant to the KMIP V1.2 protocol, such as administrator login to the admin app, administrator authorization by the admin app or logging of activity by KMC and KMS are not included in this use case.

#### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise specific capabilities in a shared KMS that is hosted by BestCloud CSP. This entails Xerxes submitting administrative requests to the system (comprising the administrative application and various key-management-related components) and receiving back confirmation that the request has been received and processed. Such a request might be to have the KMS set a particular key rotation policy as the default for all keys created in the KMS.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

Other user stories corresponding to this use case could entail other operations (, modify key policy, etc), other objects (asymmetric keys, certificates, PGP keys, etc) and other attributes (operation policy, permissions, etc).

#### Goal or Desired Outcome

The goal of this use case is to have a valid administrative request, initiated by a tenant administrative user for an enterprise, be processed by the KMS component in the CSP. This includes not only the ability of the KMS to support the message exchange protocol, but also to perform the operations, understand the objects and interpret the attributes received from the KMC on behalf of the tenant administrative user.

#### Notable Categorizations and Aspects

Same as KCUC-2b, except using Deployment Model 3.

#### Process Flow

1. Xerxes requests admin app to set key rotation policy for KMS
   1. Admin application component sends admin request to KMC component.
   2. KMC component sends admin request to KMS component via secure channel.
2. Xerxes receives response from system
   1. KMS component processes request, creates response message and sends message to KMC component via established secure channel.
   2. KMC component sends response to admin application component.
   3. Admin application component (in system) displays response to Xerxes.

#### Extensions

tbd

### Use case KCUC-3c: CSP Administrator provisions resources for a new tenant and sets up credentials for tenant administrator

As described in Use Case KCUC-3a, the CSP admin is responsible for administrative tasks related to tenant resources. This use case describes additional tasks related to tenant administrators.

In order to perform these administrative tasks, however, the CSP admin may have visibility into the credentials used by tenant administrators. This use case, therefore, implies that administrative messages sent via should be protected against this potential for access by the CSP admin or attackers in any of a number of ways (for example, message encryption at the protocol level, channel encryption, and so on).

#### Description / User Story

Zander is a CSP administrator at BestCloud CSP who is responsible for establishing and managing tenant administrator credentials and privileges in the CSP environment. BestCloud has been selected by the AllPC company to host both its AllPC application and the key management capabilities for that application (and potentially other applications) within a portion of the BestCloud shared KMS dedicated to AllPC. Zander uses tools provided by the BestCloud physical and virtual environment to perform such activities as loading credentials for tenant administrators into known locations, defining access privileges related to those credentials and so on.

Zander should not be able to interact with the tenant-specific administrative tools for the AllPC application and KMS. However, interactions such as those described in use case KCUC-1a above take place across the CSP infrastructure and within CPU resources allocated in the CSP infrastructure.

Other user stories corresponding to this use case could entail other operations (modify tenant administrator, monitor tenant administrator, remove tenant administrator).

#### Goal or Desired Outcome

Same as KCUC-3a

#### Notable Categorizations and Aspects

Same as KCUC-3a

#### Process Flow

1. Zander receives request to enable a tenant administrator access to the shared CSP infrastructure, such as access to directory locations for storing KMC entity credentials for application access to tenant-dedicated KMS.
   1. Zander verifies request.
   2. Zander performs requested action.

#### Extensions

tbd

### Use case KCUC-3d: Tenant admin provisions clients and generates credentials for clients to access KMS at CSP (admin\_client-to-server)

As described in Use Case KCUC-3b, in the deployment model for this use case, the tenant administrator is responsible for the same kinds of tasks as when the KMS is deployed within the enterprise. This includes such activities as establishing credentials for KMCs that need to access a shared KMS located in the CSP. In this model, however, those activities are performed by means of a KMS management application residing either within the enterprise or within the tenant-dedicated portion of the CSP infrastructure.

These administrative activities are not currently instrumented through objects and operations in KMIP. However, such objects and operations are being considered for V1.2 and, if defined, the particular considerations for the security of such objects and operations in this deployment model would need to be addressed. These considerations include potential vulnerability of administrative operations to capture, corruption or subversion by adversaries.

Note that tenant administrative capabilities or activities not expected to be relevant to the KMIP V1.2 protocol, such as administrator login to the admin app, administrator authorization by the admin app or logging of activity by KMC and KMS are not included in this use case.

#### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the tenant-related capabilities in a shared KMS that is hosted by BestCloud CSP. This entails Xerxes submitting administrative requests to the system (comprising the administrative application and various key-management-related components) and receiving back confirmation that the request has been received and processed. Such a request might be to have the KMS to generate and accept a particular X.509 certificate associated with a KMC as the credential for the KMS to use when authenticating and authorizing requests from that KMC.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

Other user stories corresponding to this use case could entail other operations (remove client credential).

#### Goal or Desired Outcome

Same as KCUC-3b

#### Notable Categorizations and Aspects

Same as KCUC-3b

#### Process Flow

1. Xerxes requests admin app to set X.509 certificate as credential for KMS
   1. Admin application component sends admin request to KMC component.
   2. KMC component sends admin request to KMS component via secure channel.
2. Xerxes receives response from system
   1. KMS component processes request, creates response message and sends message to KMC component via established secure channel.
   2. KMC component sends response to admin application component
   3. Admin application component (in system) displays response to Xerxes

#### Extensions

tbd

### Use case KCUC-3e: End-user requests key (KMC in enterprise requests managed object from KMS in CSP)

This use case is similar to KCUC-1a. In this case, however, the request is sent from a KMC in the enterprise to a shared KMS in the CSP infrastructure.

The request for KMIP operations by an KMC residing in an enterprise to a KMS residing in a CSP infrastructure has important implications in terms of the trust establishment model (for example, TLS channel may be proxied by CSP), protection of keys (for example, a possible requirement for key wrapping) and the additional context that may need to be exchanged (for example, finer-grained application-related information used in authenticating and authorizing the request).

#### User Story and Component Interaction

The user story is the same as for KCUC-1a.

Within the system with which Alice interacts, the ALLPC order processing application component and the KMC reside within the enterprise. The ALLPC application requests the KMC component to request the key securely from the shared KMS component at the CSP. The KMS securely returns the key to the KMC.

Other user stories corresponding to this use case could entail other operations (register, revoke, modify attributes, etc), other objects (asymmetric keys, certificates, PGP keys, etc) and other attributes (operation policy, lease, state, etc).

#### Goal or Desired Outcome

The goal of this use case is to have a valid key-related request, initiated by an end user via an application within the enterprise be processed a KMS component in the CSP. This includes not only the ability of the KMS to support the message exchange protocol, but also to perform the operations, understand the objects and interpret the attributes it receives from the KMC.

#### Notable Categorizations and Aspects

Same as KCUC-2e, except using Deployment Model 3.

#### Process Flow

1. Alice requests new laptop from system
   1. Alice enters laptop request in end-user application.
   2. Application component in enterprise sends key request to KMC component.
   3. KMC component securely sends request to KMS component at CSP.
2. Alice receives response from system
   1. KMS component at CSP processes request, creates response message and securely sends message to KMC component in enterprise.
   2. KMC component sends response to application component.
   3. Application component displays response to Alice.

#### Extensions

tbd

### Use case KCUC-3f: End-user requests key (KMC in CSP requests managed object from KMS in CSP)

This use case is similar to KCUC-3e. In this case, however, the request is sent from a tenant KMC in a CSP to a tenant portion of a shared KMS also in a CSP.

The request for KMIP operations by an KMC residing in a CSP to a KMS also residing in that CSP has important implications in terms of the trust establishment model (for example, TLS channel may not be point-to-point), protection of keys (for example, a possible requirement for key wrapping) and the additional context that may need to be exchanged (for example, finer-grained application-related information used in authenticating and authorizing the request).

#### User Story and Component Interaction

User story is the same as for KCUC-3c.

Within the system with which Alice interacts, the ALLPC order processing application component and the KMC reside within the CSP. The ALLPC application requests the KMC component to request the key securely from the KMS component at the CSP. The KMS securely returns the key to the KMC.

Other user stories corresponding to this use case could entail other operations (register, revoke, modify attributes, etc), other objects (asymmetric keys, certificates, PGP keys, etc) and other attributes (operation policy, lease, state, etc).

#### Goal or Desired Outcome

Same as KCUC-3e

#### Notable Categorizations and Aspects

Same as KCUC-3e.

#### Process Flow

1. Alice requests new laptop from system
   1. Alice enters laptop request in end-user application at CSP.
   2. Application component sends key request to KMC component.
   3. KMC component securely sends request to KMS component also at CSP.
2. Alice receives response from system
   1. KMS component processes request, creates response message and securely sends message to KMC component.
   2. KMC component sends response to application component.
   3. Application component displays response to Alice.

#### Extensions

tbd

### Use case KCUC-3g: Tenant admin requests enterprise KMS to export tenant keys to KMS at CSP (server-to-server)

This use case is a special case of KCUC-3b, resulting in cryptographic objects being sent from an enterprise KMS to a KMS at a CSP. It is similar to KCUC-2g, except that the key request is initiated by the tenant admin goes to a shared KMS in the CSP instead of to a tenant-specific KMS at the CSP.

This use case entails implications both for what is included in the exported material and how that material is protected.

#### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS and also the associated tenant-specific KMS that is hosted by BestCloud CSP. Xerxes submits an administrative request to the system (comprising the administrative application and various key-management-related components) to export keys from the enterprise KMS to the tenant-specific KMS at the CSP. He receives back confirmation that the request has been received and processed.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS performs the key export operation, then securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

#### Goal or Desired Outcome

Same as KCUC-3b.

#### Notable Categorizations and Aspects

Same as KCUC-3b.

#### Process Flow

1. Xerxes requests admin app to export keys to KMS at CSP
   1. Admin application component in enterprise sends admin request to KMC component.

1.2 KMC component in enterprise securely sends admin request to KMS component in CSP.

1. Xerxes receives response from system
   1. KMS component prepares keys for export and sends them to KMS in enterprise.
   2. KMS in enterprise processes exported keys and returns response.
   3. KMS in CSP creates response message and securely sends message to KMC component in enterprise.
   4. KMC component sends response to admin application component.
   5. Admin application component (in system) displays response to Xerxes.

#### Extensions

[tbd]

### Use case KCUC-3h: Tenant Admin requests Enterprise KMS to export policy to KMS at CSP

This use case is a special case of KCUC-2b, resulting in cryptographic policy being sent from an enterprise KMS to a KMS at a CSP. It is similar to KCUC-2h, except that the key request is initiated by the tenant admin goes to a shared KMS in the CSP instead of to a tenant-specific KMS at the CSP.

This use case entails implications both for what is included in the policy and how that policy is protected from tampering or other attacks.

#### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS and also the associated tenant-specific KMS that is hosted by BestCloud CSP. Xerxes submits an administrative request to the system (comprising the administrative application and various key-management-related components) to send cryptographic policy from the enterprise KMS to the tenant-specific KMS at the CSP. He receives back confirmation that the request has been received and processed.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS performs the policy import operation, then securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

#### Goal or Desired Outcome

Same as KCUC-3b.

#### Notable Categorizations and Aspects

Same as KCUC-3b.

#### Process Flow

1. Xerxes requests admin app to send policy to KMS at CSP
   1. Admin application component in enterprise sends admin request to KMC component.
   2. KMC component in enterprise securely sends admin request to KMS component in CSP.
2. Xerxes receives response from system
   1. KMS in CSP processes policy, creates response message and securely sends message to KMC component in enterprise.
   2. KMC component sends response to admin application component.
   3. Admin application component (in system) displays response to Xerxes.

#### Extensions

tbd

### Use case KCUC-3i: Enterprise KMS sending key-specific attribute changes to KMS at CSP (server-to-server)

This use case is a special case of KCUC-3b, resulting in cryptographic object attributes being sent from an enterprise KMS to a KMS at a CSP. It is similar to KCUC-2i, except that the key request is initiated by the tenant admin goes to a shared KMS in the CSP instead of to a tenant-specific KMS at the CSP.

This use case entails implications both for what is included in the exported material and how that material is protected.

#### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS and also the associated tenant-specific KMS that is hosted by BestCloud CSP. Xerxes submits an administrative request to the system (comprising the administrative application and various key-management-related components) to send attributes from the enterprise KMS to the tenant-specific KMS at the CSP. He receives back confirmation that the request has been received and processed.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS performs the requested operation, then securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

#### Goal or Desired Outcome

Same as KCUC-3b.

#### Notable Categorizations and Aspects

Same as KCUC-3b

#### Process Flow

1. Xerxes requests admin app to send attributes to KMS at CSP
   1. Admin application component in enterprise sends admin request to KMC component.
   2. KMC component in enterprise securely sends admin request to KMS component in CSP.
2. Xerxes receives response from system
   1. KMS in CSP processes attributes, creates response message and securely sends message to KMC component in CSP.
   2. KMC component sends response to admin application component.
   3. Admin application component (in system) displays response to Xerxes.

#### Extensions

tbd

### Use case KCUC-3j: CSP KMS exporting keys to Enterprise KMS (server-to-server)

This use case is a special case of KCUC-3b, resulting in cryptographic object attributes being sent from an KMS at a CSP to enterprise KMS. It is the inverse of use case KCUC-3e. It is similar to KCUC-2j, except that the key request is initiated by the tenant admin directly to the CSP KMS.

This use case entails implications both for what is included in the exported material and how that material is protected.

#### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS and also the associated tenant-specific KMS that is hosted by BestCloud CSP. Xerxes submits an administrative request to the system (comprising the administrative application and various key-management-related components) to export keys from the tenant-specific KMS at the CSP and have them sent to the the enterprise KMS. He receives back confirmation that the request has been received and processed.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS performs the requested operation, then securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

#### Goal or Desired Outcome

Same as KCUC-3b

#### Notable Categorizations and Aspects

Same as KCUC-3b

#### Process Flow

1. Xerxes requests admin app to send key export request to KMS at CSP
   1. Admin application component in CSP sends admin request to KMC component.
   2. KMC component in CSP securely sends admin request to KMS component in CSP.
2. Xerxes receives response from system
   1. KMS in CSP processes export request, creates response message and securely sends keys to KMS in enterprise.
   2. KMS in enterprise sends message to KMS component in CSP.
   3. KMS in CSP sends message to KMC in CSP.
   4. KMC component sends response to admin application component
   5. Admin application component (in system) displays response to Xerxes

#### Extensions

### Use case KCUC-3k: Tenant Admin requests CSP KMS to export policy to KMS at Enterprise

This use case is a special case of KCUC-3b, resulting in cryptographic object attributes being sent from KMS at the CSP to enterprise KMS. It is the inverse of use case KCUC-3f. It is similar to KCUC-2l, except that the key request is initiated by the tenant admin directly to the CSP KMS.

This use case entails implications both for what is included in the exported material and how that material is protected.

#### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS and also the associated tenant-specific capabilities in a shared KMS that is hosted by BestCloud CSP. Xerxes submits an administrative request to the system (comprising the administrative application and various key-management-related components) to export keys from the tenant-specific KMS at the CSP and have them sent to the enterprise KMS. He receives back confirmation that the request has been received and processed.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS performs the requested operation, then securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

#### Goal or Desired Outcome

Same as KCUC-3b.

#### Notable Categorizations and Aspects

Same as KCUC-3b.

#### Process Flow

1. Xerxes requests admin app to send policy export request to KMS at CSP.
   1. Admin application component in CSP sends admin request to KMC component.
   2. KMC component in CSP securely sends admin request to KMS component in CSP.
2. Xerxes receives response from system
   1. KMS in CSP processes export request, creates response message and securely sends keys to KMS in enterprise.
   2. KMS in enterprise sends message to KMS component in CSP
   3. KMS in CSP sends message to KMC.
   4. KMC component sends response to admin application component.
   5. Admin application component (in system) displays response to Xerxes.

#### Extensions

tbd

### Use case KCUC-3l: CSP Admin requests CSP KMS to send key-specific attribute changes to KMS at Enterprise (server-to-server)

This use case is a special case of KCUC-3a, resulting in cryptographic object attributes being sent from an KMS at a CSP to enterprise KMS. It is the inverse of use case KCUC-3f. It is similar to KCUC-2l, except that the key request is initiated by the tenant admin directly to the CSP KMS.

This use case entails implications both for what is included in the exported material and how that material is protected.

#### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the tenant-specific KMS that is hosted by BestCloud CSP. Xerxes submits an administrative request to the system (comprising the administrative application and various key-management-related components) to export key attributes from the tenant-specific KMS at the CSP and have them sent to the enterprise KMS. He receives back confirmation that the request has been received and processed.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS performs the requested operation, then securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

#### Goal or Desired Outcome

Same as KCUC-3b.

#### Notable Categorizations and Aspects

Same as KCUC-3b.

#### Process Flow

1. Xerxes requests admin app to send key attribute export request to KMS at CSP
   1. Admin application component in CSP sends admin request to KMC component.
   2. KMC component in CSP securely sends admin request to KMS component in CSP.
2. Xerxes receives response from system
   1. KMS in CSP processes export request, creates response message and securely sends keys to KMS in enterprise.
   2. KMS in enterprise sends message to KMS component in CSP.
   3. KMS in enterprise sends message to KMC component in CSP.
   4. KMC component sends response to admin application component.
   5. Admin application component (in system) displays response to Xerxes.

#### Extensions

Tbd

### Use case KCUC-3m: Tenant Administrator requests registration for tenant KMC using shared KMS at CSP

This use case is a special case of the manual and automated use cases defined in the registration use cases later in this document.

#### Description / User Story

Zander is a CSP administrator at BestCloud CSP who is responsible for managing physical and virtual resources in the CSP virtual environment (CPU, storage, network). BestCloud has been selected by the AllPC company to host both its AllPC application and to provide key management capabilities supporting that application (and potentially other applications) by means of a shared KMS in the BestCloud virtual infrastructure.

Xerxes is the administrator for a KMC at either an enterprise or within the CSP. Xerxes requests Zander to register a new KMC, associated with an existing or new application, with the CSP KMS for which Zander is responsible.

Other user stories corresponding to this use case could include:

* Setting up automated system for registering KMC with CSP KMS

#### Goal or Desired Outcome

The goal of this use case is to have the CSP administrator register the tenant KMC for access to the tenant capabilities within the CSP KMS. The KMC and KMS are then able to communicate securely to perform the other use cases described in this section.

#### Notable Categorizations and Aspects

Same as KCUC-3a

#### Process Flow

1. Xerxes requests a new client registration from Zander
2. Xerxes is contacted by Zander
   1. Zander assesses the KMC registration request
   2. Zander chooses the KMIP registration option based on solution needs and company security policy
   3. Zander creates a registration packet for the KMC and sends it to Xerxes. This registration packet is specific to the KMC and is time sensitive.
3. Xerxes initiates the client’s KMIP registration feature, using the registration packet Zander provided.
   1. KMC stores KMS information regarding location, mutual authentication and so on.
   2. KMC sends its registration information to KMS, using information in the provided registration package.
   3. KMS completes registration process

#### Extensions

Tbd

### Use case KCUC-3n: Tenant Administrator requests revocation of registration for tenant KMC using shared KMS at CSP

This use case is a special case of the manual and automated use cases defined in the registration use cases later in this document.

#### Description / User Story

Zander is a CSP administrator at BestCloud CSP who is responsible for managing physical and virtual resources in the CSP virtual environment (CPU, storage, network). BestCloud has been selected by the AllPC company to host both its AllPC application and to provide key management capabilities supporting that application (and potentially other applications) by means of a shared KMS in the BestCloud virtual infrastructure.

Xerxes is the administrator for a KMC at either an enterprise or within the CSP. Xerxes requests Zander to de-register a KMC, associated with an existing or new application, from the CSP KMS for which Zander is responsible.

Other user stories corresponding to this use case could include:

* Setting up automated system for de-registering KMC with CSP KMS

#### Goal or Desired Outcome

The goal of this use case is to have the CSP administrator de-register the tenant KMC for access to the tenant capabilities within the CSP KMS. Attempts by the KMC to request keys or perform other operations should be rejected by the KMS.

#### Notable Categorizations and Aspects

Same as KCUC-3a

#### Process Flow

1. Xerxes requests a client de-registration from Zander
2. Xerxes is contacted by Zander
   1. Zander assesses the KMC de-registration request
   2. Zander chooses the KMIP de-registration option based on solution needs company security policy
   3. Zander creates a de-registration packet for the KMC and sends it to Xerxes. This registration packet is specific to the KMC and is time sensitive.
   4. KMS completes its de-registration client by removing or changing status of KMC information.
3. Xerxes initiates the client’s de-registration feature, using the packet Zander provided.
   1. KMC removes KMS information according to de-registration packet
   2. KMS completes registration process

#### Extensions

Tbd

### 

### Use case KCUC-3o: Tenant Administrator requests assurance that tenant KMC using shared KMS at CSP is acting within specified policies

This use case is a special case of the administrative use case KCUC-3b.

#### Description / User Story

Zander is a CSP administrator at BestCloud CSP who is responsible for managing physical and virtual resources in the CSP virtual environment (CPU, storage, network). BestCloud has been selected by the AllPC company to host both its AllPC application and to provide key management capabilities supporting that application (and potentially other applications) by means of a shared KMS in the BestCloud virtual infrastructure.

Xerxes is the administrator for a KMC at either an enterprise or within the CSP. Xerxes requests Zander to provide assurance that tenant partition of KMS is acting in accordance with defined policies.

#### Goal or Desired Outcome

The goal of this use case is to have the CSP administrator provide authoritative information to the tenant administrator to assure that the CSP KMS is acting in accordance with agreed key policies for the tenant.

#### Notable Categorizations and Aspects

Same as KCUC-3a

#### Process Flow

1. Xerxes requests KMS policy assurance from Zander
   1. Zander assesses the KMC assurance request
   2. Zander chooses the KMIP assurance option based on solution needs and company security policy
   3. Zander creates assurance packet for the KMC and sends it to Xerxes. This assurance packet is specific to the KMC and is time sensitive.

#### Extensions

Tbd

### Use case KCUC-3p: Shared KMS responds to increase in client resource needs (scaling)

This use case is a special case of the administrative use case described in KCUC-3b, related to CSP admin establishing appropriate resource allocation for tenant partitions of the CSP KMS.

#### Description / User Story

Zander is a CSP administrator at BestCloud CSP who is responsible for managing physical and virtual resources in the CSP virtual environment (CPU, storage, network). BestCloud has been selected by the AllPC company to host both its AllPC application and to provide key management capabilities supporting that application (and potentially other applications) by means of a shared KMS in the BestCloud virtual infrastructure.

Zander establishes mechanisms to have the resources automatically allocated to the CSP KMS as client requirements increase.

#### Goal or Desired Outcome

The goal of this use case is to have the CSP administrator ensure that sufficient resources will be available for processing tenant KMC requests.

#### Notable Categorizations and Aspects

Same as KCUC-3a

#### Process Flow

1. Zander establishes mechanism for automated allocation of resources to tenant portion of KMS if required.
2. KMS detects insufficient resources to provide actual or projected tenant KMC requests.
   1. KMS mechanism for resource allocation is invoked.

#### Extensions

Tbd

### Use case KCUC-3q: Tenant KMC manages cached keys from shared KMS at CSP

This use case pertains to automated actions between KMC and KMS. It assumes setup of the mechanisms has been performed through administrative used cases such as KCUC-2b.

#### Description / User Story

Zander is a CSP administrator at BestCloud CSP who is responsible for managing physical and virtual resources in the CSP virtual environment (CPU, storage, network). BestCloud has been selected by the AllPC company to host both its AllPC application and to provide key management capabilities supporting that application (and potentially other applications) by means of a shared KMS in the BestCloud virtual infrastructure. Xerxes is the administrator for a KMC at either an enterprise or within the CSP.

Xerxes and Zander have jointly established policies and mechanism to ensure appropriate caching of keys and other objects at the KMC. This use case describes the resultant management of cached object between the KMC and KMS.

#### Goal or Desired Outcome

The goal of this use case is to have the KMC and KMS coordinate the use of keys and other objects cached at the KMC in order to ensure currency and validity of the objects.

#### Notable Categorizations and Aspects

Same as KCUC-3a

#### Process Flow

1. Application sends request to KMC for a key or other object from KMC.
   1. KMC determines if object is cached. If necessary, based on parameters such as freshness specification, it communicates with KMS to get new instance of object and caches the new instance.
2. KMC returns requested object to application.
   1. KMC create response message and sends to application.

#### Extensions

Tbd

# Use Cases for Enterprise Server-to-Server Key Management

Use cases in this area include migration of keys, key policy and attributes from one enterprise KMS to another.

## Use case KSUC-1: Admin Requests Transfer of Keys from one KMS to Another

This use case entails implications both for what is included in the exported material and how that material is protected.

### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS. Xerxes receives notification that encrypted production data has been copied to the data analytics warehouse, served by a separate key management system, Xerxes submits an administrative request to the production key management system (comprising the administrative application and various key-management-related components) to export keys from the enterprise KMS to the data warehouse KMS within the enterprise. He receives back confirmation that the request has been received and processed.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS performs the key export operation, and then securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

Other user stories that apply to this use case include:

* Migration of all keys from one KMS to another, such as in consolidation of KMS’s within an enterprise
* Migration of keys from an existing KMS to a new KMS, such as in creation of division-level KMS
* Propagation of a single specified key, by administrative request, from one KMS to another

### Goal or Desired Outcome

The goal of this use case is to have a valid administrative request, initiated by a tenant administrative user, be processed by the KMS component. This includes not only the ability of the KMS to support the message exchange protocol, but also to perform the operations, understand the objects and interpret the attributes received from the KMC on behalf of the tenant administrative user.

### Notable Categorizations and Aspects

|  |  |
| --- | --- |
| Categories Covered:   * Trust establishment * KMC request * KMS export operations * KMS response | Applicable Deployment and Service Models:   * Intra-enterprise (between two enterprise KMS) * Inter-enterprise (between KMS in two enterprises) |
| Actors:   * KMS tenant administrator (Xerxes) | System:   * System includes the following components * Administrative Application (in enterprise) * KMC (in administrative app) * Primary KMS * Secondary KMS |
| Notable Services:   * Request/response for key management administration operation * Export operation | |
| Dependencies:   * Servers and clients have credentials that can be used to establish a mutually-authenticated communication between them. * Client has been configured to identify target server and to accept server credential * Server has been configured to accept client credential and request. * Client and server systems are active and able to initiate (client) and receive (server) messages. * Enterprise KMS has access to target KMS | |
| Assumptions:   * Client and server credentials contain unique identifiers that can be used to authenticate and authorize communication between them. | |

### Process Flow

1. Xerxes requests admin app to export keys and send them to alternative KMS
   1. Admin application component in enterprise sends admin request to KMC component.
   2. KMC component in enterprise securely sends admin request to KMS component
   3. KMS exports keys and securely sends them to second KMS.
2. Xerxes receives response from system
   1. Second KMS component processes exported keys and returns response.
   2. KMS in enterprise creates response message and securely sends message to KMC component.
   3. KMC component sends response to admin application component
   4. Admin application component (in system) displays response to Xerxes

### Extensions

[tbd]

## Use Case KSUC-2: Admin Requests Transfer of Key Policy from one KMS to Another

This use case entails implications both for what is included in the exported material and how that material is protected.

### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS. This includes managing key policy such as key rotation intervals at a master KMS, which propagates any changes in key policy to subordinate KMS.

Xerxes receives notification that key rotation interval for keys used in application encryption throughout all Key Management Servers in the enterprise should be changed from its current default setting of monthly to a new default setting of weekly, Xerxes submits an administrative request to the production key management system (comprising the administrative application and various key-management-related components) to set this policy in throughout the enterprise Key Management Servers. He receives back confirmation that the request has been received and processed.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS performs the key policy distribution operation, and then securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

Other user stories in which this server-to-server propagation of policy could occur include:

* Interaction between Key Management Servers in two different enterprises
* Propagation of policy to a new KMS within the enterprise.

### Goal or Desired Outcome

The goal of this use case is to have a valid administrative request, initiated by a tenant administrative user, be processed by multiple KMS components. This includes not only the ability of each KMS to support the message exchange protocol, but also to perform the operations, understand the objects and interpret the message received from the KMC on behalf of the tenant administrative user.

### Notable Categorizations and Aspects

|  |  |
| --- | --- |
| Categories Covered:   * Trust establishment * KMC request * KMS policy distribution operations * KMS response to KMC * Seccondary KMS response to primary KMS | Applicable Deployment and Service Models:   * Intra-enterprise (between two enterprise KMS) * Inter-enterprise (between KMS in two enterprises) |
| Actors:   * KMS tenant administrator (Xerxes) | System:   * System includes the following components   + Administrative Application (in enterprise)   + KMC (in administrative app)   + Priomary KMS   + Secondary KMS |
| Notable Services:   * Request/response for key management administrative operation * Key distribution operation | |
| Dependencies:   * Servers and clients have credentials that can be used to establish a mutually-authenticated communication between them. * Client has been configured to identify target server and to accept server credential * Server has been configured to accept client credential and request. * Client and server systems are active and able to initiate (client) and receive (server) messages. * Primary KMS has access to target KMS | |
| Assumptions:   * Client and server credentials contain unique identifiers that can be used to authenticate and authorize communication between them. | |

### Process Flow

1. Xerxes requests admin app to propagate key policy to all KMS
   1. Admin application component in enterprise sends admin request to KMC component
   2. KMC component in enterprise securely sends admin request to primary KMS component
   3. KMS securely sends key policy to second KMS.
2. Xerxes receives response from system
   1. Second KMS component processes key policy and returns response.
   2. Primay KMS creates response message and securely sends message to KMC component.
   3. KMC component sends response to admin application component
   4. Admin application component (in system) displays response to Xerxes

### Extensions

[tbd]

## Use Case KSUC-3: Admin Requests Transfer of Key Attributes from one KMS to Another

This use case entails implications both for what is included in the transferred material and how that material is protected.

### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS. This includes managing key attributes such as key ownership throughout the enterprise. He does this through communicating with a primary KMS any changes in key attributes to any other KMS in the enterprise.

Xerxes receives notification that responsibility for certain sets of encrypted information has been transferred from one data owner to another. Ownership of the keys related to this encrypted data throughout all Key Management Servers in the enterprise should be changed from its current owner to the new owner, Xerxes submits an administrative request to the key management system (comprising the administrative application and various key-management-related components) to make this change in ownership throughout the enterprise Key Management Servers. He receives back confirmation that the request has been received and processed.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS performs the distribution operation for the changed attribute, and then securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes.

Other user stories in which this server-to-server propagation of key attributes could occur include:

* Interaction between Key Management Servers in two different enterprises.

### Goal or Desired Outcome

The goal of this use case is to have a valid administrative request, initiated by a tenant administrative user, be processed by multiple KMS components. This includes not only the ability of each KMS to support the message exchange protocol, but also to perform the operations, understand the objects and interpret the message received from the KMC on behalf of the tenant administrative user.

### Notable Categorizations and Aspects

|  |  |
| --- | --- |
| Categories Covered:   * Trust establishment * KMC request * KMS policy distribution operations * KMS response to KMC * Secondary KMS response to primary KMS | Applicable Deployment and Service Models:   * Intra-enterprise (between two enterprise KMS) * Inter-enterprise (between KMS in two enterprises) |
| Actors:   * KMS tenant administrator (Xerxes) | System:   * System includes the following components   + Administrative Application (in enterprise)   + KMC (in administrative app)   + Primary KMS   + Secondary KMS |
| Notable Services:   * Request/response for key management administrative operation * Key attribute distribution operation | |
| Dependencies:   * Servers and clients have credentials that can be used to establish a mutually-authenticated communication between them. * Client has been configured to identify target server and to accept server credential * Server has been configured to accept client credential and request. * Client and server systems are active and able to initiate (client) and receive (server) messages. * Primary KMS has access to target KMS | |
| Assumptions:   * Client and server credentials contain unique identifiers that can be used to authenticate and authorize communication between them. | |

### Process Flow

1. Xerxes requests admin app to propagate key ownership change to all KMS
   1. Admin application component in enterprise sends admin request to KMC component
   2. KMC component in enterprise securely sends admin request to primary KMS component
   3. KMS securely sends key ownership to second KMS.
2. Xerxes receives response from system
   1. Second KMS component processes key ownership and returns response.
   2. Primary KMS creates response message and securely sends message to KMC component.
   3. KMC component sends response to admin application component
   4. Admin application component (in system) displays response to Xerxes

### Extensions

[tbd]

## Use Case KSUC-4: Admin Sets Up Automated Transfer of Individual Keys from one KMS to Another

This use case entails implications both for what is included in the transferred material and how that material is protected.

### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS. This includes managing key distribution throughout the enterprise. He does this through communicating with a primary KMS to set up mechanisms for automatically and appropriately distributing keys across all Key Management Servers in the enterprise as they are created. For example, any key created in a KMS must be sent to a primary KMS for backup or archival purposes.

Xerxes submits an administrative request to the key management system (comprising the administrative application and various key-management-related components) to establish the automated mechanisms and policies for distributing individual keys as they are created throughout the enterprise Key Management Servers. He receives back confirmation that the request has been received and processed.

Within the system with which Xerxes interacts, the administrative application component requests the KMC component to request the administrative operation securely from the KMS component. The KMS performs the setup operations, and then securely returns the response appropriate to that request to the KMC, which returns it to the application, which returns it to Xerxes. The KMS’s in the enterprise can then execute the established automation mechanism as each new key is created.

Other user stories in which this set up of single key distribution mechanism include:

* Set up for automated propagation of policy changes or key attributes.

### Goal or Desired Outcome

The goal of this use case is to have a valid administrative request, initiated by a tenant administrative user, be processed by multiple KMS components. This includes not only the ability of each KMS to support the message exchange protocol, but also to perform the operations, understand the objects and interpret the message received from the KMC on behalf of the tenant administrative user.

### Notable Categorizations and Aspects

|  |  |
| --- | --- |
| Categories Covered:   * Trust establishment * KMC request * KMS policy distribution operations * KMS response to KMC * Secondary KMS response to primary KMS | Applicable Deployment and Service Models:   * Intra-enterprise (between two enterprise KMS) * Inter-enterprise (between KMS in two enterprises) |
| Actors:   * KMS tenant administrator (Xerxes) | System:   * System includes the following components   + Administrative Application (in enterprise)   + KMC (in administrative app)   + Priomary KMS   + Secondary KMS |
| Notable Services:   * Request/response for key management administrative operation * Key distribution mechanism establishment operation | |
| Dependencies:   * Servers and clients have credentials that can be used to establish a mutually-authenticated communication between them. * Client has been configured to identify target server and to accept server credential * Server has been configured to accept client credential and request. * Client and server systems are active and able to initiate (client) and receive (server) messages. * Primary KMS has access to target KMS | |
| Assumptions:   * Client and server credentials contain unique identifiers that can be used to authenticate and authorize communication between them. | |

### Process Flow

1. Xerxes requests admin app to establish automated key distribution mechanism across all KMS
   1. Admin application component in enterprise sends admin request to KMC component
   2. KMC component in enterprise securely sends admin request to primary KMS component
   3. KMS securely sends key distribution establishment message to second KMS.
2. Xerxes receives response from system
   1. Second KMS component processes key distribution establishment and returns response.
   2. Primary KMS creates response message and securely sends message to KMC component.
   3. KMC component sends response to admin application component

### Extensions

[tbd]

# Use Cases for Hardware Security Module Key Management

[tbd]

# Use Cases for PGP Key Management

The following section presents the set of use cases relevant to working within a PGP ecosystem of products.

## Registration of a new Managed User

This use case covers the registration of an internal user, that is one managed by the enterprise and whose key should be actively managed and maintained by the key management system.

### Description / User Story

Alice works with his company-issued MacBook Pro laptop every day for email, web surfing and other activities. His company decides to begin securing their email with PGP keys. Alice’s company’s IT staff sends out directions to Alice about how to download and install software on to his laptop to meet this requirement.

Alice follows the IT staff’s instructions, downloading the software and installing it. Upon first running the software, he is prompted to enter his company credentials. The software, in conjunction with the KMIP server, verifies those credentials and then issues Alice a PGP key. That key is escrowed on the server and the private portion is held on the client as well. Alice almost never manages this key directly; he just realizes his email is now being secured at certain times by various notifications he gets as he uses his normal email program.

### Goal or Desired Outcome

A key must be created for Alice containing his proper credentials (email addresses, user names, photo, etc.).

The key must be signed by the correct organization key.

The key must be stored on both the server and the client, both public and private portions, such that they may be used by both server and client for encryption and signing.

### Environment

|  |  |
| --- | --- |
| Categories Covered:   * Desktop-based enrollment | Applicable Deployment and Service Models:   * On premises |
| Actors:   * *Alice* – a normal knowledge worker with little to no cryptographic experience * *Client Software* – software designed to integrate with laptop / desktop endpoints and provide security for email messages | Systems:   * *Client Software Management Server* – a server that administrators use to establish the security policy for the Client Software. (This server could be, and often is, the same as the KMIP server below.) * *KMIP Server* – a server managing the key material for an enterprise * *Directory server* – Active Directory or another LDAP-type server holding passwords, identities, and organizational attributes for an enterprise |
| Notable Services:   * None | |
| Dependencies:   * Alice should already be enrolled in whatever directory service exists for an enterprise. * The KMIP server must be properly configured to confer with that directory service. | |
| Assumptions:   * None | |

### Process Flow

1. Alice downloads and installs software on his laptop to use PGP keys with emails.
2. Alice runs the software for the first time.
3. The software contacts its management server for its security policy. The security policy indicates emails must be secured on the desktop.
4. The software prompts Alice to enter his credentials. Alice provides his Directory Server identity and password.
5. The software transmits Alice’s credentials to the KMIP server.
6. The KMIP server re-transmits and verifies Alice’s credentials with the Directory Server. The Directory Server responds positively that Alice is who he claims to be, and provides all email addresses, user names, and/or photos of Alice
7. The KMIP server creates a key for Alice based on his information from the Directory Server.
8. The KMIP server stores Alice public and private keys.
9. The KMIP server transmits the public / private keypair to the client software.
10. The client software stores the public / private keypair.

### Extensions

[tbd]

## Registration of a Managed User with Pre-Existing Key Material

This use case covers the registration of an internal user that already has an existing PGP key that should now be managed by the server.

### Description / User Story

Bob has been an avid PGP user for many years. He exchanges sensitive emails with his brokerage clients by using PGP software he bought off the shelf for herself. …

### Goal or Desired Outcome

### Environment

### Process Flow

## Registration of an Unmanaged User

### Description / User Story

### Goal or Desired Outcome

### Environment

### Process Flow

## Key Lookup by Email Address

### Description / User Story

### Goal or Desired Outcome

### Environment

### Process Flow

## Key Lookup via Key ID

### Description / User Story

### Goal or Desired Outcome

### Environment

### Process Flow

## Arbitrary Key Signing

### Description / User Story

### Goal or Desired Outcome

### Environment

### Process Flow

## Decipher a Data Encryption Key

### Description / User Story

Bob works with Alice on sensitive projects. During the course of their work, they share documents to which they occasionally allow other people access. These documents are encrypted to a key managed by a server. Bob and Alice are given access to the key by virtue of their credentials and permissions established in the key management system. Software on Bob’s system mediates access to the encrypted document such that the encryption is largely transparent to her.

### Goal or Desired Outcome

Bob can create documents to share with a group of colleagues that are encrypted only to that group. Bob doesn’t have to worry about the technical details behind that encryption.

Administrators of the key management system can set up shared group keys with proper permissions.

The key management system must unlock files for Bob upon her properly identifying herself.

### Environment

|  |  |
| --- | --- |
| Categories Covered:   * Desktop-based enrollment | Applicable Deployment and Service Models:   * On premises |
| Actors:   * Bob – a reasonably sophisticated encryption user * Alice – a normal knowledge worker with little to no cryptographic experience * Client Software – software designed to integrate with laptop / desktop endpoints and provide security for email messages | Systems:   * Client Software Management Server – a server that administrators use to establish the security policy for the Client Software. (This server could be, and often is, the same as the KMIP server below.) * KMIP Server – a server managing the key material for an enterprise * Directory server – Active Directory or another LDAP-type server holding passwords, identities, and organizational attributes for an enterprise |
| Notable Services:   * None | |
| Dependencies:   * The KMIP administrator must have configured a group key for Bob and Alice’s group. * Bob and Alice must have already enrolled (see section 1.1). | |
| Assumptions:   * None | |

### Process Flow

1. Bob’s software intercepts a call to open a file encrypted to a group key managed by the server. Metadata within the file header contains the ID of that key, as well as the fact that it’s a shared key.
2. The software contacts the KMIP server and identifies itself as acting on behalf of Bob via a stored credential or ticket.
3. The KMIP server validates Bob’s credential.
4. The Client Software transports an encrypted data key to the server, with the key ID of the group key, and requests the KMIP server to unlock the encrypted data key.
5. The KMIP server checks Bob’s right to the shared group key and confirms she is part of the requested group.
6. The KMIP server uses the private portion of the group key to decrypt the encrypted data key.
7. The KMIP server returns the bare data key to the client. Presumably this interaction happens over a secure TLS / SSL tunnel, so the private key is not bare on the network itself.
8. The Client Software uses the data key to decrypt the file on behalf of the opening application. It transparently uses the same key to encrypt any changes applied.

## Trust Validation

### Description / User Story

### Goal or Desired Outcome

### Environment

### Process Flow

# Use Cases for Registration of Key Management Clients

[tbd]

# Use Cases for Continuous Stream and Interval Delivery of Keys

The use cases presented in this section address two new methods of key delivery by KMIP servers to KMIP clients. The first set of use cases (see *Deliver Key Stream Use Cases*) provides KMIP clients with continuous streams of identical one-time pad key material. The second set of use cases (see *Deliver Keys Asynchronously Use Cases*) provides KMIP clients with new keys at configurable intervals.

In both sets of use cases, key material, or discrete keys are delivered to the KMIP client by the KMIP server, without the need for multiple request messages to maintain the flow of key material, or keys.

Although the use cases were originally developed to satisfy the needs of link encryption systems incorporating quantum key distribution (QKD), their application goes beyond QKD-based systems. For example, the streaming key material use cases can be applied to virtual zeroisation storage systems; the asynchronous delivery of keys use cases can be applied to any system where re-keying, key rollover, or certificate replacement, is a feature of the system.

Although detailed specification of the network link between KMIP Client and Server (message format, transport protocol, use of TLS, etc.) is out of scope for this document, it must be stated that there is no requirement for KMIP messages to be transported over a secure link. Physical security is sufficient for many deployments employing these use cases.

This document describes an Administrator actor, and administrative use cases. These may be out of scope for KMIP, but are presented for completeness.

The following actors are defined for the use cases presented in this document:

|  |  |
| --- | --- |
| Cryptographic Consumer | A consumer of key material that uses the key material to cryptographically protect (e.g. encrypt, sign) and/or process (e.g. decrypt, verify signature) information. |
| Cryptographic Protector | A Cryptographic Consumer that uses the key material to cryptographically protect information; e.g. the encryption function within a link encryption device. |
| Cryptographic Processor | A Cryptographic Consumer that uses the key material to cryptographically process information; e.g. the decryption function within a link encryption device. |
| Administrator | An entity that manages system installation, configuration, maintenance, etc. |

## Deliver Key Stream Use Cases



*Figure 7: Deliver Key Stream Use Case Model*

### Configure Key Stream Service

An Administrator configures KMIP Servers within the system to enable key stream delivery to KMIP Clients. Usually, two key streams consisting of identical key material, will be configured: one will be used for protecting information, the other for processing information.

#### Pre-Condition

The system must support the one-time pad cipher.

For systems where more than one KMIP Server is delivering the same key stream to KMIP Clients, the KMIP Servers must be able to securely replicate, or generate (the same) true random, one-time key material.

#### Guarantee

Identical, secure, synchronized, true random key streams will be available for delivery to the Cryptographic Protector and the Cryptographic Processor.

#### Trigger

Administrator initiates service configuration.

#### Main Success Scenario

1. Administrator creates linked key stream service entries in relevant KMIP Servers.
2. Administrator associates KMIP Clients with the key stream service entries.
3. Administrator enables the key stream service.

#### Extensions

TBD

### Request Key Stream

KMIP Client issues request to KMIP Server to start flow of key material from server to client.

#### Pre-Condition

*Configure Key Stream Service* has completed successfully for the Cryptographic Protector and the Cryptographic Processor.

#### Guarantee

The KMIP server, or servers, will be ready to *Deliver Key Stream*.

#### Trigger

Cryptographic Protector or Cryptographic Processor requests that the key stream delivery begins.

#### Main Success Scenario

1. Cryptographic Protector and Cryptographic Processor prepare to receive key streams.
2. Cryptographic Protector or Cryptographic Processor request that key stream delivery begins.

#### Extensions

TBD

### Deliver Key Stream

KMIP Servers connect to KMIP Clients that have requested key streams, and deliver key streams to the clients.

#### Pre-Condition

Cryptographic Protector or Cryptographic Processor successfully *Request Key Stream*.

#### Guarantee

Identical, secure, synchronized, true random key streams will be transmitted to the Cryptographic Protector and the Cryptographic Processor.

After transmission, the key streams will be destroyed on the KMIP Server(s).

#### Trigger

The System successfully connects to the Cryptographic Protector or Cryptographic Processor.

#### Main Success Scenario

1. System connects successfully to Cryptographic Protector or Cryptographic Processor.
2. System delivers key stream to Cryptographic Protector or Cryptographic Processor.
3. Concurrent with step 2, system destroys copies of delivered key stream.

#### Extensions

1a. System permanently fails to connect to Cryptographic Protector or Cryptographic Processor.

.1 Key stream service entry is disabled.

.2 If a linked Cryptographic Protector or Cryptographic Processor has successfully connected, the system drops the connection.

.3 System destroys all copies of delivered key material.

2a. Connection is permanently lost between system and Cryptographic Protector.

.1 System destroys all copies of delivered key material.

2b. Connection is permanently lost between system and Cryptographic Processor.

.1 Key stream service entry is disabled.

.2 If a linked Cryptographic Protector is successfully connected, the system drops the connection.

.3 System destroys all copies of delivered key material.

### Cancel Key Stream

An Administrator or KMIP Client requests that key stream delivery cease. KMIP Server stops delivery of key material.

#### Pre-Condition

*Deliver Key Stream* or *Request Key Stream* is active.

#### Guarantee

Key stream delivery will halt.

All delivered key material still held within the system will be destroyed.

#### Trigger

Administrator, Cryptographic Protector, or Cryptographic Processor requests that the key stream be cancelled.

#### Main Success Scenario

1. Administrator cancels key stream.
2. System stops delivering key stream to Cryptographic Protector and Cryptographic Processor.
3. System disconnects from Cryptographic Protector and Cryptographic Processor.
4. System destroys all copies of delivered key material.
5. System disables key stream service entries.

#### Extensions

1a. Cryptographic Protector requests that key stream delivery be stopped.

.1 System stops delivery to Cryptographic Protector.

.2 System disconnects from Cryptographic Protector.

.3 System destroys all copies of delivered key material.

1b. Cryptographic Processor permanently closes connection with system.

.1 System stops delivering key stream to Cryptographic Protector.

.2 System disconnects from Cryptographic Protector.

.3 Return to MSS step 4.

## Deliver Keys Asynchronously Use Cases



*Figure 8: Deliver Keys Asynchronously Use Case Model*

### Configure Asynchronous Delivery Service

An Administrator configures a KMIP Server within the system to enable periodic, asynchronous delivery of discrete keys (or other managed objects, such as certificates) to a KMIP Client.

#### Pre-Condition

#### Guarantee

#### Trigger

Administrator initiates service configuration.

#### Main Success Scenario

1. Administrator creates asynchronous delivery service entry in KMIP Server.
2. Administrator associates KMIP Clients with the asynchronous service delivery.
3. Administrator enables the asynchronous delivery service.

#### Extensions

TBD

### Request Asynchronous Delivery

KMIP Client issues request to KMIP Server to start asynchronous delivery of keys from server to client.

#### Pre-Condition

*Configure Asynchronous Delivery Service* has completed successfully for the Cryptographic Consumer.

#### Guarantee

The KMIP server will be ready to *Deliver Keys Asynchronously*.

#### Trigger

Cryptographic Consumer requests that asynchronous delivery of keys begins.

#### Main Success Scenario

1. Cryptographic Consumer prepares to receive keys asynchronously.
2. Cryptographic Consumer requests that asynchronous delivery of keys begins.

#### Extensions

TBD

### Deliver Keys Asynchronously

KMIP Server connects to KMIP Client. KMIP Server periodically delivers keys to KMIP Client. Connection may, or may not remain open between key deliveries.

#### Pre-Condition

Cryptographic Consumer successfully completes *Request Asynchronous Delivery*.

#### Guarantee

Keys will be delivered asynchronously and periodically to the Cryptographic Consumer.

#### Trigger

Time for first scheduled delivery of a key arrives.

#### Main Success Scenario

1. System connects successfully to Cryptographic Consumer.
2. System delivers key to Cryptographic Consumer.
3. Connection remains open until next scheduled key delivery.
4. Loop back to step 2.

#### Extensions

3a. Connection is closed.

.1 System waits until it is time for the next scheduled key delivery.

.2 System connects successfully to Cryptographic Consumer.

.3 System delivers key to Cryptographic Consumer.

.4 Loop back to 3a.

### Cancel Asynchronous Delivery

An Administrator or KMIP Client requests that asynchronous delivery of keys cease. KMIP Server stops delivery of keys.

#### Pre-Condition

*Deliver Keys Asynchronously* or *Request Asynchronous Delivery* is active.

#### Guarantee

Asynchronous delivery will halt.

#### Trigger

Administrator or Cryptographic Consumer requests that asynchronous delivery be cancelled.

#### Main Success Scenario

1. Administrator cancels asynchronous delivery.
2. System stops asynchronous delivery to Cryptographic Consumer.
3. If connected, System disconnects from Cryptographic Consumer.
4. System disables asynchronous delivery service entry.

#### Extensions

1a. Cryptographic Consumer requests that asynchronously delivery be stopped.

.1 Got to MSS step 2.

# Use Cases for Key Management for Storage Environments

Use cases in this area include:

* Device requests a new key from the key manager; key is returned along with label or identifier and UUID
* Device generates a key and registers the key with the key manager
* Device adds attributes associated as meta-data with the key
* Device requests an encryption or authentication key from key manager
* Admin requests deletion of key

The first four of these use cases assume that an admin has set up the mechanism by which these interactions initiated by a storage device automatically occur. An example of that admin use case is described in the server-to-server use cases, specifically KSUC-4.

## Use case KSTUC-1: Storage Device Requests Key from KMS

This use case assumes that the admin user has set up the mechanism by which the interaction within the components described in this use case take place. The use case entails implications both for what is included in the transfer of key material and how that material is protected.

### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS. Xerxes has set up mechanisms by which a storage device requests new keys as it needs them from a KMS.

Within the system with which Xerxes interacts, the storage device component requests the KMC component (potentially incorporated into the storage device) to request the key creation operation securely from the KMS component. The KMS performs the key creation operation, and then securely returns the key to the KMC, which returns it to the storage device.

### Goal or Desired Outcome

The goal of this use case is to have a valid key creation request, initiated by a storage device, be processed by the KMS component.

### Notable Categorizations and Aspects

|  |  |
| --- | --- |
| Categories Covered:   * Trust establishment * KMC request * KMS create key operation * KMS response | Applicable Deployment and Service Models:   * Intra-enterprise (between KMC and KMS) * Inter-enterprise (between KMC and KMS) |
| Actors:   * KMS tenant administrator (Xerxes) | System:   * Storage device * KMC * KMS |
| Notable Services:   * Request/response for key creation operation | |
| Dependencies:   * Servers and clients have credentials that can be used to establish a mutually-authenticated communication between them. * Client has been configured to identify target server and to accept server credential * Server has been configured to accept client credential and request. * Client and server systems are active and able to initiate (client) and receive (server) messages. | |
| Assumptions:   * Client and server credentials contain unique identifiers that can be used to authenticate and authorize communication between them. | |

### Process Flow

1. Storage device requests key creation
   1. Storage device sends key creation request to KMC component.
   2. KMC component in enterprise securely sends a request to KMS component.
2. Storage device receives response from system
   1. KMS creates key and response message and securely sends message to KMC component.
   2. KMC component sends response to storage device component

### Extensions

[tbd]

## Use case KSTUC-2: Storage Device Registers Key with KMS

This use case assumes that the admin user has set up the mechanism by which the interaction within the components described in this use case take place. The use case entails implications both for what is included in the transfer of key material and how that material is protected.

### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS. Xerxes has set up mechanisms by which a storage device registers new keys with a KMS, as the storage device creates them.

Within the system with which Xerxes interacts, the storage device component requests the KMC component (potentially incorporated into the storage device) to request the key register operation securely from the KMS component. The KMS performs the key register operation, and then securely returns the response to the KMC, which returns it to the storage device.

### Goal or Desired Outcome

The goal of this use case is to have a valid key registration request, initiated by a storage device, be processed by the KMS component.

### Notable Categorizations and Aspects

|  |  |
| --- | --- |
| Categories Covered:   * Trust establishment * KMC request * KMS register operation * KMS response | Applicable Deployment and Service Models:   * Intra-enterprise (between KMC and KMS) * Inter-enterprise (between KMC and KMS) |
| Actors:   * KMS tenant administrator (Xerxes) | System:   * Storage device * KMC * KMS |
| Notable Services:   * Request/response for key register operation | |
| Dependencies:   * Servers and clients have credentials that can be used to establish a mutually-authenticated communication between them. * Client has been configured to identify target server and to accept server credential * Server has been configured to accept client credential and request. * Client and server systems are active and able to initiate (client) and receive (server) messages. | |
| Assumptions:   * Client and server credentials contain unique identifiers that can be used to authenticate and authorize communication between them. | |

### Process Flow

1. Storage device requests key register operation
   1. Storage device sends key register request, including key, to KMC component.
   2. KMC component in enterprise securely sends a request to KMS component.
2. Storage device receives response from system
   1. KMS registers key, creates response message and securely sends message to KMC component.
   2. KMC component sends response to storage device component

### Extensions

[tbd]

## Use case KSTUC-3: Storage Device Sends Key Attributes KMS

This use case assumes that the admin user has set up the mechanism by which the interaction within the components described in this use case take place. The use case entails implications both for what is included in the transfer of key material and how that material is protected.

### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS. Xerxes has set up mechanisms by which a storage device sends new or changed attributes for a specific key to the KMS that has that key.

Within the system with which Xerxes interacts, the storage device component requests the KMC component (potentially incorporated into the storage device) to request the key attribute operation securely from the KMS component. The KMS performs the key attribute operation, and then securely returns the response to the KMC, which returns it to the storage device.

### Goal or Desired Outcome

The goal of this use case is to have a valid key attribute request, initiated by a storage device, be processed by the KMS component.

### Notable Categorizations and Aspects

|  |  |
| --- | --- |
| Categories Covered:   * Trust establishment * KMC request * KMS key attribute creation/modification/deletion operation * KMS response | Applicable Deployment and Service Models:   * Intra-enterprise (between KMC and KMS) * Inter-enterprise (between KMC and KMS) |
| Actors:   * KMS tenant administrator (Xerxes) | System:   * Storage device * KMC * KMS |
| Notable Services:   * Request/response for key attribute operation | |
| Dependencies:   * Servers and clients have credentials that can be used to establish a mutually-authenticated communication between them. * Client has been configured to identify target server and to accept server credential * Server has been configured to accept client credential and request. * Client and server systems are active and able to initiate (client) and receive (server) messages. | |
| Assumptions:   * Client and server credentials contain unique identifiers that can be used to authenticate and authorize communication between them. | |

### Process Flow

1. Storage device requests key attribute operation
   1. Storage device sends key attribute request, including key identifier, to KMC component.
   2. KMC component in enterprise securely sends request to KMS component.
2. Storage device receives response from system
   1. KMS performs key attribute operation, creates response message and securely sends message to KMC component.
   2. KMC component sends response to storage device component

### Extensions

[tbd]

## Use case KSTUC-4: Storage Device Requests Encryption or Authentication Key from KMS

This use case assumes that the admin user has set up the mechanism by which the interaction within the components described in this use case take place. The use case entails implications both for what is included in the transfer of key material and how that material is protected.

### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS. Xerxes has set up mechanisms by which a storage device requests encryption, authentication or other keys from a KMS, as the storage device creates them.

Within the system with which Xerxes interacts, the storage device component requests the KMC component (potentially incorporated into the storage device) to perform the key request operation securely with the KMS component. The KMS performs the key request operation, and then securely returns the key and response to the KMC, which returns it to the storage device.

Alternative user stories for this use case include:

* Storage device requests key deletion (rather than requested by admin, as in use case KSTUC\_5)

### Goal or Desired Outcome

The goal of this use case is to have a valid key retrieval request, initiated by a storage device, be processed by the KMS component.

### Notable Categorizations and Aspects

|  |  |
| --- | --- |
| Categories Covered:   * Trust establishment * KMC request * KMS retrieval operation * KMS response | Applicable Deployment and Service Models:   * Intra-enterprise (between KMC and KMS) * Inter-enterprise (between KMC and KMS) |
| Actors:   * KMS tenant administrator (Xerxes) | System:   * Storage device * KMC * KMS |
| Notable Services:   * Request/response for key retrieval operation | |
| Dependencies:   * Servers and clients have credentials that can be used to establish a mutually-authenticated communication between them. * Client has been configured to identify target server and to accept server credential * Server has been configured to accept client credential and request. * Client and server systems are active and able to initiate (client) and receive (server) messages. | |
| Assumptions:   * Client and server credentials contain unique identifiers that can be used to authenticate and authorize communication between them. | |

### Process Flow

1. Storage device requests key retrieval operation
   1. Storage device sends key retrieval request, including key identifier, to KMC component.
   2. KMC component in enterprise securely sends request to KMS component.
2. Storage device receives response from system
   1. KMS retrieves key, creates response message and securely sends message to KMC component.
   2. KMC component sends response to storage device component

### Extensions

[tbd]

## Use case KSTUC-5: Admin Deletes Key from KMS

The use case entails implications both for what is included in the transfer of key material and how that material is protected.

### Description / User Story

Xerxes is a key management administrator for the AllPC enterprise who is responsible for administrator tasks related to managing the enterprise KMS.

The admin application with which Xerxes interacts requests the KMC component (potentially to request the key deletion operation securely from the KMS component. The KMS performs the key deletion operation, and then securely returns the response to the KMC, which returns it to the admin application.

### Goal or Desired Outcome

The goal of this use case is to have a valid key deletion request, initiated by an administrator, be processed by the KMS component.

### Notable Categorizations and Aspects

|  |  |
| --- | --- |
| Categories Covered:   * Trust establishment * KMC request * KMS deletion operation * KMS response | Applicable Deployment and Service Models:   * Intra-enterprise (between KMC and KMS) * Inter-enterprise (between KMC and KMS) |
| Actors:   * KMS tenant administrator (Xerxes) | System:   * Admin application * KMC * KMS |
| Notable Services:   * Request/response for key deletion operation | |
| Dependencies:   * Servers and clients have credentials that can be used to establish a mutually-authenticated communication between them. * Client has been configured to identify target server and to accept server credential * Server has been configured to accept client credential and request. * Client and server systems are active and able to initiate (client) and receive (server) messages. | |
| Assumptions:   * Client and server credentials contain unique identifiers that can be used to authenticate and authorize communication between them. | |

### Process Flow

1. Xerxes requests key deletion operation
   1. Xerxes requests Admin application to send key deletion request, including key identifier, to KMC component.
   2. KMC component in enterprise securely sends request to KMS component.
2. Xerxes receives response from system
   1. KMS deletes key, creates response message and securely sends message to KMC component.
   2. KMC component sends response to admin application component
   3. Admin application displays response to adminsitrator

### Extensions

[tbd]

# Implementation Conformance

This document is intended to be informational only and as such has no conformance clauses. The conformance requirements for the KMIP Specification can be found in the [KMIP\_SP] document itself, at the URL noted in the “Normative References” section of this document.

1. Acknowledgements

The following individuals have participated in the creation of this specification and are gratefully acknowledged:

Original Authors of the initial contribution:

[tbd]

1. Acronyms

The following abbreviations and acronyms are used in this document:

[tbd]

1. Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Date** | **Editor** | **Changes Made** |
| wd-01 | 2012-6-20 | Denis Pochuev | Initial version of document (glossary, etc) |
| wd-02 | 2012-6-28 | Robert Griffin | Re-format and addition of draft use cases |
| wd-03 | 2012-7-05 | Denis Pochuev | Major revision of Admin use-cases |
| wd-06 | 2012-9-06 | Mike Allen | Incorporated contributions from Bob Griffin, formatting changes from Kirin. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |