Service and Content Protection for Mobile Broadcast Services

Candidate Version 1.0 – 29 May 2007

Open Mobile Alliance

OMA-TS-BCAST_SvcCntProtection-V1_0-20070529-C

Contents

Figures

Tables

1. Scope

This document specifies the service protection and content protection systems, and affiliated mechanisms, which support various business models of OMA BCAST enabled mobile broadcast services delivery. On behalf of broadcast service providers and content providers, means are provided to protect the access to, and control the consumption of, broadcast content in either streaming or file delivery format. Two main systems can be used to provide service protection and/or content protection: the DRM Profile and the Smartcard Profile.

Fundamental components of the service and content protection systems consist of various content encryption mechanisms, protection signaling, and key management related messages which may carry rights objects, other post-reception consumption attributes (such as recording permission), key material, and parental rating criteria. In addition to server-client (i.e., network-to-terminal) interactions, this document also normatively specifies the server-side interfaces pertaining to service and content protection.

2. References

2.1 Normative References

The version and release numbers specified for the 3GPP and 3GPP2 references in this section are the minimum version and release numbers that can be used. The references are not meant to be restricted to these versions and releases; subsequent versions and releases can also be used because they are required to be backward compatible. For example, the minimum version of 3GPP TS 33.222 is the release 6 but the use of the relesae 7 is acceptable as well.

[3GPP TS 23.003 v6] "Numbering, Addressing and Identification (Release 6)", 3rd Generation Partnership Project, 3 GPP TS 23.003,
URL: http://www.3gpp.org/

[3GPP TS 23.032 v6] "Universal Geographical Area Description (GAD) (Release 6)", 3rd Generation Partnership Project, 3 GPP TS 23.032,
URL: http://www.3gpp.org/

[3GPP TS 26.346 v7] "Multimedia Broadcast/Multicast Service (MBMS), Protocols and codecs (Release 7)", Technical Specification Group Services and System Aspects, 3rd Generation Partnership Project, 3GPP TS 26.346,
URL: http://www.3gpp.org/

[3GPP TS 31.101 v6] "UICC-terminal interface; Physical and logical characteristics (Release 6)", 3rd Generation Partnership Project, Technical Specification 3GPP TS 31.101,
URL: http://www.3gpp.org/

[3GPP TS 31.102 v6] "Characteristics of the Universal Subscriber Identity Module (USIM) application (Release 6)", 3rd Generation Partnership Project, Technical Specification 3GPP TS 31.102,
URL: http://www.3gpp.org/

[3GPP TS 31.103 v6] "Characteristics of the IP Multimedia Services Identity Module (ISIM) application (Release 6)", 3rd Generation Partnership Project, Technical Specification 3GPP TS 31.103,
URL: http://www.3gpp.org/

[3GPP TS 31.111 v6] "Universal Subscriber Identity Module (USIM) Application Toolkit (USAT) (Release 6)", 3rd Generation Partnership Project, Technical Specification 3GPP TS 31.111,
URL: http://www.3gpp.org/

[3GPP TS 33.220 v6] "Generic Authentication Architecture, Generic Bootstrapping Architecture (Release 6)", 3rd Generation Partnership Project, Technical Specification 3GPP TS 33.220,
URL: http://www.3gpp.org/

[3GPP TS 33.222 v6] "Generic Authentication Architecture, Access to network application functions using Hypertext Transfer Protocol over Transport Layer Security (HTTPS) (Release 6)", 3rd Generation Partnership Project, Technical Specification 3GPP TS 33.222,
URL: http://www.3gpp.org/

[3GPP TS 33.246 v7] "3G Security; Security of Multimedia Broadcast/Multicast Service (Release 7)", Technical Specification Group Services and System Aspects, 3rd Generation Partnership Project, 3GPP 33.246,
URL: http://www.3gpp.org/

[3GPP TS 51.011 v4] "Specification of the Subscriber Identity Module - Mobile Equipment (SIM-ME) interface (Release 4)", rd Generation Partnership Project, Technical Specification 3GPP TS 51.011,
URL: http://www.3gpp.org/

[3GPP2 C.S0002-0] "Physical Layer Standard for cdma2000 Spread Spectrum Systems", 3rd Generation Partnership Project 2, Technical Specification 3GPP2 C.S0002-0,
URL: http://www.3gpp2.org/

[3GPP2 C.S0005-D] "Upper Layer (Layer 3) Signaling Standard for cdma2000 Spread Spectrum Systems", 3rd Generation Partnership Project 2, Technical Specification 3GPP2 C.S0005-D,

URL: http://www.3gpp2.org/

[3GPP2 C.S0023-C] "Removable User Identity Module for Spread Spectrum Systems", 3rd Generation Partnership Project 2, Technical Specification 3GPP2 C.S0023-C,
URL: http://www.3gpp2.org/

[3GPP2 C.S0024-A] "cdma2000 High Rate Packet Data Air Interface Specification", 3rd Generation Partnership Project 2, Technical Specification 3GPP2 C.S0024-A,
URL: http://www.3gpp2.org/

[3GPP2 C.S0035-A] "CDMA Card Application Toolkit (CCAT)", 3rd Generation Partnership Project 2, Technical Specification 3GPP2 C.S0035-A,
URL: http://www.3gpp2.org/

[3GPP2 C.S0054-0] "cdma000 High Rate Broadcast-Multicast Packet Data Air Interface Specification", 3rd Generation Partnership Project 2, Technical Specification 3GPP2 C.S0054-0,
URL: http://www.3gpp2.org/

[3GPP2 C.S0065-0] "cdma2000 Application on UICC for Spread Spectrum Systems", 3rd Generation Partnership Project 2, Technical Specification 3GPP2 C.S0065-0,
URL: http://www.3gpp2.org/

[3GPP2 C.S0069-0] "ISIM Application on UICC for cdma2000 Spread Spectrum Systems", 3rd Generation Partnership Project 2, Technical Specification 3GPP2 C.S0069-0,
URL: http://www.3gpp2.org/

[3GPP2 C.S0072-0] "Mobile Station Equipment Identifier (MEID) Support for cdma2000 Spread Spectrum Systems", 3rd Generation Partnership Project 2, Technical Specification 3GPP2 C.S0072-0,
URL: http://www.3gpp2.org/

[3GPP2 S.S0083-A] "Broadcast-Multicast Service Security Framework", 3rd Generation Partnership Project 2, Technical Specification 3GPP2 S.S0083-A,
URL: http://www.3gpp2.org/

[3GPP2 X.S0022-A] "Broadcast and Multicast Service in cdma2000 Wireless IP Network", 3rd Generation Partnership Project 2, Technical Specification 3GPP2 X.S0022-A,
URL: http://www.3gpp2.org/

[BCAST10-Architecture] "Mobile Broadcast Services Architecture", Open Mobile Alliance™, OMA-AD- BCAST-V1_0,
URL: http://www.openmobilealliance.org/

[BCAST10-BCMCS-Adaptation] "Broadcast Distribution System Adaptation – 3GPP2/BCMCS", Open Mobile Alliance™, OMA-TS-BCAST_BCMCS_Adaptation-V1_0,
URL: http://www.openmobilealliance.org/

[BCAST10-Distribution] "File and Stream Distribution for Mobile Broadcast Services ", Open Mobile Alliance™, OMA-TS-BCAST_Distribution-V1_0,
URL: http://www.openmobilealliance.org/

[BCAST10-DVBH-IPDC-Adaptation] "Broadcast Distribution System Adaptation – IPDC over DVB-H", Open Mobile Alliance™, OMA-TS-BCAST_DVB_Adaptation-V1_0,
URL: http://www.openmobilealliance.org/

[BCAST10-MBMS-Adaptation] "Broadcast Distribution System Adaptation – 3GPP/MBMS", Open Mobile Alliance™, OMA-TS-BCAST_MBMS_Adaptation-V1_0,
URL: http://www.openmobilealliance.org/

[BCAST10-Services] "Mobile Broadcast Services", Open Mobile Alliance™, OMA-TS-BCAST_Services-V1_0,
URL: http://www.openmobilealliance.org/

[BCAST10-SG] "Service Guide for Mobile Broadcast Services", Open Mobile Alliance™, OMA-TS-BCAST_ServiceGuide-V1_0,
URL: http://www.openmobilealliance.org/

[BCAST10-XMLSchema-SPCP-Backend]	"Mobile Broadcast Services – XML Schema SP/CP Backend Messages", Open Mobile Alliance™,OMA-SUP-XSD_bcast_spcp_backend-V1_0, URL: http://www.openmobilealliance.org/
[draft-mikey-oma]	IETF Internet draft draft-dondeti-msec-mikey-genext-oma, work in progress
[DRM Enabler-v2.0]	OMA-DRM-V2_0 enabler, Open Mobile Alliance™, URL: http://www.openmobilealliance.org/
[DRMCF-v2.0]	"DRM Content Format V2.0", Open Mobile Alliance™, OMA-DRM-DCF-V2_0, URL: http://www.openmobilealliance.org/
[DRMDRM-v2.0]	"DRM Specification V2.0", Open Mobile Alliance™, OMA-DRM-DRM-V2_0, URL: http://www.openmobilealliance.org/
[ETSI EN 300 468 V1.6.1]	*Digital Video Broadcasting (DVB); Specification for Service Infor-mation (SI) in DVB systems*, November 2004, URL: http://www.etsi.org/
[ETSI TS 102.221]	"Smart Cards; UICC-Terminal interface; Physical and Logical Characteristics", URL: http://www.etsi.org/
[ETSI TS 102.484]	"Secure Channel between a UICC and an End Point Terminal", ETSI SmartCard Platform, URL: http://www.etsi.org/
[FIPS197]	*ADVANCED ENCRYPTION STANDARD (AES)*, Federal Information Processing Standards Publication 197, URL: http://csrc.nist.gov/publications/fips/
[FIPS198]	*The Keyed-Hash Message Authentication Code (HMAC)*, Federal Information Processing Standards Publication 198, URL: http://csrc.nist.gov/publications/fips/
[IOPPROC]	"OMA Interoperability Policy and Process", Version 1.1, Open Mobile Alliance™, OMA-IOP-Process-V1_1, URL: http://www.openmobilealliance.org/
[ISMACRYP11]	"ISMA 1.0 Encryption and Authentication, Version 1.1", release version, URL: http://www.isma.tv
[ISO-3166]	"Codes for the representation of names of countries and their subdivisions", URL: http://www.iso.org/iso/en/prods-services/iso3166ma/index.html
[OMA MLP]	"Mobile Location Protocol 3.2", Open Mobile Alliance™, OMA-TS-MLP-V3_2-20051124-C, URL: http://www.openmobilealliance.org/
[OMA Push]	"OMA Push V2.1", Open Mobile Alliance™, OMA-ERP-Push-V2_1-20051122-C, URL: http://www.openmobilealliance.org/
[RFC2045]	"Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", N. Freed, N. Borenstein, November 1996, URL: http://www.ietf.org/rfc/rfc2045.txt
[RFC2104]	"HMAC: Keyed-Hashing for Message Authentication", H. Krawczyk, M. Bellare, R. Canetti, February 1997, URL: http://www.ietf.org/rfc/rfc2104.txt
[RFC2119]	"Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997, URL: http://www.ietf.org/rfc/rfc2119.txt
[RFC2246]	"The TLS Protocol, Version 1.0", T. Dierks, C.Allen, January 1999, URL: http://www.ietf.org/rfc/rfc2246.txt
[RFC2327]	"SDP: Session Description Protocol", M. Handley, V. Jacobson, April 1998, URL: http://www.ietf.org/rfc/rfc2327.txt

[RFC2401] "Security Architecture for the Internet Protocol", S. Kent, R. Atkinson, November 1998, URL: http://www.ietf.org/rfc/rfc2401.txt

[RFC2404] "The Use of HMAC-SHA-1-96 within ESP and AH", C. Madson, R. Glenn, November 1998, URL: http://www.ietf.org/rfc/rfc2404.txt

[RFC2406] "IP Encapsulating Security Payload (ESP)", S. Kent, R. Atkinson, November 1998, URL: http://www.ietf.org/rfc/rfc2406.txt

[RFC2451] "The ESP CBC-Mode Cipher Algorithms", R. Pereira, R. Adams, November 1998, URL: http://www.ietf.org/rfc/rfc2451.txt

[RFC2617] "HTTP Authentication: Basic and Digest Access Authentication", J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, L. Stewart, June 1999, URL: http://www.ietf.org/rfc/rfc2617.txt

[RFC3237] "Requirements for Reliable Server Pooling", M. Tuexen, Q. Xie, R. Stewart, M. Shore, L. Ong, J. Loughney, M. Stillman, January 2002, URL: http://www.ietf.org/rfc/rfc3237.txt

[RFC3566] "The AES-XCBC-MAC-96 Algorithm and Its Use With IPsec", S. Frankel, H. Herbert, September 2003, URL: http://www.ietf.org/rfc/rfc3566.txt

[RFC3602] "The AES-CBC Cipher Algorithm and Its Use with IPsec", S. Frankel, R. Glenn, S. Kelly, September 2003, URL: http://www.ietf.org/rfc/rfc3602.txt

[RFC3640] "RTP Payload Format for Transport of MPEG-4 Elementary Streams", J. van der Meer, D. Mackie, V. Swaminathan, D. Singer, P. Gentric, November 2003, URL: http://www.ietf.org/rfc/rfc3640.txt

[RFC3664] "The AES-XCBC-PRF-128 Algorithm for the Internet Key Exchange Protocol (IKE)", P. Hoffman, January 2004, URL: http://www.ietf.org/rfc/rfc3664.txt

[RFC3711] "The Secure Real-time Transport Protocol (SRTP)", M. Baugher, D. McGrew, M. Naslund, E. Carrara, K. Norrman, March 2004, URL: http://www.ietf.org/rfc/rfc3711.txt

[RFC3830] "MIKEY: Multimedia Internet KEYing", J. Arkko, E. Carrara, F. Lindholm, M. Naslund, K. Norrman, August 2004, URL: http://www.ietf.org/rfc/rfc3830.txt

[RFC4281] "The Codecs Parameter for Bucket Media Types", R.Gellens, D. Singer, P. Frojdh, November 2005, URL: http://www.ietf.org/rfc/rfc4281.txt

[RFC4301] "Security Architecture for the Internet Protocol", S. Kent and K. Seo, December 2005, URL: http://www.ietf.org/rfc/rfc4301.txt

[RFC4568] "Session Description Protocol (SDP) Security Descriptions for Media Streams", F. Andreasen, M. Baugher, D. Wing, July 2006, URL: http://www.ietf.org/rfc/rfc4568.txt

[RFC4771] "Integrity Transform Carrying Roll-Over Counter for the Secure Real-time Transport Protocol (SRTP)", V. Lehtovirta, M. Naslund, K. Norrman, January 2007, URL: http://www.ietf.org/rfc/rfc4771.txt

[SIMULCRYPT] ETSI TS 103 197 (V1.5.1): Digital Video Broadcasting (DVB); Head-end implementation of DVB SimulCrypt

[SSL30] "SSL 3.0 Specification", Netscape Communications, November 1996, URL: http://wp.netscape.com/eng/ssl3/draft302.txt

[XBS DRM extensions-v1.0] "OMA DRM v2.0 Extensions for Broadcast Support", Open Mobile Alliance™, OMA-TS-DRM-XBS-V1_0,
URL: http://www.openmobilealliance.org/

2.2 Informative References

[ISO-14496-12] "Information technology – Coding of audio-visual objects, Part 12: ISO base media file format", ISO/IEC 14496-12,
URL: http://www.iso.org

[OMA SUPL] "OMA Secure User Plane Location V 1.0", Open Mobile Alliance™, OMA-ERP-SUPL-V1_0-20070122-C,
URL: http://www.openmobilealliance.org/

[OMNA] Open Mobile Naming Authority, Open Mobile Alliance™,
URL: http://www.openmobilealliance.org/tech/omna

3. Terminology and Conventions

3.1 Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

All sections and appendixes, except “Scope”, are normative, unless they are explicitly indicated to be informative.

3.2 Definitions

(U)SIM	An SIM or a USIM application residing in the memory of the UICC.
BCAST Permissions Issuer	The BCAST Permissions Issuer (BCAST PI, or simply PI) is a logical entity that issues to BCAST terminals key material and/or consumption rules, the latter in the form of permissions and constraints. These rules in turn allow and control a user’s consumption of live or stored content pertaining to broadcast services. For the DRM Profile, consumption rules are defined by Generalized Rights Objects (GRO) as specified in [XBS DRM extensions-v1.0], and the BCAST Permissions Issuer is synonymous with the “Rights Issuer” in OMA DRM. For the Smartcard Profile, such rules are defined by the contents of the EXT BCAST payload included in the LTKM, and may indicate the number of times the SEK/PEK can be used to replay content.
Broadcast Device	A device that does not support an interactive communication channel and cannot communicate with other entities except using the broadcast channel. Note that a Broadcast Device can still have an implicit return channel: it may present information, triggers and dialogs to the user who may “implement” the interactive channel in various ways (e.g. telephone, web portal, service desk).
Broadcast Rights Object	This is a Rights Object used by DRM Profile of the Service and Content Protection for rights delivered over the broadcast channel. Encoding of the BCRO is specified in [XBS DRM extensions-v1.0].
Content Encryption	The cipher algorithm is applied on the data before packetization for transport or encapsulations occur.
Content Protection	This involves the protection of content (files or streams) during the complete lifetime of the content i.e. it is NOT an access control mechanism as it involves post-acquisition rules. Content protection is enabled for encrypted content through the use of appropriate rules or rights, e.g. using DRM Profile or Smartcard Profile based solution for file and stream distributed content. Content remains protected in the Terminal. Usage rules are enforced at "consumption time" (based on DRM or Smartcard Profile). In addition to subscription and pay-per-view, typically associated with Service Protection, Content Protection enables more fine-grained usage rules, such as for displaying, saving in unencrypted form, printing, processing, re-distributing, etc.
CSIM	Acronym for ‘cdma2000 Subscriber Identify Module’ corresponding to an application defined in [3GPP2 C.S0065-0] residing on the UICC to register services provided by 3GPP2 mobile networks with the appropriate security.
DRM Profile	The DRM Profile uses the Service and Content Protection solution for BCAST receivers in which the long term key management and registration of devices is based on OMA DRM and the broadcast extensions [XBS DRM extensions-v1.0]. The Service & Content Protection solution for the DRM Profile is described in Section 5.
Generalized Rights Object	This term is used in this document as a more generic term whenever an RO or a BCRO is meant.
Interactive Device	A device that supports an interactive communication channel and can communicate with other entities without using the broadcast channel for the communication. For example, an Interactive Device can execute interactive protocols, like the DRM 2.0 ROAP protocol or HTTP towards a Broadcast Permissions Issuer.
ISIM	An IP Multimedia Services Identity Module is an application defined in [3GPP TS 31.103 v6] and [3GPP2 C.S0069-0] residing in the memory of the UICC, providing IP service identification, authentication and ability to set up Multimedia IP Services.

Long-Term Key Message	Collection of keys and possibly, depending on the profile, other information like permissions and/or other attributes that are linked to items of content or services.
MIKEY (Multimedia Internet KEYing)	IETF defined key management protocol to support multimedia security protocols, as defined in [RFC3830]
Program	A logical portion of a service or content with a distinct start and end time. In the case the program is not free-to-air, it can be offered individually for purchase, such as "Pay-Per-View", or as part of a parent service (e.g. subscription service).
Rights Object	This is the Rights Object used by the DRM Profile of the Service and Content Protection for rights delivered over the interactive channel. Encoding of the RO is specified in [DRMDRM-v2.0], and some extensions are specified in [XBS DRM extensions-v1.0].
R-UIM	Acronym for 'Removable User Identity Module' corresponding to a non-UICC platform based standalone module as defined in [3GPP2 C.S0023-C] to register services provided by 3GPP2 mobile networks with the appropriate security.
Secure Storage Entity	The secure storage entity protects sensitive data such as cryptographic keys introduced by either the DRM Profile or the Smartcard Profile. Only an authorized agent is allowed to access the sensitive data. To ensure that the sensitive data is not manipulated fraudulently, it is integrity protected. The sensitive data are also cryptographically protected to guarantee its confidentiality. The secure storage entity can be implemented on either the Smartcard or the terminal.
Service Protection	This involves protection of content (files or streams) during its delivery i.e. it is an access control mechanism only. In the absence of any subsequent Content Protection, content is freely available (thus unencrypted) once it is securely delivered. For the benefit of allowing Content Protection to be provided for the same service, Service Protection is limited to immediate consumption / rendering only.
Short-Term Key Message	Message delivered alongside a protected service, carrying key material to decrypt and optionally authenticate the service, and access rights to delivered content.
SIM	A Subscriber Identity Module is a standalone module defined in [3GPP TS 51.011 v4] to register services provided by 2G mobile networks with the appropriate security.
Smartcard	A non-UICC secure function platform which may contain the SIM or R-UIM module, or a UICC-based secure function platform which may contain one or more of the following applications: a 3GPP USIM, 3GPP2 CSIM or 3GPP/3GPP2 ISIM. Note that the set of applications/modules residing on the Smartcard are typically governed by the affiliation of the Smartcard to 3GPP or 3GPP2 specifications, as indicated by the definition below for "Smartcard Profile".
Smartcard Profile	Alias for a set of Smartcard-based technologies and mechanisms which provide key establishment and key management, as well as permission and token handling for the Service and Content Protection solution for BCAST Terminals. In particular, subscriber key establishment and both short and long term key management may be based on GBA mechanisms and a Smartcard with (U)SIM/ISIM as defined by 3GPP, or based on a pre-provisioned shared secret key and a Smartcard with R-UIM/CSIM/ISIM or a UIM as defined by 3GPP2. The Smartcard Profile is described in Section 6.
Transport Encryption	The cipher algorithm is applied on the data that have been packetized for transport on a network.
UICC	A Universal Integrated Circuit Card is a physically removable secured device as defined in [3GPP TS 31.101 v6] for communication purposes not restricted to mobile convenience only. It is a platform to all the resident applications (e.g. USIM, CSIM or ISIM)
UIM	Acronym for 'User Identity Module', representing a standard device or functionality which provides secure procedures in support of registration, authentication, and privacy functions in mobile telecommunications. In the context of BCAST, the UIM refers specifically to the non-removable version of this standard device or functionality which is employed by (some) mobile terminals which operate according to 3GPP2 specifications. In addition, Smartcard Profile based service and content protection functionality can be provided on UIM-equipped BCAST Terminals.
USIM	A Universal Subscriber Identity Module is an application defined in [3GPP TS 31.102 v6] residing in the memory of the UICC to register services provided by 3GPP mobile networks with the appropriate

security.

3.3 Abbreviations

3GPP	3rd Generation Partnership Project
3GPP2	3rd Generation Partnership Project 2
AES	Advanced Encryption Standard
AU	Access Unit
AVC	Advanced Video Codec
BAK	BCMCS Access Key
BCAK	BCRO Authentication Key
BCD	Binary Coded Decimal
BCI	Binary Content ID
BCMCS	Broadcast and Multicast Services
BCRO	Broadcast Rights Object
BDS	Broadcast Distribution System
BDS-SD	BDS Service Distribution
BM-SC	Broadcast-Multicast Service Centre
BSDA	BCAST Service Distribution and Adaptation
BSF	Bootstrapping Server Functionality
bslbf	Bit String, Left Bit First
BSM	BCAST Subscription Management
CSIM	cdma2000 subscriber Identify Module
DCF	DRM Content Format
DK	Device Key
GBA	Generic Bootstrapping Architecture
GBA_ME	ME-based GBA
GBA_U	GBA with UICC-based enhancements
GMK	Group Management Key
GRO	Generalized Rights Object
H-AAA	Home Authentication, Authorization and Accounting
HMAC	Hashed Message Authentication Code
ICC	Integrated Circuit(s) Card
IPsec	IP Security
ISIM	IP Multimedia Services Identity Module
ISMA	Internet Streaming Media Alliance
KV	Key Validity
LSB	Least Significant Bit
LTKM	Long Term Key Message
MAC	Message Authentication Code
MBMS	Multimedia Broadcast Multicast Service

ME	Mobile Equipment
MIKEY	Multimedia Internet KEYing
MJD	Modified Julian Date
mjdutc	Modified Julian Date Coordinated Universal Time
MK	Master Key
MKI	Master Key Index
MRK	MBMS Request Key
MSK	MBMS Service Key
MTK	MBMS Transport Key
MTU	Maximum Transmission Unit
MUK	MBMS User Key
NAF	Network Application Function
NALu	Network Abstraction Layer Unit
OMA	Open Mobile Alliance
OMNA	Open Mobile Naming Authority
PAK	Program Authentication Key
PAS	Program Authentication Seed
PDCF	Packetized DCF
PEAK	Program Encryption / Authentication Key
PEK	Program Encryption Key
PKI	Public Key Infrastructure
PPT	Pay Per Time
PPV	Pay Per View
PRF	Pseudo Random Function
REK	Rights Encryption Key
RFC	Request For Comments
RIAK	Right Issuer Authentication Key
RK	Registration Key
RO	Rights Object
ROAP	Rights Object Acquisition Protocol
RTP	Real-time Transport Protocol
R-UIM	Removable User Identity Module
SA	Security Association
SAC	Secure Authenticated Channel
SAK	Service Authentication Key
SAS	Service Authentication Seed
SCK	SmartCard Key
SEAK	Service Encryption / Authentication Key
SEK	Service Encryption Key
SG	Service Guide

SHA-1	Secure Hash Algorithm
SIM	Subscriber Identity Module
SK	Short-term Key (appears in 3GPP2 BCMCS specifications)
SKI	Symmetric Key Infrastructure
SM	Subscription Manager
SMK	Subscriber Management Key
SPI	Security Parameters Index
SRK	Subscriber Request Key
SRTP	Secure Real-time Transport Protocol
STKM	Short Term Key Message
TAK	Traffic Authentication Key
TAS	Traffic Authentication Seed
TBK	Terminal Binding Key
TEK	Traffic Encryption Key
TK	Temporary Key
TKM	Traffic Key Message
UDN	Unique Device Number
UE	User Equipment
UICC	Universal Integrated Circuit(s) Card
UIM	User Interface Module
uimsbf	Unsigned Integer Most Significant Bit First
URI	Uniform Resource Indicator
USIM	Universal Subscriber Identity Module
UTC	Universal Time, Co-ordinated
XBS	Extensions for Broadcast Support

3.4 Symbols

E{K}(M)	Encryption of message ‘M’ using key ‘K’
D{K}(M)	Decryption of message ‘M’ using key ‘K’
A || B	Concatenation of A and B
LSBm(X)	The bit string consisting of the m least significant bits of the bit string X.
MSBm(X)	The bit string consisting of the m most significant bits of the bit string X.
HEX(X)	The hexadecimal presentation of the parameter containing hexadecimal characters 0-9 and a-f (in lowercase) with possible preceding zeros. As an example, for a 16 bit value 2748, HEX() returns "0abc". Note that two characters are always generated for each byte.

4. Introduction

4.1 Overview of the Solution

An architectural overview of Service Protection and Content Protection appears in [BCAST10-Architecture].

This specification describes the Service Protection and Content Protection system for OMA BCAST services. Not only does such system enable the restriction of access to services to authorised users during broadcast delivery, it also controls the consumption of the associated content throughout its lifetime.

OMA BCAST has requirements to provide both protection for broadcast content and services. However, the protection of broadcast content and services are required for different purposes:

- Content Protection: This involves the protection of content (files or streams) during the complete lifetime of the content. Content providers require securing the content not only at the present time of broadcasting, but also in the future. Some content providers might want to determine post-acquisition usage rules or so called digital rights. These can be obtained on an individual basis by the end user. Other content providers have content to offer, for which they do not require technical restrictions but limit it to fair use cases and rely on copyright acts.

- Service Protection: This involves protection of content (files or streams) during its delivery. Service providers require a secure access mechanism. They are only concerned with managing access to the content at the time of broadcasting. This is independent of the offered content and independent of the presence of digital rights for certain types of content. Only an access/no-access mechanism is required to distinguish between subscribed and not-subscribed users.

Therefore, Service Protection and Content Protection may be handled by two different security mechanisms. The complete protection system consists of Service and/or Content Protection. The possible key management systems and encryption are as defined in this document. There are two possibilities:

◊ DRM Profile: OMA DRM based solution for managing the keys. This is described in Section 5. The DRM Profile is derived from, and almost identical to, DVB-H 18Crypt.

  - For file download delivered over the broadcast channel, the Service or Content Protection is as per OMA DRM 2.0 specifications or using DCF or IPsec as specified in this specification. In this case normal usage rules are as defined in the OMA DRM 2.0 Rights Object.

  - For real-time broadcast streaming using RTP, Service or Content Protection is applied using the relevant broadcast extensions and appropriate encryption (IPsec, SRTP, ISMACryp). Post delivery usage rules associated with the service and / or specific program content are delivered in Rights Objects and STKMs. These rules can apply to content recorded in an appropriate file format, as defined in this specification for broadcast streams, which may be recorded either encrypted or unencrypted.

◊ Smartcard Profile: Smartcard based solutions for managing the keys. These are described in Section 6.

  - For file download delivered over the broadcast channel, the Service or Content Protection uses DCF or IPsec as specified in this specification. In this case normal usage rules are as defined in the LTKMs and STKMs.

  - For real-time broadcast streaming using RTP, Service or Content Protection is applied using the appropriate encryption (SRTP or ISMACryp). Post delivery usage rules associated with the service and/or specific program content MAY be delivered in LTKMs and STKMs. These rules can apply to content recorded in an appropriate file format, as defined in this specification for broadcast streams, which may be recorded either encrypted or unencrypted.

In addition to the key management, the encryption solution can operate on one of the following ways:

◊ The Internet Protocol (IP) layer based on the IPsec security standard, in which case it is transparent to IP based receiver applications like video players.

◊ The transport layer, based on the SRTP security standard.

◊ The content level, i.e. by encrypting Access Units before packetization occurs (ISMACryp).

For Service or Content Protection, both IPsec and SRTP allow the solution to be completely independent of the content format by protecting content at the transport level. On the other hand, content encryption is provided at the content level by using ISMACryp, allowing the solution to be completely independent of formats used on the transport level. Service or Content Protection may include message authentication/integrity protection and detecting replay attacks.

A service provider may use content level encryption instead of transport level encryption for streaming to provide Service Protection and support Content Protection for the same encrypted stream. In this case, the service offered depends on the nature of implicit or explicit rights delivered (access-only right or post-acquisition rights). To allow this scenario, recording of content-encrypted content shall be allowed in encrypted format only if content encryption is used for the purpose of providing optional Content Protection.

An OMA BCAST Terminal MAY implement Service Protection and MAY implement Content Protection, as shown in Table 1.

Table 1: Service Protection and Content Protection in OMA BCAST Terminals

	BCAST Terminal
Service Protection	OPTIONAL
Content Protection	OPTIONAL

For BCAST Terminals with Service Protection:

Table 2 summarises the possible scenarios. At least one Profile SHALL be implemented. Both Profiles MAY be implemented.

- A BCAST Terminal with a cellular radio interface and a Smartcard SHALL implement the Smartcard Profile. The DRM Profile is OPTIONAL. Hence terminals MAY implement both profiles.
- A BCAST Terminal with a cellular radio interface and no Smartcard SHALL implement the DRM Profile (the Smartcard Profile is not applicable).
- A BCAST Terminal that does not have a cellular radio interface SHALL implement the DRM Profile (the Smartcard Profile is not applicable based on current technology).

Table 2: OMA BCAST Terminal Profile Support for Service Protection

	DRM Profile	Smartcard Profile
Terminal without cellular radio interface or without Smartcard	MANDATORY	N/A
Terminal with cellular radio interface and Smartcard	OPTIONAL	MANDATORY

For BCAST Terminals with Content Protection:

Table 3 summarises the possible scenarios. At least one profile SHALL be implemented. Both profiles MAY be implemented.

- A BCAST terminal with a cellular radio interface and a Smartcard MAY implement the Smartcard Profile or MAY implement the DRM Profile. The Terminal SHALL implement at least one profile. Hence terminals MAY implement both profiles.

- A BCAST terminal with a cellular radio interface and no Smartcard SHALL implement the DRM Profile (the Smartcard Profile is not applicable).
- A BCAST terminal that does not have a cellular radio interface SHALL implement the DRM Profile (the Smartcard Profile is not applicable).

Note that 'Terminal Implementation' of a content protection profile means that the Terminal is capable of it, but does not necessarily mandate its use. Decision to use (or not to use) an implemented content protection profile is made at the time of service deployment.

Table 3: OMA BCAST Terminal Profile Support for Content Protection

	DRM Profile	Smartcard Profile
Terminal without cellular radio interface or without Smartcard	MANDATORY	N/A
Terminal with cellular radio interface and Smartcard	OPTIONAL	OPTIONAL

Adaptations of the described service and content protection mechanisms to underlying Broadcast Distribution Systems (BDSs) are possible and are described in Section 15 and in the respective adaptation specifications, e.g. [BCAST10-MBMS-Adaptation], [BCAST10-BCMCS-Adaptation], and [BCAST10-DVBH-IPDC-Adaptation].

4.2 Selected Technologies

These are the main standards on which the solution is based:

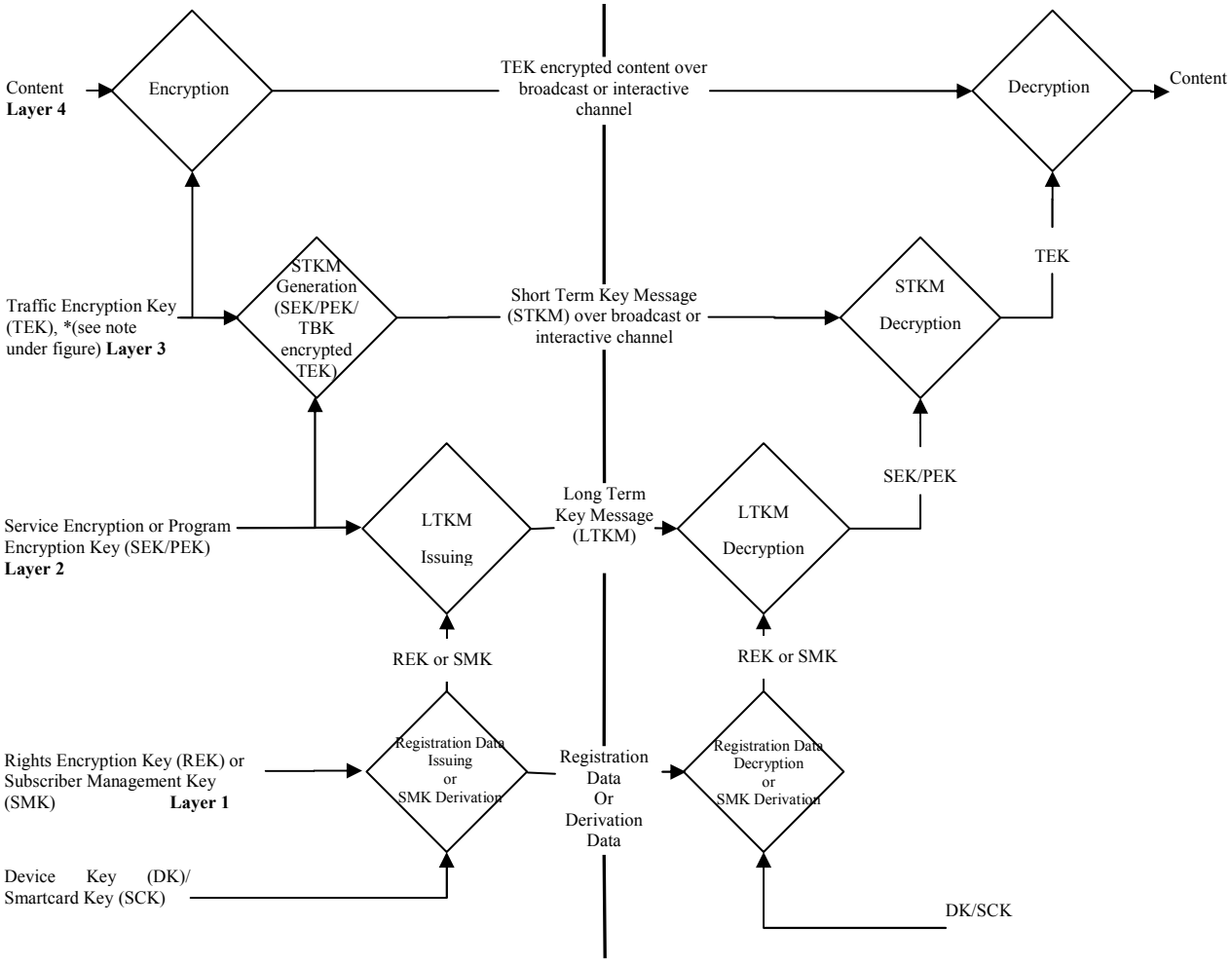
- Advanced Encryption Standard (AES, see [FIPS197]) in the Cipher Block Chaining mode with 128 bit keys, for actual content encryption. Furthermore, OMA DRM uses AES-WRAP in its Rights Objects and optionally AES CBC-MAC. AES-ECB is also used by the terminal binding scheme to protect the TEKs.
- Secure Internet Protocol (IPsec, see [RFC2406]) using the Encapsulating Security Payload (ESP) protocol, for implementing transport encryption and decryption as a function of the IP stack. Only transport mode is used.
- Secure Real Time Protocol (SRTP, see [RFC3711]) for implementing service protection at the transport layer. SRTP uses AES-CTR (counter mode).
- Content encryption as specified in [ISMACRYP11]. Appropriate extensions are provided in this specification for codec agnostic RTP transport of ISMACryp protected streams.
- Traffic Encryption Key (TEK) delivery protocol and management is specified in this document.
- Terminal Binding Key (TBK) delivery protocol and use is specified in this document.
- Open Mobile Alliance (OMA) Digital Rights Management version 2.0 [DRM Enabler-v2.0] for service and content protection, managing rights and the associated service and program encryption keys, and the cryptographic protection of those keys themselves. This specification makes some adaptations to OMA DRMv2 for OMA BCAST, mainly for devices without interactive channel.
- DRM Rights Object delivery and device registration over the OMA BCAST channel, without using an interaction channel, are also newly specified. They are described in this document and [XBS DRM extensions-v1.0]. (Devices with access to the interactive channel do not need to implement those extensions for broadcast-only devices, as they typically do registration and Rights Object acquisition over the interactive channel only.)
- GBA [3GPP TS 33.220 v6] for (U)SIM and pre-provisioning [3GPP2 S.S0083-A] for (R-)UIM/CSIM for establishing a shared secret. Their applications for service protection are as specified in this document.

The reasons for choosing these particular technologies as the basis of the solution include the following:

- AES is an open standard symmetric encryption algorithm which is widely used in various standards including OMA DRM v2.0.
- IPsec ESP is the standard way of keeping service decryption at receiving devices within the IP stack, invisible to the receiving applications, which thus remain independent of service protection and the carriers of the IP packet streams (of which IPDC may be only one). IPsec ESP has been used in various existing applications.
- SRTP is a standard way of performing service decryption at receiving devices within the transport layer. SRTP has been used to protect all common forms of streaming content.
- ISMACryp allows encrypted content to be streamed. This means encrypted content stored in a file can be streamed at the server side and directly recorded in a file at the terminal side, without the need for decryption and re-encryption. Content encryption may be used to protect content during its complete lifetime, not only during transport.
- TEK management framework and protocol are specified in this document. Guidelines are provided on TEK management based on two different assumptions:
  - First, where the terminal is untrusted, the solution is made robust by using a key delivery protocol and management scheme for frequently changing TEKs to make it expensive for a misbehaving terminal to share TEKs with unauthorized devices.
  - The alternative is to trust the terminals to behave according to certain rules. In the context of the Smartcard Profile, the terminal is expected to delete TEKs after use, cache TEKs only for authorized use, for example to rewind and play content, and never transmit TEKs to external entities.
- GBA is a general architecture that allows to share securely a secret between a server and a client; it has already been used in 3GPP MBMS. Currently, 3GPP2 uses pre-provisioning to establish shared secret between (R-)UIM/CSIM and home network.

4.3 Overview of Operations for Streaming of Content

Streaming can be done with content coming either from a live source or from a file. Protection of streamed content can be done using service protection or content protection. Both protection mechanisms use the 4-layer model of Figure 1.



*Note: For the Smartcard profile, Traffic Encryption Key (TEK) may be encrypted by Terminal Binding Key (TBK) before encryption by the SEK/PEK for the STKM Generation.

Figure 1 – Protection via the 4-Layer Model

4.3.1 The 4-Layer Model

As illustrated in Figure 1, the solution is based on a 4-layer model key management architecture, with an optional optimisation to provide both secure subscription and pay-per-view purchase options for a single service. Traffic Encryption Keys (TEKs) are applied to the actual content (Layer 4) following different mechanisms depending on the actual encryption method used.

The TEKs are themselves sent encrypted by a Service or Program Encryption Key (SEK/PEK). These messages carrying TEKs are called Short Term Key messages (STKMs). STKMs are distributed over the same channels used by the corresponding content. When using the Smartcard Profile, the TEKs MAY optionally be encrypted with a Terminal Binding Key (TBK) before being encrypted by the SEK/PEK, to provide for terminal binding.

Separate SEK and PEK keys can have different lifetimes and can be used to provide, for a single service, different granularities of purchase periods to different customers. This allows for the efficient implementation of both subscription and pay-per-view business models for the same service. Pay-per-view customers are provided with a PEK that is only valid for a single program while subscribers would be provided with a SEK, valid for reception of the service for some longer period. For the DRM Profile, within the STKM, the TEK is encrypted with a PEK, and the PEK is also carried in the STKM, encrypted with the SEK. Thus, pay-per-view customers can directly decrypt the TEK, while subscription-based customers can decrypt the PEK by using the SEK, which can then be used to decrypt the TEK.

For DRM profile, STKMs contain extension of content IDs for the program or service. Devices use this ID to identify which Long Term Key Message (LTKM) contains the necessary keys to use for decryption of Short Term Key messages. For the Smartcard Profile, STKMs contain the SEK or PEK ID directly to identify the SEK/PEK used to protect the STKM; LTKMs also contain a flag indicating whether or not a TBK is used. The LTKMs are delivered over the broadcast or interactive channel in case of the DRM Profile and over the Interactive channel in case of the Smartcard Profile.

Where the service and program functionality differentiation is not required or supported, the TEK can be directly encrypted with the SEK, and the SEK-encrypted PEK can be omitted from the STKM.

For the Smartcard Profile key management, please refer to Section 6.

Depending on the key management profile (DRM Profile or Smartcard Profile), either the Rights Encryption Key (REK) or the Subscriber Management Key (SMK) is used to protect the LTKM delivery. The key material (REK or SMK) and meta-data are delivered as a result of the registration phase (DRM Profile) / Subscriber Key Establishment phase (Smartcard Profile).

Cryptographic keys introduced by the 4-layer model SHALL be stored securely within a secure storage entity to guarantee the access control, the confidentiality and the integrity of the sensitive data and SHALL never be exposed outside of the secure storage.

Only the TEK among cryptographic keys MAY be allowed to be exposed outside the secure storage upon request from authorized applications.

4.3.2 Streaming Using Service Protection

For service protection, encryption is carried out using the AES algorithm with 128 bit symmetric traffic keys. TEKs are retrieved from the secure storage entity and are applied:

- as part of standard IPsec security associations (SAs), or
- as an SRTP master key, from which the session key is derived as per SRTP specification, or
- directly to encrypt the content, presented as Access Units (AUs), before packetization for transport occurs (ISMACryp).

Depending on the chosen encryption, the keys are used to perform decryption automatically before passing the packets to the receiving application.

The SEKs or PEKs are transmitted to each receiving device within Long Term Key messages (LTKMs) and SHALL be stored within the secure storage entity, and SHALL never be exposed outside of the secure storage. If OMA DRM 2.0 extensions [XBS DRM extensions-v1.0] are used, the LTKMs are referred to as Generalized Rights Objects. Such transmission of LTKMs can be done in two different ways, depending on whether the receiving device can make use of an interactivity channel:

- Via broadcast over OMA BCAST broadcast channel, or
- Via an interactivity channel.

As already mentioned, there are two key management systems:

- Using OMA DRM 2.0 Extensions [XBS DRM extensions-v1.0]. When delivering LTKMs over the OMA BCAST broadcast channel in the form of Rights Objects (ROs), bandwidth is a major constraint. This specification addresses this problem in two complimentary ways. Firstly, a new binary form of an RO, called a Broadcast Rights Object (BCRO), is defined. Secondly, a method is described for securely delivering BCROs to groups of devices at the same time. Valuable portions of ROs are protected by group or unit keys, and when necessary, broadcast encryption can be used to allow messages to be decrypted only by arbitrary sets of devices within a larger group. When delivering ROs to devices that have access to an interactive channel, implementation complexity is a major constraint. Thus, such devices, which are expected to support OMA DRM 2.0 for interactive content services, use standard OMA DRM 2.0 mechanisms as much as possible, e.g. they acquire ROs for broadcast content via the interactive channel using the DRM 2.0 ROAP protocol, as they would do for non-broadcast content as well. This specification defines also an efficient and user-friendly process for the registration of devices which do not have an interactivity channel. Rights Encryption Keys (REKs) are also

delivered to receive-only devices during a device registration process protected using the public key of the individual devices. When an interactivity channel is available, the registration process is according to standard OMA DRM v2.0.

- Using Smartcard Profile. An overview of operation is given in Section 6.

4.3.3 Streaming Using Content Protection

For content protection, encryption is carried out according to AES using 128 bit symmetric traffic keys. While service protection provides protection of the stream only at the time of service reception, content protection provides protection of the content even after the service reception, i.e. content remains stored protected in the Terminal. On one hand, content protection may be achieved by using TEKs to encrypt the content before packetization for transport or when encapsulation in a file occurs (ISMACryp). On the other hand, content protection may also be provided using transport encryption (SRTP or IPsec) and appropriate measures in the receiving device to protect content inside the device.

4.4 Overview of Operation for Download of Content

Protection of files is as defined by OMA DRM 2.0 specifications [DRM Enabler-v2.0] for the DRM Profile. For the Smartcard Profile, a modified version of the DCF file format is defined in this specification.

The mechanisms supported for the protected download of content using file delivery are dependent on the profile used.

For the DRM Profile the protection of files is achieved

- as defined by the OMA DRM 2.0 specifications [DRM Enabler-v2.0], or
- using an additional box in the extended headers field of the DCF file format and encryption by TEKs as defined in [XBS DRM extensions-v1.0], or
- using IPsec.

For the Smartcard Profile the protection of files is achieved

- using an additional box in the extended headers field of the DCF file format and encryption by TEKs as defined in [XBS DRM extensions-v1.0], or
- using IPsec.

Note that combining the above methods allows compatibility with OMA DRMv2 DCF file format and operation with both DRM Profile and Smartcard Profile.

4.4.1 Content Download Using Service Protection

Content download by using Service Protection is specified in Section 5.6.2.1 for the DRM Profile and Section 6.8.2.1 for the Smartcard Profile.

4.4.2 Content Download Using Content Protection

Content download by using Content Protection is specified in Section 5.6.2.2 for the DRM Profile and Section 6.8.2.2 for the Smartcard Profile.

4.5 Key Management

The 4-layer model described in [BCAST10-Architecture] allows different key management systems to be used. (See also Section 4.3.1.) This section outlines the key management profiles defined for BCAST 1.0, namely the Smartcard Profile and the DRM Profile.

The Smartcard Profile defines a key management system based on the symmetric key model used by either the 3GPP MBMS [3GPP TS 33.246 v7] security model based on the (U)SIM or 3GPP2 BCMCS [3GPP2 S.S0083] security model based on (R

)UIM/CSIM. An overview of the Smartcard Profile is provided in Section 4.5.2, while a full description is provided in Section 6.

The DRM Profile defines a key management system based on the Public Key Infrastructure (PKI) offered by OMA DRM v2.0 [DRMDRM-v2.0]. An overview of the DRM Profile is provided in Section 4.5.1, while a full description is provided in Section 5.

In order to ensure maximum interoperability, OMA BCAST defines a common layer for traffic encryption (Layer 4) and allows the other layers of key management to be implemented using either the DRM Profile or the Smartcard Profile.

Adaptation of the 4-layer model used in OMA BCAST to underlying BDSes is specified for 3GPP MBMS, 3GPP2 BCMCS and IPDC over DVB-H. This adaptation allows the existing functionalities provided by the underlying BDS to be re-used. Information on the appropriate adaptation is provided in Section 15.

4.5.1 DRM Profile Overview

The DRM Profile is based on public key based mechanisms. It uses the OMA DRMv2.0 Enabler [DRMDRM-v2.0] to support key management. For non-interactive devices, broadcast extensions for OMA DRMv2.0 as specified in [XBS DRM extensions-v1.0] are used.

The key management for the DRM Profile is based on the BCAST 4-layer model key hierarchy introduced in Section 4.3.1.

Layer 1 is for registration purpose and uses the public/private key pair stored in the BCAST terminal. The public key is used to secure the delivery of the Rights Encryption Key (REK), and, with the corresponding private key, the Generalized Rights Objects (GROs) can be processed. The REK may be delivered over an interactive or broadcast channel. In case of delivery via the broadcast channel the REK may refer to several keys, which are delivered to the BCAST terminal using the registration process specified in [XBS DRM extension-v1.0]. In case of an interactive channel, the ROAP registration procedure [DRMDRM-v2.0] is applied. Note that the actual provisioning of the public/private key pair is out of scope for this specification.

In Layer 2, the Long Term Key Messages (LTKMs) are delivered. The LTKM is a Generalized Rights Object, which may take two alternative formats. In case LTKM is delivered over a broadcast channel, the format used is of a Broadcast Rights Object (BCRO) as specified in [XBS DRM extensions-v1.0]. If an interactive channel is used, the GRO is a Rights Object (RO) as specified for OMA DRMv2.0 [DRMDRM-v2.0]. The LTKM transports the Service or Program Encryption Key (SEK/PEK), as well as permissions and attributes. SEK/PEK is encrypted using the keys delivered or broadcasted during the Layer 1 registration procedure.

Layer 3 securely transports short term keys, i.e. the Traffic Encryption Keys (TEK), in the Short Term Key Message (STKM) that are broadcasted over the same network as the media streams. Furthermore, data can be protected in case of streaming and file delivery respectively for both service and content protection. In the case where the TEK is encrypted with a PEK, the STKM may also carry the SEK-encrypted PEK.

Finally, Layer 4 is responsible for traffic encryption using the TEK for stream or file delivery respectively for both service and content protection.

The DRM Profile key management is described in detail in Section 5.

4.5.2 Smartcard Profile Overview

The Smartcard Profile is based on existing security technologies and standards defined for 3GPP or 3GPP2 broadcast/multicast services.

In the context of the BCAST 4-layer model key hierarchy, the Smartcard Profile provides a key management solution that uses a Smartcard and an interactive cellular radio interface. Assuming key provisioning has taken place, this solution enables authentication and Subscriber Key Establishment (Layer 1), LTKM delivery (Layer 2) and STKM delivery (Layer 3), as specified in Section 6. Access to the protected content (Layer 4) is supported irrespective of the type of encryption used (SRTP or ISMACryp or IPsec[H.A.A.1]), as specified in Section 9.

This specification defines two variants of the Smartcard Profile. The two variants are referred to as the (U)SIM Smartcard Profile and the (R-)UIM/CSIM Smartcard Profile respectively. The two variants differ in the way that the Smartcard establishes the Layer 1 key(s) but are otherwise the same (Layers 2, 3 and 4).

The Subscriber Key Establishment layer (Layer 1) makes use of a secret key stored on a Smartcard based identity module. This key is referred to as "SmartCard Key" (SCK) in the Smartcard Profile. The SCK is a pre-provisioned secret key that is shared between the Smartcard and the Smartcard issuer. If the Smartcard issuer is not also the broadcast service provider, then the SCK is unknown to the broadcast service provider.

The SCK is used to create the Layer 1 key, the Subscriber Management Key (SMK), using the Generic Bootstrapping Architecture (GBA) as defined in [3GPP TS 33.220 v6] for the (U)SIM Smartcard Profile, or using the pre-shared key mechanism as defined in [3GPP2 S.S0083] for the (R-)UIM/CSIM Smartcard Profile. The SMK is established between the broadcast service provider and the Smartcard or the terminal depending on the key management implementation. If the smart-card contains the key management system, the SMK SHALL be established between the broadcast service provider and the smart-card (using GBA-U, or respectively pre-shared key mechanism of 3GPP2); otherwise it is established with the terminal (using GBA-ME or 2G GBA).

The SMK SHALL be stored on the Smartcard or the terminal depending on the variant of the Smartcard Profile key management implemented. For the (U)SIM Smartcard Profile, the SMK SHALL be stored on a USIM when using GBA_U, and on a terminal when using GBA_ME or 2G GBA. For the (R-)UIM/CSIM Smartcard Profile, the SMK SHALL be stored on a (R-)UIM/CSIM.

The SMK is a user-specific key used to protect the Long Term Key Messages (LTKM) that are delivered in Layer 2. Depending on the service configuration, within the LTKM a Program Encryption Key (PEK) or a Service Encryption Key (SEK), used respectively for pay per view or subscription customers, is delivered protected by SMK. For the (U)SIM Smartcard Profile, the SEK or PEK SHALL be stored on a USIM when using GBA_U or on a terminal when using GBA_ME or 2G GBA. For the (R-)UIM/CSIM Smartcard Profile the SEK or PEK SHALL be stored on a (R-)UIM/CSIM.

Layer 3 delivers the Short Term Key Message (STKM) within which Traffic Encryption Keys (TEKs) are protected using SEK or PEK, as well as optionally by a Terminal Binding Key (TBK).

Layer 4 is for traffic encryption using the TEK for stream or file delivery respectively for both service and content protection.

Table 4 gives a brief outline of the 4-layer model key hierarchy:

Table 4: Smartcard Profile key hierarchy model

Key layer	Key name	Key hierarchy		Storage location
Key Provisioning	SmartCard Key (SCK)	SCK	Pre-provisioned secret key shared with the Smartcard issuer. Provisioning of this key is out of the scope of this specification.	Smartcard
Layer 1: Subscriber Key Establishment	Subscriber Management Key (SMK)	SMK	For the (U)SIM Smartcard Profile SMK is generated as a result of a successful run of the GBA bootstrapping procedure. For (R-)UIM/CSIM Smartcard Profile, SMK is derived from the SCK. SMK is equivalent to the MBMS User Key (MUK).	Smartcard (for GBA_U or if security is based on registration key RK) or Terminal (for GBA_ME or 2G GBA)
Layer 2: LTKM	Service / Program Encryption Key (SEK/PEK)	SMK[SEK] or SMK[PEK]	Protected by SMK and sent to the Smartcard via the terminal using a point to point channel (in the case of GBA_U). SEK/PEK is equivalent to the MBMS Service Key (MSK).	Smartcard (for GBA_U or if security is based on registration key RK) or Terminal (for GBA_ME or 2G GBA)
Layer 3: STKM	Traffic Encryption	SEK[TEK] or PEK[TEK]	Protected by SEK or PEK ((U)SIM variant) or derived from SEK or PEK ((R-)UIM/CSIM variant), and sent over	Terminal

	Key (TEK)		the broadcast channel. Optionally also encrypted with TBK. TEK is equivalent to the MBMS Traffic Key (MTK).	
Layer 4: traffic encryption		TEK[content]	TEK encrypted content; traffic encryption with SRTP, ISMACryp, or IPSec	

The Smartcard Profile key management is described in detail in Section 6.

5. DRM Profile

5.1 Introduction

OMA BCAST DRM Profile uses OMA DRMv2.0 specified solutions [DRMDRM-v2.0] for the registrations and rights management over the interactive channel and specifies a set of protocols for use in broadcast [XBS DRM extensions-v1.0] and out-of-band channels.

The following sections describe the four layers of the 4-layer model key hierarchy, as well as key provisioning required to access the first layer for DRM Profile. Section 5.2 briefly describes key provisioning. Section 5.3 describes registration. Section 5.4 describes the LTKM structure, while Section 5.5 describes that of the STKM. Section 5.6 and Section 5.6.2 describe how to protect data in case of streaming and file delivery respectively for both service and content protection. Recording aspects are described in Section 5.7, while SG signalling is explained in Section 5.8.

5.2 Key Provisioning

The OMA DRM Profile uses PKI-based mechanism. Access to the registration layer (Layer 1) is implemented using a device key (or public/private key pair) that is stored in the mobile device. How the device key is provisioned is out of scope for this specification.

5.3 Layer 1: Registration

The device must first register with the Rights Issuer to receive protected broadcast service. Registration can be performed either via an interaction or broadcast channel.

In the case that an interaction channel is used, the registration protocol is as defined in OMA DRMv2.0 [DRMDRM-v2.0] and right encryption keys (used to protect Layer 2 RO) are delivered protected with the public key of the device. In this case, the registration procedure is initiated by the device, e.g. on reception of a ROAP Registration Trigger, typically returned whenever an unregistered device executes any of the procedures for interactive service provisioning defined in [BCAST10-Services], or in response to an out-of-band mechanism.

For the devices that do not support an interaction channel, an alternative process for the registration is defined in [XBS DRM extension-v1.0] and a set of keys (used to protect Layer 2 BCRO) are delivered over the broadcast channel protected with the public key of the device.

OMA DRM Profile supports a notion of Broadcast Domains and Interactive Domains to facilitate sharing of content and services among the registered terminals, see [XBS DRM extensions-v1.0].

5.4 Layer 2: Long Term Key Message – LTKM

For the DRM Profile service encryption key (SEK)/program encryption key (PEK) is packaged in a special LTKM format. This special format is called Rights Object (RO) and in addition to the provided keys, it may contain permissions and attributes linked to the protected content. The profile supports the delivery of ROs over interactive and broadcast channel.

Before a device can start receiving LTKMs, it must be subscribed to the service or pay-per-view program that the LTKs protect. For devices that support an interaction channel, this is e.g. done with a “Service Request” or “LTKM Renewal Request” message as defined in [BCAST10-Services]. For devices that support only the broadcast channel, an out-of-band procedure is used (see Section 5.4.4.1.2 in [BCAST10-Architecture]). The information needed to perform the subscription is announced in the Service and various purchase-related fragments of the Service Guide (see Section 5.8). Services and pay-per-view programs that are available for purchase are generically referred to as “purchase items”.

Section 5.4.1 introduces and describes use of ROs. Section 5.4.2 gives OMA DRMv2.0 extensions for BCRO. Section 5.4.3 describes how ROs are used for service protection at Long Term Key Delivery layer

5.4.1 Use of ROs and BCROs

Service Encryption Keys (SEK) and Program Encryption Keys (PEK) described in Layer 2 of the Key Hierarchy for Service Protection MAY be transmitted to each terminal within Generalized Rights Objects (GROs). Two formats are available for the purpose. One is the format of an OMA DRM 2.0 Rights Object (RO), as specified in [DLRDRM-v2.0]. The other format is Broadcast Rights Object (BCRO) specified in XBS document [XBS DRM extensions-v1.0], and is used when GRO is delivered over a broadcast channel. In addition to SEKs/PEKs, GROs also contain permissions and other attributes linked to protected service. SEKs would typically be utilized for subscription services. Each SEK protects a single subscription service that can be purchased as a unit. A unit is the minimum granularity of services that a service provider offers to an end user, and a unit, therefore, MAY correspond to a single program channel, to a portion of a channel, or to a collection of program channels that are all purchased as a unit. The SEK is an intermediate key, i.e. it does not directly encrypt the content but instead encrypts a Traffic Encryption Key (TEK) or PEK. The SEKs themselves are encrypted by keys transmitted in Layer 1 of the Key Hierarchy. PEKs carried in GROs are encrypted by keys transmitted in Layer 1, and used to decrypt the TEK. In the context of BCAST Enabler, these GROs are called Long Term Key Messages (LTKMs).

A terminal periodically receives a set of SEKs/PEKs that MUST be encrypted and authenticated. Depending on the capabilities of underlying transport networks, multiple SEKs/PEKs MAY be combined into one LTKM directed to a terminal. There MAY also be multiple such messages that relay different sets of SEKs/PEKs to the same terminal.

SEKs SHOULD be periodically updated so that when someone drops a subscription, their access to a service will be terminated cryptographically once a SEK changes. For example, SEKs MAY change once per billing period (e.g., on a monthly basis). PEKs, when provided, SHOULD change once per program.

The transmission of LTKM to a terminal can be done over an interaction channel or over a broadcast channel, depending on whether the terminal has access to an interaction channel or not.

If the LTKM is transmitted over the broadcast channel, then the GRO MUST be encoded using a suitable binary encoding or compression. A GRO thus encoded is called a BCRO. The syntax for BCRO is introduced in XBS document [XBS DRM extensions-v1.0].

In addition, if the LTKM is transmitted over the broadcast channel, then digital signatures or MACs over the GRO MAY be verifiable over the BCRO itself without having to decode or de-compress the BCRO.

In addition, if the LTKM is transmitted over the broadcast channel, then all content of the LTKM other than the BCRO MUST be compressed or encoded.

If the LTKM is transmitted over the interaction channel, then the LTKM, including the GRO, digital signatures or MACs, MAY be encoded, compressed, or text-based.

5.4.2 OMA DRM v2.0 Extensions for Broadcast Rights Objects

Extensions to OMA DRM v2.0 for broadcast rights objects including design and format, appear in the OMA DRM v2.0 Extensions for Broadcast Support document [XBS DRM extension-v1.0][*].

An alternative Content Protection solution to that depicted in OMA DRM v2.0 Extensions for Broadcast Support document, or appropriate modifications thereto, is specified in Section 6.

5.4.3 GROs in Long Term Key Delivery Layer for service protection

In case of **subscription**, the Service Encryption and Authentication Key material (SEAK) associated with the service is securely delivered to the authorized terminal in a GRO. Such a GRO is called a **Service RO**. SEAK consists of 128 bits SEK (Service Encryption Key) and 128 bits SAS (Service Authentication Seed). SAS is used as a seed in a generic authentication

[*] (Informative Footnote) Where Generalized Rights Objects (particularly for post-acquisition rights associated with BCAST Stream Delivery of protected content) are stored in secure removable Smartcards, i.e. (U)SIM/(R)UIM/CSIM in 2G/3G mobile terminals, an alternative Content Protection scheme to handle such Broadcast Rights Objects may be applicable as an option. Such an alternative is explored in the OMA DRM working group.

function to derive SAK (Service Authentication Key). In general, a Service RO will contain key material associated with more than one service (when associated with a service bundle).

In case of **pay-per-view**, the Program Encryption and Authentication Key material (PEAK) associated with a pay-per-view event is securely delivered to the authorized terminal directly within a GRO. Such a GRO is called a **Program RO**. PEAK consists of 128 bits PEK (Program Encryption Key) and 128 bits PAS (Program Authentication Seed). PAS is used as a seed in a generic authentication function to derive PAK (Program Authentication Key).

The ID of GROs that contain SEAKs or a PEAK needs to be structured, to allow for the management of purchase transactions in the device, or more specifically, to create an association between the purchase item in the service guide and the successful completion of the purchase transaction (when the GRO related to the purchase has finally been received in the device). This is valid for both connected and especially for unconnected operation (see [DRMDRM-v2.0] for the definition of "connected" and "unconnected"), where the GRO may be received by the device much later than the purchase transaction is initiated. A connected device has a direct 2-way connection to the Rights Issuer (RI) through interaction channel. On the other hand, the unconnected devices do not have access to the RI through an interaction channel but they are capable of making connection via an intermediary interactive device.

Defining a structured ID for GRO will also allow the device to check later on whether GROs for all subscribed services are available (and have been renewed). The rekeying_period_number is an increasing number by which the ID of the GRO related to the same purchase item can be made unique.

The ID of a GRO linked with subscription (Service RO) or pay-per-view (Program RO), and bound to a device or to a domain, SHALL be constructed respectively as follows:

deviceRoID = "E" || ***deviceID*** || ***"_S"*** || ***stringtomakeitunique*** || ***"_I"*** || ***purchaseItemID*** || ***"_"*** || ***HEX(rekeying_period_number)***

domainRoID = "O" || ***domainID*** || ***"_S"*** || ***stringtomakeitunique*** || ***"_I"*** || ***purchaseItemID*** || ***"_"*** || ***HEX(rekeying_period_number)***

- **deviceID** is the Unqiue Device Number (UDN) as discussed in [XBS DRM extensions-v1.0].
- **stringtomakeitunique** Note that 'deviceRoID' and 'domainRoID' shall be globally unique. Note further that because of the specification of 'purchaseItemID' in the OMA BCAST SG, the global uniqueness is already guaranteed and therefore, 'stringtomakeitunique' shall be the empty string.
- **purchaseItemID** is the GlobalPurchaseItemID associated with the purchase item and signalled in the Purchase Item Fragment of the SG(see Section 5.8).
- **rekeying_period_number** is a 7-bit counter that is used to differentiate between different ROs with the same purchase_item_id (defined in Section 7.2 of [XBS DRM extensions-v1.0])

In the case of BCROs, the link with the corresponding subscription (Service RO) or pay-per-view (Program RO) is obtained by using the BCRO fields purchase_item_id and rekeying_period_number ([XBS DRM extensions-v1.0]).

A **Service RO** SHALL contain at least one (<CID>, <SEAK>) pair. The <CID> (Content Identifier) SHALL be constructed as specified in the paragraph defining the Short Term Key Message (see Section 5.5).

The <SEAK> contains SEK and SAS. SEK and SAS are obtained from a GRO as specified in Sections A.12.2.1 and A.12.2 of [XBS DRM extensions-v1.0]).

A **Program RO** SHALL contain at least one (<CID>, <PEAK>) pair. The <CID> SHALL be constructed as specified in the paragraph defining the traffic key message (see Section 5.5).

The <PEAK> contains PEK and PAS. PEK and PAS are obtained from a GRO as specified in Sections A.12.2.1 and A.12.2 of [XBS DRM extensions-v1.0]).

5.5 Layer 3: Short Term Key Message - STKM

This Section describes the format and role of STKM (Short Term Key Message) in the transport of short term traffic keys (TEKs) for DRM Profile at the Short Term Key Delivery layer.

Each STKM SHALL be encapsulated in exactly 1 UDP packet.

In order to keep access times low for devices that start accessing a service, a STKM SHALL be transmitted periodically.

The STKM SHALL be transported over the same network as the media streams that are protected with the traffic keys contained in the STKM. The STKM stream MAY be transported in an own session, e.g. transported in an own IP stream.

If the traffic_protection_protocol equals to TKM_ALGO_DCF, then the STKM MAY be delivered as a separate object inside a FLUTE session, together with the protected traffic, having its own FDT entry.

Table 5: Format of STKM for DRM Profile

Short_Term_Key_Message_Description	Length (in bits)	Type
short_term_key_message() {		
selectors_and_flags {		
protocol_version	**4**	**uimsbf**
protection_after_reception	**2**	**uimsbf**
reserved_for_future_use	**1**	**bslbf**
access_criteria_flag	**1**	**uimsbf**
traffic_protection_protocol	**3**	**uimsbf**
traffic_authentication_flag	**1**	**uimsbf**
next_traffic_key_flag	**1**	**uimsbf**
timestamp_flag	**1**	**uimsbf**
program_flag	**1**	**uimsbf**
service_flag	**1**	**uimsbf**
}		
if (traffic_protection_protocol == TKM_ALGO_IPSEC) {		
security_parameter_index	**32**	**uimsbf**
if (next_traffic_key_flag == TKM_FLAG_TRUE) {		
next_security_parameter_index	**32**	**uimsb**
}		
}		
if (traffic_protection_protocol == TKM_ALGO_SRTP) {		
master_key_index_length	**8**	**uimsbf**
master_key_index	**8*master_key_index_length**	**uimsbf**
reserved_for_future_use	**5**	**bslbf**
next_master_key_index_flag	**1**	**uimsbf**
next_master_salt_flag	**1**	**uimsbf**
master_salt_flag	**1**	**uimsbf**
if (master_salt_flag == TKM_FLAG_TRUE) {		
master_salt	**112**	**bslbf**
}		
if (next_traffic_key_flag == TKM_FLAG_TRUE) {		
if (next_master_key_index_flag == TKM_FLAG_TRUE) {		
next_master_key_index	**8*master_key_index_length**	**uimsbf**
}		
if (next_master_salt_flag == TKM_FLAG_TRUE) {		
next_master_salt	**112**	**bslbf**
}		
}		
}		
if (traffic_protection_protocol == TKM_ALGO_ISMACRYP) {		
key_indicator_length	**8**	**uimbf**
key_indicator	**8*key_indicator_length**	**uimsbf**
if (next_traffic_key_flag == TKM_FLAG_TRUE) {		
key_indicator	**8*key_indicator_length**	**uimsbf**
}		
}		
if (traffic_protection_protocol == TKM_ALGO_DCF) {		
key_identifier_length	**8**	**uimsbf**

key_identifier	**8*key_identifier_length**	**bit string**
}		
encrypted_traffic_key_material_length	**8**	**uimsbf**
encrypted_traffic_key_material	**8*encrypted_traffic_key_material_length**	**bslbf**
if (next_traffic_key_flag == TKM_FLAG_TRUE) {		
next_encrypted_traffic_key_material	**8*encrypted_traffic_key_material_length**	**bslbf**
}		
reserved_for_future_use	**4**	**bslbf**
traffic_key_lifetime	**4**	**uimsbf**
if (timestamp_flag == TKM_FLAG_TRUE) {		
Timestamp	**40**	**mjdutc**
}		
if (access_criteria_flag == TKM_FLAG_TRUE) {		
reserved_for_future_use	**8**	**bslbf**
number_of_access_criteria_descriptors	**8**	**uimsbf**
access_criteria_descriptor_loop() {		
access_criteria_descriptor()		
}		
}		
if (program_flag == TKM_FLAG_TRUE) {		
program_selectors_and_flags {		
reserved_for_future_use	**7**	**bslbf**
permissions_flag	**1**	**uimsbf**
}		
if (permissions_flag == TKM_FLAG_TRUE) {		
permissions_category	**8**	**uimsbf**
}		
if (service_flag == TKM_FLAG_TRUE) {		
encrypted_PEK	**128**	**bslbf**
}		
program_CID_extension	**32**	**uimsbf**
program_MAC	**96**	**bslbf**
}		
if (service_flag == TKM_FLAG_TRUE) {		
service_CID_extension	**32**	**uimsbf**
service_MAC	**96**	**bslbf**
}		
}		

Reserved_for_future_use – These bits are reserved for future use, and SHALL be set to zero when not used.

5.5.1 Coding and Semantics of Attributes

Section 7 introduces the coding and semantics of all Attributes common between the DRM Profile and the Smartcard Profile. Any DRM Profile specific attributes are introduced below.

next_traffic_key_flag – indicates whether or not the Short Term Key Message contains the next traffic key material:

TKM_FLAG_FALSE	The Short Term Key Message contains only the current traffic key material.
TKM_FLAG_TRUE	The Short Term Key Message contains both the current and the next traffic key material.

The next traffic key material SHALL be included at least 1 second before it becomes current. This is to enable the devices to process the traffic key material and put the necessary security associations in place before the media packets that are encrypted with the next traffic encryption key start arriving.

The above time SHALL be relative to the moment of transmission of the key stream messages.

If PEK is used to protect the traffic key material, then next traffic key material that protects a program different from the current program SHALL NOT be included.

timestamp_flag – indicates whether or not the STKM contains a timestamp:

TKM_FLAG_FALSE	The STKM does not contain a timestamp.
TKM_FLAG_TRUE	The STKM contains a timestamp.

program_flag – indicates whether or not the program key layer is present in the Short Term Key Message:

TKM_FLAG_FALSE	The PEK is not present, i.e. the optional program key layer is not used for the service.
TKM_FLAG_TRUE	The PEK is present, i.e. the optional program key layer is used for the service.

<program_flag> and <service_flag> SHALL NOT both be 0. All other combinations are allowed, indicating that either one or both of the key layers are present.

service_flag – indicates whether or not the service block is present in the Short Term Key Message:

TKM_FLAG_FALSE	The SEK is not present, i.e. the optional service key layer is not used for the service.
TKM_FLAG_TRUE	The SEK is present, i.e. the optional service key layer is used for the service.

<program_flag> and <service_flag> SHALL NOT both be 0. All other combinations are allowed, indicating that either one or both of the key layers are present.

security_parameter_index – provides the link to the IPsec ESP header:

Upon reception of a protected IP packet, the terminal SHALL use the security parameter index (SPI) to identify (look up) the correct security association and thereby find the decryption and authentication keys to be used for the received IPsec ESP packet. The SPI value SHALL be in the range 0x00000100 – 0xFFFFFFFF. An incoming ESP packet containing the SPI value specified in this field SHALL use the keymaterial provided in the encrypted traffic key material field as keymaterial for the decryption operation.

next_security_parameter_index – provides the link to the IPsec ESP header:

This field is present in the packet only if next traffic key flag is set to true. This field then contains the IPsec SPI value corresponding to the next_encrypted_traffic_key_material field. The value of the SPI SHALL be in the range 0x00000100 – 0xFFFFFFFF. An incoming ESP packet containing the SPI value specified in this field SHALL use the keymaterial provided in the next encrypted traffic key material field as keymaterial for the decryption operation.

master_key_index_length – provides the length of the master_key_index field

This field gives the length of the master_key_index field in bytes.

master_key_index – provides the link to the SRTP header:

Upon reception of a protected RTP packet, the terminal SHALL use the master key index (MKI) to identify (look up) the correct security association and thereby find the decryption and authentication keys to be used for a received SRTP packet.

This field is a sequence of Octets. The sequence consists of master_key_index_length bytes. The bytes are in the same order that they will be in an SRTP packet and SHALL be in SRTP [RFC3711] network byte-order when extracting the MKI value.

next_master_key_index_flag – specifies if the master key index (MKI) for the next TEK is explicitly included in the SRTP parameters (as the next_master_key_index field). In the case that the next_master_key_index is not present in the message, the value of current MKI+1 SHALL be assumed. In the case when the next_traffic_key_flag is false there is no information related to the next traffic key included in the message and this parameter does not apply.

next_master_salt_flag – specifies if the next SRTP master salt value corresponding to the next TEK is explicitly included in the SRTP parameters (as the next_master_salt field). In the case that the next_master_salt is not present in the message, the same value as for the current master salt SHALL be assumed. In the case when the next_traffic_key_flag is false there is no information related to the next traffic key included in the message and this parameter does not apply.

master_salt_flag – specifies if the master salt is included in the SRTP parameters. In the case that the master salt is not present in the message, a NULL value consisting of 112 0-bits SHALL be assumed.

master_salt – SRTP master salt that is used along with the master key to derive SRTP session keys as defined by SRTP [RFC3711].

next_master_key_index – provides the link to the SRTP header:

This field is present in the packet only if the next_traffic_key_flag and the next_master_key_index_flag are both set to true. This field then contains the SRTP MKI value corresponding to the next_encrypted_traffic_key_material field. An incoming protected RTP packet containing the MKI value specified in this field SHALL use the key material provided in the next encrypted traffic key material field as key material for the decryption operation.

next_master_salt – next value of the SRTP master salt that is used along with the next master key to derive SRTP session keys as defined by SRTP [RFC3711].

This field is present in the packet only if the next_traffic_key_flag and the next_master_salt_flag are both set to true. This field then contains the SRTP master salt value corresponding to the next_encrypted traffic key material field. An incoming protected RTP packet containing the next MKI value SHALL use the next master salt value provided in this field during the SRTP session key derivation.

key_indicator – value of the KeyIndicator used to identify the TEK transported in the STKM. This is used to identify the particular TEK key needed to decrypt AUs (as indicated in the OMABCASTAUHeader).

key_identifier_length – indicates the length in bytes of the key_identifier. For ISMACryp, key_indicator_length is signaled in SDP. For DRM Profile, the key_indicator_length is also signaled in STKM. Note that the Smartcard Profile STKM does not contain such field for ISMACryp. The key_indicator_length parameter is part of the Session Description Protocol (SDP) and is described in Section 10.2.

key_identifier – value of the identifier used to identify the TEK transported in the STKM. This is used to identify the particular TEK needed to decrypt DCF encoded files.

encrypted_traffic_key_material_length – is the length in bytes of the encrypted traffic key material.

The length of the traffic key material depends on the encryption and authentication algorithm, and is obtained by adding the respective key sizes. Encryption MAY require the clear-text key material to be padded.

encrypted_traffic_key_material – is the key material currently used for encryption and optional authentication of the traffic, encrypted using AES-128-CBC, with fixed IV 0, and with 0 padding in the last block, if needed.

If <program_flag> == TKM_FLAG_TRUE, the traffic key material is encrypted with the Program Encryption Key (PEK).

If <program_flag> == TKM_FLAG_FALSE and <service_flag> == TKM_FLAG_TRUE, the traffic key material is encrypted with the Service Encryption Key (SEK).

After decryption (and discarding any padding), the Traffic Encryption Key (TEK) and the Traffic Authentication Key (TAK) are obtained in a way that depends on the protocol used for traffic protection:

IPsec: If no traffic authentication is used, the IPsec encryption key is identical to the decrypted traffic key material (16 bytes).

If traffic authentication is used, IPsec encryption key and Traffic Authentication Seed (TAS) are obtained by splitting the decrypted traffic key material into two parts, where the IPsec encryption key is identical to the first 16 bytes, and the TAS is identical to the second 16 bytes. The TAK (20 bytes) is derived from the TAS, as described in Section 9.1.

SRTP: The master key is identical to the decrypted traffic key material and SHALL always be a 16-byte key. How the keys for traffic decryption and authentication are derived from the master key is defined by SRTP.

ISMACRYP: If no traffic authentication is used, the decrypted traffic key material is identical to the key used for the AES-CTR decryption and its length is 16 bytes. If authentication is used, the first 16 bytes of the decrypted traffic key material are used as the 128 bit master key (MK) together with the 112 bit master_salt (MS) to derive encryption and authentication keys as described by STRP.

For the DRM Profile, when traffic authentication is used, the MS, from which the actual salt keys are derived, SHALL be signalled via SDP. When traffic authenticatoin is not used, the salt keys as such are signaled in SDP.

Note that, for the Smartcard Profile, the MK is sent in the MIKEY STKM, and the MS is also sent in the MIKEY STKM.

DCF: If no traffic authentication is used, the encryption key is identical to the decrypted traffic key material (16 bytes).

If traffic authentication is used, the encryption key and the Traffic Authentication Seed (TAS) are obtained by splitting the decrypted traffic key material into two parts, where the encryption key is identical to the first 16 bytes, and the TAS is identical to the second 16 bytes. The authentication key (20 bytes) is derived from the TAS in the same way as specified for IPsec (see Section 9.1, Authentication for IPsec).

next_encrypted_traffic_key_material – is the encrypted key material used for encryption and optional authentication of the traffic after the current crypto period is over and the next crypto period starts. The structure of this attribute is the same as for the encrypted_traffic_key_material attribute.

timestamp – Field containing a timestamp at the point of sending the STKM. The timestamp SHALL be used as a reliable time of reception of the associated media stream for post-acquisition permissions. The device SHALL not use the timestamp as a reliable source for DRM time.

The format of the 40-bit mjdutc field is specified in Section 14. This 40-bit field contains the timestamp of the STKM in Universal Time, Co-ordinated (UTC) and Modified Julian Date (MJD). This field is coded as 16 bits giving the 16 LSBs of MJD followed by 24 bits coded as 6 digits in 4-bit Binary Coded Decimal (BCD).

As an example, 93/10/13 12:45:00 is coded as "0xC079124500".

permissions_flag – indicates whether or not permissions category is defined for the program:

TKM_FLAG_FALSE	No permissions category is defined.
TKM_FLAG_TRUE	Permissions category is defined.

permissions_category – indicates the permissions category for the program:

0x00	No permissions category, RO applies as such,
0x01...0x3F	Permissions_category is included in the post- acquisition permissions lookup.
0x40...0xEF	Reserved for future standardization.
0xFF	No post-acquisition content protection (export in plaintext is allowed)

If permissions_category is in the range 0x01...0x3F,

- In case of a RO that is not a BCRO, the device SHALL use as service_CID for post-acquisition permissions lookup the text string

  service_CID = “cid:” || stringtomakeitunique || "#S" || baseCID || "@" || HEX(service_CID_extension) || "_" || HEX(permissions_category)

and then apply the permissions specified in the service RO for this asset. Note that ‘service_CID’ shall be globally unique. Note further that because of the specification of ‘baseCID’ in the OMA BCAST SG, the global uniqueness is already guaranteed and, therefore, ‘stringtomakeitunique’ shall be the empty string.

- In case of BCRO, the device SHALL look up the permissions specified in the service BCRO for the asset that has a matching permissions_category field.

If permissions_category is in the (reserved for future standardization) range 0x40...0xEF, and the device does not support it, the device SHALL drop (i.e. ignore) all post-acquisition permissions (like play, redistribute etc.) indicated in the service RO, or if the device cannot do such permissions dropping, allow real-time rendering of the streaming content only (i.e. refuse to record the content, or to redistribute it in real time). Permissions_category has no impact on a Program RO. The permissions delivered in a Program RO apply as such.

If permissions_category = 0xFF, there is no need to protect the content after service protection has been removed; in other words, export in plaintext is allowed. This is comparable to setting protection_after_reception to 0x03. If protection_after_reception = 0x03 and permissions_category value is included in the STKM, the permission_category SHALL be set to 0xFF.

encrypted_PEK – is the Program Encryption Key (PEK) used within the current STKM to decrypt the traffic key material, encrypted using AES-128-CBC with a fixed IV equal to 0. The PEK is encrypted with the SEK.

program_CID_extension – is the extension of the program_CID which allows to identify the program key material that has been delivered to the device within a LTKM for a program.

Note that for BCRO, a binary, fixed-size version of the content ID (CID) is needed. This ID is called BCI in this specification.

The CID/BCI of the service key is constructed as:

program_CID = “cid:” || stringtomakeitunique || "#P" || baseCID || "@" || HEX(program_CID_extension)

program_BCI = hash("cid:" || stringtomakeitunique || "#P" || baseCID || "@") || program_CID_extension

The baseCID is a string value and is part of the service guide. Upon reception of a STKM, the terminal can assemble the program_CID/BCI and look up the PEK (wrapped inside a LTKM). Note that 'program_CID' shall be globally unique. Note further that because of the specification of 'baseCID' in the OMA BCAST SG, the global uniqueness is already guaranteed and, therefore, 'stringtomakeitunique' shall be the empty string.

The HEX() function is a hexadecimal presentation of the parameter containing hexadecimal characters 0-9 and a-f (in lowercase) with possible preceding zeros. As an example, for a 16 bit value 2748, HEX() returns "0abc". Note that two characters are always generated for each byte.

The hash function for the construction of program_BCI is SHA1-64. It doesn't depend on the contents of the STKM, and can thus be pre-computed.

program_MAC – is the HMAC-SHA-1-96 according to [RFC2104] and [RFC2404] calculated over all preceding fields of the Short Term Key Message. It is used to authenticate the relevant part of the STKM in case of pay-per-view, where a PEK from a LTKM for a program is used to directly decrypt the traffic key material.

In case the terminal is accessing the STKM with a LTKM for a program, the terminal SHALL compute the program MAC, and drop the message if authentication fails. In this case, <program_MAC> MAY also be used to detect and drop duplicates (it can be expected that a particular STKM is repeated multiple times, in order to keep access times short for terminals that newly start receiving a broadcast transmission).

In case the terminal is accessing the STKM with a LTKM for a service, it will not be able to compute the program MAC, and there is no need for it to do so.

service_CID_extension – is the extension of the service_CID which allows to identify the service key material that has been delivered to the device within a LTKM for a service.

Note that for BCRO, a binary, fixed-size version of the content ID (CID) is needed. This ID is called BCI in this specification.

The CID/BCI of the service key is constructed as:

service_CID ::= "cid:" || stringtomakeitunique || "#S" || baseCID || "@" || HEX(service_CID_extension)

service_BCI ::= hash("cid:" || stringtomakeitunique || "#S" || baseCID || "@") || service_CID_extension

The baseCID is a string value announced in the service guide (see Section 5.8). Upon reception of a STKM, the terminal can assemble the service_CID/BCI and look up the SEK (wrapped inside a LTKM). Note that 'service_CID' shall be globally unique. Note further that because of the specification of 'baseCID' in the OMA BCAST SG, the global uniqueness is already guaranteed and, therefore, 'stringtomakeitunique' shall be the empty string.

The hash function for the construction of service_BCI is SHA1-64. It doesn't depend on the contents of the STKM, and can thus be pre-computed.

If the permissions_category field is present and has a nonzero value, the Service_CID of the service is constructed as specified at description of the permissions_category field.

service_MAC – is the HMAC-SHA-1-96 according to [RFC2104] and [RFC2404] calculated over all preceding fields of the Short Term Key Message. It is used to authenticate the STKM with SAK in case of subscription, where a SEK from a LTKM for a service is used to decrypt the PEK and further decrypt the traffic key material.

In case the terminal is accessing the STKM with a LTKM for a service, the terminal SHALL compute the service MAC, and drop the message if authentication fails, i.e. if the computed MAC doesn't correspond to <service_MAC>. In this case,

<service_MAC> MAY also be used to detect and drop duplicates (it can be expected that a particular traffic key message is repeated multiple times, in order to keep access times short for terminals that newly start receiving a broadcast transmission).

In case the terminal is accessing the STKM with a LTKM for a program, it need not to compute the service MAC.

5.5.2 Authentication for STKMs for OMA DRM 2.0 Extensions

A STKM can contain two MAC fields: The program MAC and the service MAC. If only one MAC field would be used, the authentication key could only be renewed when both SEK and PEK change at the same time. Having two MAC fields and two authentication keys makes it possible to authenticate the message and check for its integrity while only having one key set. The Service Authentication Key (SAK) and the Program Authentication Key (PAK) will be derived from the Service Authentication Seed and the Program Authentication Seed respectively which are transmitted together with the encryption keys in the LTKMs. (How this is carried in the BCRO and RO is explained in [XBS DRM extensions-v1.0], Section A.12.2.1 and A.12.2.2, respectively.) A RO for a service will contain Service Encryption and Authentication Keys (SEAK) and a RO for a program will contain Program Encryption and Authentication Keys (PEAK).

To obtain the SAS or PAS from the BCRO the encrypted SEAK/PEAK is decrypted with the Inferred Encryption Key (IEK, see Section 4.1 in [XBS DRM-extensions-v1.0]):

$$SAS = LSB_{128}(D\{IEK\}(E\{IEK\}(SEAK))$$

$$PAS = LSB_{128}(D\{IEK\}(E\{IEK\}(PEAK))$$

The authentication key is generated from the authentication seed:

$$SAK = f_{auth}\{SAS\}(CONSTANT\_SAK)$$

$$PAK = f_{auth}\{PAS\}(CONSTANT\_PAK)$$

where :

CONSTANT_SAK = 0x02020202020202020202020202020202 (120 bit)

CONSTANT_PAK = 0x01010101010101010101010101010101 (120 bit)

The SAK or PAK is used in the MAC generation / verification of the STKM. The algorithm used to calculate the MAC field is HMAC-SHA1-96 according to [FIPS198] and [RFC2104], using authentication keys of 160 bit in both cases.

The function f_{auth} consists of several steps:

1. Denote by PRF{key}(text) as the AES-XCBC-MAC-PRF with output blocksize 128 bits as defined by IPsec WG in IETF. Please note:
   - ◊ Refer to [RFC3566] for the AES-XCBC-MAC-PRF based key generation function.
   - ◊ Refer to [RFC3664] for the requirement NOT to truncate the generated key material.
2. Apply the generated input key according to ideas of IKEv2 to generate authentication key. Define a key generator function f-kg{key}(constant). Keying material will always be derived as the output of the negotiated PRF algorithm.. PRF^+ describes the function that outputs a pseudo-random stream of n blocks based on the inputs to a PRF as follows:

   $$T1 = AES\_XCBC\_MAC\_PRF\{AS\}(CONSTANT \,||\, 0x01)$$

   $$T2 = AES\_XCBC\_MAC\_PRF\{AS\}(T1 \,||\, CONSTANT \,||\, 0x02)$$

   ....

   $$Tn = AES\_XCBC\_MAC\_PRF\{AS\}(T1 \,||\, CONSTANT \,||\, n)$$

where *AS* is the appropriate authentication seed (be it TAS, PAS, SAS or RIAK) and *CONSTANT* is the appropriate constant as described in this section, Section 9.1 and [XBS DRM extensions-v1.0]. The amount of blocks to derive is defined by the amount of key material needed, i.e. n is the amount of needed key bits divided by 128 and rounded up.

This means that if 160 bits were needed then PRF*() would be computed as:

$$T1 \,||\, T2 = PRF^{+}\{K\}(S)$$

3. The 160 bit authentication key is taken from the generated key material as follows:

$$AK = MSB_{160}(T1 \,||\, T2)$$

The generated authentication key is applied as described in this section and Section 9.1.

5.6 Layer 4: Traffic Encryption

Layer 4 corresponds to the BCAST 4-layer key hierarchy model and describes how to protect data. The services considered for the BDS delivery are streaming sessions and file downloads, for which service and content protection is described in the following sections.

5.6.1 Streaming Delivery

5.6.1.1 Service Protection of Streams

Broadcast streams that are signalled as having service protection are securely delivered to authorized users. The service protection mechanism protects streams only at the delivery time. The streamed content after the removal of service protection can be stored in clear if post delivery content protection is not signalled.

For DRM Profile, Layer 4 protection is provided through encryption. The encryption mechanisms are described in Section 9 of this document.

5.6.1.2 Content Protection of Streams

Broadcast streams that are signalled (through protectionType value in Service Guide and protection after reception value in STKM) as having content protection may be recorded as defined in this specification. However, for recorded material having content protection, appropriate rights need to be obtained via Rights Issuer.

For terminals using the DRM Profile, the appropriate key material can be requested based on the Program or Service ID.

As the content encryption key provides access to recorded content stored in the terminal, preventing unauthorized access to content encryption key is extremely important. However, the exact storage and handling of content encryption key in the device is specific to an implementation.

5.6.2 File Delivery

5.6.2.1 Service Protection of Files

BCAST terminal and server MAY support download protection using DCF.

The same mechanism can be used to protect PDCF files. This is optional for both terminal and server.

Service protection of download data uses IPsec or DCF encryption protocol. In case of DCF encryption protocol, DCF file is used as a container for ciphered file data. The DCF container also identifies the keys used in protecting the data.

Each file is encrypted using a single TEK, as explained in Section 9.4.

The correct TEK for decrypting and verifying the integrity of the download data is indicated by the KeyID field in the Key Info box.

For the DRM Profile, KeyID takes its value as follows:

- If SEK is used for protecting STKMs, KeyID is defined as the base64 encoded concatenation (service_CID_extension || ";" || TEK ID).
- If PEK is used in protectig STKMs and the PEK is not protected by an SEK, KeyID is defined as the base64 encoded concatenation (program_CID_extension || ";" || TEK ID).
- If PEK is used in protecting STKMs and the PEK is protected by an SEK, KeyID is defined as the base64 encoded concatenation (service_CID_extension || ";" || program_CID_extension || ";" || TEK ID).

The RightsIssuerURL MAY be indicated within the Key Info box in the KeyIssuerURL, or MAY be indicated in the RightsIssuerURL in the OMADRMCommonHeaders box.

5.6.2.2 Content Protection of Files

When using the DRM Profile, Content Protection for files SHALL follow OMA DRM 2.0 specification [DRMCF-v2.0].

For audio or video content either the PDCF or the DCF formats SHALL be used.

5.7 Recording

Please refer to Section 8 for details on recording.

5.8 SG Signaling

SG signalling is described in [BCAST10-SG]. The relevant fragments linking SG signalling to service and content protection are the Service, Content, Access, Purchase Item, Purchase Data and Purchase Channel Fragments.

The Access Fragment describes how the service may be accessed during the validity time of the access fragment. The fragment links to Session Description and indicates the delivery method. KeyManagementSystem element identifies the type of KMS that can be used to contact the RI. The value of this element for DRM Profile is oma-bcast-drm-pki. The associated attributes are ProtectionType and RightsIssuerURI. The ProtectionType attribute specifies the protection type (service protection only, content protection only or both service & content protection) offered by the DRM Profile. The RightsIssuerURI specifies the URI of RightsIssuer that should be contacted to obtain ROs.

The Purchase Channel Fragment represents a system from which access and content rights can be purchased by the terminal. The associated attribute RightsIssuerURI specifies the identity of the rights issuer associated with the BSM. For DRM Profile, RightsIssuerURI SHALL be specified.

For devices that support an interaction channel, the PurchaseURL in the Purchase Channel Fragment specifies the URL to which the interactive service provisioning messages defined in [BCAST10-Services] are to be addressed. An interactive service ordering procedure will result in the delivery of a ROAP trigger to the device, which in turn uses the trigger to initiate a Rights Object Acquisition as specified in [DRMDRM-v2.0].

For broadcast-only devices, the Purchase Channel contains information on how to initiate an out-of-band purchase. For an overview of the purchase message flow, see [BCAST10-Architecture].

The Purchase Item fragment contains the GlobalPurchaseItemID, used to refer to the services, service bundles or pay-per-view programs when subscribing via the BSM.

The Purchase Data fragment contains additional information on how the purchase item can be subscribed to. Depending on the chosen purchase data, the resulting LTKM will contain different access rights.

To identify the asset in the RO needed for a service or a program, the following parameter is used in SG: baseCID. The parameter baseCID is announced in the Service fragment and Content fragment of the SG.

5.9 Usage Metering for DRM Profile

Extensions to OMA DRM v2.0 for usage metering appear in the OMA DRM v2.0 Extensions for Broadcast Support document.

6. Smartcard Profile

Caution: The term "Smartcard" is used in this document in the restricted sense specified in the definition provided in Section 3.2.

6.1 Introduction

The Smartcard Profile is based on an existing security framework for service protection defined for broadcast/multicast services based on smartcards defined by 3GPP MBMS [3GPP TS 33.246 v7] and may include the key provisioning mechanism defined for 3GPP2 BCMCS [3GPP2 S.S0083-A]. The solution requires an interactive channel to obtain key material.

Two variants of the Smartcard Profile are defined in this specification: the (U)SIM Smartcard Profile and the (R-)UIM/CSIM Smartcard Profile. The two variants differ in the way that the Layer 1 key(s) are established (see Section 6.5 but are otherwise the same (Layers 2, 3 and 4).

The following sections describe the four layers of the 4-layer model key hierarchy, as well as the key provisioning required to access the first layer.

Section 6.4 briefly describes the provisioning of the SmartCard Key (SCK). Section 6.5 describes Subscriber Key Establishment. Section 6.6 details the structure and delivery of the LTKM while Section 6.7 describes those of the STKM. Section 6.8.1 and Section 6.8.2 describe how to protect content in case of streaming and file delivery respectively for both service and content protection. Recording aspects are detailed in Section 6.9 while SG signalling is explained in Section 6.10.

6.2 Relationship between MBMS Security and the Smartcard Profile

As stated above, the Smartcard Profile uses the key management defined by 3GPP MBMS [3GPP TS 33.246 v7]. To clarify the relationship between the two specifications the following tables provide a mapping between the keys and key IDs used in [3GPP TS 33.246 v7] and this specification. The remainder of this specification uses the terminology defined for the Smartcard Profile.

Table 6: Mapping between MBMS keys and Smartcard Profile Keys

MBMS key	Smartcard Profile key
MBMS User Key (MUK)	Subscriber Management Key (SMK)
MBMS Registration Key (MRK)	Subscriber Request Key (SRK)
MBMS Service Key (MSK)	Service Encryption Key (SEK) [1]
MBMS Traffic Key (MTK)	Traffic Encryption Key (TEK)

[1] The Smartcard also supports the concept of a Program Encryption Key (PEK); see Section 11.1 for further details.

Table 7: Mapping between MBMS key IDs and Smartcard Profile Key IDs

MBMS		Smartcard Profile	
Key ID	**Construction**	**Key ID**	**Construction**
MUK ID	MUK ID is received by combining IDi and IDr, where IDi is the identity of the initiator and the IDr is the identity of the responder. IDr is Bootstrapping – Transaction ID (B-TID) and IDi is the Network Application Function ID (without the Ua security	SMK ID	As for MUK ID

	protocol identifier), as defined in [3GPP TS 33.246 v7].		
MRK ID	The B-TID is used as the username when MRK is used as the password within HTTP digest and so can be thought of as the MRK ID (although it is never defined as such within the MBMS specification). See [3GPP TS 33.246 v7] for further details.	SRK ID	As for MRK ID
MSK ID	MSK ID is 4 bytes long and with byte 0 and 1 containing the Key Group part, and byte 2 and 3 containing the Key Number part. Every MSK is uniquely identifiable by its Key Domain ID and MSK ID where Key Domain ID = Mobile Country Code || Mobile Network Code, and is 3 bytes long (see [3GPP TS 33.246 v7] for further details).	SEK/PEK ID	As for MSK ID
MTK ID	MTK ID is 2 bytes long sequence number and is used to distinguish MTKs that have the same Key Domain ID and MSK ID. Every MTK is uniquely identifiable by its Key Domain ID, MSK ID and MTK ID (see [3GPP TS 33.246 v7] for further details).	TEK ID	As for MTK ID

The Smartcard Profile BSM provides the functionality that in MBMS is provided by the MBMS Broadcast-Multicast Service Centre (BM-SC) security functions. As such the Smartcard Profile BSM SHALL support the following MBMS BM-SC security functions:

- Key Management function
    - Key Request function
    - Key Distribution function
- Membership function

as defined in [3GPP TS 33.246 v7], with the modifications described in this specification. Note that the Session and Transmission functionality is not required to be supported by the BSM as this functionality is provided by the BSDA.

MBMS uses the Generic Bootstrapping Architecture (GBA) [3GPP TS 32.220] to establish a MUK and MRK between the BM-SC, an instance of a GBA Network Application Function (NAF), and the USIM/terminal. GBA requires the implementation of a Bootstrapping Server Function (BSF) to enable the bootstrapping procedure required to establish MUK and MRK. Within this specification the (U)SIM Smartcard Profile BSM is assumed to support BSF functionality requied to establish SMK and SRK, which are equivalent to the MBMS MUK and MRK respectively. However, the BSF may be shared between the BSM NAF and NAFs for other services.

The (R-)UIM/CSIM Smartcard Profile derives SMK and SRK from the SmartCard Key (i.e. RK) pre-provisioned on the (R-)UIM/CSIM and in the BCMCS Subscription Manager (SM) function. Within this specification the (R-)UIM/CSIM Smartcard Profile BSM is assumed to support the SM functionality requied to establish SMK and SRK.

Smartcard Profile terminals SHALL support the key management functionality specified for MBMS terminals, as defined in [3GPP TS 33.246 v7], with the modifications described in this specification.

(U)SIM Smartcard Profile Smartcards (i.e. (U)SIMs) SHALL support all key management functionality specified for MBMS capable (U)SIMs, as defined in [3GPP TS 31.102 v6] and MAY support the additions and modifications described in this specification.

(R-)UIM/CSIM Smartcard Profile Smartcards SHALL support the key management functionality specified for MBMS capable (U)SIMs related to the processing of MBMS MSK and MTK messages , as defined in [3GPP TS 33.102] and MAY support the additions and modifications described in this specification. (R-)UIM/CSIM Smartcard Profile Smartcards SHALL

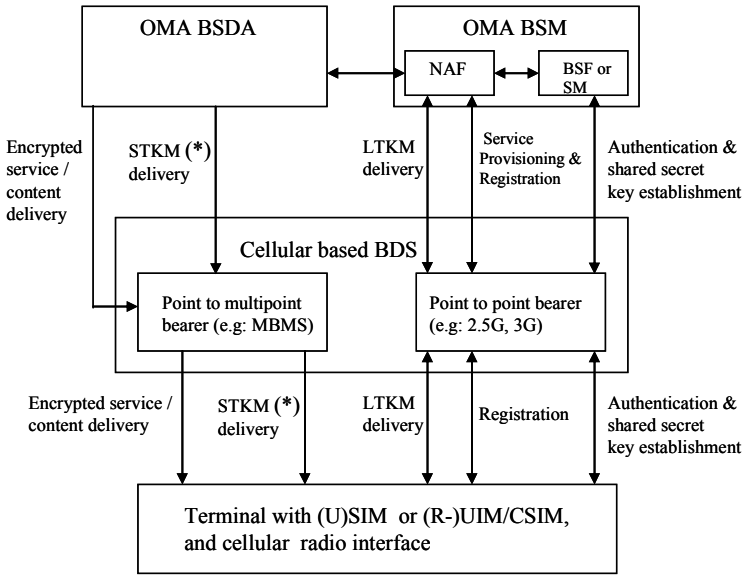
support the functionality defined in [3GPP2 S.S0083-A] to derive the Temporary Key (TK) and Authentication Key (Auth-Key), which correspond to the SMK and SRK respectively, from the pre-provisioned Registration Key (RK).

6.3 Use of the Smartcard Profile for Various BDS Architectures

Different BDS architectures that can be used with the Smartcard Profile using MBMS key management. The Smartcard Profile is applicable to cellular based BDS architectures, which natively can use a point-to-multipoint or point-to-point bearer, and also to broadcast-only BDS architectures with the additional support of a cellular interaction channel.

6.3.1 Smartcard Profile using a pure Cellular Based BDS

In the pure cellular based BDS case, both multicast/broadcast and unicast bearers are available.



(*) Short-term key message may be delivered over the point-to-point bearer instead.

Figure 2 – Pure Cellular based BDS Scenario

As a clarifying note for the scenario shown in, there may exist one or more broadcast service providers, each represented by a separate instance of the BSM. One of these broadcast service providers also serves as the cellular BDS network operator, such that the BDS-SD is functionally integrated with the BSDA and BSM.

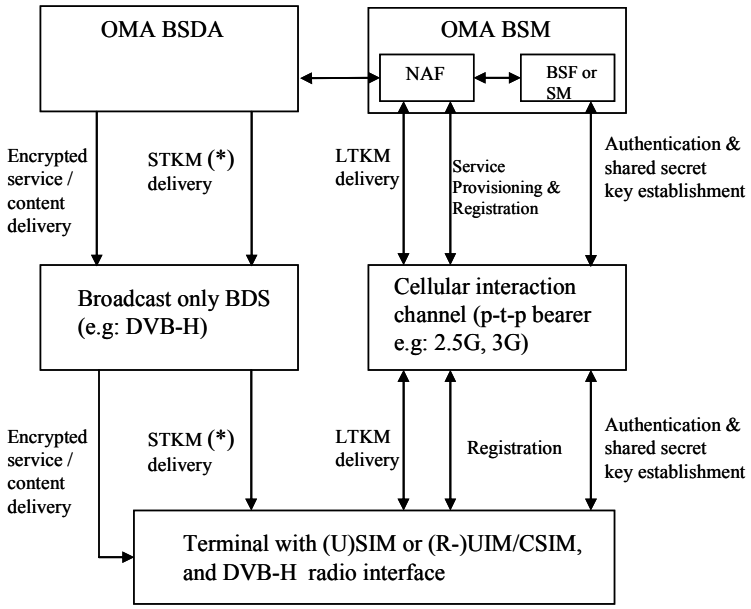
A basic overview of the operations of the Smartcard Profile in this scenario is as follows:

- Broadcast Service Discovery: A user selects a protected service on the BCAST service guide available over the cellular based BDS.
- Authentication and Subscriber Key Establishment: This corresponds to Layer 1 of the BCAST 4-layer model key hierarchy and takes place via the point-to-point bearer. See Section 6.5 for details of how Authentication and Subscriber Key Establishment is handled for the two variants of the Smartcard Profile.
- LTKM Delivery: This corresponds to Layer 2 of the BCAST 4-layer model key hierarchy. After Layer 1 keys have been established between the BSM and the Smartcard/terminal and the terminal has subscribed to one or more services, the terminal may request the relevant LTKMs from the BSM, or alternatively, the BSM may send them automatically. LTKMs are delivered by the BSM to the terminal, via the point-to-point bearer. The construction, delivery and processing of Smartcard Profile LTKMs is explained in Section 6.6.
- STKM Delivery: This corresponds to Layer 3 of the BCAST 4-layer model key hierarchy. STKMs are used to deliver the TEKs and may be delivered over the point-to-multipoint bearer or the point-to-point bearer to the terminal. The construction, delivery and processing of Smartcard Profile STKMs is explained in Section 6.7.

- Access to protected content: This corresponds to Layer 4 of the BCAST 4-layer model key hierarchy. The cellular based BDS delivers a service, e.g. a file download or streaming session, which may be transmitted over the cellular network in unicast, multicast or broadcast mode.

6.3.2 Smartcard Profile using a broadcast BDS and cellular interactive channel

In a mixed or hybrid scenario (e.g. IPDC over DVB-H + cellular interaction channel) a pure broadcast BDS is complemented with an additional interaction channel given by a cellular network.



(*) Short-term key message may be delivered over the cellular interaction channel instead.

Figure 3 – Broadcast-only BDS with Cellular Interaction Channel Scenario, using either GBA or derivation of Layer 1 Key from RK

The same clarification note as indicated for applies here as well.

A basic overview of the operation of the BCAST Smartcard Profile in this scenario can be the following:

- Broadcast Service Discovery: As for cellular BDS above but available over the broadcast BDS (e.g. IPDC over DVB-H).
- Authentication and Registration: As for cellular BDS above i.e. via the cellular interaction channel.
- LTKM Delivery: As for cellular BDS above i.e. via the cellular interaction channel
- STKM Delivery: As for cellular BDS above but STKMs may be delivered over the broadcast only BDS (e.g. IPDC over DVB-H) or via the cellular interaction channel.
- Access to protected content: As for cellular BDS above but available over the broadcast BDS (e.g. IPDC over DVB-H).

6.4 Use of Pre-provisioned Keys

The Smartcard Profile uses a pre-provisioned secret key - the "SmartCard Key" (SCK) - stored on the Smartcard to establish the shared Layer 1 key(s) between the BSM and the Smartcard/terminal, as described in Section 6.5. The SCK corresponds to the authentication key "K" stored on 3GPP compliant USIMs [3GPP TS 31.102 v6] or ISIMs [3GPP TS 31.103 v6], to the authentication key "Ki" on 3GPP compliant SIMs [3GPP TS 31.111 v6], and to the key "RK" on 3GPP2 compliant (R-)UIM/CSIMs [3GPP2 C.S0023-C] or CSIM [3GPP2 C.S0065-0].

How the SCK is provisioned is out of scope of this specification.

6.5 Layer 1: Subscriber Key Establishment

6.5.1 Subscriber Key Establishment using a (U)SIM

This layer enables the establishment of two shared keys to secure communiction between the BSM and the terminal: The Subscriber Management Key (SMK), which is used to protect the delivery of SEK/PEKs within LTKM from the BSM to the terminal, and the Subscriber Request Key (SRK), which is used to secure communication between the terminal and the BSM. The SMK corresponds to the MBMS User Key (MUK) while the SRK corresponds to the MBMS Request Key (MRK), where the MBMS keys are as defined in [3GPP TS 33.246 v7].

The (U)SIM Smartcard Profile is based on MBMS security and therefore SMK and SRK are derived by running the GBA bootstrap procedure, as defined in Section 6.1 “Using GBA for MBMS” of [3GPP TS 33.246 v7]. The relationship of the BSM to the GBA NAF and BSF elements is described in Section 4.5.2.

6.5.2 Subscriber Key Establishment using a (R-)UIM/CSIM

BCMCS uses pre-provisioning to establish a unique 128-bit Registration Key (RK) in the (R)-UIM/CSIM and the Subscription Manager (a functional entity, SM) prior to providing service. This is referred to as the SmartCard Key (SCK) within this specification. The SM performs accounting, authentication and authorization for BCMCS. The SM also calculates the "Auth-Key", derived from the RK, which is used to secure communication between the terminal and the BSM. (The RK is functionally equivalent to the SMK and, therefore, the MUK in MBMS. The "Auth-Key" is functionally equivalent to the SRK and, therefore, the MRK in MBMS.) The SM may be the subscriber’s home AAA (H-AAA) or an independent entity.

6.6 Layer 2: Service Provisioning and LTKM Delivery

To access a protected service a terminal must obtain the necessary LTKM(s). To receive the LTKM(s) the terminal must subscribe to or purchase a BCAST purchase item. Subscription MAY be achieved using one of the following Service Provisioning messages, as defined in [BCAST10-Services]:

- “Service Request”
- “Token Purchase Request”

Alternatively, subscription MAY be achieved via other channels, e.g. the user may subscribe to the service via a web portal/shop (see Section 6.10.3 for more details).

The BSM SHALL authenticate the sender of the Service Provisioning and/or Registration message(s) sent by the terminal, by following the HTTP DIGEST authentication procedure defined in section 6.3.2.1A of [3GPP TS 33.246 v7], e.g. the BSM shall ensure that a valid SRK is used for in the HTTP DIGEST authentication. If authenticated is successful the request SHALL be acknowledged using an HTTP 200 OK message. Note that the requirement for a valid SRK also ensures that a valid SMK has been established.

Following a successful service registration, the LTKMs corresponding to the services to which the terminal is subscribed SHALL be delivered by the BSM to the terminal as a result of a push or pull procedure as defined in sections 6.3.2.2 and 6.3.2.3 of [3GPP TS 33.246 v7]. This provides support for the scenarios described below:

- The BSM MAY push an LTKM to the terminal in order to provide a new SEK/PEK to a terminal or to update the Key Validity data (KV) associated with a SEK or PEK that has previously been delivered. Pushing LTKMs to registered terminals allows the BSM to spread the delivery of SEKs/PEKs required by a large number of users to manage network congestion, e.g. the BSM determines when the LTKM is pushed to the terminal. Pushing an LTKM to a terminal to update the KV data provides a means to extend or reduce the lifetime of a SEK or PEK.

  The “Registration” message, as defined in [BCAST10-Services], SHALL be sent by the terminal after the application is started and the terminal re-establishes connectivity to the interactive network associated with its service provider. This message indicates to the BSM that the terminal is available to receive any LTKMs that it may

have missed while it was unreachable. Note that when a terminal establishes connectivity with an interactive network that is not associated with its service provider, e.g. in the case of roaming between cellular networks, the terminal MAY send the Registration message.

The sending of the "Registration" message also ensures that the terminal establishes the necessary IP connectivity required to enable the BSM to push the LTKM(s) over UDP. Note that the "Registration Request" message corresponds to the MBMS "User Service Registration" message, as defined in [3GPP TS 33.246 v7].

When the BSM wishes to push an LTKM, if the network is able to retrieve a valid IP address for the terminal, the LTKM can be pushed over UDP. Otherwise, the BSM can use the BSM Solicited Pull Procedure Initiation over SMS Bearer feature described below to deliver the LTKM.

- The terminal SHALL request the LTKM associated to a particular service when the terminal realises that it has missed an LTKM update, e.g. due to being out of coverage. The terminal SHALL use the "LTKM Request" message, as defined in [BCAST10-Services], to request the missed LTKM. Note that the "LTKM Request" message corresponds to the MBMS "MSK Request message", as defined in [3GPP TS 33.246 v7].

- The BSM MAY trigger the terminal to request the current LTKM for a particular service. This process SHALL be as defined for the "BM-SC solicited pull procedure" in section 6.3.2.2.4 of [3GPP TS 33.246 v7]. The solicited pull procedure can be used to provide a means to update the terminal with a new SEK when:

    - the SMK is no longer valid, e.g. the BSM can respond to the "Registration Request" message from the terminal with an HTTP 401 WWW Authenticate message, thereby initiating a new run of GBA;

    - the terminal is not trusted to provide acknowledgment of LTKM delivery, e.g. with the solicited pull procedure the BSM can assume successful delivery if the terminal does not repeat the "Registartion Request" message;

    - the UE has just registered to a User Service, and needs to initiate the delivery of the SEK/PEK

When a terminal has successfully unsubscribed from a BCAST service using an "Unsubscribe Request", as defined in [BCAST10-Services], or via other means like a web shop, the BSM MAY invalidate the SEK/PEKs on the terminal that are associated with the relevant purchaseItemID and that are not used by any other purchase items to which the device is subscribed. The BSM invalidates SEKs/PEKs by sending an LTKM with invalid Key Validity data, i.e. the lower bound is greater than the upper bound, where the bounds define the allowed range of either TEK or TimeStamp values.

Note that once a purchaseItemID has expired, i.e. the content associated with this purchaseItemID has been broadcast, the terminal SHALL remove the purchaseItemID from all subsequent "RegistrationRequest" and "Deregistration request" messages, as defined in [BCAST10-Services], sent to the BSM.

Table 8 below shows the MIKEY message format used for LTKMs in the Smartcard Profile. The message structure SHALL be identical to the MIKEY message used by MBMS to deliver the MBMS Service Key (MSK) (defined by [3GPP TS 33.246 v7]) apart from the addition of the (optional) EXT BCAST payload as defined in [draft-mikey-oma]. The EXT BCAST payload is described in Section 6.6.2.

Table 8: The Logical Structure of the MIKEY Message used for LTKMs. The use of brackets is according to section 1.3 of RFC 3830 (MIKEY)

Common HDR
EXT MBMS
{EXT BCAST}
TS*
MIKEY RAND
IDi

IDr
{SP}
KEMAC

* The TS (timestamp) field in the LTKM pertains strictly to the LTKM, for the purpose of LTKM replay detection. It is not the same timestamp that exists in the STKM (the latter serves for replay detection of STKMs).

The structure of the EXT MBMS payload, depicted in Table 8 above, is defined in Section 6.4.4 "General extension payload" of [3GPP TS 33.246 v7] and reproduced below for convenience. For the Smartcard Profile LTKM the EXT MBMS payload is the extension payload defined in [3GPP TS 33.246 v7] for use with MBMS MIKEY MSK message.

Table 9: The Logical Structure of the EXT MBMS Payload

Key Domain ID sub-payload
Key Type ID sub-payload (MSK ID)

All fields in the Smartcard Profile LTKM, with the exception of the EXT BCAST payload and the modifications defined in this section, SHALL be populated as defined in [3GPP TS 33.246 v7] for the MBMS MSK message. The mappings described in Section 6.2 for the Smartcard Profile parameters SHALL be used, i.e. SEK/PEK ID is mapped to MSK ID and SEK/PEK is mapped to MSK.

All fields in the BCAST MIKEY LTKM SHALL be populated as defined in [3GPP TS 33.246 v7] with the above mapping for BCAST parameters.

The EXT BCAST for LTKMs SHALL be populated as defined in Section 6.6.2. If the LTKM message includes the EXT BCAST payload and the security_policy_ext_flag is set to LTK_FLAG_TRUE or the consumption_reporting flag is set to LTK_FLAG_TRUE, then the Key Validity data subfield of the KEMAC payload in the LTKM defines the Key Validity interval for the SEK/PEK in terms of a specified interval of STKM Timestamp values:

From (32-bits): Lower limit of STKM Timestamp ("TS low")

To (32bits): Upper limit of STKM Timestamp ("TS high")

It should be noted that the use of STKM Timestamps to define Key Validity, as described above, is a deviation from the MBMS Security specification.

SP is present only when the LTKM addresses a streaming service which uses SRTP. If the LTKM message does not include the EXT BCAST payload or when the security_policy_ext_flag is set to LTK_FLAG_FALSE and the consumption_reporting flag set to LTK_FLAG_FALSE, the standard Key Validity data is constructed as per [3GPP TS 33.246 v7], i.e. the Key Validity data subfield in the KEMAC payload is defined in terms of sequence number interval (i.e. lower limit of MTK ID and upper limit of MTK ID, where MTK ID is a 16 bits identifier).

In the sections below, the following terms are used: (Note that, collectively, these apply to both LTKM and STKM processing)

Resending check: Resending LTKMs with the same key material is allowed provided the 32-bit counter timestamp in the TS field is increased, as mandated by MBMS [3GPP TS 33.246 v7].

Anti-replay processing or verification: This procedure is used to detect replay attacks for both LTKMs and STKMs, and for the case of STKM, presence of terminal intent to play back recorded content. This procedure operates by comparing the TS field in either the STKM or LTKM with the corresponding anti-replay counter in the Smartcard. Messages with a counter (TS field) less than or equal to the current counter raise a freshness failure. Depending on the security_policy_extension type, this failure results in a discard of the message or a detection of intent to play back recorded content.

Anti-replay verification failure (or freshness failure): This occurs when LTKM or STKMs contain a counter in TS field less than or equal to the current counter.

Replay protection: This is the function which processes the anti-replay verification.

Key Validity Data check: The Key Validity Data check verifies that the SEK/PEK key is still valid. This procedure uses the Key Validity Data field in the KEMAC payload of the LTKM. For MBMS key management the Key Validity Data is defined as an interval of sequence number (i.e. lower limit of MTK ID and upper limit of MTK ID). For BCAST key management (LTKM with a security-policy-extension in the OMA BCAST EXT payload), the Key Validity Data is defined as an interval of STKM timestamps (i.e. Lower limit of Timestamp ("TS low") and Upper limit of Timestamp ("TS high"). A failure occurs when the timestamp in the STKM received is higher than the TS high or lower than the TS low.

Message validation: The Message Validation check consists of the verification of integrity of the message, using the SMK for the LTKM and SEK/PEK for STKM. This procedure is described in the Section 7.1.1.6 for MSK messages and Section 7.1.1.8 for MTK messages of the [3GPP TS 31.102 v6].

Play-back counter: An internal counter in the USIM that contains the number of play-backs authorized.

SEK/PEK key group: A group of SEK/PEKs that are identified by the same Key group part of the SEK/PEK ID. The SEK/PEK key group is uniquely identifiable by its Key Domain ID and Key group part of the SEK/PEK ID.

Note: Although Section 6.6 pertains to LTKM, anti-replay detection and message validation procedures apply equally to processing of LTKM and STKM. Therefore references to STKM for these functions are also indicated in the above. Section 6.7.2 describes in detail the Smartcard Profile security policies which depend jointly on the contents of the LTKM and STKM.

6.6.1 BSM Solicited Pull Procedure Initiation over SMS Bearer

The first message in the BSM solicited pull procedure referenced in Section 6.6 is sent over UDP, requiring the terminal to have a valid IP address. In networks in which it cannot be guaranteed that terminals maintain a valid IP address the BSM MAY trigger a solicited pull procedure by sending an LTKM over SMS to the terminal. The LTKM SHALL be encoded according to this specification, with MSK ID Key Number part = 0, KEMAC Encr Data Len = 0, and V-bit in HDR not set. The SMS SHALL satisfy the following conditions:

- The SMS carries a WAP connectionless push (WDP/WSP encoding) as defined in [OMA Push] ;
- The WSP content type header contains the Content Type code registered by OMNA for 'application/mikey', i.e. the binary value 0x52 (to be assigned by [OMNA]);
- The WSP X-Wap-Application-Id header contains the binary code registered by OMNA for the PUSH Application ID identifying the BCAST Push client, as specified in [BCAST10-Distribution].

The terminal SHALL process this LTKM carried over SMS exactly as it processes the LTKMs carried over UDP that initiate a BSM solicited pull procedure.

Support for this BSM solicited pull procedure initiation over SMS bearer is:

- for the BSM: optional;
- for BCAST terminal: mandatory if Smartcard Profile using (U)SIM is supported, optional otherwise.

6.6.2 EXT BCAST for LTKM

To include Smartcard Profile specific information in LTKMs that can not be supported by the MBMS MSK message, a new MIKEY Extension payload MAY be included in the LTKM. For LTKMs this payload is referred to as the EXT BCAST for LTKMs. The EXT BCAST for LTKMs is used to transport additional information governing the use of the SEK/PEK carried within the LTKM. The EXT BCAST for LTKMs enables the following functionality:

- Pay-Per-View
- Pay-Per-Time
- Number of times the SEK/PEK can be used

- Send tokens to be added to a purse in the USIM
- Service/program termination for a user and SEK/PEK ID key deletion

The BSM and BSD/A SHALL support the use of the EXT BCAST for LTKMs.The terminal SHALL support the use of the EXT BCAST for LTKMs. The Smartcard MAY support the use of the EXT BCAST for LTKMs.

The EXT BCAST payload is an instance of the General Extension Payload for MIKEY defined in Section 6.15 of [RFC3830] and reproduced below for convenience:

Table 10: Logical Structure of the MIKEY General Extension Payload

Next Payload
Type
Length
Payload Data

Note that the same fields are present in the EXT BCAST for LTKMs and the EXT BCAST for STKMs (defined in Section 6.7.1). The difference between the two EXT BCAST payloads is the Payload Data that they contain.

For the EXT BCAST for LTKMs the MIKEY General Extension Payload fields SHALL be populated as defined below:

Next Payload (8 bits): This field SHALL be populated as defined in [RFC3830]. No change is required.

Type (8 bits): This field defines a new type for MIKEY in addition to the existing types for MIKEY. The new type is named "OMA BCAST STKM/LTKM MIKEY General Extension" and is assigned the value of 5.

Note: The actual value can be different than 5 as there are currently several extension header proposals in IETF and by the time an IETF draft is submitted the number allocated here (5) may already be taken by other proposals.

Length (16-bits): This field SHALL be populated as defined in [RFC3830]. No change is required.

Payload Data (Variable length): This field SHALL be populated with the Smartcard Profile LTKM Management Data as defined in Table 11.

Table 11: Format of Smartcard Profile LTKM Management Data

Smartcard Profile Management Data	Length (in bits)	Type
Long_term_key_message_extension() {		
protocol_version	**4**	**uimsbf**
security_policy_ext_flag	**1**	**bslbf**
Consumption_reporting_flag	**1**	**bslbf**
reserved_for_future_use	**1**	**uimsbf**
terminal_binding_flag	**1**	**bslbf**
if (security_policy_ext_flag == LTK_FLAG_TRUE) {		
security_policy_extension	**4**	**uimsbf**
purse_flag	**1**	**bslbf**
access_control_flag	**1**	**uimsbf**
reserved_for_future_use	**2**	**uimsbf**
cost_value	**16**	**uimsbf**
if (security_policy_extension == 0x06 || 0x07 || 0x08 || 0x09) {		
number_play_back	**8**	**uimsbf**

}		
if (purse_flag == LTK_FLAG_TRUE) {		
purse_mode	**1**	**bslbf**
token_value	**31**	**uimsbf**
}		
If (access_control_flag == LTK_FLAG_TRUE) {		
Reserved_for_future_use	**8**	**bslbf**
number_of_access_control_descriptors	**8**	**uimsbf**
access_control_descriptor_loop() {		
access_control_descriptor()		
}		
}		
}		
if (terminal_binding_flag == LTK_FLAG_TRUE) {		
TerminalBindingKeyID	**32**	**uimsbf**
RightsIssuerURILength	**8**	**uimsbf**
RightsIssuerURI	**8*RightsIssuerURILength**	**bslbf**
}		
if (Consumption_reporting_flag == LTK_FLAG_TRUE) {		
Security_policy_extension	**4**	**uimsbf**
reserved_for_future_use	**4**	**uimsbf**
}		
}		

6.6.2.1 Constant Values

LTK_FLAG_FALSE 0
LTK_FLAG_TRUE 1

6.6.2.2 Coding and Semantics of Attributes

protocol_version (4 bits): This field indicates the protocol version of this LTKM.

The device SHALL ignore messages that have a protocol_version number it doesn't support.

If the protocol version is set to 0x0 the format specified in this version of the specification SHALL be used. If set to anything else than 0x0, then the format is beyond the scope of this version of the specification.

security_policy_ext_flag (1 bit): This field indicates whether or not a security extension payload is carried in this LTKM. LTK_FLAG_FALSE indicates no security extension payload is present and the MBMS security policy in section 6.5.3 "MSK processing" in [3GPP TS 33.246 v7] applies (i.e. according to the parameters carried in the SP payload in Table 8), and LTK_FLAG_TRUE indicates a security extension payload is present and that a Smartcard Profile specific security policy associated with the security_policy_extension applies.

Specifically, if the security_policy_ext_flag is equal to LTK_FLAG_TRUE, the counter in TS field in STKMs is used to detect replay attacks and facilitate key validity data check (both procedures associated with the TEK) while the TEK ID field of the EXT payload is used to detect the resending of the same TEK.

consumption_reporting_flag (1 bit): This field indicates whether or not a consumption reporting payload is carried in this LTKM. If the consumption_reporting_flag is equal to LTK_FLAG_TRUE, the security_policy_ext_flag and the terminal_binding_flag shall be set to LTK_FLAG_FALSE, the V bit in the common header of LTKM SHALL be set and a verification message containing the status of the purse, the number of playbacks and the cost value of the content associated to the SEK/PEK ID and the security policy defined in the consumption_reporting extension.

terminal_binding_flag (1 bit): This field indicates whether or not terminal binding applies for the STKM streams protected by the SEK or PEK transported in this LTKM. LTK_FLAG_FALSE indicates it is not used, LTK_FLAG_TRUE indicates it is used.

security_policy_extension (4 bits): This field indicates the security policy extension.

Let in the following CxN be the value computed as “cost_value" times "number_play_back".

In the table below:

Purse associated to a service means that the purse is associated to a SEK/PEK key group.

User purse means that the purse is associated to the SMK key

It is assumed that TEK decryption is performed only as long as the associated SEK/PEK is deemed valid (i.e. the TS value in the received STKM is within the range of the key validity data stored in the Smartcard).

Table 12: Security Policy Extension Values

Value	Description
0x00	**Pay Per Time with purse associated to a service and play-back of content not allowed** In this mode, the Smartcard uses the purse associated with the SEK/PEK Key Group to implement charging based on the number of TEKs decrypted. STKM replay protection is enabled.
0x01	**Pay Per Time with purse associated to a service and play-back of content allowed** In this mode, the Smartcard uses the purse associated with the SEK/PEK Key Group to implement charging based on the number of TEKs decrypted. STKM replay protection is disabled.
0x02	**Pay Per Time with user purse and play-back of content not allowed** In this mode, the Smartcard uses the purse associated with the SMK to implement charging based on the number of TEKs decrypted. STKM replay protection is enabled.
0x03	**Pay Per Time with user purse and play-back of content allowed** In this mode, the Smartcard uses the purse associated with the SMK to implement charging on the number of TEKs decrypted. STKM replay protection is disabled.
0x04	**Subscription mode for a single play of content** In this mode the LTKM is processed as defined for MBMS in [3GPP TS 33.246 v7] but using the OMA BCAST STKM replay protection mechanism and key validity data format. No purse is used and the fields token_value and cost_value in the OMA BCAST Extension payload of LTKM SHALL NOT be processed. STKM replay protection is enabled.
0x05	**Subscription mode for unlimited play-back of content** In this mode The LTKM is processed as defined for MBMS in [3GPPTS 33.246] but using the OMA BAST key validity data format and with STKM replay protection disabled. This allows a user to watch recorded content an unlimited number of times, provided that the SEK/PEK is not deleted. No purse is used and the fields token_value and cost_value in the OMA BCAST Extension payload of LTKM SHALL NOT be processed.
0x06	**Pay Per View with purse associated to a service and deduction of tokens prior to play-back of content** In this mode, the Smartcard decreases the purse associated with the SEK/PEK Key Group by the value CxN at the reception of the right (i.e. LTKM reception). The play-back counter associated with the SEK/PEK is updated with the value number_play_back
0x07	**Pay Per View with user purse and deduction of tokens prior to play-back of content** In this mode, the Smartcard decreases the purse associated with SMK by the value CxN at the reception of the right (i.e. LTKM reception). The play-back counter associated with the SEK/PEK is updated with value number_play_back
0x08	**Pay Per View with purse associated to a service and play-back of content allowed** In this mode, the Smartcard decreases the purse associated with the SEK/PEK Key Group by the cost_value when a freshness failure is detected.
0x09	**Pay Per View with user purse and play-back of content allowed**

	In this Pay per view mode, the Smartcard decreases the purse associated with the SMK by the cost_value when a freshness failure is detected.
0x0A	**Service/Program termination** In this mode, a service/program termination is identified by the smart card through the equal to zero values of upper and lower limits of the Timestamp interval applicable to a certain SEK/PEK ID. As a result, the USIM/(R-)UIM/CSIM deletes all the previously stored SEK/PEK and related key material corresponding to that SEK/PEK ID.
0x0B …0x0D	Reserved for future standardization.
0x0E …0x0F	Reserved for proprietary implementation

Cost_value (16 bits): If the security_policy_extension is set to 0x00 or 0x01 or 0x02 or 0x03, the cost_value field indicates the number of tokens per TEK decrypted by the USIM to decrement from the associated purse. If the security_policy_extension is set to 0x06 or 0x07 or 0x08 or 0x09, the cost_value field indicates the number of tokens per play-back to decrement from the associated purse. For security_policy_extension values 0x04, 0x05 this field is not used.

Number_play_back (8bit): If the security_policy_extension is set to 0x06 or 0x07, or 0x08 or 0x09 the number_play_back field indicates the maximum number of times content recorded under a SEK/PEK can be played back, otherwise this field is not used.

purse_flag (1 bit): This field indicates whether or not a purse extension is carried in this LTKM. LTK_FLAG_FALSE indicates no purse data is present, LTK_FLAG_TRUE indicates purse data is present and the USIM SHALL perform appropriate update of the purse indicated by the purse_mode.

purse__mode (1 bit): This field indicates the purse update mode.

Table 13: Purse Update Mode Indication

Value	Description
0x00	**Set mode** In this mode, the purse SHALL be set to token_value.
0x01	**Add mode** In this mode, the token_value SHALL be added to the purse.

In order to detect overflow in the purse when an update occurs, the following SHALL apply:

When the purse is present in LTKM (purse_flag = LTK_FLAG_TRUE) and the purse_mode is set, the V bit in the common header of LTKM SHALL be set and a verification message containing the status of the update SHALL be sent by the secure function according to Section 6.6.3.

token_value (31 bits): This field indicates the number of tokens to be added or set (according to purse_mode value) in the purse.

access_control_flag (1bit): This field indicates whether or not an access_control_descriptor is carried in this LTKM. LTK_FLAG_FALSE indicates that no access_control_descriptor is present, LTK_FLAG_TRUE indicates that at least one access_control_descriptor is present in the LTKM.

Server MAY support access_control_descriptor. In case server doesn't support access_control_descriptor, the access_control_flag SHALL be set to LTK_FLAG_FALSE.

Number_of_access_control_descriptor (8bits): This field indicates the number of access_control_descriptor present in the LTKM

Access_control_descriptor_loop

Tag	8	uimsbf
Length	8	uimsbf
Value	8xlength	bit string

The Access Criteria Descriptor loop is an extension mechanism to allow the addition of new access controls associated to access_criteria carried in STKM in future versions of this specification. The secure function SHALL ignore Access Criteria Descriptors that it doesn't support. It is OPTIONAL for the BCAST Terminal to support Access Criteria Descriptors.

A single Access Criteria Descriptor can carry one or more access controls.

The following Access Criteria Descriptors have been defined:

parental_control – This descriptor enables an access restriction based on a content rating. The descriptor tag for this descriptor is 1. The value for this descriptor is encoded as follows:

Table 14: parental_control Access Criteria Descriptor

parental_control descriptor	Length	Type
update_mode	**1**	
parental_control_PIN_flag	**1**	
Reserved_for_future_use	**1**	
number_of_rating_types	**5**	**uimsbf**
For (i=0; i < number_of_rating_types; i++) {		
rating_type	**8**	**uimsbf**
level_granted	**8**	**uimsbf**
}		
if (parental_control_PIN_flag == LTK_FLAG_TRUE) {		
parental_control_PIN	**128**	**uimsbf**
}		

update_mode (1 bit): This field indicates whether the rating_value and rating_type pairs should be added to or deleted from the list stored in the secure function. If the update mode is set to LTK_FLAG_TRUE the rating_value and rating_type pairs are added to the list. If the update mode is set to LTK_FALG_FALSE the rating pairs are deleted from the list.

parental_control_PIN_flag (1 bit): This field indicates the presence of a pincode in the message.

number_of_rating_types (5bits): this field indicates the number of rating types transmitted within the descriptor.

rating_type (8bits): This field indicates the rating_type. See Table 31 in Section 7.1 for the coding of this field.

level_granted (8 bits): This field is an integer with the meaning that is dependent on the rating_type.

For rating_type different from 0x00, this level_granted is the maximum rating_value authorized to the user. The rating values are defined in Table 31 in Section 7.1.

The level_granted associated to the rating_type 0x00 is the age of the user and for this case the coding of rating_value is two digits in 4-bit Binary Coded Decimal (BCD)

As an example 11 year's old is coded as "0x11"

parental_control_PIN (128 bits): The parameter is used to initialize of the UNBLOCK PIN for the parental control PINCODE in the secure function. The UNBLOCK PIN is used if the parental PINCODE has been forgotten by the user. This field contains the cryptogram of the PIN.

The ciphering of the PIN SHALL be based on the SMK.

The ciphering algorithm used SHALL be AES-CM-128

The input message SHALL consist of the UNBLOCK PIN padded with 0 for the 64 Most Significant Bits to obtain an input message of 128 bits: 0 (64 bits) || PIN

TerminalBindingKeyID (32 bits): This field contains the identifier of the Terminal Binding Key. See Section 12 for further details. This field is ignored by the USIM as it is used only by the terminal.

RightsIssuerURILength (8 bits): This field specifies the length in bytes of the Rights Issuer URI specified below. This field is ignored by the USIM as it is used only by the terminal.

RightsIssuerURI (Variable Length): This field is the URI of the Rights Isssuer that can be contacted to obtain the Terminal Binding Key. See Section 12 for further details. This field is ignored by the USIM as it is used only by the terminal.

6.6.3 LTKM Verification Message Structure

The processing described in this section is handled by a secure function located on either the Smartcard or terminal. Following MBMS [3GPP TS 33.246 v7], for the Smartcard Profile the location of the secure function is determined by whether GBA_U or GBA_ME is used to establish SMK/SRK. Where GBA_U is used the secure function is located on the Smartcard and where GBA_ME is used the secure function is located on the terminal.

If the V-bit in the MIKEY common header is equal to 1 in a LTKM, the secure function SHALL send a verification message as a response. If the EXT BCAST payload is not present in the incoming LTKM, then the verification message SHALL be constructed as defined in [3GPP TS 33.246 v7]. Otherwise the following applies:

The verification message shall consist of the following fields: HDR || EXT BCAST || TS || IDr || V

Table 15: Logical Structure of the LTKM Verification Message

Common HDR
EXT BCAST
TS
Idr
V

The MAC included in the verification payload (V), shall be computed over both the initiator's and the responder's ID as well as the timestamp in addition to be computed over the response message as defined in RFC 3830 [9]. The key used in the MAC computation is the authentication key derived from SMK as described in RFC 3830 [9].

The terminal SHALL forward the LTKM verification messages received from the secure function to the BSM/Permissions issuer.

The EXT BCAST payload of the verification message SHALL be formatted as follows

Table 16: Format of the EXT BCAST of the Verification Message

Verification_Message_BCAST Extension _Description	Length (in bits)	Type
Verification_message_BCAST extension() {		
Consumption_reporting_flag	**1**	**bslbf**
Overflow_flag	**1**	**bslbf**
Reserved_for_future_use	**6**	**uimsbf**
if (Consumption_reporting flag == LTK_FLAG_TRUE) {		
Security_policy_extension	**4**	**uimsbf**
purse_flag	**1**	**bslbf**
reserved_for_future_use	**3**	**uimsbf**
Cost_value	**16**	**uimsbf**
if (security_policy_extension == 0x06 || 0x07 || 0x08 || 0x09) {		
Number_play_back	**8**	**uimsbf**
}		
if (purse_flag == LTK_FLAG_TRUE) {		
reserved_for_future_use	**1**	**bslbf**
token_value	**31**	**uimsbf**
}		
}		
}		

The Overflow flag shall be set to LTK_FLAG_FALSE except when explicitly required by the processing of the LTKM (Section 6.6.4).

The Security_policy_extension shall be a copy of the Security_policy_extension received in the LTKM message.

The purse flag shall be set to LTK_FLAG_TRUE if the Security_policy_extension is 0x00, 0x01, 0x02, 0x03, 0x06, 0x07, 0x08 or 0x09. Otherwise, it shall be set to LTK_FLAG_FALSE.

The Cost_value shall be a copy of the Cost_value received in the LTKM message.

If present, the Number_of_play_back shall be the remaining number of play-back allowed after the LTKM processing.

If present, the token_value shall be the remaining number of tokens after the LTKM processing.

6.6.4 OMA BCAST LTKM Processing

LTKMs are processed by a secure function located on either the Smartcard or terminal. Following MBMS [3GPP TS 33.246 v7], for the Smartcard profile the location of the secure function is determined by whether GBA_U or GBA_ME is used to establish SMK/SRK. Where GBA_U is used the secure function is located on the Smartcard and where GBA_ME is used the secure function is located on the terminal.

When the OMA BCAST LTKM arrives at the terminal, the terminal first checks whether an EXT BCAST payload is present. If the EXT BCAST payload is not present or the EXT BCAST payload is present and the security_policy_extension_flag is set to 0x00, the terminal shall process the LTKM as defined in section 6.4.6.1 of [3GPP TS 33.246 v7]. In all other cases the terminal shall process the LTKM as defined in section 6.4.6.1 of [3GPP TS 33.246 v7] omitting the MBMS replay protection check, i.e. the terminal SHALL NOT check the Time Stamp (TS) payload in the LTKM against the stored replay counter associated with the given SMK. If the terminal processing of LTKM is successful the LTKM shall be transported to the secure function for further processing. If the LTKM does not include an EXT BCAST payload, the secure function performs the MBMS security procedure (MSK Update Mode) according to Section 6.5.3 of [3GPP TS 33.246 v7]. Otherwise, the following procedure applies:

First, the secure function performs the message validation using the SMK (MUK) as described in [3GPP TS 31.102 v6]. If the message validation is successful, then the replay counter associated with the SEK/PEK (for subsequent detection of attempted STKM replay) SHALL be initialized with the value "TS Low" in the Key Validity Data. In addition, the secure function processes the LTKM based on the value of the security_policy_extension as follows:

- If the consumption_reporting flag equals to 1, the only action to perform is to send a Verification message as described in Section 6.6.3. The Overflow flag SHALL be set to LTK_FLAG_FALSE. This mode allows the server to retrieve in

the secure function, remaining tokens associated to a given purse, cost_value of a given SEK/PEK and/or remaining play-back number without change and update of SEK/PEK.

Otherwise:

- If the security_policy_extension is set to 0x00 or 0x01, the secure function stores the received SEK/PEK (if KEMAC Key Data sub-payload is present), the validity data, the security_policy_extension and the cost_value. If the purse_flag is set to 1, the secure function then updates the purse associated with the SEK/PEK Key Group with the "token_value" according to the received purse_mode value. If an overflow occurs on the purse during this update (i.e. purse value > 0x7FFFFFFF), then the purse SHALL remain unchanged and a verification message with the Overflow flag and the consumption_reporting_flag set to LTK_FLAG_TRUE SHALL be sent.

- If the security_policy_extension is set to 0x02 or 0x03, the secure function stores the received SEK/PEK (if KEMAC Key Data sub-payload is present), the validity data, the security_policy_extension and the cost_value. If purse_flag is set to 1, the secure function then updates the purse associated with the SMK with the token_value according to the received purse_mode value. If an overflow occurs on the purse during this update (i.e. purse value > 0x7FFFFFFF), then the purse SHALL remain unchanged and a verification message with the Overflow flag and the consumption_reporting_flag set to LTK_FLAG_TRUE SHALL be sent.

Let, in the following, CxN be the value computed as "cost_value" times "number_play_back".

- If the security_policy_extension is set to 0x06, check the purse-flag first. If the purse_flag is set to 1, then the secure function SHALL update the purse associated with the SEK/PEK Key Group with the token_value according to the received purse_mode value. If an overflow occurs on the purse during this update (i.e. purse value > 0x7FFFFFFF) the purse SHALL remain unchanged and a verification message with the Overflow flag and the consumption_reporting_flag set to LTK_FLAG_TRUE SHALL be sent.

Otherwise, if the purse_flag is set to 0 and adequate token balance exists, then the secure function SHALL decrease the purse associated with the SEK/PEK Key Group ID by the CxN value. That balance SHALL include, if applicable, update of the purse associated with the SEK/PEK Key Group (indicated by received purse_flag = 1 and the purse_mode value). In addition, the secure function SHALL store the received SEK/PEK (if KEMAC Key Data sub-payload is present), the key validity data and the security_policy_extension, and SHALL set the SEK/PEK play-back counter to number_play_back value.

Otherwise, should this operation fail due to insufficient token balance in the purse associated with the SEK/PEK Key Groups, the secure function SHALL return the response to the terminal for the current AUTHENTICATE command corresponding to OMA BCAST operation for security_policy_extension operation (see Appendix E) with a status code corresponding to 'lack of credit in the service purse' or return the status word '91xx' to indicate that a proactive command is pending (e.g. DISPLAY TEXT, SEND SMS) in order to inform the user of the lack of credit in the purse.

In this mode, the purse associated with SEK/PEK Key Group is decreased by the cost of the multiple play-back at the reception of the current LTKM. This is a pre-paid mode. The purse is a service-level purse, with its tokens valid for consumption of a specific service under the subscription.

- If the security_policy_extension is set to 0x07, check the purse-flag first. If the purse_flag is set to 1, then the secure function SHALL update the purse associated with the SEK/PEK Key Group with the token_value according to the received purse_mode value. If an overflow occurs on the purse during this update (i.e. purse value > 0x7FFFFFFF) the purse SHALL remain unchanged and a verification message with the Overflow flag and the consumption_reporting_flag set to LTK_FLAG_TRUE SHALL be sent.

Otherwise, if the purse_flag is set to 0 and adequate token balance exists, the secure function SHALL decrease the purse associated with the SMK by the CxN value. That balance SHALL include, if applicable, update of the purse associated with the SMK (indicated by received purse_flag = 1 and the purse_mode value). In addition, the secure function SHALL store the received SEK/PEK (if KEMAC Key Data sub-payload is present), the key validity data and the security_policy_extension and SHALL set the SEK/PEK play-back counter to number_play_back value.

Otherwise, should this operation fail due to insufficient balance in the associated SMK token purse, the secure function SHALL return the response to the terminal for the current AUTHENTICATE command corresponding to OMA BCAST operation for security_policy_extension operation (see Appendix E) with a status code corresponding to 'lack of credit in the

service purse' or return the status word '91xx' to indicate that a proactive command is pending (e.g. DISPLAY TEXT, SEND SMS) in order to inform the user of the lack of credit in the purse.

In this mode, the purse associated with SMK is decreased by the cost of the multiple play-back at the reception of the current LTKM. This is a pre-paid mode. The purse is a user-level purse, with its tokens valid for consumption of any service accessible by the user.

- If the security_policy_extension is set to 0x08, the secure function stores the received SEK/PEK (if KEMAC Key Data sub-payload is present), the key validity data, the security_policy_extension, the cost_value, and sets the play-back counter to number_play_back value. If the purse_flag is set to 1, then the secure function updates the internal purse associated with the SEK/PEK Key Group with the token_value according to the received purse_mode value. If an overflow occurs on the purse during this update (i.e. purse value > 0x7FFFFFFF) the purse SHALL remain unchanged and a verification message with the Overflow flag and the consumption_reporting_flag set to LTK_FLAG_TRUE SHALL be sent.

In this mode, the purse associated with SEK/PEK is decreased only at the consumption of the content (at each play-back). The purse is a specific purse associated to a service.

- If the security_policy_extension is set to 0x09, the secure function stores the received SEK/PEK (if KEMAC Key Data sub-payload is present), the key validity data, the security_policy_extension, the number_play_back and the cost_value. If the purse_flag is set to 1, then the secure function updates the internal user purse associated with the SMK with the token_value according to the received purse_mode value. If an overflow occurs on the purse during this update (i.e. purse value > 0x7FFFFFFF) the purse SHALL remain unchanged and a verification message with the Overflow flag and the consumption_reporting_flag set to LTK_FLAG_TRUE SHALL be sent.

In this mode, the purse associated with SMK is decreased only at the consumption of the content (at each play-back). The purse is a user purse, not associated to a specific service.

- If the security_policy_extension value indicates that the replay protection is enabled (i.e. security_policy_extension equals to 0x00 or 0x02 or 0x04), the secure function shall set the anti-replay counter (used to detect STKM replay attacks) to a value equal to "TS low" found in the Key Validity Data subfield of the KEMAC payload in the OMA BCAST MIKEY LTKM.

- If the security_policy extension is set to 0x0A, and the Key Validity Data subfield in the KEMAC payload of the OMA BCAST MIKEY LTKM message indicates the Key Validity interval for a given SEK/PEK ID is "TS high"= "TS low"=0, the (U)SIM/(R-)UIM/CSIM SHALL delete all the previously stored SEK/PEK with the same SEK/PEK ID and their related key material (i.e: key validity data, security policy extensions, number_play_back and cost_value).

- If access_control_flag is set to LTK_FLAG_TRUE, the LTKM contains at least one access_control_descriptor. For the present specification, only one access_control_descriptor is defined for LTKM: the Parental_control_descriptor.

    - If the parental_control_descriptor is present in the LTKM,

      If update_mode is set to LTK_FLAG_TRUE, the rating_type and associated level_granted transmitted in the LTKM are added to any rating_type and associated level_granted values that exist in the secure function. For a given rating_type received in the LTKM, if a level_granted value is already stored for this same rating_type, the incoming level_granted value SHALL replace the old one.

      If the update_mode is set to LTK_FLAG_FALSE, all rating_type values stored in the secure function that match a rating_type value transmitted in the LTKM SHALL be erased. Deleting a rating_type from the secure function SHALL also erase the associated level_granted value.

      This list of rating_type/level_granted pairs is associated to the SMK used to protect the LTKM.

      A default setting for rating_type/level_granted pairs specific to service provider MAY be possible during the card manufacture.

      If the parental_control_PIN is present in the message (parental_control_PIN_flag set to LTK_FLAG_TRUE, the secure function re-initializes the UNBLOCK PIN of the PIN used for parental control with the deciphered value of parental_control_PIN received in LTKM.

The response of the AUTHENTICATE command in the case of parental_control_descriptor is present in the LTKM with an update of the list of rating_type/level_granted pairs is the response of AUTHENTICATE command corresponding to an MSK Update Mode for OMA BCAST operation for security_policy_extension operation (see Appendix E) with the list of the rating_type and level_granted values stored in the Smartcard after this update.

Finally, if the V-bit in the HDR field of the received LTKM is set then the secure function SHALL produce a LTKM Verification Message as described in Section 6.6.3.

After the LTKM processing by the secure function, if the terminal didn't receive an error due to integrity or validation or bootstrapping failure, and if the terminal_binding_flag of the LTKM is set to 1, the terminal stores the TerminalBindingKeyID and the PermissionsIssuerURI.

NOTE: The first STKM sent by the network (BSD/A) for the associated SEK/PEK SHALL contain a timestamp value equal to or greater than "TS Low" + 1

NOTE: The Key Validity Data subfield in the KEMAC payload in the LTKM defines the Key Validity interval for SEK/PEK in terms of STKM TIMESTAMP interval:

From (32-bits): Lower limit of Timestamp ("TS low")

To (32bits): Upper limit of Timestamp ("TS high")

NOTE: There is one internal anti-replay counter per SEK/PEK to support replay protection for STKM delivery

NOTE: To avoid security failures during the key validity data and anti replay verifications, the Timestamp field (TS) in STKM associated with a SEK/PEK key group SHALL only be reset when the SEK/PEK is updated.

NOTE: If the secure function is located on the Smartcard, for SEKs/PEKs without a defined security policy extension, the policy of deleting SEK/PEK records to free up space in the file in EF_{MSK} SHALL be controlled by the Terminal, e.g. the Terminal SHOULD delete any SEKs/PEKs that are no longer needed. How the terminal decides which SEKs/PEKs are no longer needed is implementation specific. Note that the file EF_{MSK} is only used for SEKs/PEKs without a defined security policy extension. For SEKs/PEKs with a defined security policy extension it should be possible to store a variable number (e.g. typically 2) of SEK/PEK per Key Domain ID/Key Group ID pair in the Smartcard. In the case where a security policy extension is defined for a SEK/PEK, the management (deletion) of the SEKs/PEKs is implementation specific.

The following table gives the association between Smartcard Profile parameters used in the processing of LTKM or STKM and SEK/PEK, SEK/PEK key group, or SMK, i.e. it indicates which parameters can be linked to which key identifiers.

Table 17: Association between Smartcard Profile Parameters and Key Identifiers

	SEK/PEK	SEK/PEK key group	SMK
SEK/PEK ID	X		
Key validity data (STKM TS low & TS high)	X		
Security_policy_extension	X		
Cost_value	X		
SEK/PEK Key data	X		
Play_back_counter	X		
Anti-replay-counter	X		
Token purse		X	X

For example, Cost_value and Play-back_counter can only be linked to a specific SEK/PEK, whereas Token purse can be either linked to a SEK/PEK key group **or** an SMK. Note that a Token purse linked to an SMK can therefore be used for **all** SEK/PEKs linked to that SMK.

6.6.5 Service purses and global purse

The pay-per-view and pay-per-time modes use purses in the Smartcard.

Several purses are defined:

1. Purse associated to a service and then linked to a SEK/PEK key group
2. Global purse linked to the SMK that has been established during the registration step

According to security-policy-extension value, the purse associated to the service or the purse associated to the SMK will be used either during the LTKM processing and/or STKM processing (see Section 6.6.4 and Section 6.7.2).

If security policy extensions stored for a same SEK/PEK ID imply that service purse or global purse could be used (e.g. security-policy-extension 0x00 and 0x02 available for a SEK/PEK), the service purse SHALL be used before the SMK purse.

If for some reason, the SMK (established during the last GBA run) is renewed, the remaining purse value associated to the old SMK SHALL be copied and associated to the new SMK.

Update of the purses is made at the reception of a LTKM with a security-policy-extension and when the purse_flag is set.

In this case the number of token indicated in the token_value will be

⇒ added to the purse indicated by the security-policy-extension and the SEK/PEK ID, if the purse_mode is 0x01

⇒ or used to set the purse if the purse_mode is 0x00

This update SHALL not be executed if:

⇒ security_policy_ext_flag is set

⇒ AND purse_flag is set

⇒ AND purse_mode is set

⇒ AND V-bit flag in the MIKEY common header is set

⇒ AND token_value is set to 0x7FFFFFFF

⇒ AND cost_value is set to 0xFFFF

⇒ AND no KEMAC Key Data sub-payload

Such case is used for the reporting of token consumption (see Section 6.6.3).

6.7 Layer 3: Short Term Key Message - STKM

Unless specified in this section, terms defined in the LTKM section also apply to this section. The following applies for STKMs:

Resending check: Resending of the same STKM/TEK SHALL be detected by the terminal using the TEK_ID (MTK ID) field of the MBMS EXT payload. The terminal SHALL NOT forward duplicate STKMs to the Smartcard.

This shall not be confused with the detection of replay attacks which uses the TS field in the STKM message.

Resending of the same STKM/TEK allows faster changing of channels because the terminal does not have to wait for the arrival of a new STKM/TEK. For example, a new STKM/TEK may only be sent every minute, but the STKM/TEK is resent every 500ms meaning that the terminal has to wait much shorter for a matching STKM/TEK after a channel change.

In MBMS for each STKM sent the TS field is increased, even if this STKM carries the same TEK as the previous STKM message.

However, in BCAST the server MAY resend the same STKM, containing the same TEK, without increasing the TS field. This avoids the need for generating new STKMs within the same crypto period.

Note: this is an improvement to the MBMS specification version 6 since BM-SC handling needs less processing for building subsequent authenticated STKM with the same key material included.

Filtering at the terminal side keeps the solution consistent with the MBMS replay protection, since in the terminal resending of the STKM/TEK is detected by checking the TEK_ID (MTK ID) field of the MBMS EXT payload.

The table below shows the MIKEY message format used for STKMs in the Smartcard Profile. The message structure SHALL be identical to [AMP-R002] the MIKEY message used by MBMS to deliver the MBMS Traffic Key (MTK), as defined by [3GPP TS 33.246 v7], with the addition of the EXT BCAST payload. The EXT BCAST payload is described in Section 6.6.2.

Table 18: Logical Structure of the MIKEY Message Used

Common HDR
EXT MBMS
EXT BCAST
TS
KEMAC

The EXT MBMS payload, depicted in Table 19, is defined in section 6.4.4 “General extension payload” of [3GPP TS 33.246 v7] and reproduced below for convenience. For the Smartcard Profile STKM the EXT MBMS payload is the extension payload defined in [3GPP TS 33.246 v7] for use with MBMS MIKEY MTK message.

Table 19: EXT MBMS Used within the MBMS MTK Message

Key Domain ID sub-payload
Key Type ID sub-payload (MSKID)
Key Type ID sub-payload (MTK ID)

All fields within the STKM SHALL be populated as defined in [3GPP TS 33.246 v7] for the MBMS MTK message, with the exception of the EXT BCAST payload. The mappings described in Section 6.2 for the Smartcard Profile parameters, i.e. SEK/PEK ID is mapped to MSK ID and SEK/PEK is mapped to MSK, TEK ID is mapped MTK ID and TEK is mapped to MTK.

Each STKM stream MUST only be secured using a single SEK/PEK. In some cases multiple STKM streams can deliver the same TEKs secured by different SEKs/PEKs. The Terminal MUST use the Service Guide (SG) to locate the relevant STKM stream for the encrypted traffic stream it needs to decrypt.

The EXT BCAST payload SHALL be populated as defined in Section 6.7.1.

6.7.1 EXT BCAST for STKMs

To include Smartcard Profile specific information in STKMs that can not be supported by the MBMS MTK message, a new MIKEY Generic Extension Header payload (described in Section 6.6.2) is included in the STKM. For STKMs this payload is reffered to as the EXT BCAST for STKMs. The EXT BCAST for STKMs is used to transport information related to the use of the TEKs contained within the STKM.

The terminal SHALL support the processing of all fields included in the EXT BCAST for STKM with the exception of the Access Criteria Descriptor. The terminal MAY support the processing of Access Criteria Descriptors.

The terminal SHALL implement all necessary processing and SHALL support associated messaging to handle Smartcard based access criteria enforcement:

- For the Smartcard based parental control enforcement, the terminal SHALL implement pincode requested processing (described in the requesting pincode part in Section 6.7.2.1), operation on Pincode (described in the operation on pincode in Smartcard part in Section 6.7.2.1) and associated messaging to handle Smartcard parental control with the related processing as defined in Section 6.7.2 and Section 6.7.2.1 (i.e: proactive command DISPLAY TEXT, response of AUTHENTICATE command corresponding to OMA BCAST operation for parental control operation (see Appendix E), VERIFY PIN, CHANGE PIN, UNBLOCK PIN defined in [ETSI TS 102.221]).
- For Smartcard based location_based_restriction enforcement, the terminal SHALL support the proactive command PROVIDE LOCAL INFORMATION, the proactive command DISPLAY TEXT and the response of AUTHENTICATE command corresponding to OMA BCAST operation for location based restriction operation (see Appendix E).

The terminal SHALL support the processing of the all fields included in the EXT BCAST for STKM with the exception of the Access Criteria Descriptor. If the Smartcard supports the use of the EXT BCAST payloads, the Smartcard MAY process and enforce the access criteria (if it is transmitted in the EXT BCAST for STKM) as defined in Section 6.7.2. Note that MBMS MIKEY implementations [3GPP TS 33.246 v7] ignore the EXT BCAST for STKMs and therefore do not support the enforcement of the access criteria.

If the Parental Rating access criteria are transmitted in the STKM the following applies:

- If the Smartcard supports the use of EXT BCAST payloads and supports the enforcement of parental rating, parental rating enforcement SHALL be done by the Smartcard as explained in Section 6.7.2.1. Note that MBMS MIKEY implementations [3GPP TS 33.246 v7] will ignore the EXT BCAST for STKMs and therefore will not support the enforcement of the access criteria.
- If the Terminal supports the enforcement of the Parental Rating Access Criteria Descriptor it SHALL do so by not sending an STKM to the Smartcard if the STKM contains a rating more restrictive than that which the user of the device is authorised to access. Whether or not Terminal enforcement is used is implementation specific. For example a terminal MAY choose to enforce the Parental Rating Access Criteria when it identifies that the Smartcard does not support the EXT BCAST payloads. Alternatively, Terminal enforcement MAY be used in parallel with the Smartcard enforcement mechanism to provide an additional, locally controlled restriction on access. Note that in this case the most restricted level from the smartcard or the terminal will apply.

If the location_based_restriction access criteria are transmitted in the STKM the following applies:

- If the Smartcard supports the use of EXT BCAST payloads and supports the enforcement of location_based_restriction, this enforcement SHALL be done by the Smartcard as explained in Section 6.7.2.2. Note that MBMS MIKEY implementations [3GPP TS 33.246 v7] will ignore the EXT BCAST for STKMs and therefore will not support the enforcement of the access criteria.

For the EXT BCAST for STKMs the fields of the MIKEY Generic Extension Header MUST contain the following data:

Next Payload (8 bits): This field SHALL be populated as defined in [RFC3830]. No change is required.

Type (8 bits): This field defines a new type for MIKEY in addition to the existing types for MIKEY. The new type is named "OMA BCAST STKM/LTKM MIKEY General Extension" and is assigned the value of 5.

Note: The actual value can be different than 5 as there are currently several extension header proposals in IETF and by the time an IETF draft is submitted the number allocated here (5) may already be taken by other proposals.

Length (16-bits): This field SHALL be populated as defined in [RFC3830]. No change is required.

Payload Data (Variable Length): This field SHALL contain Smartcard Profile STKM Management Data defined in Section 6.7.1.1.

6.7.1.1 STKM Management Data

The format below SHALL be used for the Smartcard Profile STKM Management Data.

Table 20: Format of Smartcard Profile STKM Management Data

Smartcard Profile STKM Management Data	Length (in bits)	Type
short_term_key_message() {		
selectors_and_flags {		
protocol_version	**4**	**Uimsbf**
protection_after_reception	**2**	**Uimsbf**
terminal_binding_flag	**1**	**Uimsbf**
access_criteria_flag	**1**	**uimsbf**
traffic_protection_protocol	**3**	**uimsbf**
traffic_authentication_flag	**1**	**uimsbf**
}		
traffic_key_lifetime	**4**	**uimsbf**
if (access_criteria_flag == TKM_FLAG_TRUE) {		
reserved_for_future_use	**8**	**bslbf**
number_of_access_criteria_descriptors	**8**	**uimsbf**
access_criteria_descriptor_loop() {		
access_criteria_descriptor()		
}		
}		
}		

6.7.1.2 Coding and Semantics of Attributes

Section 7 introduces the coding and semantics of all attributes common between the DRM Profile and the Smartcard Profile. Any Smartcard Profile specific attributes are introduced below.

terminal_binding_flag – indicates whether or not terminal binding is required for the Smartcard Profile. 0 indicates it is not required, 1 indicates it is required.

6.7.2 OMA BCAST STKM Processing

STKMs are processed by a secure function located on either the Smartcard or terminal. Following MBMS [3GPP TS 33.246 v7], for the Smartcard profile the location of the secure function is determined by whether GBA_U or GBA_ME is used to establish SMK/SRK. Where GBA_U is used the secure function is located on the Smartcard and where GBA_ME is used the secure function is located on the terminal.

When the STKM arrives at the terminal, the terminal SHALL perform the resending check according to 3GPP MBMS [3GPP TS 33.246 v7] before sending the STKM to the secure function, i.e. if the TEK ID extracted from the EXT MBMS payload is less than or equal to the current TEK ID (associated to the SEK/PEK used to protect the STKM) buffered in the terminal, the message SHALL NOT be sent to the secure function. The terminal SHALL NOT perform the MBMS replay protection check (defined in section 6.4.3 of [3GPP TS 33.246 v7]), i.e. the terminal SHALL NOT check the Time Stamp (TS) payload of the STKM against the stored replay counter associated with the given SEK/PEK. Note that unlike MBMS this check is completed by the secure function.

If the resending check is passed, the STKM is sent to the secure function and the following procedure applies:

On reception of the STKM, the secure function first retrieves, from the EXT MBMS payload, the SEK/PEK with the Key Domain ID and the SEK/PEK ID. It then performs the message validation according to [RFC3830]. If the message validation is successful, the secure function checks for the presence of a stored security policy extension associated to the SEK/PEK used to authenticate the STKM.

If several policy extensions are available for the given SEK/PEK ID, the following priorities SHALL apply on the security policy extension used to handle the incoming STKM:

1. Policy extension for a subscription mode (0x04 or 0x05)

2. Policy extension for a pay-per-view mode (0x06 or 0x07 or 0x08 or 0x09)
3. Policy-extension for a pay-per-time mode (0x00 or 0x01 or 0x02 or 0x03)

If a purse associated to the SEK/PEK is available, this purse will be used for the pay-per-time or pay-per-view before the global purse (associated to the SMK). Hence the following priority SHALL apply on the pay-per-view and pay-per-time security-policy-extension:

- Security-policy-extension 0x00 will take precedence to 0x02
- Security-policy-extension 0x01 will take precedence to 0x03
- Security-policy-extension 0x06 will take precedence to 0x07
- Security-policy-extension 0x08 will take precedence to 0x09

If no security policy extension is present for the SEK/PEK ID extracted from the STKM, the SEK/PEK is processed according to [3GPP TS 33.246]; otherwise the following procedure applies:

First, the secure function checks the presence of access criteria in the message and controls that the access criteria are met using internal information. This internal information depends on the type of access criteria. For the current version of the specification, access criteria defined are for parental control and location based restriction. See Section 6.7.2.1 for the enforcement of parental control and Section 6.7.2.2 for the enforcement of location based restriction.

If the processing of the parental control access criteria ends with failure the secure function SHALL abort the processing of the STKM. If the secure function is located on the Smartcard it SHALL either send a response to the terminal for the current AUTHENTICATE command corresponding to OMA BCAST operation for parental control operation (see Appendix E) with a status code corresponding to ‘User not authorized’ and with the rating_type corresponding to the rating_type of the current program and the level_granted for this rating_type stored in the Smartcard, or SHALL send the proactive command ‘DISPLAY TEXT’ (as described in [3GPP TS 31.111 v6] or [3GPP2 C.S0035-A]) in order to inform the user of the following:

- of the level_granted stored in the card for the rating_type received in the STKM
- and that they are not allowed to view this service as they are not authorized to view services with the associated rating transmitted in the STKM

The terminal SHALL support the proactive command ‘DISPLAY TEXT’.

If the processing of the parental control access criteria ends with success, the secure function performs a key validity data check by comparing the received Time Stamp field (TS) in the STKM with the stored key validity data, in the form of Timestamp interval “TS low” and “TS high” of the associated SEK/PEK. If the received TS is lower than “TS low” or is greater than “TS high”, then the secure function SHALL return a failure to the terminal. If the secure function is located on the Smartcard, the return message shall take the form of the status word '6985' (Conditions of use not satisfied). In this section, it is presumed that the SEK/PEK is always deemed valid upon the key validity check.

If the location based restriction ends with the status ‘blackout’, then the secure function aborts the processing of the STKM. If the secure function is located on the Smartcard, then it SHALL either send a response to the terminal for the current AUTHENTICATE command corresponding to OMA BCAST operation for location based restriction operation (see Appendix E) with a status code corresponding to ‘blackout’, or SHALL send a proactive command ‘DISPLAY TEXT’ (as described in [3GPP 31.111 v6] or [3GPP2 C.S.0035-A]) in order to inform the user that the program can not be displayed in this area. The terminal SHALL support the proactive command ‘DISPLAY TEXT’.

If the location based restriction ends with the status ‘need specific permissions’, then the secure function checks if a security_policy_extension for a PPV is available for this content. Two cases result from this check:

- If a security_policy_extension for a PPV is available for this content, then the processing of the STKM continues as as we discuss below.
- If a security_policy_extension for a PPV is not available for this content, then the secure function aborts the processing of the STKM. If the secure function is located on the Smartcard, then it SHALL either send a response

to the terminal for the current AUTHENTICATE command corresponding to OMA BCAST operation for location based restriction operation (see Appendix E) with a status code corresponding to 'need specific permissions', or SHALL send a proactive command 'DISPLAY TEXT' (as described in [3GPP 31.111 v6] or [3GPP2 C.S.0035-A]) in order to inform the user that the program can not be displayed in this area without a specific permission. The terminal SHALL support the proactive command 'DISPLAY TEXT'.

If the security_policy_extension is equal to 0x00 or 0x02 or 0x04, the secure function SHALL compare the received Time Stamp field (TS) with the stored anti replay value of the associated SEK/PEK. If the received TS is equal or lower than the internal stored anti replay value, then the secure function SHALL return a failure to the terminal. If the secure function is located on the Smartcard, the return message shall take the form of the status word '9865' (Key freshness failure). Otherwise, the stored anti replay counter SHALL be set to the received TS value.

Note: Less than or equal is to be taken in the meaning of RFC1982. If the less than or equal relation is undefined in the sense of RFC1982, the message should be considered as being replayed and shall be discarded.

If the security_policy_extension is equal to 0x06 or 0x07 or 0x08 or 0x09, the secure function SHALL perform the anti replay verification described above except that the Key freshness failure SHALL NOT be returned to the terminal.The Key freshness failure in these cases is used to detect the play-back of content and the action performed depends on the security_policy_extension value.

If the security_policy_extension equals to 0x00 or 0x01 or 0x02 or 0x03, the secure function SHALL decrement the purse associated with the SEK/PEK key group or SMK by the stored cost value associated with that SEK/PEK. Then, the secure function returns the decrypted TEK and Salt key (if Salt key is available).

If the security_policy_extension is equal to 0x04 or 0x05 then, the secure function returns the decrypted TEK and Salt key (if Salt key is available).

If the security_policy_extension is equal to 0x06 or 0x07 and the play-back counter is not equal to zero and an anti-replay verification failure is detected by the secure function, the play-back counter SHALL be decreased by one and the anti-replay counter of the associated SEK/PEK SHALL be set to the lower limit (TS low) of the SEK/PEK key validity data payload. Then, the secure function returns the decrypted TEK and Salt key (if Salt key is available). The secure function SHOULD delete the SEK/PEK when the associated play-back counter is equal to zero and a Key freshness failure is detected, this situation occurring when all play-back allowed are already consumed and a new play-back occurs.

If the security_policy_extension is equal to 0x08, then

If the play-back counter is not equal to zero, and an anti-replay verification failure is detected by the secure function, then

- If the purse associated with the SEK/PEK Key Group contains adequate balance, the following operations SHALL take place:
  - That purse SHALL be decremented by the cost_value,
  - the play-back counter SHALL be decreased by one,
  - the anti-replay counter of the associated SEK/PEK SHALL be set to the lower limit (TS low) of the associated key validity data, and
  - the secure function returns the decrypted TEK and Salt key (if Salt key is available).
- Otherwise (i.e. the token balance is less than the cost_value), the purse SHALL NOT be decremented, the Smartcard SHALL NOT return the decrypted TEK and Salt. In addition, the Smartcard SHALL return the status word '91xx' to indicate that a proactive command is pending (e.g. DISPLAY TEXT, SEND SMS) in order to inform the user of inadequate credit.

Otherwise, if the play-back counter equals zero, then regardless of token balance or occurrence of key freshness failure, the secure function SHOULD delete the SEK/PEK.

If the security_policy_extension is equal to 0x09, then

If the play-back counter is not equal to zero, and an anti-replay verification failure is detected by the secure function, then

- If the purse associated with the SMK contains adequate balance, the following operations SHALL take place:
  - That purse SHALL be decremented by the cost_value,
  - the play-back counter SHALL be decreased by one,
  - the anti-replay counter of the associated SEK/PEK is set to the lower limit (TS low) of the key validity data, and
  - the Smnartacrd returns the decrypted TEK and Salt key (if Salt key is available).
- Otherwise (i.e. the token balance is less than the cost_value), the purse SHALL NOT be decremented, the Smartcard SHALL NOT return the decrypted TEK and Salt. In addition, the Smartcard SHALL return the status word '91xx' to indicate that a proactive command is pending (e.g. DISPLAY TEXT, SEND SMS) in order to inform the user of inadequate credit.

Otherwise, if the play-back counter equals zero, then regardless of token balance or occurrence of key freshness failure, the secure function SHOULD delete the SEK/PEK.

For the security_policy_extension modes using the purse or the play-back counter, the secure function SHALL ignore the nominal operation in cases whereby the purse cannot be decreased by the cost_value, or play-back counter equals zero. If the secure function is located on the Smartcard, when the purse of the service cannot be decreased the Smartcard SHALL return the response to the terminal for the current AUTHENTICATE command corresponding to OMA BCAST operation for security_policy_extension operation (see Appendix E) with a status code corresponding to 'lack of credit in the service purse'. When the user purse cannot be decreased the Smartcard SHALL return the response to the terminal for the current AUTHENTICATE command corresponding to OMA BCAST operation for security_policy_extension operation (see Appendix E) with a status code corresponding to 'lack of credit in the user purse'. When the play-back counter is invalid or equals zero, the Smartcard SHALL return the response to the terminal for the current AUTHENTICATE command corresponding to OMA BCAST operation for security_policy_extension operation (see Appendix E) with a status code corresponding to 'Play_back counter invalid or equal to zero'.

The secure function SHALL NOT send a verification message as a response to an STKM even in the case where the V-bit in the STKM message is equal to 1.

When the TEK is returned by the secure function to the terminal, the TEK is in the clear unless the secure function is located on the Smartcard and the Terminal Binding Key (TBK) is used, in which case the TEK is wrapped by the TBK.

6.7.2.1 Enforcement of Parental Control

The mechanisms described in this section are only designed for Smartcard.

On reception of an STKM that contains an access_criteria_descriptor, if the Smartcard supports the enforcement of the parental_control access descriptor, the enforcement of the parental control is processed as follow.

Enforcement of parental control access condition consists on the check between the level_granted stored in the Smartcard against the rating_value received in the STKM for the same rating_type.

The enforcement of the parental control is divided in several processing phases

1. Checking the rating_value: The rating_value transmitted in the STKM is checked against the level_granted stored in the Smartcard for the same rating_type.
2. Checking if PINCODE has been verified.
3. Depending on the outcome of 2, a PINCODE request may be performed. During this step, a PINCODE entered by the user is checked against a PINCODE stored by the Smartcard.

1. Checking of rating_value

The Smartcard SHALL first compare the rating_type received in the STKM against all of the rating_type values stored in the Smartcard.

Depending on the rating_value and the rating_type, the outcome is success or failure:

checking of rating_value: success

If there is a rating_type match, and the level_granted associated to this rating_type in the Smartcard is an equal or more restrictive value than the rating_value received in the STKM, the checking of rating_value ends with success.and the processing of STKM resumes.

If there is no rating_type match, the user is authorized to view the content and the checking of rating_value ends with success and the processing of the STKM resumes.

checking of rating_value: failure

If there is a match, and the level_granted associated to this rating_type is less restrictive than the rating_value received in the STKM, the checking of rating_value ends with failure.

If the checking of rating_value ends with failure and a PINCODE is not defined in the Smartcard, the Smartcard aborts the processing of STKM and indicates to the user that they are not allowed to view this content (see Section 6.7.2).

Note: A PIN is defined in the Smartcard, when a local PIN (as defined in [ETSI TS 102.221]) has been assigned for the parental control function at the manufacture of the Smartcard. The PINCODE function is optional in the Smartcard for the parental control.

If the checking of rating_value ends with failure and a PINCODE is defined in the Smartcard, Step 2 (Checking if PINCODE has been Verified) is executed (see below).

Example of STKM filtering based on STKM Parental control rating and Smartcard rating

	Smartcard rating	**none**	**-10**	**-12**	**-14**	**-16**	**-18**
STKM parental rating	**BCAST "rating"**	**0 (least restrictive)**	**1**	**2**	**3**	**4**	**5 (most restrictive)**
none	**0 (least restrictive)**	**O**	**O**	**O**	**O**	**O**	**O**
-10	**1**	**X**	**O**	**O**	**O**	**O**	**O**
-12	**2**	**X**	**X**	**O**	**O**	**O**	**O**
-14	3	X	X	X	O	O	O
-16	4	X	X	X	X	O	O
-18	5 (most restrictive)	X	X	X	X	X	O

where:

Rating -16 means forbidden to those under 16

X means Smartcard **blocks** the STKM unless a **valid** PINCODE is provided

O means Smartcard accepts the STKM

Note that, in the text, the term 'more restrictive' means that there are more constraints on having access to the content. This typically means the user age is higher, i.e. in the table above rating "-18" is more restrictive than "-10". Note that actual numerical values of rating_value for certain rating_types do not always follow a linear scale, either from less restrictive to more restrictive or vice-versa. The corresponding logical order (from least restrictive to most restrictive) is indicated in the rating_type table in order to indicate explicitly how these should be rated.

Table 21 shows the order of restrictiveness for each of the rating_types range of rating_values.The meaning of the rating_value is defined in Table 31 in Section 7.1.

Table 21: Order of Restrictiveness

rating_type	rating_value in order of increasing restrictiveness
0	1, 2, 3, 4, 5,…etc
1	4, 1, 2, 3
2	6, 5, 4, 3, 2, 1
3	6, 1, 2, 3, 4, 5
4	6, 5, 4, 3, 2, 1
5	2, 1
6	6, 5, 4, 3, 2, 1
7	7, 1, 2, 3, 4, 5, 6
8	6, 5, 4, 3, 2, 1
9	1, 2, 3, 4, 5
10	0x00, ….0xFF

2. Checking if PINCODE has been verified

If the checking of rating_value ends with failure and a PINCODE is defined in the Smartcard, the Smartcard SHALL check if the PINCODE has been verified previously for the same content. Two possibilities exist: the PINCODE has already been checked successfully, or the PINCODE check failed:

checking of PINCODE: success

If the PINCODE has been previously verified with success, the parental control ends with success and the processing of STKM resumes.

The Smartcard SHALL NOT request that a PINCODE is entered if the PINCODE has been previously verified with success for the same content (i.e. when the SEK/PEK_ID and rating_type/rating_value pair is the same in the STKM). Information that the PINCODE has been verified SHALL be stored in the Smartcard and SHALL be reset if the content changes (SEK/PEK_ID or rating_type/rating_value change in the incoming STKM), if the terminal is switched off and if the transmission of STKM has been interrupted. This interruption in the transmission MAY be detected by a gap in the timestamp value in the incoming STKM (width of the gap MAY be adjusted by the service provider at the manufacture stage of the Smartcard) against the value stored in the replay counter of the SEK/PEK_ID.

checking of PINCODE: failure

If the PINCODE has not been verified or the verification process ended with failure, the Smartcard proceeds to request the PINCODE, as per Step 3 (Requesting PINCODE) below.

3. Requesting PINCODE

If the Smartcard needs to request a PINCODE the following applies:

PINCODE has not been initialised:

If the PINCODE has not been initialized in the Smartcard, The Smartcard aborts the STKM processing sending a response to the terminal for the current AUTHENTICATE command corresponding to OMA BCAST operation for parental control operation (see Appendix E) with a status code corresponding to 'PINCODE not initialized' and with the key reference corresponding to the PIN used for parental control in order to request to the terminal a PINCODE initialization processing. At the reception of this response, the terminal asks the user to initialize the PINCODE and sends this new PINCODE to the card using the APDU command CHANGE PIN defined in [ETSI TS 102.221] on the PIN corresponding to the key reference value transmitted in the response of AUTHENTICATE command. After this step, the PINCODE is initialized

NOTE: The PINCODE is initialized in the Smartcard when the user has entered his proprietary PINCODE, deleting then the default PINCODE.

PINCODE has been initialised:

If the PINCODE is already initialized, the Smartcard aborts the STKM processing sending a response to the terminal for the current AUTHENTICATE command corresponding to OMA BCAST operation for parental control operation (see Appendix E) with a status code corresponding to 'PIN required' and with the key reference corresponding to the PIN used for parental control in order to request to the terminal a PINCODE verification processing. At the reception of this response, the terminal asks the user to enter a PINCODE and sends this PINCODE to the card using the APDU command VERIFY PIN defined in [ETSI TS 102.221] on the PIN corresponding to the key reference value transmitted in the response of AUTHENTICATE command.

Depending on whether the VERIFY PIN command is successful or fails, the following applies:

VERIFY PIN: success

If the VERIFY PIN ends with success, the terminal SHALL resend the STKM to the card for the computing of keys.

VERIFY PIN: failure

If the VERIFY PIN ends with failure, the terminal MAY request another entry of PINCODE (3 false entries will block the PIN), or wait until the next STKM is received.

Operation on PINCODE in the Smartcard:

The PIN associated to the parental control is defined as a second application PIN (local PIN) as defined in [ETSI TS 102.221]. The key reference used for this PIN is transmitted in the response of the AUTHENTICATE command corresponding to unsuccessful command for parental control operation (see Appendix E).

The change of the PINCODE is controlled by the terminal using the APDU command CHANGE PIN defined in [ETSI TS 102.221] and using the key reference received in the response of AUTHENTICATE command.

If the PINCODE is blocked in the Smartcard the terminal MAY unblock the PINCODE using the command UNBLOCK PIN defined in [ETSI TS 102.221] and using the key reference received in the response of AUTHENTICATE command. The PINCODE for this function is the UNBLOCK PIN.

6.7.2.2 Enforcement of location_based_restriction

The enforcement of the location_based_restriction is processed as follow:

Beforehand, the secure function has requested to the terminal current location information sending a proactive command 'PROVIDE LOCAL INFORMATION' as described in [3GPP 31.111 v6] or [3GPP2 C.S.0035-A]. To request this

information the secure function, on a polling of the terminal returns a status code '91XX' to the terminal to indicate that a proactive command is pending. The terminal fetches the pending proactive command 'PROVIDE LOCAL INFORMATION', performs it and sends to the secure function the response of the proactive command execution. The terminal SHALL support proactive command 'PROVIDE LOCAL INFORMATION' as described in [3GPP 31.111 v6] or [3GPP2 C.S.0035-A].

The secure function will be able to handle some target_area_type that relies on information provided by the proactive command 'PROVIDE LOCAL INFORMATION'.

If target_area_type is 0x2, then the country_code of the current location of the terminal is used and is compared to the country_code specified in the STKM. The comparison results in the following cases:

o If there is a match and if interpretation flag in the STKM is set to 0 ("normal") and if the override flag is set to 0, then the location_based_restriction ends with the status 'blackout', i.e. the terminal SHALL not render the media streams and then the STKM processing SHALL ends without the processing of the TEK and then is aborted (see Section 6.7.2).

o If there is no match and if interpretation flag in the STKM is set to 1 ("spotbeam") and if the override flag is set to 0, the location_based_restriction ends with the status 'blackout' i.e. the terminal SHALL not render the media streams and then the STKM processing SHALL ends without the processing of the TEK and then is aborted (see Section 6.7.2).

o If there is a match and if interpretation flag in the STKM is set to 0 ("normal") and if the override flag is set to 1, the location_based_restriction ends with the status 'need specific permissions', i.e. the restriction may be ignored if the terminal is able to obtain the necessary permissions. Then the STKM processing resume to check if a PPV for this program is active (see Section 6.7.2).

o If there is no match and if interpretation flag in the STKM is set to 1 ("spotbeam") and if the override flag is set to 1, the location_based_restriction ends with the status 'need specific permissions', i.e. the restriction may be ignored if the terminal is able to obtain the necessary permissions. Then the STKM processing resume to check if a PPV for this program is active (see Section 6.7.2).

o Otherwise the location_based_restriction enforcement ends without blackout, i.e. the terminal MAY render the media streams and then the STKM processing SHALL resume.

If target_area_type is 0x5, then depending of the cell_target_area_type, the Cell Global Identifier or the location Area Identifier or the SID, or the SID+NID…of the current location of the terminal is used and compared to the cell_area_values received in the STKM. The comparison results in the following cases:

o If there is a match and if interpretation flag in the STKM is set to 0 ("normal") and if the override flag is set to 0, then the location_based_restriction ends with the status 'blackout', i.e. the terminal SHALL not render the media streams and then the STKM processing SHALL ends without the processing of the TEK and then is aborted (see Section 6.7.2).

o If there is no match and if interpretation flag in the STKM is set to 1 ("spotbeam") and if the override flag is set to 0, the location_based_restriction ends with the status 'blackout' i.e. the terminal SHALL not render the media streams and then the STKM processing SHALL ends without the processing of the TEK and then is aborted (see Section 6.7.2).

o If there is a match and if interpretation flag in the STKM is set to 0 ("normal") and if the override flag is set to 1, the location_based_restriction ends with the status 'need specific permissions', i.e. the restriction may be ignored if the terminal is able to obtain the necessary permissions. Then the STKM processing resume to check if a PPV for this program is active (see Section 6.7.2).

o If there is no match and if interpretation flag in the STKM is set to 1 ("spotbeam") and if the override flag is set to 1, the location_based_restriction ends with the status 'need specific permissions', i.e. the restriction may be ignored if the terminal is able to obtain the necessary permissions. Then the STKM processing resume to check if a PPV for this program is active (see Section 6.7.2).

o Otherwise the location_based_restriction enforcement ends with the status 'without blackout', i.e. the terminal MAY render the media streams and then the STKM processing SHALL resume.

6.7.3 STKMs and traffic encryption protocols

STKM can be sent over UDP. It is possible to multiplex STKM/UDP with FLUTE packets (on the same IP transport address but on a separate IP port – refer to Section 10.1 on how this is signaled)

If the traffic_protection_protocol equals to TKM_ALGO_DCF, then the STKM MAY be delivered as a separate object inside a FLUTE session, together with the protected traffic, having its own FDT entry.

SRTP

3GPP MBMS security [3GPP TS 33.246 v7], on which the Smartcard Profile is based, is designed for use with SRTP. It follows that the STKM defined in Section 6.7 is compatible with SRTP. SRTP encryption SHALL be indicated by the traffic_protection_protocol value in the STKM. The SRTP Master Key (MK, 128 bits) and Master Salt (MS, 112 bits) SHALL be sent within the STKM. For compatibility with the DRM Profile a NULL MS MAY be sent.

The correct TEK to use to decrypt the data is indicated using the MKI (Master Key identifier) field, which SHALL be included in the SRTP packets as defined in [RFC 3711]. The MKI SHALL be the TEK ID, unless compatibility with MBMS terminals is required in which case the MKI SHALL be a concatenation of SEK/PEK ID and TEK ID, i.e. MKI = (SEK/PEK ID || TEK ID). See Section 11.3.2 for further details on the requirements related to sharing protected traffic streams

The key derivation rate MAY be zero.

ISMACRYP

The Smartcard Profile STKM, defined in Section 6.7, is compatible with ISMAcryp. For content encryption, the use of ISMACryp SHALL be signaled by traffic_protection_protocol value in the STKM. The Smartcard Profile TEK ID corresponds to the key_indicator in the DRM Profile STKM. The key_indicator sent in the OMABCASTAUHeader as part of the encrypted stream SHALL correspond to the TEK ID (2 bytes) sent in the EXT MBMS payload of the STKM. Note that, unlike for SRTP, there is no requirement for compatability with MBMS only terminals and therefore there is no case in which the key_indicator SHALL be a concatenation of SEK/PEK ID and TEK ID, i.e. SEK/PEK ID || TEK ID. The 128 bit TEK SHALL be transported, as for SRTP, in the KEMAC field of the STKM.

If no SRTP authentication is used the 128 bit encryption key SHALL be sent instead of the MK. The MS is not used. Salt keys SHALL be signaled in SDP. No SRTP key derivation is done in ISMACryp.

If SRTP authentication is used, MK (128 bits) and MS (112 bits) SHALL be sent within the STKM and used to derive encryption and authentication keys as per SRTP.

IPsec

IPsec encryption SHALL be signaled by traffic_protection_protocol value in the BCAST STKM. The 4-byte SPI sent in IPsec packets SHALL consist of constant prefix 0x0001 followed by the 2-byte MTK ID. In other words, SPI = (0x0001 || MTK ID).

The security policy information is as specified in this document, and the CS ID map type SHALL be set to value '1' as defined in [RFC4563].

The MTK SHALL be transported in the KEMAC field. The 16-byte IPsec encryption key (the key for ESP encryption) SHALL be derived from the MTK as specified by MIKEY ([RFC3830], Section 4.1.3) using the "encryption key" constant. If traffic authentication is used, the 16-byte Traffic Authentication Seed (TAS) SHALL be derived from the MTK as specified by MIKEY using the "authentication key" constant. The IPsec authentication key (TAK, 20 bytes) is derived from the TAS, as described in Section 9.1. No salt is used.

6.8 Layer 4: Traffic Encryption

Layer 4 corresponds to the BCAST 4-layer model key hierarchy. The protection of data in case of streaming and file delivery respectively for both service and content protection is described for the Smartcard Profile.

6.8.1 Streaming Delivery

6.8.1.1 Service Protection of Streams

Broadcast streams that are signalled as having service protection by the SG via the protection_after_reception field are encrypted by TEKs using IPsec, SRTP or ISMACryp.

How to obtain the relevant information from the SG to request the appropriate SEK or PEK (used for TEK protection) to access with the TEK the protected stream is explained in Section 6.10. If the LTKM extension payload is absent, and the "protection_after_reception" field in the STKM = 0x03 (i.e. "Service Protection"), then upon obtaining the TEK from the Smartcard, the terminal can either store the TEK or record the content in the clear or do both.

6.8.1.2 Content Protection of Streams

Broadcast streams that are signaled as having content protection may be recorded as defined in this specification. However, for recorded material having content protection, appropriate rights need to be obtained via a Broadcast Permissions Issuer.

For terminals using the Smartcard Profile, the appropriate key material can be requested based on the Program or Service ID.

The Permissions Issuer can provide content protection for the Smartcard Profile allowing an implicit play once right. Once the server issues the appropriate SEK or PEK to the terminal / Smartcard, the terminal SHALL interpret the obtained keys relating to the recorded stream as being "play once" unless otherwise indicated by a a security policy extension contained in the EXT BCAST payload in the LTKM (see Section 6.6.2). If the EXT BCAST payload is not present in the LTKM or does not contain a security policy extension, it SHALL not be possible to use the SEK/PEK to access the same content more than once. This is achieved through the processing described in Section 6.6.4.

As the key material provides access to recorded content stored in the terminal, preventing unauthorized access to these keys is extremely important. It is therefore recommended that they are stored in a secure storage area and protected appropriately during their limited lifetime. For an implementation using GBA_U, the Smartcard can deliver TEKs to the terminal if the adapted PDCF is used to record a TEK key stream. For content protection, the terminal-Smartcard interface SHOULD be secured. This includes appropriate terminal authentication to the Smartcard.

For (U)SIM Smartcard Profile, the Smartcard-terminal interface SHOULD comply to [ETSI TS 102.484] and [3GPP TS 33.110]. For (R-)UIM/CSIM Smartcard Profile terminals, the Smartcard-terminal interface is expected to comply to relevant 3GPP2 specifications to be developed.

6.8.1.3 Permissions Management using the Smartcard Profile for Content Protection of Streams

If the EXT BCAST payload is not present in the LTKM, the SEK/PEK in Smartcard Profile is based on an implicit "play once" permission. This "play once" functionality can be used by the BSM to enable more complex constraints relating to the use of a SEK/PEK, e.g. unlimited access to the appropriate SEK/PEK for a given time period or controlled access to the SEK/PEK for a given number of times. In all cases, where the EXT BCAST payload is not present in the LTKM, the Smartcard Profile terminals are forced to request a new SEK/PEK for every access to content.

If the EXT BCAST payload is also present in the LTKM, the security_policy_extension value defines the rights applicable to the LTKM. The security_policy_ext enables the provision of extended rights such as Pay Per View (PPV) and Pay Per Time. The types of extended rights that can be offered are described in Section 6.6.2. In order to request the tokens and / or consumption rules required to provide PPT and PPV functionality, the "Token Purchase Request" message defined in BCAST in [BCAST10-Services] SHALL be used.

If broadcast streams are protected and need content protection consumption rules, this is signaled via ProtectionType in the SG and via the protection_after_reception values in the STKM. For the Smartcard Profile, this means there SHALL be mutual terminal-server authentication and there SHALL be a secure authenticated channel as described above and there SHALL be the standard Smartcard BSM authentication (Section 13), before the delivery of the LTKM.

Hence the following steps SHOULD be followed when requesting key material for content protected streams:

1. Identify the Permissions Issuer URI and SEK/PEK ID

2. Initiate mutual terminal BSM authentication (see Section 6.5)

3. Initiate mutual Smartcard BSM authentication (see Section 6.11.2)

4. Establish / enable the secure authenticated channel between the Smartcard and terminal (see Section 6.11.2)

5. Request the appropriate SEK or PEK using the "Token/LTKM Purchase Request" message in [BCAST10-Services] (see Section 6.6). The requested key identifier is the SEK / PEK ID.

6.8.2 File Delivery

6.8.2.1 Service Protection of Download Data using DCF

This section contains material from MBMS text in [3GPP TS 33.246 v7]. The mechanism described in this section was adopted from [3GPP TS 33.246 v7] and adapted to BCAST needs.

BCAST terminals SHALL support download protection using DCF. BCAST servers MAY support download protection using DCF.

The same mechanism can be used to protect PDCF files. This is optional for both terminal and server.

Service protection of download data uses IPsec or DCF encryption protocol. In case of DCF encryption protocol, DCF file is used as a container for ciphered file data. The DCF container also identifies the keys used in protecting the data. Use of IPsec for Service Protection of download data is Optional.

Each file is encrypted using a single TEK, as explained in Section 9.3.2.

For the Smartcard Profile, KeyID takes its values as follows:

- KeyID is defined as the base64 encoded concatenation of (SEK or PEK ID || TEK ID) (i.e. equivalent to Key Domain ID || SEK/PEK ID || TEK ID).

Keys can be acquired by using the PermissionsIssuerURI indicated via the KeyIssuerURL in the Key Info box.

6.8.2.2 Content Protection of Download Data using DCF for Smartcard Profile

BCAST terminals SHALL support download protection using DCF. BCAST servers MAY support download protection using DCF.

The same mechanism can be used to protect PDCF files. This is optional for both terminal and server.

The DCF format defined in Section 6.8.2.1 above can also be used for content protection for the Smartcard Profile. Content protection rules are identified to the terminal by the protection_after_reception value in the STKM, and to the Smartcard by the security_policy_extension value in the LTKM.

Keys can be acquired by using the PermissionsIssuerURI indicated via the KeyIssuerURL in the Key Info box.

OMA DRM v2.0 MAY be used for download content protection together with the Smartcard Profile.

6.9 Recording

Please refer to Section 8 for details on recording.

6.9.1 Playback of Content Protected Recorded Streams

This section describes how streamed content encrypted at the content level using ISMACryp and recorded in the adapted PDCF together with STKM track can be re-read locally. Content protection is indicated to the terminal by the protection_after_reception value in the STKM. The Smartcard Profile mechanisms for service protection can be used as described briefly for content protection. Unless indicated otherwise standard MBMS mechanisms are used. The description below mentions only GBA_U, but GBA_ME can be used also for Content Protection of recorded material using the (U)SIM.

1. Read the first STKM from the STKM track and send it to the Smartcard if using GBA_U via the Secure Authenticated Channel (SAC) between the terminal and Smartcard as defined in [ETSI TS 102.484] and [3GPP TS 33.110] (unless the Terminal Binding Key is required in which case the SAC is optional) or move to step 5
2. If the TEK is returned then decrypt the encrypted content. Otherwise, go to step 4. Note: This condition indicates that the SEK/PEK corresponding to the desired TEK is not available at the Smartcard, such that the terminal must request its delivery from the BSM.
3. Repeat 1 to 2 until the end of the file or until the TEK is not returned (this is indicated by a failure message sent by the Smartcard)
4. Identify the MBMS Service Protection Description via the RightsIssuerURL in the OMADRMCommon HeadersBox. Obtain the appropriate information regarding the BSM for the service and the User Service ID (this is the GlobalPurchaseItemID || PurchaseDataID) and KeyDomainID.
5. Identify the SEK/PEK from the recorded STKM track
6. Identify the Timestamp field (TS) from the current STKM in the STKM track
7. Identify the Timestamp (TS) from the last STKM in the STKM track
8. Mutually authenticate with the Permissions Issuer (BSM) and establish an HTTPS tunnel as described in Section 6.11
9. Request the SEK/PEK from the BSM using the "Token Purchase Request" message, as defined in [BCAST10-Services].
10. Receive the LTKM with the requested SEK/PEK from the BSM
11. Read the first STKM from the STKM track and send it to the Smartcard if using GBA_U.
12. Receive the TEK and decrypt the encrypted content
13. Read the next STKM from the STKM track and send it to the USIM if using GBA_U
14. Receive the TEK and decrypt the encrypted content
15. Repeat 13 to 14 until the end of the file

The following must be noted when using the above mechanism:

- The "Token Purchase Request" message in step 9 above is specific to a BCAST client and can only be understood by a BCAST BSM. Hence the Service Provider must ensure the RightsIssuerURL allows the BSM to know the request is from a BCAST client. The LTKM containing tokens and / or PPV / PPT permissions, received in response to the "Token Purchase Request" message, has to be understood by the Smartcard.
- The MBMS replay protection mechanisms mean any "rewind" forces a new SEK/PEK request unless TEKs are buffered in the terminal. Hence buffering is recommended until end of play.
- The OMA General Extension and the new Smartcard security policies permit to authorize replay content according the security policy associated to the SEK/PEK.
- The Timestamp (TS) lower and upper limits allow a finer management of rights on the server side rather than basing charging on the full duration of the program defined by the TEK ID.
- The Permissions Issuer (BSM) must keep a history of SEK/PEKs.
- The delivery of the STKM must only be done through a Secure Authenticated Channel to ensure TEKs are returned via a secure channel and not in the clear, unless the Terminal Binding Key is used in which case the SAC is optional.

The above mechanisms SHOULD be implemented for terminals using content protection with the Smartcard Profile (see Section 6).

6.10 SG Signalling (Description of Service Access)

6.10.1 SG Signalling for SEK/PEK Acquisition

The Service Guide (SG) provided by OMA BCAST provides information regarding available services and allows a user to subscribe to or acquire purchase items. For example, information regarding available services is delivered via the Service fragments, and information regarding available purchase items is delivered via the PurchaseItem fragments and PurchaseData fragments. Each fragment contains its own unique identifier. GlobalPurchaseItemID is defined in the PurchaseItem fragment and PurchaseDataID is defined in the PurchaseData fragment. The concatenation of GlobalPurchaseItemID and PurchaseDataID results in a parameter equivalent to the MBMS User Service ID defined in MBMS.

Having completed subscription/purchase of a purchase item/broadcast service, to subsequently enable rendering of service/content, the appropriate SEK/PEK must be acquired by the BCAST Terminal. The Access fragment clearly identifies the type of protection offered (service protection or content protection or both) and the supported key management systems, e.g. the DRM Profile or the Smartcard Profile. In the case of the Smartcard Profile two possible means exist for acquiring the appropriate SEK/PEK, via the LTKM:

- via the MBMS USD contained in the Session Description fragment (Section 6.10.1.1) or
- via session description information extended to include BCAST protection-specific information; these may be provided directly in the Access fragment or in the Session Description fragment (Section 6.10.1.2).

The PurchaseChannel fragment can be linked to a PurchaseItem fragment to provide further information via the PortalURL or indicate to the terminal that it must contact the PortalURL for any subscription (see Section 6.10.3), rather than send a Service Request directly to the PurchaseURL.

The association between the protected services or contents and the corresponding keys (SEK/PEK) necessary to access them is done via the “ProtectionKeyID” element, which can be present in the Service, Content or Purchase Item fragments. This allows the mapping of keys with a specific service, content or group of services/contents, respectively, so that the terminal can determine whether it already has valid access keys or not.

6.10.1.1 MBMS USD Method for Acquiring SEK/PEK

As specified in [BCAST10-SG], the SessionDescription fragment, referenced by the Access fragment, may contain MBMS User Service Description (USD), the latter specified by [3GPP TS 26.346 v7]. If the MBMS USD is used as the entry point, it SHALL contain the relevant service information required by the terminal to register to for the services that it is advertising. For convenience these steps are summarised below:

1. During the MBMS announcement procedure, the terminal receives the full domain name of the BSM (BM-SC) from which it can deduce the IP address to send the “Registration Request” and “LTKM Request” messages, as defined in [BCAST10-Services]. Note that the Smartcard Profile “Registration Request” and “LTKM Request” messages correspond to the MBMS “User Service Registration” and “MSK Request” messages respectively.

2. The terminal sends a “Registration Request” message to the BSM (BM-SC) for the services to which it is subscribed. As defined in [BCAST10-Services], the following information SHALL be included in the “Registration Request” message:

- Indication that the UE requests to register to the MBMS User Service;
- One or more MBMS User Service ID(s), where each MBMS User Service ID corresponds to a concatenation of GlobalPurchaseItemID and PurchaseDataID.

In this situation the RightsIssuerURI contained in the Access fragment and the BaseCID contained in the Service fragment are to be ignored as the relevant parameters are provided in the MBMS USD fragments. This is summarised in the table below.

Table 22: Parameters used when using MBMS USD

Parameter	Value / Description
Session Description Fragment contents	MBMS USD.
RightsIssuerURI (in Access fragment)	Not used / ignored. MBMS USD contains a Service Protection Description, which identifies the key management server which the terminal should register with, and request SEK/PEK from.
BaseCID (in Service or Content fragment)	Not used / ignored as this applies to DRM Profile only. Note: Equivalent identifier in Smartcard Profile is provided in two possible ways: 1) in the MBMS USD (represented by the serviceID attribute of userServiceDescription element in the User Service Description; this serviceID is equivalent to the MBMS User Service ID); 2) in the PurchaseItem and PurchaseData fragments of the BCAST Service Guide [BCAST10-SG].

6.10.1.2 Session Description Method for Acquiring SEK/PEK

In this scenario, session description information, either embedded in the Access fragment or provided in a standalone Session Description fragment, and containing Smartcard Profile specific protection information (in addition to nominal session information) is used. The session description is formatted according to the syntax of Session Description Protocol (SDP). The BCAST Service Guide provides the global purchase item identifier (*globalPurchaseItemID* of Purchase Item fragment) and purchase data identifier (*id* attribute of Purchase Data fragment). These two identifiers are concatenated to create the MBMS User Service ID.

In this method, the SDP file provides information on the data and STKM streams, as well as other service protection parameters equivalent to those found in MBMS USD's Service Protection Description. This would typically be the case for a non-MBMS bearer used to deliver the data, with the interactive communication channel being used to provide LTKMs. The TEK delivery could be done in-band with the data. Depending on the bearer, this could be an MBMS or non-MBMS network.

Registration to the service is achieved by sending the "Service Registration" message as explained above in Section 6.10.1.1.

The relevant parameters are summarised in the table below.

Table 23: Parameters used when using Session Description

Parameter	Value / Description
Session Description information type	SDP in Access fragment, or SDP-formatted data in Session Description fragment referenced by Access fragment.
RightsIssuerURI (in Access fragment)	Used to reference the key management server, i.e. the BSM, which the terminal should register with and to request SEK/PEK from.
BaseCID	Not used / ignored as applies to DRM Profile only. Note: Equivalent identifier is provided in the PurchaseItem and PurchaseData fragments of the BCAST Service Guide [BCAST10-SG].

6.10.2 Description of Service Access for Smartcard Profile using BCMCS Information Acquisition

Section 5 of BCMCS specification [3GPP2 X.S0022-A] describes BCMCS service discovery, subscription and registration procedures using BCMCS information acquisition process. For terminals with (R)-UIM smartcards, those procedures are used for OMA BCAST application information and security parameters, including keys. The RightsIssureURI and SubscriptionManagerURI may be pre-provisioned or acquired via BCMCS Information Acquisition exchange(s).

When using BCMCS Information Acquisition, the RightsIssuerURI contained in the Access fragment and the BaseCID contained in the Service fragment are to be ignored as the relevant parameters are provided in the BCMCS Information Acquisition elements. This is summarised in the table below.

Table 24: Parameters used when using BCMCS Information Acquisition

Parameter	Value / Description
Session Description Reference Type	BCMCS Information Acquisition
RightsIssuerURI	Not used / ignored. Such information is provided by BCMCS Security Parameters.
BaseCID	Not used / ignored. BCMCS Application Information and BCMCS Security Parameters are linked.

6.10.3 Web Portal used as Entry Point

While the Service Guide can provide all the information to obtain information on available services as well as information relating to acquisition of LTKMs, as explained above, another possibility for terminals having access to an interaction channel is to use a Web Portal.

If the PortalURL in the PurchaseChannel fragment linked to a PurchaseItem indicates that the PortalURL should be contacted to obtain further information and subscribe to services, the terminal SHOULD contact the PortalURL. Furthermore, the terminal can send the GlobalPurchaseItemID, along with the idRef of the PurchaseData fragment, to the web portal associated with the PortalURL to indicate to the portal the PurchaseItem of interest.

When the user attempts to subscribe to a service via the portal, two scenarios are possible (this is also described in Section 5.1.8 of [BCAST10-Services] and Sections 5.4.7.2.6 and 5.4.7.2.7 of [BCAST10-Architecture]):

1. The portal is unable to determine whether or not the Smartcard/Terminal has established a valid SMK and SRK with the BSM (e.g. whether or not the bootstrapping procedure has been run in the case of (U)SIM, or whether TK and Auth-Key have been derived from the pre-provisioned RK in the case of (R-)UIM/CSIM). In this case, the portal sends the Terminal the Smartcard Profile trigger as defined in section 5.1.8.1 of [BCAST10-Services] to force the terminal to run the Registration procedure, which in turn requires that the bootstrapping procedure has been run in the case of (U)SIM. The message flow for this scenario is described in section 5.4.7.2.7 of [BCAST10-Architecture].

2. The the portal is able to determine that theSmartcard/Terminal has established a valid SMK and SRK with the BSM (e.g. whether or not the bootstrapping procedure has been run in the case of (U)SIM, or whether TK and Auth-Key have been derived from the pre-provisioned RK in the case of (R-)UIM/CSIM). Note that how this may be achieved is out of scope of this specification).

6.11 BCAST Client ID for Smartcard Profile

This section describes how a BCAST Client identifier MAY be sent by the Terminal or MAY be requested by the BSM (Permissions Issuer) during MBMS User Service Registration.

This MAY allow the BSM (Permissions Issuer) to check software / firmware versions and make a decision as to whether or not access can be granted to the terminal requesting the service.

The mechanisms described in this section are OPTIONAL for the network to use and MANDATORY for the terminal to support if they have a BCAST client ID, a terminal certificate and if they support the Smartcard Profile for service protection. The mechanisms are MANDATORY for the terminal to support for the Smartcard Profile for content protection, i.e. the BCAST client ID and terminal certificate are MANDATORY.

Security: Message integrity and authentication is guaranteed by using certificate-based mutual authentication between Terminal and Application Server for access to NAF using HTTPS as specified in [3GPP TS 33.222 v6], where HTTPS SHALL be based on .SSL3.0 [SSL30] and TLS 1.0 [RFC2246].

6.11.1 BCAST Client Identifier

The format defined below SHALL be used as a unique BCAST client identifier for the Smartcard Profile.

The BCAST_Client_ID SHOULD be stored in the BCAST Management Object (BCAST MO) as specified in [BCAST10-Services]. Note that the BCAST_Client_ID may be stored elsewhere in the device.

Note that it is NOT mandatory for every terminal to have a BCAST Client Identifier for service protection, it is only MANDATORY for content protection.

Table 25: BCAST Client ID

BCAST_Client_ID	Length (in bits)	Type
TerminalIdentifierType	1	byte
TerminalIdentifier	16	byte
TerminalFirmwareVersionNo	2	byte
ClientManufacturerCode	2	byte
ClientModelNo	2	byte
ClientSerialNo	3	byte
ClientSoftwareVersionNo	2	byte

Coding and Semantics of Attributes:

The Terminal identifiers are specific to the actual device used to receive mobile broadcast services and are defined in the Table below:

Table 26: Terminal Identifiers

Parameter	Definition	
TerminalIdentifierType	**Value**	**Type**
	0	IMEI (International Mobile Equipment Identity) as defined in [3GPP TS 23.003 v6].
	1	MEID (Mobile Station Equipment Identifier) as defined in [3GPP2 C.S0072-0]
	2	Globally Unique Identifier (GUID)
	3	Media Access Control (MAC) address in EUI-48 or EUI-64 format
	4-127	*for future use*
	128-255	*for private use*
TerminalIdentifier	The identifier of the terminal, in the format specified through TerminalIdentifierType.	

	Identifiers that occupy less space than 16 bytes are padded with leading zeros to fill 16 bytes after padding.
TerminalFirmwareVersion	Version number indicating the firmware version of the terminal. This version number is assigned by the Terminal manufacturer. This version number SHALL be increased following a secure firmware upgrade.

The Client identifiers are specific to the BCAST client installed in the Terminal allowing access to the BCAST services and are indicated in the Table below:

Table 27: BCAST Client Identifiers

Parameter	Definition
ClientManufacturerCode	Indicates the BCAST client manufacturer. Values for ClientManufacturerCode are available in an OMNA [OMNA] registry.
ClientModelNo	Model number for a specific manufacturer code. Numbering assignment is left to the manufacturer.
ClientSerialNo	Unique serial number specific to the BCAST client manufacturer code and model number. Serial number assignment is left to the manufacturer. Note that this is unique for a given ClientManufacturerCode and ClientModelNo pair
ClientSoftwareVersion	Version number indicating the software (or firmware) version of the terminal. This version number is assigned by the BCAST client manufacturer. This version number SHALL be increased following a secure software (or firmware) upgrade.

6.11.2 Signalling Protocols used for Smartcard Profile

This section explains how the information presented above MAY be sent or requested for the Smartcard Profile during the BCAST service provisioning message sequence as defined in [BCAST10-Services]. The Figures below summarise the possible messages exchanged. Italics are used to indicate the parameters / messages related to the BCAST_Client_ID. The first Figure illustrates the case where the BSM/Permissions Issuer requests the BCAST_Client_ID. The second Figure illustrates the case where the Terminal sends its BCAST_Client_ID to the BSM/Permissions Issuer.

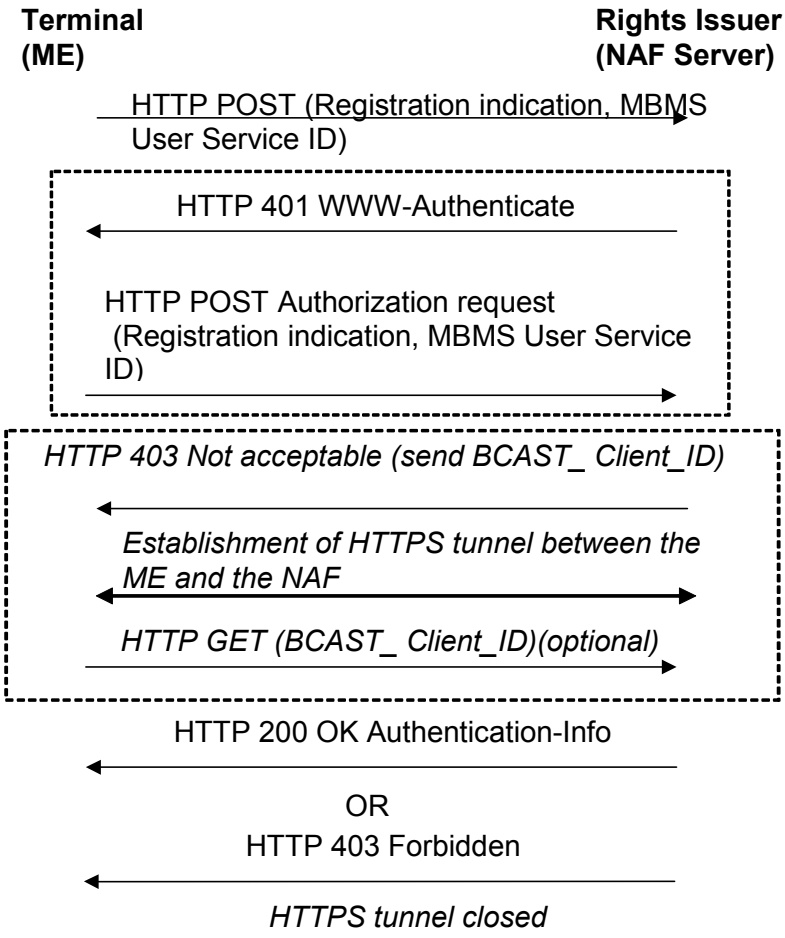


Figure 4 – Mutual Authentication and BCAST Service Provisioning Messages when BSM/Permissions Issuer requests BCAST_Client_ID

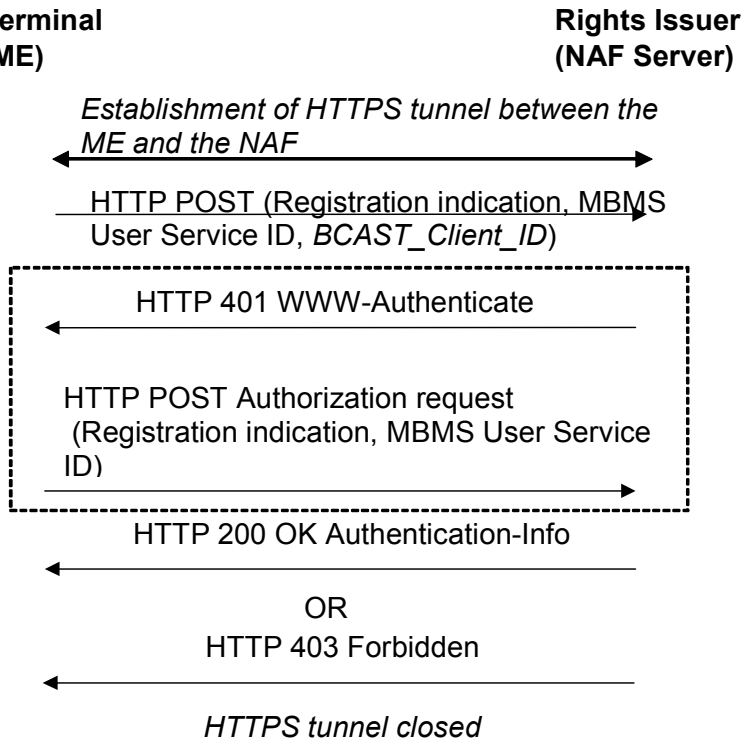


Figure 5 – Mutual Authentication and BCAST Service Provisioning Messages when Terminal sends BCAST_Client_ID

The following sections describe the messages in more detail.

6.11.2.1 Certificate-based Mutual Authentication between Terminal and BSM/Permissions Issuer

Before initiating a BCAST Service Provisioning message request, the Terminal and the BSM/Permissions Issuer MAY establish a HTTPS tunnel with certificate-based mutual authentication between the Terminal and the application server as described in TS [3GPP TS 33.222 v6] Section 5.5 "Certificate based mutual authentication between Terminal and application server" for Release 6. This SHALL be done if the Terminal intends to send a BCAST Client ID at the start of BCAST Service Provisioning message request as described below.

6.11.2.2 Terminal Sending BCAST_Client_ID at start of BCAST Service Provisioning Message Request

The BCAST_Client_ID identifier MAY be sent automatically by the Terminal in the initial HTTP Post Request during the start of the BCAST Service Provisioning message request as defined in [BCAST10-Services]. If it sends a BCAST_Client_ID it SHALL do so through an HTTPS tunnel, as described above in Section 6.11.2.

The BCAST_Client_ID SHALL be sent using the following notation:

User-Agent:BCAST_Client_ID=*BCAST_Client_ID*

Where:

"BCAST_Client_ID=" is text allowing the server to identify the BCAST client ID and *BCAST_Client_ID* is the actual value.

BCAST_Client_ID is Base64 encoded.

If the Terminal does not send the BCAST_Client_ID in the HTTP Post Request, then the BSM/Permissions Issuer MAY ask for it as described below in Section 6.11.2.

6.11.2.3 BSM/Permissions Issuer Requesting BCAST_Client_ID

If the Terminal has NOT sent the BCAST_Client_ID in the HTTP Post Request, the BSM/Permissions Issuer MAY request the BCAST client identifier using the following request:

Table 28: BSM/Permissions Issuer Requesting BCAST_Client_ID

```
HTTP/1.1 403 Not acceptable
Server: BCAST BSM
Date: Thu, 08 Jan 2004 10:13:18 GMT
send_BCAST_Client_ID
```

where send_BCAST_Client_ID is text.

6.11.2.4 Terminal Sending BCAST_Client_ID to BSM/Permissions Issuer following Request

Upon reception of the request for the BCAST client id, the terminal MAY be incapable of sending this identifier (as it is optional).

If the terminal recognizes the request for a BCAST_Client_ID (as defined in Section 6.11.1), it SHOULD establish an HTTPS tunnel between the Terminal and BSM/Permissions Issuer as described in Section 6.11.2 and then send the BCAST_Client_ID using the following response:

Table 29: Terminal Sending BCAST_Client_ID to BSM/Permissions Issuer

```
GET / HTTP/1.1
User-Agent: BCAST_Client_ID=BCAST_Client_ID
Date: Thu, 08 Jan 2004 10:13:18 GMT
```

Where "BCAST_Client_ID=" is text allowing the server to identify the BCAST client ID and *BCAST_Client_ID* is the actual value.

BCAST_Client_ID is Base64 encoded.

If the BCAST Client does not have a BCAST_Client_ID it is recommended that the above message is sent using an empty User-Agent entry without establishing an HTTPS tunnel. Note that this may result in the BSM/Permissions Issuer refusing registration.

6.11.2.5 BSM/Permissions Issuer Accepting BCAST_Client_ID

If the BCAST_Client_ID provided by the terminal to the BSM/Permissions Issuer is deemed acceptable, then the normal BCAST Service Provisioning Response message is sent, i.e. HTTP 200 OK as defined in [BCAST10-Services].

6.11.2.6 BSM/Permissions Issuer Refusing Access to Terminal

If the Terminal does not send a BCAST_Client_ID following the request from the BSM/Permissions Issuer, it MAY refuse access to the Terminal by sending an HTTP 403 Forbidden message.

If the BSM/Permissions Issuer refuses access to the Terminal after receiving its BCAST_Client_ID, it SHALL send an HTTP 403 Forbidden message.

6.11.3 Security Requirements on BCAST_Client_ID and Terminal Private Key

The BCAST_Client_ID SHALL be stored securely and updated accordingly following secure upgrades.

The Terminal private key SHALL also be stored securely.

If the BCAST_Client_ID is used by the BSM/Permissions Issuer to check the BCAST client version, then clearly the information provided by the terminal must be trusted. The BCAST_Client_ID SHALL be transported over a secure, authenticated channel between the terminal and the BSM/Permissions Issuer, as described in [3GPP TS 33.222 v6].

6.12 Terminal-Smartcard Interface

The interface between the Terminal and the 3GPP (U)SIM smartcard SHALL comply with the specifications in [3GPP TS 31.101 v6].

7. Short Term Key Message – Common Attributes

STKMs of the DRM Profile and the Smartcard Profile share a set of common attributes. These common attributes are introduced below.

Section 7.1 introduces the descriptors for access_criteria_descriptor_loop. Section 7.2 introduces used constant values. Section 7.3 introduces coding and semantics of the common STKM attributes.

7.1 Descriptors for access_criteria_descriptor_loop

Tag	8	uimsbf
Length	8	uimsbf
Value	8xlength	bit string

The Access Criteria Descriptor loop is an extension mechanism to allow the addition of new access criteria in the future versions of this specification. The device SHALL ignore Access Criteria Descriptors that it doesn't support. It is OPTIONAL for the BCAST Terminal to support Access Criteria Descriptors.

A single Access Criteria Descriptor can carry one or more access criteria.

The following Access Criteria Descriptors have been defined:

parental_rating – is the parental rating of the program. The descriptor tag for this descriptor is 1. The value for this descriptor is encoded as follows:

Table 30: parental_rating Access Criteria Descriptor

parental_rating descriptor	Length (in bits)	Type
rating_type	**7**	**uimsbf**
country_code_flag	**1**	**uimsbf**
rating_value	**8**	**uimsbf**
if (country_code_flag == TKM_FLAG_TRUE) {		
number_of_country_codes	**8**	**uimsbf**
for (i = 0; i < number_of_country_codes; i++) {		
country_code	**16**	**uimsbf**
}		
}		

The optional list of **country_code** specifies that the rating is for a specific list of one or more countries, which is analogous to the MPEG-7 definition of the ParentalGuidanceType. Each country code is a 2 ascii-character value that must be compliant with [ISO-3166].

The **rating_type** with values 0 through 8 specifies one of the content rating systems that are defined by MPEG-7 and **rating value** is an integer with the meaning that is dependent on the rating_type. The rating values for rating type 0 through 8 are exactly as they had been defined by MPEG-7. Rating type 9 is for the parental rating for the German system. Some values of the **rating_type** are reserved for future use.

Table 31: Rating Types

rating_type	Name	Description	rating_value
0	N/A	ETSI EN 300 468 for the parental_rating_descriptor in DVB systems	Minimum allowable age.
1	JapaneseAdmCommMotionPictureCodeEthicsParentalRatingCS	Japanese Motion Picture Parental Rating	1=PG12 2=R-15 3=R-18 4=None

2	ICRAParentalRatingCS	Internet Content Rating Association Parental Rating	1=Level4 2=Level3 3=Level2 4=Level1 5=Level0 6=None
3	MPAAParentalRatingCS	MPAA Parental Rating	1=G 2=PG 3=PG-13 4=R 5=NC-17 6=NR
4	ICRAParentalRatingNudityCS	Internet Content Rating Association Parental Rating for Nudity	1=Level4 2=Level3 3=Level2 4=Level1 5=Level0 6=None
5	RIAAParentalRatingCS	RIAA Parental Rating	1=Parental advisory 2=None
6	ICRAParentalRatingSexCS	Internet Content Rating Association Parental Rating for Sex	1=Level4 2=Level3 3=Level2 4=Level1 5=Level0 6=None
7	MPAAParentalRatingTVCS	MPAA Parental Rating for TV	1=TVY 2=TVY7 3=TVG 4=TVPG 5=TV14 6=TVMA 7=None
8	ICRAParentalRatingViolence		1=Level4 2=Level3 3=Level2 4=Level1 5=Level0 6=None
9	GermanyFSK	German Freiwillige Selbstkontrolle der Filmwirtschaft Rating System	1=0 (Freigegeben ohne Altersbeschränkung) 2=6 (Freigegeben ab 6 Jahren) 3=12 (Freigegeben ab 12 Jahren) 4=16 (Freigegeben ab 16 Jahren) 5=18 (Keine Jugendfreigabe)
10	BCAST rating		0x00: least restrictive … 0xFF: Most restrictive

location_based_restriction – location-based restrictions on the rendering of content based on [BCAST10-SG].

An alternative service can be specified in the service guide [BCAST10-SG]. It is possible to specify an alternative service as a blank screen with a burnt-in text notifying the user of the blackout. In this case, the burnt-in text can be conveyed as a subtitle in 3GPP Timed Text format as described in [BCAST10-Services].

If a terminal supporting a location_based_access_descriptor receives an STKM with this descriptor and the terminal is not able to obtain its current location or is not able to process the STKM, then the terminal MUST NOT decrypt the Traffic Key and possible Program Key contained in this STKM and MUST NOT decrypt the corresponding content. A terminal MAY be capable of determining at least its cell ID using a native bearer signalling mechanism. A terminal MAY in addition utilize a

suitable position location protocol to determine its position. Examples are SUPL [OMA SUPL] or MLP [OMA MLP]. In the case that a terminal is not capable of determining location information other than a cell ID, additional location information (other than a list of blacked out cell IDs) provided in the location_based_restriction Access Criteria Descriptor MAY be ignored. In the case that the terminal is able to detect multiple cell IDs using native bearer signalling mechanisms, for the purpose of checking against a possible blackout it MAY select the same cell ID that is being used to receive the protected service.

The descriptor tag for this descriptor is 2. The value for this descriptor is encoded as follows:

Table 32: location_based_restriction Access Criteria Descriptor

location_based_restriction descriptor	Length	Type
version	32	Uimsbf
interpretation	1	Uimsbf
override	1	Uimsbf
reserved_for_future_use	6	Bslbf
number_of_target_areas	8	uimsbf
for (i=0; i < number_of_target_areas; i++) {		
target_area_type	4	bslbf
reserved_for_future_use	4	bslbf
if (target_area_type == 0x1) {		
shape()		
}		
if (target_area_type == 0x2) {		
country_code	16	uimsbf
}		
if (target_area_type == 0x3) {		
name_area_length	8	uimsbf
name_area	8*name_area_length	bslbf
}		
if (target_area_type == 0x4) {		
zip_code_length	8	uimsbf
zip_code	8*zip_code_length	bslbf
}		
if (target_area_type == 0x5) {		
cell_target_area()		
}		
hor_acc	16	uimsbf
}		

interpretation – flag specifying how the restriction is interpreted. If set to 0 (“normal”), a terminal residing within the defined area may not render the associated media streams. If set to 1 (“spotbeam”), a terminal located outside of the defined area may not render the associated media streams.

override – flag specifying whether the location-based restriction may be ignored by a properly authorized terminal. If set to 0, the restriction must be obeyed. If set to 1, the restriction may be ignored if the terminal is able to obtain the necessary permissions (e.g., PPV Rights Object for the corresponding Program Key).

The override option allows the service provider to signal to the terminal that it may render restricted content regardless of its physical location, as long as the terminal has been pre-authorized to do so. This could also be used to notify unauthorized terminals of the ability to purchase rights to circumvent the restriction.

version – tells the terminal if the contents of this descriptor have changed since the last STKM. When the version number is the same as in previous STKMs and the terminal has already processed a descriptor with this same version number, it MAY ignore the contents of this descriptor and assume that geographical restrictions are the same as in previous STKMs with this same descriptor version number.

number_of_target_areas – the number of TargetAreas that define the geographical area. TargetArea is specified in OMA Service Guide for Mobile Broadcast Services [BCAST10-SG] as an XML element. It is adapted here with modifications such that it can be used for blackout restrictions.

target_area_type – specifies the type of area as specified in [BCAST10-SG]. The following values are possible:

0x1 = shapes used to represent a geographic area as defined by the shape descriptor below

0x2 = country code, 1-3 digits, e.g. 355 for Albania as specified in [OMA MLP]

0x3 = geopolitical name of area such as "Seoul" as specified in [OMA MLP]

0x4 = zip code

0x5 = a set of "cell_area_values" as defined by the cell_target_area descriptor below.

shape – adapted from shapes used to represent a geographical area specified in [OMA MLP]. The value for this descriptor is encoded as follows:

Table 33: shape Descriptor

shape descriptor	Length	Type
shape_type	4	bslbf
reserved_for_future_use	4	bslbf
if (shape_type == 0x3) {		
shape_polygon()		
}		
if (shape_type == 0x5) {		
shape_circular_area()		
}		
if (shape_type == 0x7) {		
shape_elliptical_area()		
}		

shape_type - specifies the type of shape as specified in [OMA MLP]. The following values are possible:

0x3 = a polygon

0x5 = a circular area

0x7 = an elliptical area

Note: Some shape types defined in [OMA MLP] are not included as they are not as applicable for use in blackout restriction.

shape_polygon – a polygon, which is a connected surface defined by an outer boundary and zero or more inner boundaries. The value for this descriptor is encoded as follows:

Table 34: shape_polygon Descriptor

shape_polygon descriptor	Length	Type
outerBoundarys		shape_linear_ring
number_of_innerBoundarys	8	uimsbf
for (i=0; i < number_of_innerBoundarys; i++) {		
innerBoundarys		shape_linear_ring
}		

Note: The shape_polygon, adapted from [OMA MLP], is a super-set of the polygon definition in [3GPP TS 23.032 V6].

outerBoundarys – the outer boundary of the polygon, defined by the shape_linear_ring type.

number_of_innerBoundarys – the number of inner boundaries, has to be larger than or equal to 0.

innerBoundarys – an inner boundary of the polygon, defined by the type shape_linear_ring.

shape_linear_ring – a closed, simple piece-wise linear path which is defined by a list of coordinates that are assumed to be connected by straight line segment. The value for this descriptor is encoded as follows:

Table 35: shape_linear_ring Descriptor

shape_linear_ring descriptor	Length	Type
number_of_coords	8	uimsbf
for (i=0; i < number_of_coords; i++) {		
coord()		
}		

number_of_coords – the number of coordinate points that define the linear ring. The number has to be larger than or equal to 3.

coord – a geographical coordinate. In [OMA MLP], a coordinate is specified by the tuple (x, y, z). The "z" component (specifying the altitude, if present) is not included as it is not applicable for blackout restriction use. For simplicity, the "x" and "y" components are represented as latitude and longitude as defined in [3GPP TS 23.032 V6]. The value for this descriptor is encoded as follows.

Table 36: coord Descriptor

coord descriptor	Length	Type
latitude	24	bslbf
longitude	24	bslbf

latitude – latitude coordinate, encoded as defined in [3GPP TS 23.032 V6].

longitude – longitude coordinate, encoded as defined in [3GPP TS 23.032 V6].

The definitions of latitude and longitude in [3GPP TS 23.032 V6] are quoted below for completeness:

- The latitude is coded with 24 bits: 1 bit of sign and a number between 0 and 2^{23}-1 coded in binary on 23 bits. The relation between the coded number N and the range of (absolute) latitudes X it encodes is the following (X in degrees):

$$N \le \frac{2^{23}}{90} X < N + 1$$

  except for N=2^{23}-1, for which the range is extended to include N+1.

- The longitude, expressed in the range -180°, +180°, is coded as a number between -2^{23} and 2^{23}-1, coded in 2's complement binary on 24 bits. The relation between the coded number N and the range of longitude X it encodes is the following (X in degrees):

$$N \le \frac{2^{24}}{360} X < N + 1$$

shape_circular_area – a set of points on the ellipsoid which are at a distance from the point of origin less than or equal to the radius. The value for this descriptor is encoded as follows:

Table 37: shape_circular_area Descriptor

shape_circular_area descriptor	Length	Type
origin		coord
radius	16	uimsbf
distance_unit	2	uimsbf

reserved_for_future_use	6	bslbf

Note: shape_circular_area corresponds to the type "ellipsoid point with uncertainty circle" specified in [3GPP TS 23.032 V6].

origin – specifies the coordinate of the origin.

radius – specifies the length of radius of the circular area.

distance_unit – specifies the distance unit used. The following values are possible:

0x0 = meter

0x1 = kilo-meter

0x2 = yard

0x3 = mile

shape_elliptical_area – a set of points on the ellipsoid, which fall within or on the boundary of an ellipse. The value for this descriptor is encoded as follows:

Table 38: shape_elliptical_area Descriptor

shape_elliptical_area descriptor	Length	Type
origin		coord
angle	10	uimsbf
semi_major	16	uimsbf
semi_minor	16	uimsbf
angular_unit	2	uimsbf
distance_unit	2	uimsbf
reserved_for_future_use	2	bslbf

Note: shape_elliptical_area corresponds to the type "ellipsoid point with uncertainty ellipse" specified in [3GPP TS 23.032 V6].

origin – specifies the coordinate of the origin.

angle – specifies the angle of the ellipse.

semi_major – specifies the length of the semi major.

semi_minor – specifies the length of the semi minor.

angular_unit – specifies the angular unit used. The following values are possible:

0x0 = degree

0x1 = grad

country_code – country code, 1-3 digits as specified in [OMA MLP].

name_area_length – number of bytes used to encode the name_area field.

name_area – a geopolitical name of area as specified in [OMA MLP].

zip_code_length – number of bytes used to encode the zip_code field.

zip_code – zip code represented by a character string.

cell_target_area – the target area defined by a set of cell IDs or other area identifiers. The value for this descriptor is encoded as follows:

Table 39: cell_target_area Descriptor

cell_target_area descriptor	Length	Type
cell_target_area_type	8	uimsbf
number_of_cell_area_values	16	uimsbf
for (i=0; i < number_of_cell_area_values; i++) {		
cell_area_value	32	uimsbf
number_of_pp2_cell_ids	16	uimsbf
for (j=0; j < number_of_pp2_cell_id; j++)		
pp2_cell_id	32	uimsbf
}		
}		

cell_target_area_type – specifies the cell_target_area type as defined in [BCAST10-SG]. The following values are possible:

0x0 = Unspecified

0x1 = 3GPP Cell Global Identifier as defined in [3GPP TS 23.003 v6]

0x2 = 3GPP Routing Area Identifier as defined in [3GPP TS 23.003 v6]

0x3 = 3GPP Location Area Identifier as defined in [3GPP TS 23.003 v6]

0x4 = 3GPP Service Area Identifier (SAI) as defined in [3GPP TS 23.003 v6]

0x5 = 3GPP MBMS Service Area Identity (MBMS SAI) as defined in [3GPP TS 23.003 v6]

0x6 = 3GPP2 Subnet ID as defined in [3GPP2 C.S0054-0]

0x7 = 3GPP2 SID as defined in [3GPP2 C.S0005-D]

0x8 = 3GPP2 SID+NID as defined in [3GPP2 C.S0005-D]

0x9 – 3GPP2 SID+NID+PZID as defined in [3GPP2 C.S0005-D]

0xA = 3GPP2 SID+PZID as defined in [3GPP2 C.S0005-D]

0xB = DVB-H Cell ID (specified in section 6.3.4.1 of [BCAST10-DVBH-IPDC-Adaptation])

number_of_cell_area_values – specifies the number of cell_area_value included in the target area.

cell_area_value – specifies the BDS specific transmit area given in the format as defined by cell_target_area_type given in the format as defined by cell_target_area_type.

pp2_cell_id – If cell_target_area_type = 4, the value is Sector_ID as defined in [3GPP2 C.S0024-A]. If cell_target_area_type = 5, 6, 7, 8, the value is BASE ID as defined in [3GPP2 C.S0002-0].

hor_acc – Horizontal accuracy in meters (as specified in [OMA MLP]).

7.2 Constant Values

TKM_ALGO_IPSEC 0

TKM_ALGO_SRTP 1

TKM_ALGO_ISMACRYP 2

TKM_ALGO_DCF 3

TKM_FLAG_FALSE 0

TKM_FLAG_TRUE 1

7.3 Coding and Semantics of Attributes

protocol_version – indicates the protocol version of this STKM.

The device SHALL ignore messages that have a protocol_version number it doesn't support. The value of the protocol_version of this message is set to 0x0 (i.e. the original format).

Note: If set to 0x0 the format specified in this version of the specification is used. If set to anything else than 0x0, then the format is beyond the scope of this version of the specification.

protection_after_reception – 2-bit field defining the required protection after the removal of the service protection, according to the following table:

Table 40: Protection_after_Reception Values

Value	Description DRM Profile	Description Smartcard Profile
0x00	**Content Protection** Content only available to terminals with the Content Protection function. Device has to protect all content against access in the clear, unless such access is explicitly permitted by the GRO / permissions_category. Only the explicitly allowed types of consumption as defined in Generalized Rights Objects (GROs) that the device has for this service or program are permitted (taking also into account the impact of permissions_category value, if included in the STKM). An example permission in GROs is 'Access' for the immediate rendering of the service or program. If a GRO has explicit permissions / constraints, then these SHALL be respected, without taking into account the protection_after_reception value.	same as 0x01 described below
0x01	**Content Protection with Implicit Direct Rendering Permission** Content only available to terminals with the Content Protection function. Device has to protect all content against access in the clear, unless such access is explicitly permitted by the GRO / permissions_category, but: • Direct rendering is implicitly allowed.	**Content Protection with Implicit Direct Rendering Permission** Content only available to terminals with the Content Protection function. Device has to protect all content against access in the clear, but: • Direct rendering is implicitly allowed. LTKMs provide keys for access to live content, broadcast files and recordings.

	No Generalized Rights Object is required in the device for direct rendering; a GRO with only the service or program key but without any permissions is sufficient. The device needs to have an GRO with the appropriate permissions (and possibly constraints) for any other type of consumption. If a GRO has explicit permissions / constraints, then these SHALL be respected, without taking into account the protection_after_reception value.	When using GBA_U, recordings SHALL include STKMs. These SHALL be sent to the Smartcard for processing during playback.
0x02	**Content Protection with Implicit Direct Rendering Permission and Playback of Protected Recording** Content only available to terminals with the Content Protection function. Device has to protect all content against access in the clear, unless such access is explicitly permitted by the GRO / permissions_category, but implicitly, two types of consumption are allowed: • Direct rendering, and • Unlimited play back of protected recordings of this service or program or protected files The above two types of consumption may also be made available over appropriately protected digital output links (see Appendix D for examples). If the protection_after_reception flags are not available for a protected recording, the device SHALL assume that they have the value 0x1 for that recording. If a GRO has explicit permissions / constraints, then these SHALL be respected, without taking into account the protection_after_reception value.	**Content Protection with Implicit Direct Rendering Permission and Playback of Protected Recording** Content only available to terminals with the Content Protection function. Device has to protect all content against access in the clear, but implicitly, two types of consumption are allowed: • Direct rendering, and • Unlimited playback of protected recordings of this service or program or protected files. The above two types of consumption may also be made available over appropriately protected digital output links (see Appendix D for examples). LTKMs provide keys for access to live content, broadcast files and recordings. When using GBA_U, recordings SHALL include STKMs. These SHALL be sent to the Smartcard for processing during playback.
0x03	**Service Protection** Content available to terminals with the Service Protection or Content Protection function. This specification does not impose any protection measures for the content after the removal of service protection. If a permissions_category value is included in the STKM, it SHALL be set to 0xFF to allow exporting in plaintext. Note that for e.g. legal or other reasons, the device still might have to protect the content in some way.	**Service Protection** Content available to terminals with the Service Protection or Content Protection function. This specification does not impose any protection measures for the content after the removal of service protection. Note that for e.g. legal or other reasons, the device still might have to protect the content in some way. LTKMs provide keys for access to live content and broadcast files.

	GROs provide keys for access to live content and broadcast files.	
Note: the creation of protected recordings, except for the protected format specified by the SAVE permission for the DRM Profile, is always allowed, because the play-back (or any consumption in general) is governed by GROs or LTKMs, or by the implicit play-back of protected recordings right when the protection_after_reception field has the value 0x02.		

traffic_protection_protocol – defines the protocol used for the encryption and authentication of traffic:

TKM_ALGO_IPSEC	IPsec ESP (transport mode; encryption: AES-128-CBC [key length 128]; authentication: HMAC-SHA1-96 [key length 160] or NULL).
TKM_ALGO_SRTP	SRTP (encryption: AES_128_CTR [key length 128]; authentication: HMAC-SHA1-80 [key length 160] or NULL).
TKM_ALGO_ISMACRYP	AU encryption (encryption: AES_128_BYTE_CTR [key length 128] (refer to [XBS DRM extensions-v1.0] for details); SRTP authentication: HMAC-SHA1-80 [key length 160] or NULL).
TKM_ALGO_DCF	DCF encryption (encryption: AES-128-CBC [key length 128]; authentication: HMAC-SHA1-80 [key length 160])
Other values	Reserved for future use

Whether or not authentication is used depends on <traffic_authentication_flag>.

traffic_authentication_flag – defines whether or not the traffic is authenticated:

TKM_FLAG_FALSE	Traffic authentication is not used.
TKM_FLAG_TRUE	Traffic authentication is used, and the algorithm depends on <traffic_protection_protocol>.

access_criteria_flag – indicates whether or not access criteria are defined for the program:

TKM_FLAG_FALSE	No access criteria are defined, implying that the terminal is allowed to access program without further restrictions (provided the necessary keys are available to the terminal).
TKM_FLAG_TRUE	Access criteria are defined, implying that the terminal is allowed to access the program only if the specified access criteria are met.

Access criteria cannot change during a program, i.e. as long a PEK is valid.

traffic_key_lifetime – is the lifetime of the Traffic Encryption Key, relative to the first occurrence of an SPI or MKI.

If <traffic_key_lifetime> is n, then the actual lifetime is 2^n seconds.

Note: Although the allowed values for the traffic_key_lifetime span from seconds to hours, service providers should not use TKM key material to realize long term key functionality. The TKM messages should be considered and used strictly for short-term key signalling. Also, the lifetime of traffic keys should be considerably shorter than the lifetime of service keys and program keys, to avoid users receiving the service or PPV event (encrypted with traffic keys) even after their service key or current program key has expired.

The following scenario may help in explaining the note. The field "next_encrypted_traffic_key_material" maybe present in the STKM. The field is encrypted with the current Service Key or current Program Key. If someone subscribes to a service,

or someone purchases a PPV event, then the person obtains both the current TEK and the next TEK. At the end of the service period, or the end of a PPV event, this means that the person has also a TEK for the next service period or the next PPV event. If the person stops subscription at the end of the current service period or the end of the current PPV event, then the person still has access to the first TEK of the next service period or next PPV event. When the maximum TEK lifetime is 1.5 minutes, a subscriber can at most have 1.5 minutes of unauthorized content, which may not be considered to be excessive. If the traffic_key_lifetime becomes 2 hours, then the subscriber may have excessive access to unauthorized content, especially in the case of PPV events, because the person now may have 2 hours of unauthorized content.

The TEK can be changed frequently to mitigate the risk of end-users posting the key via the interactive channel so that non-members can download that key. The cost of the attack, i.e., extracting the key, and uploading and downloading the key should be made to be more expensive than the cost of BCAST service/content. The frequency of change depends on the value of the BCAST service/content. For high-value PPV content, the TEK SHOULD be changed frequently whereas for low-value content, the TEK MAY be changed infrequently. The exact frequency is a configurable value and does not have impact on interoperability. The option to include two consecutive keys into one STKM, using next_encrypted_traffic_key_material, should be executed with care, since it allows the end user in any case to access service for 2*traffic_key_lifetime.

In the case when a Program Event is available either through subscription or as a PPV event, a STKM containing the next TEK at the end of a PPV program would allow a PPV user to view part of the next PPV event that corresponds to the next TEK. In this case, if next_encrypted_traffic_key_material is used, it SHOULD be utilized with sufficiently short Traffic Key lifetimes so as not to provide PPV users with free access to a PPV event that has not yet been purchased.

The actual duration of the crypto period SHALL be strictly shorter than the defined lifetime of the traffic key material. Typically, an SPI or MKI appears for the first time implicitly, when the “next” traffic key material is included in a STKM. Any safety margins to cope with network and transmission delays SHALL be added by the network. A typical value for the lifetime could be three times the crypto period.

The maximal value for the crypto period duration is in practice slightly shorter than the TEK lifetime, because the TKM will include the “current” and “next” traffic key material before a change of crypto period, to allow the devices to set up the security associations.

After the lifetime has expired, the security association containing the TEK can be safely deleted by the terminal. This may help managing the security association database in the terminal or enable other optimizations.

The maximum value for the TEK lifetime is defined mainly in order to have a strict upper bound for the effect of the “sneak post view” problem: the next traffic key material is distributed under the current PEK, and allows viewers to view a program during the next crypto period. Should this possibility still be of a concern, the network MAY choose a shorter crypto period than the maximum value, or, during the crypto period where the current program ends and a new program starts, choose to distribute the current and the next traffic key material in separate STKMs, encrypted with their respective PEKs.

number_of_access_criteria_descriptors – indicates the number of Access Criteria Descriptors.

8. Recording

8.1 Recording of Protected Streams

Service protection, whether it is provided using the DRM Profile or the Smartcard Profile, is an access-control mechanism only, i.e. once the SEK or PEK has been delivered to the user, access to a given broadcast stream is typically unrestricted.

However, certain broadcast content may have premium value and recording may be allowed only in protected form. This is achieved by using the protection_after_reception parameter of the STKM, as explained in Section 7.3. Both cases are explained below.

Recording can be governed by different flags. Depending on the profile, not all flags are considered to allow recording.

- The permissions associated with broadcast RTP streams, defined in the OMA DRM v2.0 Extensions for Broadcast Support document [XBS DRM extensions-v1.0], are sent in ROs for the DRM Profile. The value of the permissions_flag and the permissions_category (Section 10.1.5 of [XBS DRM extensions-v1.0]) for a programme that is part of the STKM must also be considered.
- Protection_after_reception values in the STKM define the type of protection provided for the recorded content. These are applicable to both DRM and Smartcard Profiles.

Depending on the above, content may be recorded in the clear or in protected form, as explained below.

8.2 Recording in the Clear

If recording in the clear is allowed, this SHALL be signalled in the Short Term Key Messages by setting the protection_after_reception to 0x03.

In this case, recording of content (unencrypted AUs) is possible in the clear, using appropriate file formats (provided by the BDS specifications, from other standards bodies or using proprietary formats). For BCAST, the existing DCF or PDCF file formats as defined in OMA DRM 2.0 [XBS DRM extensions-v1.0] MAY be used for recording in the clear [DRMCF-v2.0]. Other similar formats such as ISO or 3GPP can be used.

8.3 Recording in Protected Form Only

If recording in protected form only is allowed, this SHALL be signalled by setting protection_after_reception to 0x00, 0x01 or 0x02. In such cases, recording MUST be protected against access in the clear. This MAY be done by encrypting the content and protecting the decryption key(s) against access in the clear. This MAY be done using other means to protect content against access in the clear.

Access to the recorded content depends on the value of the protection_after_reception parameter. See Section 7.3.

The broadcast stream can be encrypted at transport level (IPsec or SRTP) or content level (ISMACryp) as described in Section 9.

If the broadcast stream in encrypted at content level using ISMACryp, recording in encrypted format may be achieved by recording the encrypted AUs without decryption in the adapted PDCF file format together with the TEK stream as explained in [XBS DRM extensions-v1.0]. Other methods and file formats may also be used. Note that recording of encrypted broadcast streams is possible without having the appropriate service protection rights (i.e. SEK or PEK) when using ISMACryp. These can be acquired at a later stage using the information stored in the KeyInfo box. This allows automatic recording of programmes based on user profiles, for example, or pricing models based on the time at which rights are acquired for service protection, i.e. the value of recorded content reduces as time goes by.

If the broadcast stream is encrypted at transport level using IPsec or SRTP, then the recording may require first decryption of the content and then re-encryption in an appropriate file format. This method is only applicable in the case of DRM Profile.

Recommendations for dealing with changes in rights are given in Section 8.4.

8.3.1 Recording of Streamed Content using (P)DCF File Format

Streamed protected content MAY be stored using the DCF file format [DRMCF-v2.0]. If the PDCF file format is used instead, the protected file MAY be stored using this file format. Both file formats are defined in OMA DRM 2.0 [DRMCF-v2.0].

Recording of super-distributable OMA assets containing a recording of broadcast content that is suitable for standard DRMv2 devices is described in section 7.4 of [XBS DRM extensions-v1.0]. This involves re-encryption with a single key and hence does not require recording of the key stream.

8.3.2 Recording of ISMACryp Protected Streamed Content using Adapted PDCF File Format

When recording content from a real-time delivery service using ISMACryp, the file MAY be created according to a modified version of OMA DRM PDCF 2.0 that allows usage of multiple encryption keys (TEKs) for content encryption in a single file [XBS DRM extensions-v1.0]. This is achieved by using the Access Unit header OMABCASTAUHeaderhich signals AU encryption and provides storage for the Key Indicator and IV. The Key Indicator identifies the TEK key used to encrypt Access Unit and the IV is used for the Counter mode of AES. The STKMs are recorded in a STKM track. Note that repeated STKMs can be ignored i.e. if the same STKM is received as one already recorded, it SHOULD not be recorded. The type of STKM is indicated in the adapted PDCF.

The Table below shows the appropriate location for parameters that need to be stored in the adapted PDCF file.

Table 41: Mapping of broadcast parameters to PDCF parameters

Parameter	Source Location	Destination Location
RightsIssuerURI	SG Access Fragment	RightsIssuerURL in CommonHeadersBox or KeyIssuerURL in KeyInfoBox
Service_BCI or Service_CID or Program_BCI or Program_CID	SG Access Fragment and STKM	ContentID in CommonHeadersBox
STKMs	STKM stream	OMAKeySample in STKM track
STKM type indication	SDP	sample_type in OMAKeySampleDescriptionEntry
TerminalBindingKeyID (if TBK is used)	SG Access Fragment	entry in OMAKeySampleDescriptionEntry and KeyInfoBox
RightsIssuerURI (for TBK)	SG Access Fragment	entry in OMAKeySampleDescriptionEntry and KeyInfoBox

This applies to both DRM and Smartcard profiles.

The Table below shows the content of the CommonHeadersBox fields when using the adapted PDCF. The equivalent table when using re-encryption with a single key for a DRMv2 format can be obtained from section 7.4 in [XBS DRM extensions-v1.0]. The Table shows which parameters are used for DRM and Smartcard Profiles.

Table 42: CommonHeaders box fields for adapted PDCF

Field	Contents DRM Profile	Content Smartcard Profile
EncryptionMethod	NULL (0x00) if no encryption. AES_128__BYTE_CTR (0x03) for ISMACryp encryption with TEKs.	same
PaddingScheme	Determined by the recording device.	same
PlaintextLength	Determined by the length of the recorded asset, calculated by the recording device.	same
ContentIDLength ContentID[]	N/A	N/A

RightsIssuerURLLength RightsIssuerURL[]	RightsIssuerURI if KeyIssuerURL not used in KeyInfoBox.	N/A
TextualHeadersLength TextualHeaders[]	Determined by context information (original asset, service guide, session description protocol).	N/A
ExtendedHeaders[]	Empty.	same

In the definition of these fields, the base64() operation is defined by [RFC2045].

The Table below shows the content of the KeyIDBox fields when using the adapted PDCF. The Table shows which parameters are used for DRM and Smartcard Profiles.

Table 43: KeyInfo box fields for adapted PDCF

Field	Contents DRM Profile	Content Smartcard Profile
KeyID Type	0x00	0x01
KeyID	base64Binary(Service_BCI) for recording of stream protected by SEK base64Binary(Program_BCI) for recording of stream protected by PEK	base64Binary(Key Domain ID|MSK ID)
KeyIssuerURL	RightsIssuerURI	PermissionsIssuerURI
TBK_ID	N/A	TerminalBindingKeyID
TBKIssuerURL	N/A	RightsIssuerURI for TBK
STKM[]	N/A as have STKM track	same

The following section provides recommendations for how change of rights is handled when recording.

8.4 Change of Rights and Recommendations for Recording

The following rules SHALL be observed when recording streamed content in a PDCF:

1. If the user has a valid Rights and the end of a program / event is reached, a new track MAY be created for the new program / event. Alternatively, a new file MAY be created for the new program / event, rather than using the same file.
2. If the user has a valid Service Rights and PEKs are used to protect TEKs, then new tracks or files MAY be created when PEKs change, rather than using the same track.
3. If a program / event is being recorded for which the user has the appropriate Rights and a new program / event starts for which the user has NO valid Rights, a new track or a new file SHOULD be created, rather than using the same track.
4. If a program / event is being recorded for which the user has no Rights, a new track or file MAY be created for a new program / event, rather than using the same track, if the user still has no valid Rights for the new program / event.
5. If the user has valid Rights for the new program / event, a new track or file SHOULD be created, rather than using the same track.
6. In all cases, if different rights or a different GRO is required, a different track or file SHALL be used.

9. Encryption Protocols

This section deals with the "Traffic Encryption Layer" (Layer 4) in the 4-layer model. The encryption protocols discussed below are optional on the Network (server) side.

9.1 IPsec

IPsec [RFC4301] fulfills both the criterion to be bearer-agnostic and to be universally usable for all types of IP-based services. The Broadcast System MAY use IPsec to protect Broadcast Services. Broadcast Terminals MAY support IPsec.

The IPsec implementation in the device can be such that it does not interfere with the usage of IPsec for other applications than OMA BCAST. This implies that the SPI allocation and security association lookups can be implemented in such a way that they interoperate with existing IPsec implementations.

An IPsec Security Association (SA) consists of a tuple of the following parameters.

- Selectors (IP protocol version, source IP address, destination IP address, protocol, source port and destination port)
- SPI
- Destination IP address
- Security protocol, security protocol mode and security protocol parameters
- Algorithms and algorithm parameters
- Key material

An IPsec SA is uniquely identified by a destination IP address and SPI pair.

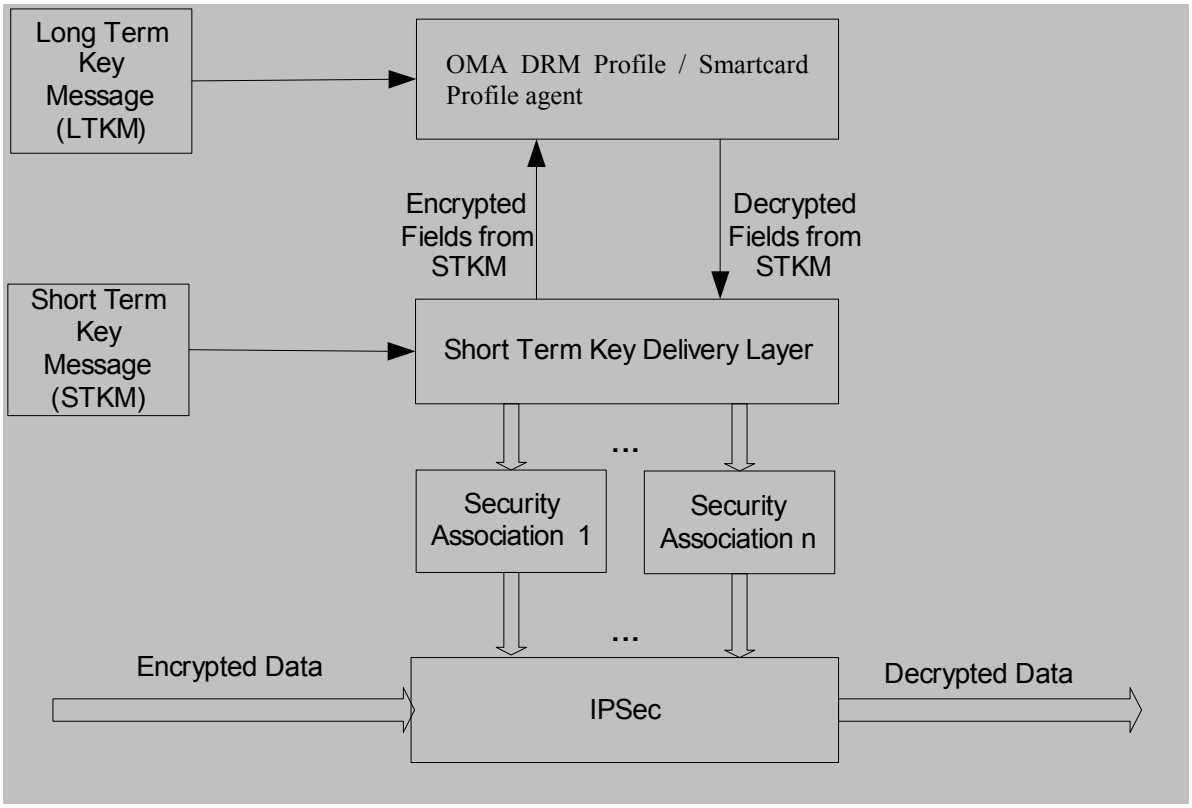


Figure 6 – IPsec Security Association Elements

Figure 6 shows the different objects and elements involved in instantiating IPsec security associations. The instantiation of security associations is performed by Layer 2 and Layer 3 messages. Given a Layer 3 message, Layer 3 extracts the encrypted fields from the message. Layer 3 passes these and other relevant fields (service-CID_extension and program_CID_extension) for Layer 2 message identification to Layer 2. For all Layer 2 messages on the device, Layer 2 examines them to see if one would be able to decrypt the fields in the Layer 3 message. If Layer 2 does find suitable Layer 2 messages, then it uses Layer 2 keys (SEK and/or PEK) in these messages to decrypt Layer 3 message fields. The decrypted fields are provided back to Layer 3, which based on the Layer 3 message and the decrypted fields instantiates a set of security associations. If Layer 2 does not find a suitable Layer 2 messages in the device, then the Layer 3 message SHOULD be silently dropped.

Selectors

Selectors are provided by the Layer 3 messages. The selectors can contain wildcards, ranges or point values, but all the other parameters SHALL be exactly defined. For transport mode all address selectors SHALL be point values and the destination address selector SHALL match the destination IP address of the SA.

Encapsulation Protocol and Mode

If IPsec is used for encryption of broadcast services, the protocol and mode SHALL be ESP in Transport Mode, according to [RFC2401] and [RFC2406]. Other IPsec encapsulation protocols or modes SHALL NOT be used.

Encryption Algorithm

The encryption algorithm for IPsec ESP SHALL be AES-128-CBC with explicit IV in each IP packet, as defined in [RFC2451] and [RFC3602]. Other encryption algorithms or key sizes or chaining modes SHALL NOT be used.

Authentication Algorithm

The authentication algorithm for IPsec ESP SHALL be HMAC-SHA-1-96, as defined in [RFC2104] and [RFC2404]. Other authentication algorithms or truncations SHALL NOT be used.

Support for the authentication algorithm as specified above is MANDATORY for both the terminal and the broadcast system. If no authentication is desired, the NULL authentication algorithm SHALL be specified. In this case, replay protection SHALL NOT be performed by the terminal.

The traffic_authentication_flag field in STKM indicates whether security transform includes integrity protection.

SA Management

The STKM Layer defines how often the IPsec encryption keys are rekeyed. This sets the following requirements:

- The IPsec encryption key SHALL be used as the key for the ESP encryption.
- The IPsec encryption key SHALL be derived from the key material contained within the STKM as follows:
    - In case of DRM Profile, the IPsec encryption key SHALL be the first 128 bits of the decrypted traffic key material. See Section 5.5.1.
    - In case of Smartcard Profile, the IPsec encryption key SHALL be obtained as described in Section 6.7.3.
    - NOTE: In Smartcard Profile, the MBMS MTK is called TEK, from which the actual encryption and authentication keys for IPsec are derived. To avoid confusion, the key for the ESP encryption is called “IPsec encryption key”.
- The IPsec authentication key TAK, which is derived from the key material in the STKM, refer to “Authentication for IPsec” below, SHALL be used as the key for the ESP message integrity code if authentication is used.
- The IPsec implementation SHALL be able to manage security associations relating to the key stream messages separately from those managed manually or by any other protocol such as IKE. This implies the ability to identify whether an SA is relating to key stream messages.

- The IPsec Security Policy (SP) SHALL be provided by the Service Guide [BCAST10-SG]. Security associations relating to STKMs SHALL be prioritized lower than those security associations that have a locally defined policy or a policy that is provided by a trustworthy party.
- Security associations relating to STKMs are simplex and SHALL be applied only to inbound traffic on the recipient side.
- An implementation SHALL be able to keep alive the security associations for at least two crypto periods (crypto period is the time span during which a specific traffic key is authorized) of the key stream.

The rekeying of existing IPsec SAs by Layer 3 SHOULD be managed on a resource basis by the Traffic Encryption Layer according to the following recommendations:

- The IPsec implementation SHOULD be able to keep alive at least the two most recently instantiated IPsec security associations for a specified set of selectors.
- The IPsec implementation SHOULD provide a least-recently-instantiated mechanism for destroying security associations as resources reserved for OMA BCAST IPsec security associations are exhausted.
- The amount of IPsec SAs required to exhaust the resources such that the cleanup mechanism is triggered SHOULD be 3 per SEK per set of IP selectors.

Authentication for IPsec

IPsec can be used with authentication. In case of authentication with IPsec the authentication data SHALL carry the TAS. The authentication mechanism SHALL create the TAK from the TAS.

For the DRM Profile, to obtain the encrypted traffic key material from the STKM the encrypted traffic key material SHALL be decrypted with the SEK or PEK:

$$TAS = D\{SEK\}(traffic\_key\_material)$$

or

$$TAS = D\{PEK\}(traffic\_key\_material)$$

For Smartcard Profile, the authentication seed TAS SHALL be obtained as described in Section 6.7.3.

The authentication key SHALL be generated from the authentication seed as follows:

$$TAK = f_{auth}\{TAS\}(CONSTANT\_KSM)$$

where:

CONSTANT_KSM = 0x040404040404040404040404040404 (120 bit)

Refer to Section 5.5.2 for details on f_{auth}.

The TAK SHALL be used in the MAC generation / verification of the IPsec data. Refer to [RFC 2406] for details.

9.2 SRTP

The Broadcast System MAY use SRTP [RFC3711] to protect Broadcast Services. Broadcast Terminals SHALL support SRTP.

An SRTP session is defined as a cryptographic context in the terminology of SRTP. A cryptographic context for SRTP, when used for service protection in OMA BCAST, consists of the following elements:

- Roll-over counter (ROC)

- Receiving sequence number
- Cipher and mode definition
- MAC method definition
- List of received packets
- MKI indicator bit
- Length of the MKI field
- Value of currently active MKI
- Array of secret master keys (MK)
- Array of counter of processed packets for each master key
- Length of encryption and authentication keys
- Master salt
- Context id

A cryptographic context is uniquely identified by its context id. The context id consists of the SSRC, destination network address and destination transport port number, as defined in [RFC3711].

Figure 7 shows a general case of key management for SRTP. Figure 8 shows a special case where Layer 3 is omitted and the necessary data is received from MKI to derive TEK (see [3GPP2 X.S0022-A] and [3GPP2 S.S0083-A]).

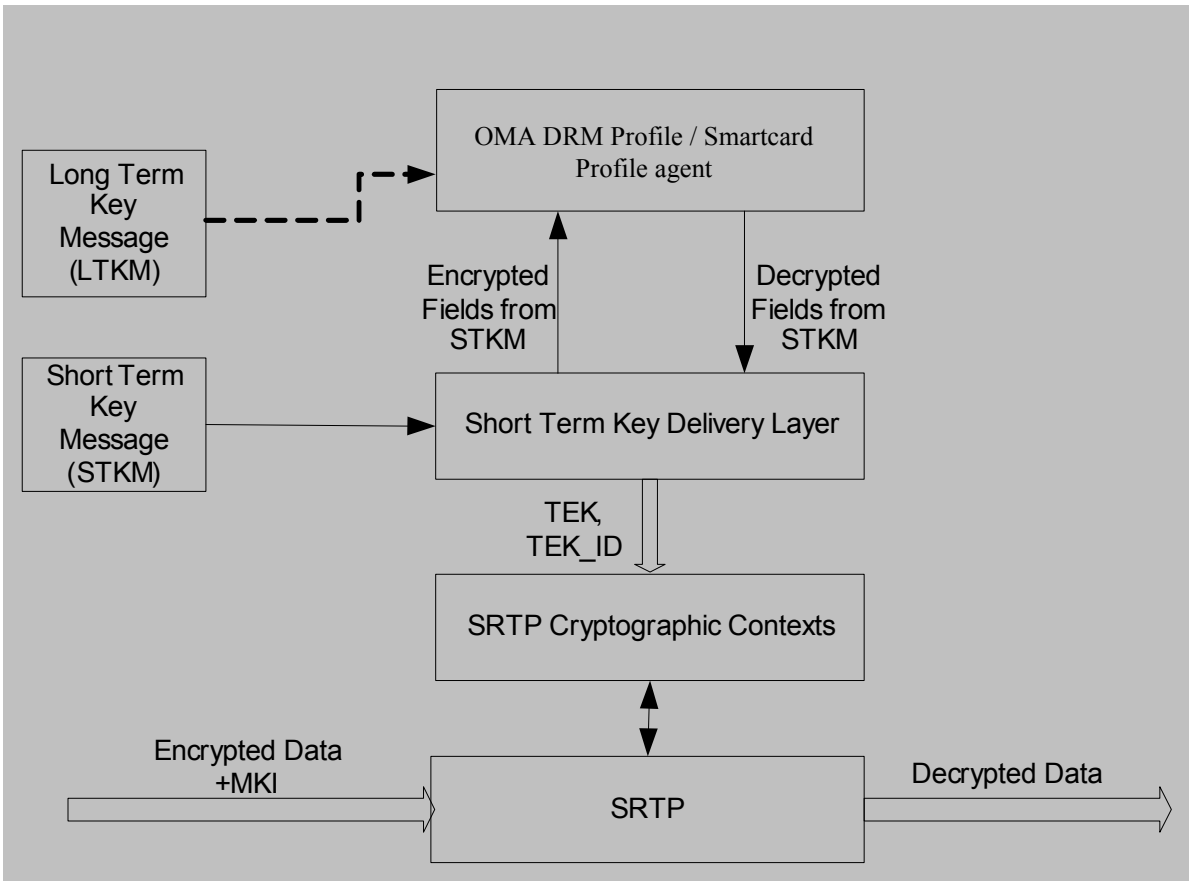


Figure 7 – SRTP Cryptographic Context Management (General Case)

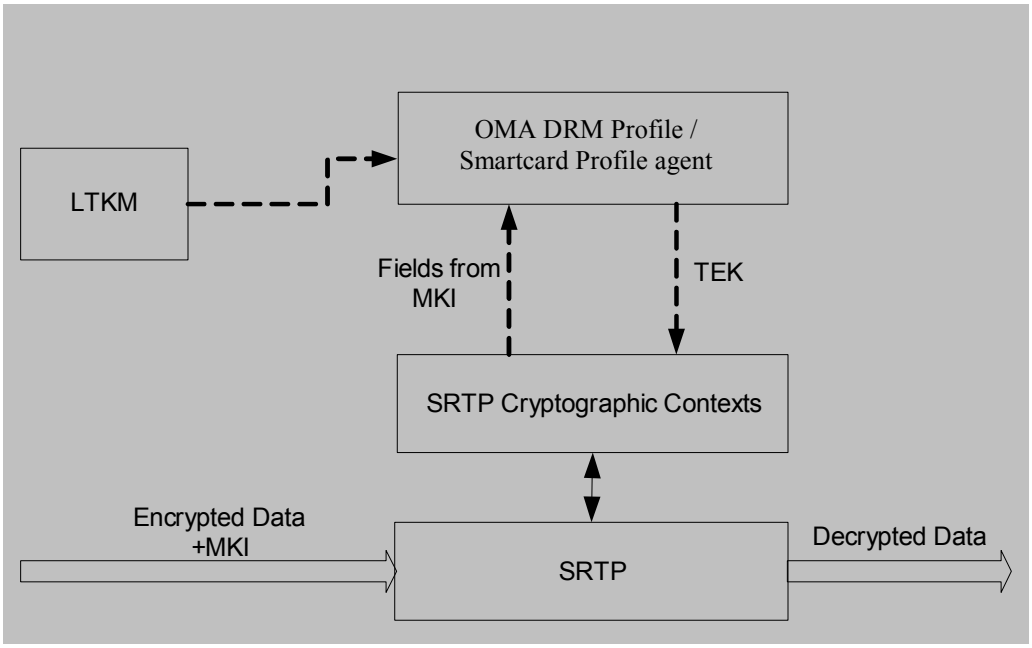


Figure 8 – SRTP Cryptographic Context Management (No Short Term Key Delivery Layer)

The instantiation of a cryptographic context is performed via the STKM and is driven by STKM and LTKMs. In the case of SRTP, an STKM includes an MKI (Master Key Index) length and MKI that is necessary to identify an SRTP cryptographic

context. STKMs may also carry Master Salt (MS) depending upon the underlying BDS (See below under "key management" for further discussions). The ROC (Roll-Over-Counter) values are carried in band in SRTP packets. STKM also carries the length of the encrypted key material (Master Key (MK) for SRTP).

The SRTP Master Key is extracted from the STKM. The traffic encryption layer then creates an SRTP session with decryption and (optionally) authentication keys that are derived from the Master Key as required by SRTP.

Key Management

The OMA BCAST SRTP application SHALL use the MKI value for looking up decryption keys. This means that a cryptographic context SHALL have the MKI indicator bit set to 1. The <From, To> value method of key lookup SHALL NOT be used.

The Master Salt (MS) MAY be used. The MS SHALL NOT be longer than 112 bits.

For the DRM Profile used independently, a NULL MS is used. A non-NULL 112 bit MS may be used for interoperability with the underlying BDS or the Smartcard Profile (see Section 11.3 for further details).

For the (U)SIM, a NULL 112 bit MS may be used for interoperability with the DRM Profile. Otherwise a non-NULL 112 bit MS is used. For interoperability with the (R-)UIM/CSIM Smartcard Profile, the 112 bit MS may correspond to the 32 bit SK_RAND with zero bit padding for the remaining 80 bits. See Section 11.3 for further details.

For the (R-)UIM/CSIM, the MS SHALL be the 32 bit SK_RAND value with zero bit padding for the remaining 80 bits. A NULL MS SHALL NOT be used as this results in a fixed encryption key for the lifetime of the SEK or PEK (see Section 11.3 for further details).

The TEK contained in the STKM SHALL be used as the SRTP master key.

The key derivation rate SHALL be 0 except for interoperability with 3GPP MBMS terminals, where the key derivation rate MAY be zero (see Section 11.3 and [BCAST10-MBMS-Adaptation]).

Layer 3 SHALL provide and update the cryptographic contexts to the SRTP implementation (excluding the ROC). Note that some fields are initialized or managed internally, such as the list of received packets used in replay protection, receiving sequence number, and the ROC.

The ROC SHALL be transferred in every R-th packet according to [RFC4771].

Because the SRTP key-deriviation rate is not used and the <From,To> values are also not used, the SRTP crypto context will be rekeyed by Layer 3.

Encryption Algorithm

The encryption algorithm for SRTP packets SHALL be AES_128_CTR, as defined in [RFC3711]. Other encryption algorithms or key sizes or chaining modes SHALL NOT be used.

Authentication Algorithm

The authentication algorithm for SRTP SHALL be as defined in [RFC4771], based on HMAC-SHA-1-80 as defined in [RFC2104] and [RFC3711]. Other authentication algorithms or truncations SHALL NOT be used.

Support of the authentication algorithm for SRTP as specified above shall be OPTIONAL for both the terminal and the broadcast system. If no authentication is desired, the NULL authentication algorithm SHALL be specified. In this case, also replay protection SHALL NOT be performed by the terminal.

Note that there must be a secure way of notifying whether a security transform includes integrity protection. This should be handled as part of the mechanism for negotiating SRTP security parameters e.g. MIKEY.

For the DRM Profile, the traffic_authentication_flag field in STKM indicates whether security transform includes integrity protection. For the Smartcard Profile, this can be handled by the corresponding Layer 3 message or some other mechanism for negotiating SRTP security parameters.

Some specific points of the implementation need to be specified to be able to share protected stream(s) between operators. Section 11.3 introduces how to be able to share a media stream among operators that implement different key management mechanisms, Section 11.3.1 with respect to 3GPP-MBMS bearer features using SRTP, and Section 11.3.2 considering access limited to BCAST terminals.

9.3 ISMACryp

For content encryption of RTP streams, content that is part of a real-time delivery service MAY be protected using ISMACryp as explained in this specification, i.e. by encrypting elementary audio video samples called Access Units (AUs). Individual AUs are encrypted using AES_128_BYTE_CTR mode. Each encrypted AU has an OMABCASTAUHeader defined in OMA DRM V2.0.

BCAST terminals MAY support content encryption using ISMACryp as specified in this section and Section 9.3.1.

Encryption Algorithm

The encryption algorithm SHALL be AES_128_BYTE_CTR. Refer to [ISMACRYP11] or [XBS DRM extensions-v1.0] for further details. Other encryption algorithms or key sizes or chaining modes SHALL NOT be used.

The TEK is sent in STKMs, the IV is in the OMABCASTAUHeader preceding the encrypted data, the salt key k_s is signalled in the SDP file and the use of the counter is described in [ISMACRYP11].

The Table below shows BCAST parameters and equivalent ISMACryp parameters.

Table 44: Equivalent BCAST and ISMACryp parameter names

OMA BCAST parameters	Equivalent ISMACryp parameters [ISMACRYP11]
TEK	key_k (encryption key)
IV	IV
k_s	k_s (salt key)

Authentication Algorithm

The default authentication algorithm is SRTP with an HMAC-SHA1 with an 80-bit output tag and a 160-bit key [RFC3711]. Other authentication algorithms or truncations SHALL NOT be used. Support of the authentication algorithm for ISMACryp shall be OPTIONAL. The authentication key to be used is derived as per SRTP using the 128 bit MK and 112 bit MS sent in STKMs. SRTP authentication is signaled using SDP security descriptions [RFC4568]. ROC is signalled as per SRTP described in Section 9.2, key indicator (see Section 5.5) is used as MKI for DRM Profile and MTK ID is used as MKI for Smartcard Profile. MKI and Authentication tag is delivered over SRTP packet according to [RFC3711].

9.3.1 RTP Transport of Encrypted AUs (ISMACryp)

Note: This technical specification of RTP transport is based on ISMACryp documents [ISMACRYP11].

Content encryption modifies data before packetization of RTP packets, thus the various RFCs defining ways to encapsulates audio and video data do not apply. In addition, some signalling is necessary in the SDP in order to enable the decryption of the data. ISMA [ISMACRYP11] has defined encapsulation specifications for MPEG-4 codecs. When applicable, these specifications SHALL be used as they are optimized for specific codecs. For any encrypted media that has a defined mapping to the ISO Media File Format ([ISO-14496-12]), the following codec-generic method SHALL be used to encapsulate the encrypted data in RTP packets.

The following defines the RTP payload formats for content encrypted before packetisation. This payload is called “enc-isoff-generic” and supports RTP transport of encrypted tracks from the ISO file format.

The Terminal SHALL fully support this payload, as specified below.

RTP Header	AU Header Section	Auxiliary Section	Access Unit Data Section

Figure 9 – Data Sections within an MPEG4-Generic RTP Packet

The RTP payload structure is based upon the RTP payload format defined in mpeg4-generic [RFC3640] shown in Figure 9. Support for encrypted media is added by adding new fields to the Access Unit (AU) header section. Each RTP packet SHALL contain either:

1. Exactly one access unit,
2. Two or more complete access units, or
3. One fragment of an access unit.

The format of enc-isoff-generic RTP packets is identical to that of mpeg4-generic: same RTP header and same RTP payload structure (AU Header section, Auxiliary Section and Access Unit Data Section).

AU-headers-length	AU Header (1)	AU Header (2)	AU Header (n)

Figure 10 – AU Header Section within an RTP packet

As defined in RFC 3640 ([RFC 3640]), if the AU-header is configured empty, the AU-headers-length field SHALL NOT be present and consequently the AU Header Section is empty. If the AU-header is not configured empty, then the AU-headers-length is a two octet field that specifies the length in bits of the immediately following AU-headers, excluding the padding bits.

Some optional fields have been added to the AU headers (see [RFC 3640)) in order to offer a codec agnostic solution:

Field name	Optional/ Mandatory
OMABCASTAUHeader	NO/TM
AU-size	NO/TM
AU-Index / AU-Index-delta	NO/TM
CTS-flag	NO/TM
CTS-delta	NO/TM
DTS-flag	NO/TM
DTS-delta	NO/TM
RAP-flag	NO/TM
Stream-state	NO/TM
Slice-start-flag	NO/TM
Slice-end-flag	NO/TM
Padding-size	NO/TM

Figure 11 – Extended AU header

AU-size, AU-Index, AU-Index-delta, CTS-flag, CTS-delta, DTS-flag, DTS-delta, RAP-flag, Stream-state fields are defined in RFC 3640 ([RFC 3640]).

For video codecs that encode the video frame as slices that can be decoded independently, a new optional fragmentation mode is defined. In this mode, the sender SHOULD try to align slice boundaries with AU fragment boundaries where possible. Slices boundaries will be indicated using special fields of the extended AU header.

Note: If the media is already encrypted, it may not be possible for the sender to know where the slice begins and this mode may not be used. However, there are two different situations in which this information can be obtained:

1. when packetization and encryption occur together: live encoding and encryption, encryption and hinting, or

2. when the PDCF has SubSampleInformationBoxes.

Constraints:

- Each slice MUST be parsed (i.e. syntax decoded) independently of other bytes in the access unit by the decoder. In other words, if a decoder receives only a slice and not the whole access unit, it will be able to decode it.

- If a slice is greater than the MTU, it is fragmented over multiple RTP packets and fragment MUST be byte aligned.

- A RTP packet MUST carry only one or several complete slice(s) or only one fragment of a slice.

Note: This definition of slices matches with the definition of video-packets for ASP, slices for H263, NALu for AVC and SVC…

Two new media fields are defined in the AU header section (see RFC 3640 [RFC 3640]) to indicate the first and the last fragment of a slice:

- `Slice-start-flag`: indicates whether it is the first fragment of a slice. A value of 1 indicates that it is the first fragment of a slice, a value of 0 means it is not.

- `Slice-end-flag`: indicates whether it is the last fragment of a slice. A value of 1 indicates that it is the last fragment of a slice, a value of 0 means it is not.

RTP receiver behaviour in case of packets loss:

If `Slice-start-flag` and `Slice-end-flag` are not used, when an IP packet is lost, the RTP receiver may drop the entire AU (depending of the decoder robustness).

If `Slice-start-flag` and `Slice-end-flag` are used, when an IP packet is lost, the RTP receiver MAY drop all slices which may be affected by the packet lost (depending of the decoder robustness). All others slices may be delivered to decoder.

9.3.1.1 Padding

MPEG decoders must be able to decode Access Units in which such padding is applied. For codecs which are not byte aligned, we have to handle the "de-padding" process.

A new optional media fields are defined in the AU header section (see RFC 3640 [RFC 3640]) to indicate number of padding bits:

`Padding-size`: indicates the number of padding bits.

9.3.1.2 OMABCASTAUHeader

OMA inserts cryptographic metadata at the beginning of each AU header. The format of the first AU header is different from the second and subsequent AU headers (similar to the treatment of AU-Index and AU-Index-Delta in mpeg4-generic). This block supplies the cryptographic context for each AU or AU fragment in an RTP packet and is defined as follows:

```
aligned(8) class OMABCASTAUHeader {
if (SelectiveEncryption == 1) {//
   bit(1)   EncryptedAU;   // Encryption indicator
   bit(7)   reserved;   // Must be zero
```

```
}
if (auNum==0) { // first AU in packet?
   unsigned int(8 * ISMACrypIVLength) IV;
   unsigned int(8 * ISMACrypKeyIndicatorLength) KeyIndicator;
}
else {
      int(8 * ISMACrypDeltaIVLength) delta_IV;
      if (ISMACrypKeyIndicatorPerAU)
         unsigned int(8 * ISMACrypKeyIndicatorLength) KeyIndicator;
}
```

Reserved fields MUST have all bits set to zero though a compliant receiver MAY choose to ignore this field. Note that in bit(n), unsigned int(n) and int(n), n is always a bit count. Note also that delta_IV is the only signed int in the above definition. See next section for the signalling of the parameter constants (ISMACrypSelectiveEncryption, ISMACrypIVLength, ISMACrypKeyIndicatorLength, ISMACrypDeltaIVlength, and ISMACrypKeyIndicatorPerAU). Diagrammatically, this means that these fields are inserted just before the AU-size field in the diagram above:

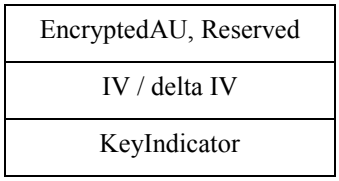


Figure 12 – OMABCASTAUHeader fields

All "Reserved" fields MUST be zero and MAY be ignored by the receiver. Note that it is possible to compute the access unit count by using the configuration parameters, and the signaled length of the access unit headers. This is because the total bit-length of the AU-headers is given in each packet, and the length of the first AU Header as well as the second and subsequent AU-headers can be computed from the signaled parameters. This is still true with this extended AU header. So we have:

AU-count = (AU-header-length – first-header-size)/subsequent-header-size + 1;

Note that this equation does not hold if either CTS or DTS timestamps are present; however, this normally applies only to video, and in that case, the payload format restricts the packet to containing only one AU or a fragment of an AU.

The fields in the OMACrypContextAU structure have the following meaning.

EncryptedAU : An optional single bit field to signal selective encryption. A 1 value signals that the corresponding access unit is encrypted, a value of 0 means it is not. The presence of this field is configured with the ISMACrypSelectiveEncryption parameter. All fragments of a single access unit SHALL have the same value for AU_is_encrypted.

IV: Contains the initial IV value for the first access unit or fragment contained in the packet. In most cases, this is the only IV in the packet. In some cases such as interleaved media, however, there MAY be an IV per AU.

delta_IV: This field contains IV data on a per-AU basis when ISMACrypDeltaIVLength is non-zero and the data are interleaved in packet payloads.

The actual IV to be used for each access unit in a packet is computed as follows, with the first access unit in a packet indexed as zero:

IV[0] := AUHeader[0].Initial_IV; // First AU in packet

IV[N+1] := IV[N] + AUSize[N] + (ISMACrypDeltaIVLength == 0 ? 0 : AUHeader[N+1].delta_IV) // Subsequent

Note: The number of access units in a packet is not signaled in this payload format. The number of access units in the packet can be deduced from the access unit header as for unencrypted modes. A packet that has the M-bit cleared contains a fragment that is not the last of an AU. If the M-bit is set, then the packet has one or more access units or the last fragment of an access unit. The access unit header indicates whether there are two or more access units in the packet. To distinguish

between one whole AU and the last fragment, compare the payload data size and the access unit size conveyed in the access unit header. The access unit size will be the size of the whole AU and not the fragment.

Note: In the simple case where there is one AU per packet, or the AUs are contiguous, this structure reduces to signalling a key indicator and an initial IV per packet.

KeyIndicator: Contains the key indicator for an access unit when ISMACrypKeyIndicatorLength is non-zero. If the ISMACrypKeyIndicatorPerAU is 0, then only the first access unit in a packet has an explicit key indicator value included in the cryptographic context; all subsequent access units SHALL have the same value for KeyIndicator as the first access unit. If ISMACrypKeyIndicatorPerAU is 1, then a value of key_indicator is included in the cryptographic header for each access unit or fragment in the packet. If SelectiveEncryption is 0 for an access unit, then the value of this field is ignored.

9.3.2 Using PDCF for ISMACryp Streaming

For video codecs that encode the video frame as slices that can be decoded independently, this specification defines a new fragmentation mode in which the packetizer (hinter or server) should try to align slice boundaries with AU fragment boundaries where possible. However if the media is already encrypted, it may not be possible for the sender to know where the slice begins. To allow such packetization, the use of SubSampleInformationBox is RECOMMENDED in the encrypted file.

```
aligned(8) class SubSampleInformationBox
        extends FullBox('subs', version, 0)
{
        unsigned int(32) entry_count;
        int i,j;
        for (i=0; i < entry_count; i++)
        {
                if ()
                {
                        unsigned int(32) sample_delta;
                }
                unsigned int(16) subsample_count;
                if (subsample_count > 0)
                {
                        for (j=0; j < subsample_count; j++)
                        {
                                if(version == 1) {
                                        unsigned int(32) subsample_size;
                                }
                                else {
                                        unsigned int(16) subsample_size;
                                }
                                unsigned int(8) subsample_priority;
                                unsigned int(8) discardable;
                                unsigned int(32) reserved = 0;
                        }
                }
        }
}
```

9.4 (P)DCF Encryption with TEK

This section describes how (P)DCF files can be protected over the broadcast channel by encrypting individual files with individual TEKs. This technique is based on material from MBMS text in [3GPP TS 33.246 v7]. The mechanism described in this section was adopted from [3GPP TS 33.246 v7] and adapted to BCAST needs.

Protection of download data uses DCF as a container for ciphered file data. The DCF container also identifies the key used in protecting the data. In this case the encryption key is a single TEK. Usage of DCF is independent of the KMS type. The same principle applies to the PDCF format for audio video data.

Data that belongs to a download Service is decrypted as soon as possible by the terminal, if the SEK or PEK needed to provide the relevant TEK are already available on the terminal or Smartcard. Storage of the STKM containing the TEK is also allowed in BCAST.

The following method is compatible with the OMA DRMv2 DCF file format as defined by [DRMCF-v2.0] as it uses the Key Info box defined in [XBS DRM extensions-v1.0] in the Extended Headers field, which is ignored by OMA DRMv2 terminals.

Access to the file SHALL respect the protection_after_reception values defined in the STKM message.

9.4.1 Integrity Protection using OMADRMSignature Box

When it is required to protect BCAST download data, OMA DRM V2.0 DCF as defined in reference [DRMCF-v2.0] shall be used. However, encryption and authentication keys are generated from TEK. For integrity protection, an OMADRMSignature as specified below MAY be attached inside the optional Mutable DRM information box ('mdri') of the (P)DCF.

The OMADRMSignature Box is an extension to OMA DRM V2.0 DCF for use by OMA BAC BCAST, and is defined by 3GPP as follows:

Table 45: OMA DRM Signature Box

```
aligned(8) class OMADRMSignature extends Fullbox('odfs', version, flags) {
   Unsigned int(8)   SignatureMethod;     // Signature Method
   Char           Signature[];           // Actual Signature
}

SignatureMethod Field:
NULL  0x00
HMAC-SHA1   0x01
```

The range of data for the HMAC calculation shall be according to section 5.3 of reference [DRMCF-v2.0].

9.4.2 Use of OMABCAST Key Info Box

BCAST has defined a specific box [XBS DRM extensions-v1.0] to provide key management information for both DRM Profile and Smartcard Profile.

The OMABCASTKeyInfo box allows the following information to be stored for the DRM Profile and / or Smartcard Profile:

- KeyID: SEK / PEK ID & TEK ID used for decrypting the (P)DCF
- KeyIssuerURL: PermissionsIssuer URL used to acquire the appropriate permissions
- TBK_ID: TerminalBindingKey ID and URL if used
- STKM containing the TEK used to decrypt the content

In order to ensure key material can be acquired, the KeyIssuerURL in the Key Info box MAY be used. If the Terminal does not have the SEK or PEK required to decrypt the TEK within the STKM, it may request it by sending the Service request described in [BCAST10-Services] to the KeyIssuerURL with the corresponding SEK or PEK ID. If the KeyIssuerURL is not present, the RightsIssuerURL in the OMADRMCommonHeader box MAY be used instead.

The STKM containing the TEK used to decrypt the DCF MAY be stored inside the OMABCASTKeyInfo box in the STKM field. Note that as the OMABCASTKeyInfo box is part of the HMAC calculation, if the OMADRMSignature box is included but the STKM is not delivered within the OMABCASTKeyInfo box, subsequently adding the STKM to the (P)DCF invalidates the hash. A terminal doing this would typically remove the OMADRMSignature box.

9.4.3 FDT Protection within DCF

In case the FDT of the FLUTE protocol needs to be protected, the FDT MAY also be wrapped in a different DCF. Confidentiality or integrity protection of FDT can be provided this way.

9.4.4 Support of OMA DRM v2 Boxes

The OMA BCAST DCF format SHALL support the following boxes specified in OMA DRM V2.0 DCF [DRMCF-v2.0]:

- Fixed DCF header;
- Mutable DRM information Box;
- OMA DRM Container Box.

10. Signaling

Access to key streams is provided in SDP.

10.1 Protection Signaling in SDP

10.1.1 Description

SDP information is used to specify streaming sessions according to [RFC2327].

Additional information is required to identify parameters relative to key management: STKM streams, KMS versions, etc. These are defined below and SHALL be used to describe encrypted media streams and key streams (STKM and LTKM). Note that, in the case of MBMS, such information can be signalled in the MBMS security description as per [3GPP TS 26.346 v7].

The table below defines the <field values> to be used for signal protection information. These parameters are used for the signalling of media, short-term key message (STKM) and long-term key message (LTKM) streams. Their usage for the different streams will be explained in the following sections. A media stream can be protected by one or more STKM streams. Some other optional and stream specific parameters are introduced in the relevant sections.

Table 46: Protection Signalling in SDP

Field name	Category	Type	Purpose
kmstype	NM/TM	String	Identifies the Key Management system (KMS) used (see Table 47 for supported KMSs)
bcastversion	NM/TM	Decimal x.y	Identifies the BCAST version x.y
serviceprovider	NM/TM	String	Identifies the service provider i.e. the name of the provider using the key stream (see Table 49 for examples)
control:streamid	NM/TM	Decimal (integer)	Unique non-zero integer identifying a particular stream. Numbers are unique within a particular SDP session i.e. no global numbering is required. Used to indicate which media stream is protected by which STKM stream.
BaseCID	NO/TO	AnyURI	For the DRM Profile, part of the Service or Program CID used to identify the corresponding asset within an OMA DRM 2.0 Rights Object. The Service or Program CID is obtained from the BaseCID as described in Section 5.5.1. The network and terminal SHALL support this field in case the DRM Profile is supported.

where, NM=Mandatory for network to support; NO=Optional for network to support; TM=Mandatory for terminal to support; TO=Optional for terminal to support

The tables below shows the corresponding <field values> for the <field names>:

Table 47: kmstype values

Value (String)	Semantics
oma-bcast-drm-pki	DRM Profile Key Management System
oma-bcast-gba_u-mbms	Smartcard Profile Key Management System, using 3GPP GBA_U to establish Layer 1 keys
oma-bcast-gba_me-mbms	Smartcard Profile Key Management System, using either 3GPP GBA_ME or 3GPP GBA_U to establish Layer 1 keys
oma-bcast-prov-bcmcs	Smartcard Profile Key Management System, using provisioned 3GPP2 BCMCS Symmetric

	Key Infrastructure

Table 48: bcastversion values

Value (Decimal x.y)	Semantics
1.0	Current version in this specification is 1.0

Table 49: serviceprovider values

Value (String)	Semantics
<provider name>	<provider name> is the name of the provider selling access to the SDP stream using the specified key stream, e.g. Pay4TV, DiscountBcast, MajorMediaGroup.

Table 50: control:streamid values

Value (Decimal)	Semantics
1, 2, 3, etc.	Each stream declared in the SDP will be uniquely numbered. Only non-zero positive integers are acceptable. While it is recommended that streams are numbered in increasing order, this is NOT mandatory. Duplicate streamids SHALL be ignored, i.e. only the first one SHALL be used.

Table 51: BaseCID values

Value (String)	Semantics
<BaseCID>	<BaseCID> is part of the Service or Program CID used to identify the corresponding asset within an OMA DRM 2.0 Rights Object. Upon reception of a STKM, the terminal can assemble the service_CID/program_CID/BCI and look up the SEK or PEK (wrapped inside a LTKM) as described in 5.5.3.

10.1.2 Short-Term Key Message Streams (STKM)

This Section gives descriptions of short-term key message (STKM) streams using SDP.

10.1.2.1 Description

To support efficient STKM carriage, each STKM Stream is carried in its own UDP stream. The mime type bcast-stkm is defined to signal a STKM Stream.

The location of a STKM stream is signaled within the SDP file used to describe the delivery parameters for a given service. The SDP file describing the service typically contains a media announcement entry for the Video and one for the Audio. In addition, to signal the associated STKM streams, one or two additional stream announcements are added.

A STKM stream is signaled in the following way:

m=data <port> UDP bcast-stkm.

MIME type parameters are signaled in the "fmtp:" line. MIME type parameters for STKM as defined in Table 52 SHALL be supported.

Table 52: Parameters of the mime type bcast-stkm

Parameter	Terminal support	Server support	Purpose
streamid	Mandatory	Mandatory	See Table 46
kmstype	Mandatory	Mandatory	See Table 46

serviceprovider	Mandatory	Optional	See Table 46
BaseCID	Optional	Optional	See Table 46

10.1.2.2 SDP Example for Short –Term Key Message Streams

This Section gives an example of SDP descriptions of short term key streams:

m=data 49230 UDP bcast-stkm

c= IP4 224.2.17.12/127

a=fmtp:bcast streamid=10; serviceprovider=DiscountBcast;kmstype=oma-bcast-drm-pki

10.1.3 Short-Term Key Message (STKM) Streams Binding

The signaling described below allows the terminal to clearly identify which STKM streams are relevant for each media stream. Several media streams may reference the same STKM stream, thereby sharing the same Traffic Encryption Keys, but each media stream may also reference a different STKM stream. An encrypted media stream must refer to one (in case only DRM or Smartcard Profile is used) or two STKM streams (one for DRM Profile and one for Smartcard Profile), each providing a secure delivery of the same Traffic Encryption Keys (TEKs) by a particular profile. Furthermore, there can be more than one STKM stream for a given profiles if there are more than one service provider.

AES in counter mode requires that the same key stream is never reused. In the case that the same STKM stream is shared among several media streams – a distinct IV must be provided for each such media stream. This is already the case for SRTP-based encryption (where each IV is based on the SSRC value in the RTP header).

In the case of IPsec, only AES in CBC mode is currently supported.

In the case of ISMACryp the effective IV value is based on the salting key k_s that can be made different for each media stream. To ensure that ISMACryp can safely allow sharing of the same STKM stream between multiple media streams, each such media stream MUST have a unique salting key k_s specified in the SDP file.

Example:

A service comprising a video stream and an audio stream, both encrypted with the same Traffic Encryption Keys, and protected by two different KMSs will make use of 4 streams: one for the video, one for the audio, one for KMS#1 (supporting DRM Profile) STKM stream and one for KMS#2 (supporting Smartcard Profile) STKM stream.

This way, the terminal will only listen to and process the STKM stream coming on the relevant IP connection. SDP [RFC3237] isused to describe the STKM stream(s) associated with each media stream. The following attribute is defined for mapping STKM streams to media streams in the SDP:

MIME type parameters are signalled in the "fmtp:" line. MIME type parameter stkmstream as defined in Table 53 SHALL be supported.

Table 53: Definition of stkm stream attribute

Attribute	Terminal support	Server support	Type	Purpose
stkmstream	Mandatory	Mandatory	Stream Reference	streamid (id of the stkm stream) indicating which STKM stream applies to this media stream .

The attribute can be at session level, in which case it applies to all media streams, or the attribute can be at media level, in which case it only applies to the specified media and would overwrite possible session level attribute.

Each session or media stream can have multiple stkmstream attributes. Using this attribute the terminal can lookup the corresponding STKM stream announcements and figure out which one to listen to and process. We note that this attribute is optional and hence would not be there for unencrypted media streams.

10.1.3.1 STKM Streams Binding Example

Below is an example where two STKM streams (10 and 11) are associated on session level with the media streams, however two other STKM streams (13 and 14) are associated to a second audio track. The stkmstream attribute on media level overwrites the stkmstream attribute on session level for that particular media stream. In this example, to decrypt the Spanish audio track, STKM stream 13 or 14 can be used.

```
v=0
o=BCAST 2890844526 2890842807 IN IP4 126.16.64.4
s=A protected Bcast stream
c=IN IP4 224.2.17.12/127
t=2873397496 2873404696
a=recvonly
a=stkmstream:10
a=stkmstream:11
m=audio 49170 RTP/AVP 0
a=lang:en
/* English Language audio track can be decrypted using STKM streams in SDP file with
streamid 10 or 11*/

m=audio 52002 RTP/AVP 0
a=lang:ES
a=stkmstream:13
a=stkmstream:14

/* Spanish Language audio track can be decrypted using STKM streams in SDP file with
streamid 13 or 14 */
```

In the case of English language audio track, this signaling announces that to gain access to the English audio stream, the terminal may use either the STKM with streamid=10, or the one with streamid=11. The terminals can then lookup in the same SDP file for both two STKM streams (identified by their streamid), and to identify the KMS and the operator each is associated with. Similarly Spanish language audio track can be decrypted using STKM streams with id 13 or 14 in the same SDP file. Then, on the basis of this information and depending on which KMS it is supporting, the terminal can decide which stream it needs to listen to in order to get the short-term key message (STKM) stream it requires.

10.1.4 Long-Term Key management Message (LTKM) Stream

This Section describes the description of LTKM stream using SDP.

10.1.4.1 Description

The mime type for long-term key management message (ltkm) streams (e.g. stream carrying rights objects/entitlements) is data/bcast-ltkm.

A key management message stream is signaled in the following way:

m=data <port> UDP bcast-ltkm.

The actual format of the key management message stream is given by the kmstype in the ‘a=fmtp:bcast-ltkm’ line. Every a=fmtp line may contain a parameter streamid which identifies the particular LTKM stream.

MIME type parameters are signalled in the “fmtp:” line. MIME type parameters for LTKM as defined in Table 54 SHALL be supported.

Table 54: Parameters of the mime type bcast-ltkm

Parameter	Terminal support	Server support	Purpose
streamid	Mandatory	Mandatory	See Table 46

kmstype	Mandatory	Mandatory	See Table 46
serviceprovider	Mandatory	Mandatory	See Table 46

10.1.4.2 SDP Example for LTKM Stream

m=data 49230 UDP bcast-ltkm

c=IN IP4 224.2.17.12/127

a=fmtp:bcast-ltkm streamid=42; kmstype=oma-bcast-drm-pki; serviceprovider=SOMEID

10.1.5 SDP Entry Examples (Informative)

This section provides several examples illustrating how the parameters defined above are signalled in an SDP file. Note that these are simplified example i.e. lots of parameters are missing, but these have been omitted for clarity.

Example 1: This example shows a video and audio stream protected by both Long Term and Short Term Key Message streams using DRM Profile.

```
m=video 49168 RTP/AVP 96          // video stream & protocol
i=video
c=IN IP6 FF15:0:0:0:0:0:81:1BC         // stream address
a=rtpmap:96 MP4V-ES/1000          // payload type
a=fmtp:96 <rtp_param>             // rtp parameters
a=control:streamid:1              // stream identifier
a=stkmstream:3
m=audio 49170 RTP/AVP 97          // audio stream & protocol
i=audio
c=IN IP6 FF15:0:0:0:0:0:81:1BC         // stream address
a=rtpmap:97 MP4A-LATM/1000        // payload type
a=fmtp:97 <rtp_param>             // rtp parameters
a=control:streamid:2              // stream identifier
a=stkmstream:3

m=data 49172 UDP bcast-stkm

c=IN IP6 FF15:0:0:0:0:0:81:1BC         // stream address
a=bcastversion:1.0
a=fmtp:bcast-stkm     control:streamid=3;kmstype=oma-bcast-drm-
pki;serviceprovider=DiscountBCAST

m=data 49173 UDP bcast-ltkm

c=IN FF15:0:0:0:0:0:81:1BC        // stream address
a=bcastversion:1.0
a=fmtp:bcast-ltkm     control:streamid=4;kmstype=oma-bcast-drm-
pki;serviceprovider=DiscountBCAST
```

Example 2: This example shows a video and audio stream protected by Short Term Key Message streams using GBA_ME MBMS.

```
m=video 49168 RTP/AVP 96          // video stream & protocol
i=video
c=IN IP4 224.2.1.1                // stream address
a=rtpmap:96 MP4V-ES/1000          // payload type
```

```
a=fmtp:96 <rtp_param>            // rtp parameters
a=control:streamid:1             // stream identifier
a=stkmstream:3

m=audio 49170 RTP/AVP 97         // audio stream & protocol
i=audio
c=IN IP4 224.2.1.1               // stream address
a=rtpmap:97 MP4A-LATM/1000       // payload type
a=fmtp:97 <rtp_param>            // rtp parameters
a=control:streamid:2             // stream identifier
a=stkmstream:3

m=data 49172 UDP bcast-stkm      // keystream port no.
c=IN IP4 224.2.1.1               // stream address
a=bcastversion:1.0
a=fmtp:bcast-stkm     control:streamid=3;kmstype=oma-bcast-gba_me-
mbms;serviceprovider=DiscountBCAST
```

Example 3: This example shows two audio streams, each protected by a different key stream

```
m=audio 49170 RTP/AVP 96         // audio stream & protocol
i=audio_english
c=IN IP4 224.2.1.1               // stream address
a=rtpmap:96 MP4A-LATM/100        // payload type
a=fmtp:96 <rtp_param>            // rtp parameters
a=control:streamid:1             // stream identifier
a=stkmstream:3             //stkm with id 3 protects english audio track

m=audio 49172 RTP/AVP 97         // audio stream & protocol
i=audio_spanish
c=IN IP4 224.2.1.1               // stream address
a=rtpmap:97 MP4A-LATM/1000       // payload type
a=fmtp:97 <rtp_param>            // rtp parameters
a=control:streamid:2             // stream identifier
a=stkmstream:4             //stkm with id 4 protects spanish áudio track

m=data 49174 UDP bcast-stkm      // keystream port no.
i=short term key messages
c=IN IP4 224.2.1.1               // stream address
a=bcastversion:1.0               // stream address
a=fmtp:bcast-stkm     control:streamid=3;kmstype=oma-bcast-drm-
pki;serviceprovider=DiscountBCAST1

a=bcastversion:1.0

m=data 49175 UDP bcast-stkm      // keystream port no.
c=IN IP4 224.2.1.1               // stream address
a=fmtp:bcast-stkm     control:streamid=4;kmstype=oma-bcast-gba_u-
mbms;serviceprovider=DiscountBCAST2
a=control:streamid:4
a=kmstype:oma-bcast-gba_u-mbms        // key management system
a=kmsversion:1.0           // KMS version
```

```
a=keystreamtype:stkm              // keystream type
a=encryptedstreamid:2             // encrypted streamid
```

Example 4: This example shows how two separate providers can use different key streams to give access to the same video stream (audio stream left out for brevity). The different key streams carry the same keys.

```
m=video 49168 RTP/AVP 96          // video stream & protocol
i=video
c=IN IP4 224.2.1.1                // stream address
a=rtpmap:96 MP4A-LATM/1000        // payload type
a=fmtp:96 <rtp_param>             // rtp parameters
a=control:streamid:1              // stream identifier
a=stkmstream:2
a=stkmstream:3

m=data 49171 UDP bcast-stkm       // STKM stream port no.
c=IN IP4 224.2.1.1                // stream address
a=bcastversion:1.0                // BCAST version
a=fmtp:bcast-stkm     control:streamid=2;kmstype=oma-bcast-gba_me-
mbms;serviceprovider=supertv

m=data 49190 UDP bcast-stkm       // STKM stream port no.
i=short term key messages
c=IN IP4 224.2.1.1                // stream address
a=bcastversion:1.0                // BCAST version
a=fmtp:bcast-stkm     control:streamid=3;kmstype=oma-bcast-drm-
pki;serviceprovider=bargaintv
```

10.2 SDP Signalling of ISMACryp

The following notation SHALL be used to indicate that streams are encrypted at the content level (content encryption) using OMA BCAST. Terminals that support ISMACryp SHALL also support the following notation.

10.2.1 Overview

OMA signalling has stream signalling parameters. The stream signalling parameters describe the encryption of the stream.

1. IV length: Describes the size of the initialization vector in bytes.
2. Key indicator length: Describes the size of the key indicator in bytes.
3. Selective encryption: Indicate whether selective encryption is used for the session or not.
4. Salt key k_s: indicates the value of the salt key (used with IV to create the counter for AES in CTR mode as per [ISMACRYP11])

Selective encryption is optional since OMA streams are not required to have unencrypted media frames.

In addition to the stream-signalling parameters, there are two optional transport parameters.

1. delta IV length: Describes the maximum size of the optional media-frames initialization vector.
2. Key indicator per AU: Indicates key rotation on a media frame basis.

The delta IV length parameter is needed when media frames are interleaved in packets and unneeded otherwise. Key indicator per AU is needed when the stream has multiple keys and the packetizer might rotate a key between two media frames that are in the same packet.

10.2.2 Session Description Protocol Signalling

This section defines SDP [RFC2327] fmtp signalling for BCAST Terminals.

The SDP fmtp signalling SHALL use enc-isoff-generic as its format. All SDP signalling parameters and names are case-insensitive.

Generic SDP signalling conforms to RFC 2327 [RFC 2327]:

- m=<media> <port>/<number of ports> <transport> <fmt list>
- a=rtpmap:<payload type> <encoding name>/<clock rate>[/<encoding parameters>]
- a=fmtp:<payload type> <ISOFF-GENERIC-PARMS> <ENC-ISOFF-GENERIC-PARAMS>

The MIME media type name depends on the track's media type (i.e. the component subtype field in the 'hdlr' box): "video" if the media type is 'vide', "audio" if the media type is 'soun', and "application" otherwise:

<media> = "audio"|"video"|"application"

The MIME subtype name is "enc-isoff-generic":

<encoding name> = "enc-isoff-generic"

ISOFF-GENERIC-PARMS are OPTIONAL parameters that are defined in mpeg4-generic [RFC3640]. Some parameters, which were added to offer a codec agnostic solution, are defined below.

Codec identification

Codec identification is required to know the result after decryption. A mandatory parameter called "codec" is added. The value of this parameter MUST be compliant with RFC4281 [RFC4281]:

codec = """<id-simple> "/" <simp-list>"""

<id-simple> are <simp-list> defined in §3.3 of RFC4281 [RFC4281].

Codec initialization

All modern codecs needs parameters for their initialization. In the ISO file format (see [ISO-14496-12]), these parameters are stored in one or more boxes stored in the *SampleDescriptionBox*.

To carry this configuration, a suite of optional parameters is added:

config.*xxxx* = <*value*>

where :

- *xxxx* : the 4cc of a corresponding box contained in the sample description box
- <*value*> : the content of the box
  - coded in base-64
  - not including the length and name fields of the box
  - including the version and flags fields (if present).

- must be present in the *fmtp* line in the same order as the corresponding boxes in the sample description.

Optional parameters

SliceStartEndIndication

For the support of slices, an optional parameter called "subSampleStartEndIndication" is added. This parameter indicates whether the Slice-start-flag and the Slice-end-flag are present in the sample-header.

subSampleStartEndIndication = "0" | "1"

If subSampleStartEndIndication=1, the Slice-start-flag and the Slice-end-flag parameters MUST be present.

paddingIndication

For the support of padding bits, an optional parameter called "paddingIndication" is added. This parameter indicates whether the Padding-size field is present in the sample-header.

paddingIndication = "0" | "1"

If present, paddingIndication=1, the Padding-size field MUST be present and its size is 3 bits.

ENC-ISOFF-GENERIC-PARAMS are mandatory for the server and terminal to support and defined below.

Table 55: Network Optional Parameters used in SDP

Descriptor	Defined values (bytes)	Default value (bytes)
ISMACrypIVLength	0..8	4
ISMACrypDeltaIVLength	0..2	0
ISMACrypSelectiveEncryption	0..1	1
ISMACrypKeyIndicatorLength	0..255	4
ISMACrypKeyIndicatorPerAU	0..1	0

If the parameters are not defined in the SDP file, the above default values SHOULD be assumed.

ISMACrypIVLength describes the byte length of the initialization vector that is conveyed initially in the packet.

ISMACrypDeltaIVLength describes the byte length of the initialization vector, if any, that is conveyed with an individual AU.

ISMACrypSelectiveEncryption declares that the media stream uses selective encryption when it is set to 1, which indicates that the selective encryption bit will appear in the OMABCASTAUheader.

When ISMACrypKeyIndicatorLength is non-zero, a key indicator will appear in the OMABCASTAUheader. ISMACrypKeyIndicatorLength can signal a key indicator field that is 0 to 255 bytes in length.

master_salt_key

For the DRM Profile, when SRTP authentication is used, the 112 bit Master Salt (MS) MAY be signalled as follows if it is not sent in the STKM:

MasterSaltKey= MS where MS is the 112 bit MS, base64 encoded.

10.2.2.1 ISMACryp SDP Examples

This section provides several examples of how the use of ISMACryp is signalled via SDP.

Note that these are simplified examples i.e. lots of parameters are missing, but these have been omitted for clarity.

Example 1 **Encrypted H.263 video**

This example shows SDP entries for an encrypted H.263 video stream.

```
m=video 0 RTP/AVP 96
a=rtpmap:96 enc-isoff-generic/90000
a=fmpt:96         codec="video/3GPP2;s263";         config.d263=VmlWaQAKAA==;
DTSDeltaLength=22; randomAccessIndication=1; ISMACrypIVLength=4;
```

In this configuration (assuming no B-frames), the RTP packet would consist of:

- AU-headers-length field (16 bits) = 40
- initial_IV (32 bits) = IV of the AU fragment
- DTS-flag (1 bit) = 0
- RAP-flag (1 bit) = 1 if the AU fragment is part of an I-frame, 0 otherwise
- padding (6 bits) = 0
- fragment of the H.263 AU

Note that the M-bit in the RTP header is set to 1 if the RTP packet contains the last fragment of an H.263 access unit.

Example 2 **Encrypted AMR-NB audio**

This example shows SDP entries for an encrypted AMR-NB audio stream (assuming silence detection is not used).

```
m=audio 0 RTP/AVP 96
a=rtpmap:96 enc-isoff-generic/8000/1
a=fmpt:96         codec="audio/3gpp;samr";         config.damr=VmlWaQEAAQAB;
constantSize=13;          constantDuration=160;          ISMACrypIVLength=4;
ISMACrypDeltaIVLength=0; ISMACrypSelectiveEncryption =1
```

In this configuration, the RTP packet would consist of:

- AU-headers-length (16 bits) = 32 + 8 * number of AUs in packet
- For the first AU:
- AU_is_encrypted (1 bit) = 1 if AU is encrypted, 0 otherwise
- reserved (7 bits) = 0
- initial_IV (32 bits) = IV of the first AU
- For the following AUs:
- AU_is_encrypted (1 bit) = 1 if AU is encrypted, 0 otherwise
- reserved (7 bits) = 0
- One or more encrypted AMR audio frames

Example 3 **Encrypted H.264 video**

This example shows SDP entries for an encrypted H.264 video stream.

```
m=video 0 RTP/AVP 96
a=rtpmap:96 enc-isoff-generic/90000
a=fmpt:96                                                codec="video/mp4;avc1";
config.avcC=AULgDf/hAAlnQuANllQKD8gBAARozjyA;  config.btrt=AADFRAAGKiAABiog;
```

```
SliceStartEndIndication=1;    DTSDeltaLength=20;    randomAccessIndication=1;
ISMACrypIVLength =4;
```

The RTP packet structure is identical to the "avc-video" mode of enc-mpeg4-generic.

Example 4 **Encrypted MPEG-4 AAC audio**

This example shows SDP entries for an encrypted MPEG-4 AAC audio stream.

```
m=audio 0 RTP/AVP 96
a=rtpmap:96 enc-isoff-generic /48000/2
a=fmpt:96     codec="audio/3gpp2;mp4a.E1";     config.esds=AAEfFgAgrGAgrgi=;
sizeLength=13; indexLength=3; indexDeltaLength=3; ISMACrypIVLength =4;
```

The RTP packet structure is identical to the "AAC-hbr" mode of enc-mpeg4-generic.

Example 5 **Encrypted MPEG-4 video**

This example shows SDP entries for an encrypted MPEG-4 video stream.

```
m=video 0 RTP/AVP 96
a=rtpmap:96 enc-isoff-generic/90000
a=fmpt:96                                    codec="video/3gpp2;mp4v.20.9";
config.esds=AAEfFgAJFfejfzeKJZKEFKgrGAgrgi=;              DTSDeltaLength=22;
randomAccessIndication=1; ISMACrypIVLength =4;
```

The RTP packet structure is identical to the "mpeg4-video" mode of enc-mpeg4-generic.

10.2.3 Traffic Authentication

If SRTP authentication is used, the SDP signalling of SRTP authentication specified in Section 10.4 SHALL be used.

10.3 Service Guide Signaling

Session Description information is contained or referenced in Access fragment of the Service Guide [BCAST10-SG].

10.4 SDP Signalling of SRTP

When SRTP is used, the following specific paramaters SHALL be included into the SDP:

- a=SRTPAuthentication:n

where n is the SRTP authentication algorithm value for the authentication algorithm to use. Only values specified in [RFC4771] are allowed, i.e. values 0 and 1 representing NULL and HMAC-SHA-1-160 are not allowed.

- a=SRTPROCTxRate:R

where R is the value of the ROC transmission rate parameter, an integer between 1 and 65535 inclusive, as specified in [RFC4771].

11. Common Keys / Sharing Streams for DRM Profile and Smartcard Profile

This section explains how different keys are mapped between the DRM Profile and the Smartcard Profile. It also explains how a protected data stream can be shared between different operators using both DRM and Smartcard Profiles.

11.1 Service and Program Encryption Keys

For the DRM Profile, Service Encryption Keys (SEKs) and Program Encryption Keys (PEKs) are as described in Section 5.4.

For the Smartcard profile the PEK and SEK map to the same key and the differentiation is based on the Key Validity data, e.g. the PEK Key Validity data will define a shorter validity period than the SEK Key Validity data. This enables the same key to be used for both subscription and Pay-Per-View service offerings.

The mapping between Smartcard Profile keys and MBMS keys is described in Section 6.2.

11.1.1 Mapping of Encryption and Authentication Keys

The SEK/PEK used within the Smartcard Profile is not used directly to secure the delivery of Traffic Encryption Keys (TEKs). Instead the MIKEY protocol [RFC3830] uses the SEK/PEK to derive an integrity key (auth_key) and encryption key (encr_key).

The DRM Profile similarly utilizes separate encryption and authentication keys to encrypt the Traffic Keys and to authenticate STKMs. However, in the case of the DRM Profile the Service Encryption Key (SEK) and the Service Authentication Key (SAK) are not derived from the same key. Likewise, the PEK and the PAK are not derived from the same key.

A more detailed mapping of the encryption and authentication keys between the DRM and Smartcard profiles is provided in the following table:

Table 56: BCAST Encryption and Authentication Key Mapping

Purpose of Key	Service or Program Key	DRM Profile Key	Smartcard Profile Key
Encryption of STM	Service	SEK (128 bits)	MIKEY encr_key (128 bits derived from SEK)
Athentication of STKM	Service	SAK (160 bits derived from 128 bit SAS)	MIKEY auth_key (160 bits derived from SEK)
Encryption of STM	Program	PEK (128 bits)	MIKEY encr_key (128 bits derived from PEK)
Athentication of STKM	Program	PAK (160 bits derived from 128 bit PAS)	MIKEY auth_key (160 bits derived from PEK)

11.2 SEK, PEK and TEK Key IDs in STKM

The table below decribes the mapping between key identifiers used in the Smartcard Profile and DRM profile:

Table 57: Mapping between Key Identifiers Used in the Smartcard Profile and DRM profile

Key to locate	DRM Profile Identifier (contained in STKM)	Smartcard Profile Identifier (contained in STKM)
SEK	service_CID_extension (32 bit)	Key Domain ID || SEK ID (3+4=7 bytes)
PEK	program_CID_extension (32 bit)	Key Domain ID || PEK ID (3+4=7 bytes)
TEK for IPsec	SPI (32 bits)	SPI (32 bits) = 0x0001 ||

		MTK ID (2 bytes)
TEK for SRTP	MKI (8*key_indicator_length bits) Note that if compatibility with the Smartcard Profile is required MKI must be 2 bytes long	MKI = TEK ID (2 bytes) for SRTP implemenations aimed only at BCAST terminals. MKI = (SEK/PEK ID) || TEK ID (6 bytes) for for SRTP implemenations that are MBMS compatibile
TEK for ISMACryp	key_indicator (8*key_indicator_length bits)	TEK ID (2 bytes)

The terminal MUST use the SDP to locate the relevant STKM stream for the encrypted traffic stream it needs to decrypt.

For information on how the parameteres in the above table should be specified for the case where both DRM Profile and Smartcard Profile provide access to the same data stream, please refer to Section 11.3 below.

11.3 Sharing SRTP Protected Data Streams

This section describes how a protected data stream can be shared between different operators using both DRM and Smartcard Profiles. Two solutions for sharing protected data streams are described below:

- Section 11.3.1 introduces a solution that is compatible with the 3GPP MBMS key management specification.
- Section 11.3.2 introduces a solution aimed at BCAST terminals only.

Section 11.3.3 discusses the properties of these solutions.

11.3.1 Sharing 3GPP-MBMS Compatible SRTP Protected Media Streams

The way in which key identifiers are used by the SRTP implementation is based on the MBMS specification (cf. SRTP). Specification details related to this section are described in [BCAST10-MBMS-Adaptation] in the section titled “Sharing 3GPP-MBMS compatible SRTP protected media streams”.

11.3.2 Sharing a Protected Media Stream where Content is Aimed only at BCAST Terminals

Compared with Section 11.3.1, this section outlines how to handle the sharing protected stream(s) between different broadcast service providers none of which are using MBMS service protection.

The management and use of key identifiers for the protected media stream is based on the BCAST specification. It simplifies the way in which the MKI is constructed allowing the use of SRTP and ISMACryp at the content encryption layer. Note that the solution is not compatible with the 3GPP MBMS key management solution.

A shared TEK ID enables the retrieval of the correct TEK required to decrypt the protected media stream. This necessitates that:

- A single TEK ID SHALL be used to enable access to the corresponding shared protected stream among different broadcast service providers.
- The TEK ID SHALL be synchronised for all broadcast service providers.
- The same TEK material SHALL be used by all broadcast service providers. This means:
  - the same 128 bit Master Key (MK) SHALL be used.
  - the same 112 bit Master Salt (MS) SHALL be used. The MS MAY be NULL.

- the key derivation rate SHALL be zero for the DRM Profile. For the Smartcard Profile the key derivation rate may be zero or non-null.

SRTP key management parameters used when an SRTP stream is to be shared between DRM Profile and Smartcard Profile Terminals are summarised in the table below.

Table 58: SRTP Parameters – to enable sharing common stream

SRTP Parameter	DRM Profile value	Smartcard Profile value
MKI	same as Smartcard	TEK ID (2 bytes)
MK	same as Smartcard	TEK (random 128 bits)
MS	same as Smartcard	random 112 bits or NULL

The figures and section below explain the MKI format used to allow DRM Profile and Smartcard Profile to be shared between different broadcast service providers.

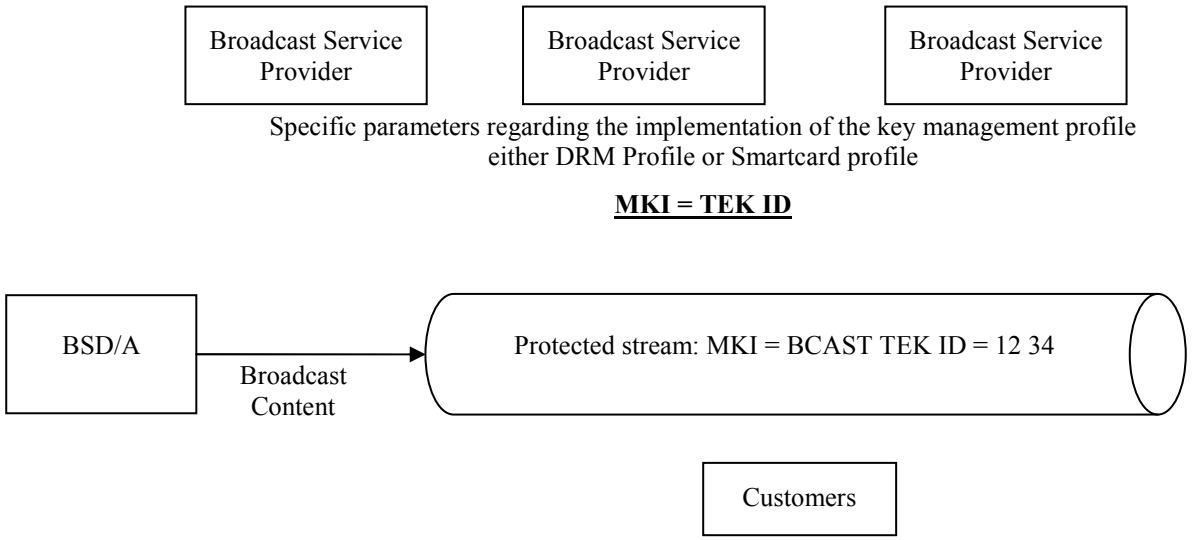


Figure 13 – Sharing a single protected media stream between several broadcast service providers using the Smartcard Profile and the DRM Profile, where there is no requirement to also share the protected stream with MBMS only terminals

Figure 13 illustrates how a single broadcast content distributed by the Service Provider (BSD/A) is shared between three broadcast service providers A, B and C. Broadcast service providers A, B and C implement either the DRM Profile or the Smartcard Profile.

The TEK material and the corresponding TEK ID must be shared among broadcast service providers. Broadcast service providers A, B and C generate and use their own SEK/PEK material to protect the shared TEKs. Each broadcast service provider constructs and broadcasts their own STKMs to their subscribers, containing the common TEK protected with the service provider specific SEK/PEK. There is no need to synchronise any key material or key identifiers above the TEK level.

The service provider can then broadcast the content encrypted with this common TEK by using SRTP. Upon reception the terminal retrieves the TEK based on the MKI where:

MKI = TEK ID (2bytes)

Each BCAST terminal can then retrieve the TEK needed to decrypt the shared protected stream from the STKM provided by their broadcast service provider.

The following can then be considered:

- A subscriber from broadcast service provider A has access to media streams 1, 2 and 3, using
  - ◊ SEK ID = SEK _IDA, key material = SEK A, and has a key validity period = 1 week. SEK A is transmitted by broadcast service provider A.
  - ◊ TEK ID = TEK _ID, key material = TEK, and has a key validity period = TEK _period, transmitted with broadcasted content.
- A subscriber from broadcast service provider B has access to media streams 1, 2 and 4, using
  - ◊ SEK ID = SEK _IDB, key material = SEK B, and has a key validity period = 1 month. SEK B is transmitted by broadcast service provider B.
  - ◊ TEK ID = TEK _ID, key material = TEK, and has a key validity period = TEK_period, transmitted with broadcasted content.

The value of the SEK/PEK ID is not shared and is specific to each broadcast service provider. The frequency of the update of SEK is up to each broadcast service provider.

In contrast, the value for TEK ID and TEK material have to be synchronised and coordinated for all broadcast service providers.

Summary:

In summary, for the Smartcard Profile to share protected media streams with DRM profile terminals, when the broadcast media is protected using SRTP, it must deviate from the MBMS specifications [TS 3GPP 33.246]. As described in Section 11.3.1, when SRTP is used in MBMS the MKI value is constructed as a concatenation of MSK ID (SEK/PEK ID) and MTK ID (TEK ID):

MKI = (MSK ID || MTK ID)

where MSK ID is 4 bytes long and MTK ID is 2 bytes long resulting in an MKI length of 6 bytes.

In contrast, when following the scheme described in this section, the MKI value is constructed using only the TEK ID (equivalent to the MTK ID in MBMS):

MKI = TEK ID = MTK ID

where TEK ID is 2 bytes long.

Restructuring the way that the MKI is formatted by omitting the SEK/PEK ID (equivalent to the MSK ID in MBMS) removes the need for broadcast service providers to synchronise the SEK/PEK ID and SEK update period as described in section 11.3.1. The deviation from the MBMS specification means that it is not possible for a media stream protected in the manner described in this section to also be shared by MBMS only terminals.

It is also necessary for terminals implementing the Smartcard Profile to recognise whether the MKI in the SRTP stream they are trying to decrypt is constructed in the way described in Section 11.3.1 or 11.3.2 in order to find the required TEK. This is achieved by looking at the MKI length: if it is 2 bytes long then the MKI corresponds to the TEK ID transported in the STKM; if the MKI length is 6 bytes long, the MKI corresponds to the SEK/PEK ID || TEK ID, both of which are transported in the STKM. Note that in the first case the MKI signalled in SRTP does not contain the SEK/PEK ID, but the SEK needed to decrypt the STKMs is still signalled in the STKM, i.e in the EXT MBMS payload.

11.3.3 Properties of the above Solutions

For the solution that is compatible with [3GPP TS 33.246 v7] the following can be stated:

- ◊ It is possible to share a protected media stream between a broadcast service provider using MBMS and a broadcast service provider using another broadcast bearer with either the Smartcard or DRM Profile.

◊ The requirement for all broadcast service providers that are sharing the same protected stream, regardless of the profile that they are using, is to use the same SEK/PEK ID. Using the same SEK/PEK ID further necessitates that the update frequency of the SEK and SEK/PEK ID must also be coordinated.

For the solution that aims protected media stream sharing at BCAST terminals only, the following can be stated:

◊ There is no need for broadcast service providers to synchronise key identifiers or key update periods above the TEK layer

◊ Broadcast service providers using MBMS service protection (following [3GPP TS 33.246 v7]) cannot share streams with broadcast service providers using the OMA BCAST service protection not relying on SRTP encryption protocol.

In summary, both solutions are possible, and signalling information within the STKM might be necessary.

11.4 Sharing Streams using ISMACryp

This section explains how a single protected media stream encrypted using ISMACryp can be accessed by different broadcast service providers.

As explained in Section 9.3, the OMABCASTAUHeader indicates the KeyIndicator (TEK ID) in the protected content stream. This KeyIndicator is used to find the relevant TEK used to decrypt the content in the STKM stream. In the head-end, the TEK and TEK ID are used during STKM generation by BSM / BSDA so that broadcast service providers can generate their own STKM streams (using DRM Profile or Smartcard Profile). These are broadcast together with the protected content stream.

The KeyIndicator can be found in the STKMs for the DRM Profile and for the Smartcard Profile. SDP signaling provides information on the relevant STKM streams (see Section 10.1) indicating whether the STKM stream is a DRM Profile stream or a Smartcard Profile stream. Furthermore, the "serviceprovider" string allows individual broadcast service providers to use their own STKM stream.

Once the correct STKM stream has been identified, the terminal can obtain the correct TEK and KeyIndicator, and hence match it to the TEK needed to decrypt the content.

To summarise:

- Content is encrypted using a key identified by KeyIndicator (TEK ID) in OMABCASTAUHeader
- STKMs are identified in SDP
- STKMs contain the TEK and associated TEK ID (KeyIndicator)
- The correct TEK is used to decrypt protected content

11.5 Sharing (P)DCF File Delivery Protection using TEK (Informative)

Section 9.4 explains how (P)DCF files can be protected by individual TEKs during file delivery.

This section highlights how the OMABCASTKeyInfo box can be used to allow both DRM Profile and Smartcard Profile signaling to be provided, allowing both to operate in parallel. Indeed, the Key Info box allows multiple "KeyInfo" entries to be present.

11.5.1 Use of OMABCASTKeyInfo Box

For the DRM Profile, the KeyIDType would indicate that information on KeyID corresponds to the DRM Profile.

For the Smartcard Profile, the KeyIDType would indicate information on KeyID corresponds to the Smartcard Profile. If multiple service providers are using the Smartcard Profile, multiple KeyInfo entries can be provided for multiple KeyIDs.

Note also that the use of the OMABCASTKeyInfo box allows OMA DRMv2 compatibility to be maintained, potentially allowing a (P)DCF to be provided that is compatible for OMA DRMv2, OMA BCAST DRM Profile and OMA BCAST Smartcard Profile.

Note also that 3GPP MBMS also uses the same approach, so that 3GPP MBMS signaling can also be provided in parallel. This allows interoperability across OMA DRM, OMA BCAST and 3GPP MBMS domains.

The OMABCASTKeyInfo box allows the STKM used to provide the TEK for decryption of DCF files to be stored and/or delivered within the DCF. Again, as this is contained within a single "KeyInfo" entry, STKMs for different key management systems can be provided in parallel.

12. Terminal Binding Key

In case of Smartcard Profile, a Permissions Issuer MAY elect to bind some or all of the content being broadcasted to valid terminals by the use of a Terminal Binding Key (TBK). This binding is in addition to the UICC binding provided by the Smartcard Profile. The binding is signalled in the SG and in the STKM and LTKM for the Smartcard Profile.

The following section and subsections are MANDATORY to support for Terminals with the Smartcard Profile for content protection. In all other cases, the sections are OPTIONAL for both server and terminal to support.

12.1 TBK Generation

If Terminal Binding is desired for any of the content being broadcasted, the Permissions Issuer will define the TBK to be a randomly, or pseudo-randomly, generated key of 128 bits. This key will be shared by all compliant non-revoked devices. For each TBK generated, the PI will issue a unique 32 bit TerminalBindingKeyID.

The TBK can be changed by the PI at will, such as when devices need to be revoked. The TBK change can occur as seldom as never once it was set, or as frequently as desired.

A single TBK can be set for the PI to use with all terminal-bounded content, or a separate TBK may be set for contents related to each SG entry. The scope and lifetime of the TBK are implementation specific.

12.2 Encrypting of TEKs with TBK

The TBK used to protect TEKs is used as follows:

Upon generation of each TEK, the PI determines if it would like to bind the TEK also to the terminal. If not, the TEK is processed as usual (encrypted by SEK/PEK). If terminal binding is desired, a TBK has already been generated, given an ID (TerminalBindingKeyID), and this ID was added to the SG entry. For each TEK generated while terminal binding is on, an *Encrypted_TEK* is computed as follows:

$$Encrypted\_TEK = \text{AES-ECB}_{TBK}(TEK)$$

Where *TBK* is fed as the 128-bit key that is used (referred to as *KEK* in AES-ECB), and *TEK* is fed as the key to be encrypted (referred to as *plaintext* in AES-ECB). The resulting *encrypted_TEK* is processed from that point onwards instead of the original, plaintext, *TEK*.

12.3 Decrypting of TEKs with TBK

When content is selected to be processed from the SG, the terminal will note the ID of the TBK that is being used with that content, if at all. If a TBK of the specified ID is not available in the terminal cache, the terminal MAY attempt to obtain it, as described in Section 12.4.

When processing a STKM, if the terminal binding flag bit is set, the terminal will fetch from its cache the correct TBK, according to the TerminalBindingKeyID specified in the SG. This fetch may occur once when processing the LTKM to avoid repeatedly retrieving the same value from the cache. The terminal will use this TBK to decrypt, using AES-ECB, encrypted TEKs that are received from the UICC, before these are used for content decryption.

The effect of this additional decryption, that is required when terminal binding is on, is that an unapproved terminal, which does not possess the correct TBK, is unable to utilize the output of the UICC to deduce meaningful TEK values. It is perceived as infeasible to obtain the correct TEK values from $\text{AES-ECB}_{TBK}(TEK)$ without knowledge of TBK.

12.4 TBK Acquisition

The PI SHALL deliver any requested TBK value to any requesting Terminal, as long as the Terminal was successfully authenticated and was positively identified as a Terminal that has not been revoked.

The protocol by which TBK values are delivered is initiated by the Terminal at any time, typically when an SG entry indicates the requirement for a TBK that is not cached by the device.

To obtain a TBK value, the Terminal starts an HTTPS session with the PI server (see Section 6.11.2 for the Smartcard Profile). The HTTPS session SHALL be based on mutual authentication using both client and server certificates. The server SHALL verify the authenticity and the validity of the client certificate and SHALL consider the identity of the Terminal to be the one indicated by the certificate.

Following the HTTPS session establishment, the Terminal SHALL send the *BCAST_Client_ID* (see Section 6.11.2). The PI server MAY use this ID information, but if doing so it SHALL assure that the identity of the terminal as reflected in the *BCAST_Client_ID* matches the identity indicated by the client certificate mentioned above.

If the terminal ID that is supplied in the *BCAST_Client_ID* does not match the ID indicated by the client certificate, or if the ID reflects a device that has been revoked, or if the identification failed, or if the HTTPS session failed, then the PI server SHALL close the connection without providing the requested TBK but while returning a "*Forbidden*" error instead.

If the version number sent in the *BCAST_Client_ID* reflects an inadequately old version, the PI server SHALL close the connection without delivering the requested TBK, and MAY indicate the URI at which an update or further information can be found (see response table).

If none of the above conditions was met, then the PI server SHALL return the required TBK over the secure connection and close the connection.

Upon reception of the requested TBK, the terminal MAY cache it. The policy and size of this cache is implementation specific.

The Figure below illustrates the steps explained above.

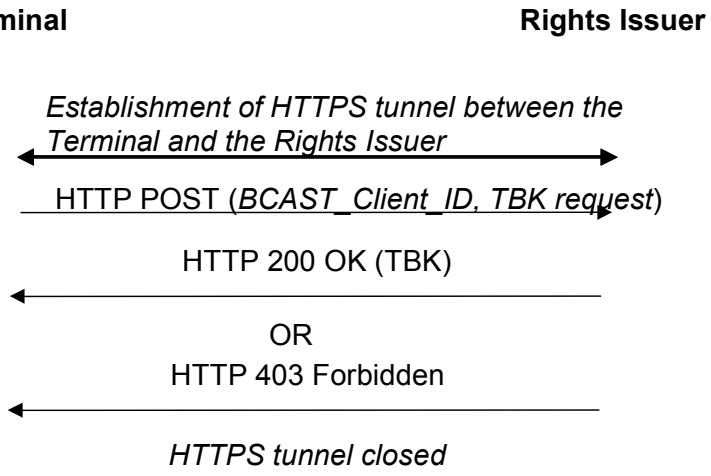


Figure 14 – Mutual Authentication, sending BCAST_Client_ID and TBK exchange

The BCAST_Client_ID and TBK request SHALL be sent using the following notation:

```
POST / HTTP/1.1
User-Agent:BCAST_Client_ID=BCAST_Client_ID
TBK_request=TerminalBindingKeyID
```

Where:

"BCAST_Client_ID=" is text allowing the server to identify the BCAST client ID and *BCAST_Client_ID* is the actual value.

BCAST_Client_ID is Base64 encoded.

"TBK_request=" is text allowing the server to identify the TBK request and TerminalBindingKeyID is the ID of the TBK key being requested.

TerminalBindingKeyID is Base64 encoded.

The Permissions Issuer response, if successful SHALL be sent using the following notation:

```
HTTP/1.1 200 OK
Server: BCAST Permissions Issuer
Date: Thu, 08 Jan 2004 10:13:18 GMT
TBK=TBK
```

Where:

"TBK=" is text indicating the TBK follows.

TBK is the actual Terminal Binding Key

TBK is Base64 encoded.

If the Permissions Issuer refuses to issue the TBK it SHALL send the following response:

```
HTTP/1.1 403 Not acceptable
Server: BCAST Permissions Issuer
Date: Thu, 08 Jan 2004 10:13: GMT
```

Or:

```
HTTP/1.1 403 Not acceptable
Server: BCAST Permissions Issuer
Date: Thu, 08 Jan 2004 10:13:18 GMT
Update_URI=Update_URI
```

Where:

"Update_URI=" is text indicating that the URI where update or further information can be obtained, follows.

Update_URI is the URI where an update or further information can be obtained.

Update_URI is Base64 encoded.

13. Server Side Interfaces and Messages

Message flows can be found in the OMA BCAST AD [BCAST10-Architecture].

13.1 Interface SP-4

Interface SP-4 has three functions:

1) To deliver Service and Program key material from SP-M in the BSM to the SP-KD in the BSD/A for the service and content protection.

2) To deliver the LTKM and/or Registration key material from SP-M in the BSM to SP-KD in the BSD/A, for subsequent broadcast distribution of these data..

3) To deliver of the STKM's from the BSM to the BSD/A.

A BSM that support service and/or content protection SHALL support Interface SP-4. A BSD/A that support service and/or content protection SHALL support Interface SP-4.

Two options are given for Interface SP-4:

- Using DVB Simulcrypt based interfaces
- Using BCAST specific interfaces

Interface SP-4 MAY support DVB Simulcrypt as specified in Section 13.1.1.

Interface SP-4 MAY support OMA BCAST specific signalling as specified in Section 13.1.2.

Interface SP-4 SHALL support either DVB Simulcrypt as specified in Section 13.1.1 or OMA BCAST specific signalling as specified in Section 13.1.2.

13.1.1 Interface SP-4: Adaptation of DVB Simulcrypt Head-End Interfaces to the OMA BCAST Environment

This section defines the use of a DVB Simulcrypt interface for SP-4. All normative text in this Section (13.1.1 and sub-sections) applies if DVB-H Simulcrypt is supported over interface SP-4. In this case, support of the interfaces defined in Section 13.1.1.2 is MANDATORY.

Simulcrypting, from a device perspective, means that signalling is available that allows the device to acquire the necessary information based on the supported Key Management Systems (KMS), either the DRM Profile or the Smartcard Profile. Such signalling is obtained from the Service Guide descriptors and/or the SDP. It also allows multiple Service Providers to generate STKMs and LTKMs i.e. multiple BSMs in the BCAST architecture.

At head-end side, supporting both profiles has implication on the architecture. The final data-cast shall include all necessary information and some data, such as the TEK, have to be shared among the various KMS to ensure well formed STKM. Simulcrypt also defines interfaces between various entities in the head-end. The original Simulcrypt specifications have been extended by DVB to allow a full support of broadcast over IP. This section describes how the DVB Simulcrypt head-end interfaces are adapted to the OMA BCAST environment, where the system is considering IP transport and encryption with IPsec, SRTP or ISMACryp.

13.1.1.1 Reference DVB Head-end Architecture

Figure 15 (extracted from [SIMULCRYPT]) illustrates the DVB Head-end reference architecture as described in [SIMULCRYPT]. For further information on the Simulcrypt system in a DVB environment, refer to this specification.

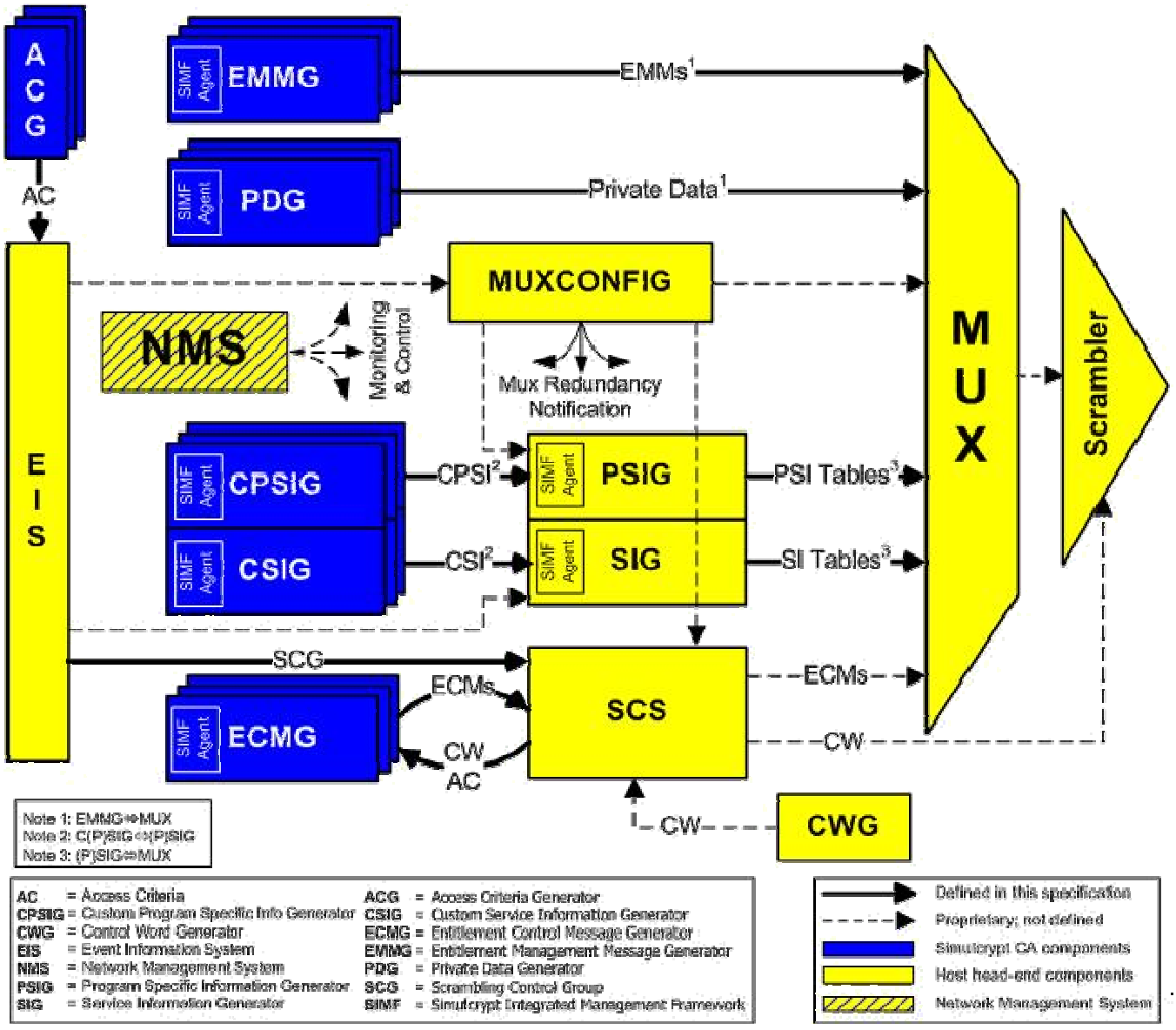


Figure 15 – Reference DVB Head-end Architecture

13.1.1.2 OMA BCAST Head-end Architecture and Interfaces

From Figure 15, the elements to take into account in the OMA BCAST head-end system architecture are (as described in [SIMULCRYPT]). Note that an STKM corresponds to an ECM in Simulcrypt:

- SCS (SimulCrypt Synchroniser). The role of the Simulcrypt Synchronizer is to:
    - Establish TCP connections with ECMGs (STKM generator) and setup one channel per connection;
    - Setup streams within channels as needed and allocate ECM_stream_id values;
    - Get the Control Words (TEK) from the CWG (TEK generator);
    - Supply the CWs to the relevant ECMGs on the relevant streams, as well as any KMS specific information, if any.
    - Acquire ECMs (STKM) from the ECMGs;

- Synchronize the ECMs (STKMs) with their associated Crypto periods according to channel parameters;
- Submit these ECMs (STKMs) to the multiplexer and request their repetition according to the channel parameters;
- Supply the CW (TEK) to the scrambler for use in the specified Crypto period.

• CWG (Control Word Generator - TEK generator). This component is responsible for generating control words used in scrambler initialization stream. The exact functionality of the Scrambler is implementer specific. The Control Word Generator shall be able to provide the SCS with control words.

• ECMG (Entitlement Control Message Generator) or BCAST STKM Generator (STKMG). The ECMG shall receive CWs in a CW provision message as well as access criteria and shall reply with an ECM or an error message. The ECMG does not support ECM repetition. This corresponds to STKM generation in BCAST.

• EMMG (Entitlement Management Message Generator) or BCAST LTKM Generator (LTKMG). This component, supplied by the KMS system provider shall interface over a specified interface to the multiplexer. The EMMG initiates connections to the multiplexer. This corresponds to LTKM generation in BCAST.

Mapping on the OMA BCAST head-end architecture with these elements is shown in Figure 16. For SP-4 to be applicable in this diagram, the entity containing the ECMG/STKMG must be the BSM.

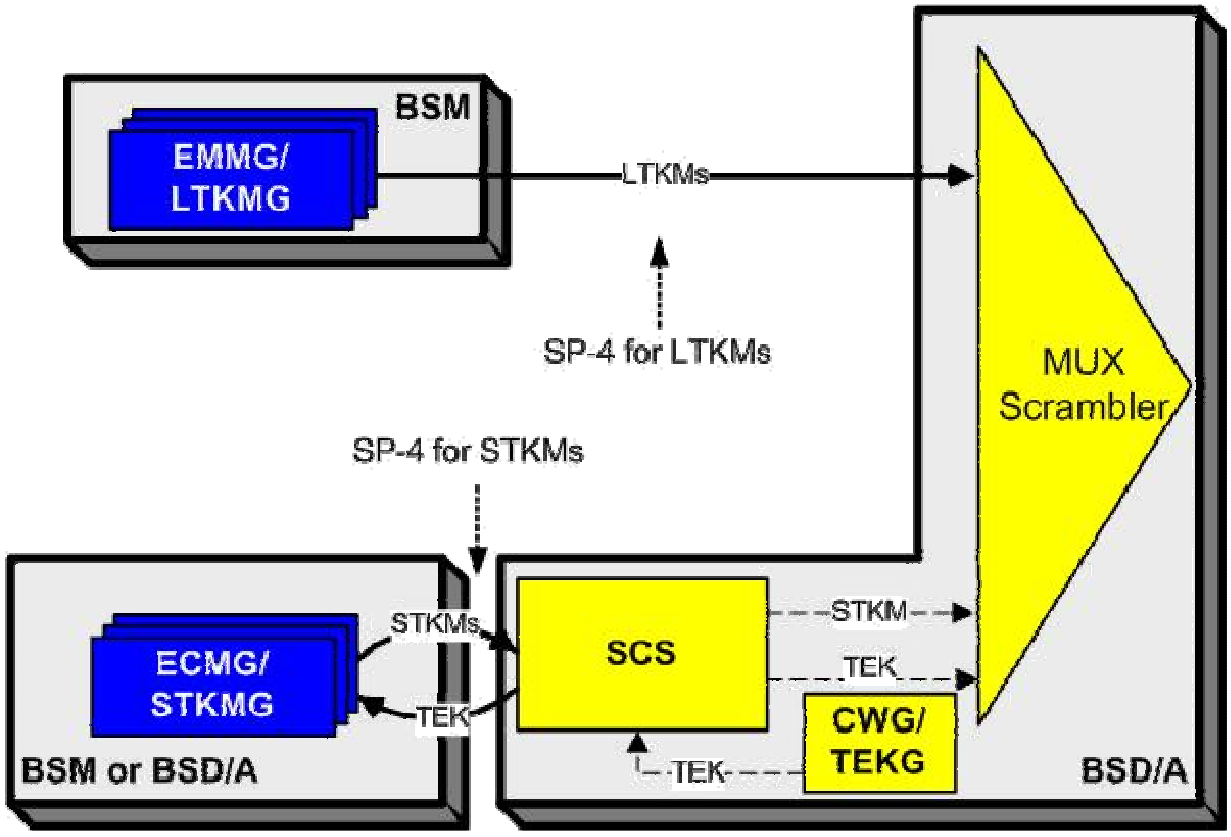


Figure 16 – OMA BCAST Head-end Architecture

The interfaces between the different elements are then as follows. The reference architecture can be mapped to BCAST head-end architecture as follows:

1. Interface ECMG/STKMG ⇔ SCS SHALL be implemented according to [SIMULCRYPT], Section 5 and the modifications in this specification, Section 13.1.1.3.1.

2. Interface EMMG/LTKMG ⇔ Multiplex MAY be implemented according to [SIMULCRYPT], Section 6 and the modifications in this specification, Section 13.1.1.3.2.

Any other interface is out of the scope of this specification and may be proprietary.

Encryption parameters for IPsec, ISMACryp or SRTP are generated by the TEK generator and sent to the SCS. The SCS transfers them to the STKMG as defined in [SIMULCRYPT], allowing the STKMG to generate the appropriate STKMs for DRM Profile and / or Smartcard Profile.

13.1.1.3 Adaptation of Simulcrypt Interfaces to OMA BCAST

13.1.1.3.1 Interface ECMG/STKMG ⇔ SCS

This interface allows a KMS system to provide a SCS with ECMs/STKMs under the control of this SCS. The following provides adaptations of some parameters on this interface. The combination Super_CAS_id + ECM_id identifies uniquely an STKM stream in the whole system. On this interface, the SCS sends TEKs to an ECMG to allow it to generate STKMs. The SCS knows which ECMG it contacts by using the Super_CAS_id. The SCS indicates to the ECMG for which STKM stream the TEKs are used for STKM generation by using the ECM_id, which links to the stkmstream value (also used in the SDP). This allows the ECMG to make the link with the Service Guide information related to the protected stream.

The Super_CAS_id is a 4-byte identifier that uniquely identifies a KMS system provider and BSM. The first 2 bytes of the Super_CAS_id for the DRM Profile are 0x01. The first 2 bytes of the Super_CAS_id for the Smartcard Profile are 0x02. The last 2 bytes are defined by the user (as an example, they can allow to identify the "serviceprovider" or BSM)

The ECM_id is a 2-byte identifier internal to a given ECMG. It is used by the MUX to map the ECM to the correct IP address and port. It is equal to the stkmstream value, as defined in Section 10.1.3 (in this case, the stkmvalue has to be unique across all SDP for a given KMS in a BSM) or it is an ID valid in the head-end only that allows both entities to uniquely identify STKM streams.

The ECM_datagram format is extended for BCAST and can also be formatted as STKM. As a consequence, the section_TSpkt_flag is equal to:

0x02	**The ECMs carried on the interface are in binary STKM format as defined for the DRM Profile or the Smartcard Profile.**
All other values are DVB reserved and SHALL not be used.	

The ECM_datagram is the actual STKM to be sent by the SCS to the MUX. Depending of the value of the Super_CAS_id, it is either a DRM Profile STKM as defined in Section 5.5 or a Smartcard Profile STKM as defined in Section 6.7.

The CP_CW_combination is the concatenation of the Crypto period number, the Key System Information (KSI) and the Traffic Key Material (TKM) from which the TEK and the optional Traffic Authentication Key (TAK) are derived. The meaning and length of the KSI and TEK are defined in the Simulcrypt specification for IPsec, ISMACryp and SRTP. BCAST has extended these for IPsec and for DCF_algo, as shown below:

IPsec Option 1, (as specified in [SIMULCRYPT], can be used with DRM Profile):

IPsec:

- The KSI = SPI, and KSI length = 4 bytes
- IPsec (no auth): TKM = TEK, TKM length = 16 bytes
  - Note: TEK = IPsec encryption key (see Section 5.5.1 and Section 9.1)
- IPsec (auth): TKM = TEK || TAS, TKM length 32 bytes
  - Note: TEK = IPSec encryption key, TAS = IPsec authentication seed (see Section 5.5.1 and Section 9.1)

IPsec Option 2 (can be used with Smartcard Profile):

- The KSI = SPI, and KSI length = 4 bytes
- TKM = TEK, TKM length = 16 bytes

    - Note: IPsec encryption key and authentication seed are derived from TEK using MIKEY PRF (see Section 6.7.3 and Section 9.1)

- Note: The Scrambler derives the encryption key and possible the authentication key from TEK, using RAND and CSB ID. How RAND and CSB ID are shared between STKMG and scrambler is outside of the scope of the current specification.

DCF_algo for DRM Profile:

- The KSI is the Key Identifier, and 1<= KSI length =< 255 bytes.
- The TKM is identical to the TEK, and the TKM length = 16 bytes.

DCF_algo for Smartcard Profile:

- The KSI is the TEK ID, and KSI length = 2 bytes.
- The TKM is identical to the TEK, and the TKM length = 16 bytes.

13.1.1.3.2 Error messages on ECMG/STKMG ⇔ SCS

Errors messages defined in [SIMULCRYPT], section 5.6, are supported. Errors messages defined in Appendix C are also supported and are in the range 0x8000 to 0xFFFF, i.e., for example, error message 000 defined in Appendix C is coded as 0x8000.

13.1.1.3.3 Using ECMG/STKMG ⇔ SCS in BCAST (Informative)

This section shows a possible method to use the ECMG/STKMG ⇔ SCS interface in the scope of BCAST. This is provided as an example only.

There is one instantiation of the SCS for each encrypted service in the BSDA. The connection between a SCS and an ECMG/STKM is established in three steps:

- A TCP connection is established as described in section 5.1.2 of [SIMULCRYPT]
- A "Channel" is set-up on top of this TCP connection. The SCS connects to an ECMG/STKMG in the BSM over a TCP connection and assigns to this connection an ECM_channel_id value. This value allows the SCS to uniquely identify the channel to the ECMG/STKMG. This first message contains the super_CAS_id that states which KSM and which service provider are considered. The ECMG/STKMG replies with a status message that contains some information, including its optimal crypto-period, the maximum number of STKM streams it can concurrently create, and the section_TSpkt_flag that defines the format of the STKM it will generate (full detail of this message is given in the channel_status message of section 5.4.3 in [SIMULCRYPT]). It can also reply with a channel_error message.
- On top of this Channel, for each STKM stream that has to be created for the service, a "Stream" is created. The SCS sends a set-up message to the ECMG/STKMG that contains the effective crypto-period duration to use for this STKM stream, the ECM_stream_id equivalent to the ECM_channel_id, and the ECM_id equal to the stkmstream. The ECMG/STKMG replies with a stream_status message or a stream_error message.

Once these three steps are completed, the ECMG/STKMG can provision the SCS with STKMs on the established stream. The CW_provision message allows the SCS to send to the ECMG/STKMG the necessary material (depending on the selected scrambling algorithm, i.e. IPsec, SRTP, or ISMACryp) to create STKMs that are send back in ECM_message response. Sending of TEKs in the CW_provision message can be optionally encrypted with a selectable algorithm.

13.1.1.3.4 Interface EMMG/LTMKG ⇔ Multiplex

This interface allows a BSM to provide a Multiplex with EMMs (LTKMs) under the control of the BSM. This interface applies only if the BSM is creating LTKMs as described for the DRM Profile. LTKMs created for the Smartcard Profile are not broadcasted, but sent over the unicast link the device. The combination {data_type + client_id + data_id} identifies uniquely this new LTKM data stream in the whole system.

The data_type is equal to 0x00, i.e. for EMM/LTKM data which is equivalent to LTKM in OMA BCAST.

The client_id is a 4-byte identifier uniquely identifying a BSM. The first 2 bytes of the client_id are 0x01 (value for the DRM Profile). The last 2 bytes are defined by the user (as an example, they can allow to identify the "serviceprovider" or BSM).

The data_id is a 2-byte identifier correspond to the ECM_id for the ECMG/STKMG, i.e. it is an internal identifier to the EMMG/LTKMG allowing the MUX to map the EMM/LTKM to a given IP address and port.

The datagram format is extended and can also be formatted as LTKM. As a consequence, the section_TSpkt_flag is equal to:

0x02	**The EMMs carried on the interface are in binary LTKM format as defined for the DRM Profile.**
All other values are DVB reserved and SHALL not be used.	

The datagram is the actual LTKM sent to the MUX.

13.1.1.3.5 Error messages on EMMG/LTMKG ⇔ Multiplex

Errors messages defined in [SIMULCRYPT], section 6.2.6, are supported. Errors messages defined in Appendix C are also supported and are in the range 0x8000 to 0xFFFF, i.e., for example, error message 000 defined in Appendix C is coded as 0x8000.

13.1.1.3.6 Using EMMG/LTMKG ⇔ Multiplex in BCAST

This section shows a possible method to use the EMMG/LTKMG ⇔ Multiplex interface in the scope of BCAST. This is provided as an example only.

A channel and then a stream are created over a TCP connection. This follows the same principle than the one presented in Section 13.1.1.3.3 for the ECMG/STKMG ⇔ SCS. Each creation level allows negotiating parameters related to this LTKM exchange, parameters are, among others, the KMS and service provider, the format of data (LTKM in this case).

A first difference appears in the optional possibility to negotiate bandwidth allocation for the LKTM stream. The EMMG/LTKMG can request a specific bandwidth, the Multiplex replies with the effective allocated bandwidth.

A second difference is that the LTKM stream can be provided to the Multiplex over UDP instead of TCP.

13.1.2 BCAST Specific Interface

This section defines a BCAST specific interface that is not compatible with the DVB Simulcrypt interface described above. All normative text below applies if interface SP-4 follows this section.

13.1.2.1 Protocol Stacks

The following protocol stack SHALL be used for messages between the BSD/A and the BSM connected via interface SP-4.

Figure 17 – Protocol Stack for SP-4-1

HTTPS that SHALL be based on .SSL3.0 [SSL30] and TLS 1.0 [RFC2246] SHALL be used to secure the interface between the BSD/A the BSM. All the messages defined over the SP-4 interface are XML documents. The XML schema definition is specified in [BCAST10-XMLSchema-SPCP-Backend].

13.1.2.2 Service and Program key material delivery

For the DRM Profile, SEAK and/or PEAK is sent from the BSM to the BSD/A.

For the Smartcard Profile, SEK and/or PEK is sent from the BSM to the BSD/A.

As these messages allow the delivery of high-level key material from BSM to BSD/A, the BSM MAY decide not to do so. This means the BSM MAY decide not to send Key_Delivery messages or MAY send an empty Key_Delivery_Confirmation message with Status code 011 "Operation not Permitted". If the BSD/A receives such a reply then the STKM Delivery section applies (see Section 13.1.2.4).

13.1.2.2.1 Message flows

Tags are defined in the following table to identify the type of each message. There are two cases for the delivery of SEAK or SEK and/or PEAK or PEK.

1. In the first case the BSD/A sends a Key_Request message to the BSM. The BSM then sends a Key_Request_Response message to the BSD/A.

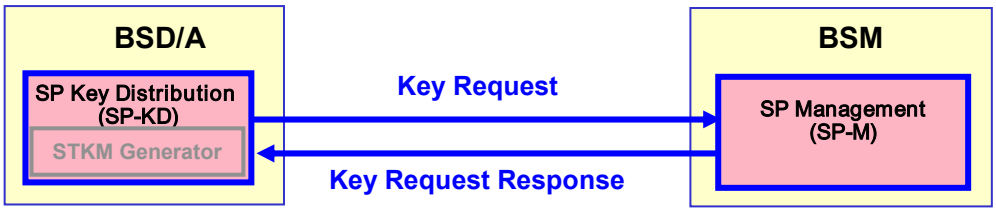


Figure 18 – Message Flow Between BSD/A and BSM for Delivery of Service and Program Key Material

2. In the second case the BSM sends a Key_Delivery message to the BSD/A. The BSD/A then sends a Key_Delivery_Confirmation message to the BSM.

Figure 19 – Alternative Message Flow Between BSD/A and BSM for Delivery of Service and Program Key

Tag	Message Type	Key
1	Key Request	SEAK/PEAK or SEK/PEK
2	Key Request Response	SEAK/PEAK or SEK/PEK
3	Key Delivery (same as Key Request Response)	SEAK/PEAK or SEK/PEK
4	Key Delivery Confirmation	SEAK/PEAK or SEK/PEK

13.1.2.2.1.1. Key Request

This message is sent from the BSD/A to the BSM for the acquisition of SEAK/PEAK or SEK/PEK, which in turn enables BSD/A to generate Short Term Key Messages (STKMs)

Name	Type	Category	Cardinality	Description	Data Type
KeyRequest	E			Key Request Message Contains the following attributes - tag - version - messageID - destination - source - time Contains the following elements - GlobalServiceID - GlobalContentID - ScheduleID - KeyStartTime - KeyEndTime	
tag	A	M	1	Identifier for the message type	unsignedByte
version	A	O	0..1	BCAST enabler version supported by this message	unsignedInt
messageID	A	M	1	This message ID	string
destination	A	M	1	BSM ID (Note: To be independent of the underlying network protocols, Destination is included in the message.)	string
source	A	M	1	BSD/A ID (Note: To be independent of the underlying network protocols, Source is included in the message.)	string
time	A	O	0..1	The time when this message is sent. This field contains the 32bits integer part of an NTP time stamp.	unsignedInt
GlobalServiceID	E1	M	1	Identifier of the service to be encrypted	anyURI
GlobalContentID	E1	O	0..1	Identifier of the content that is protected. Used if service protection is program based. Only GlobalContent ID which is related to the GlobalService ID is allowed.	anyURI
ScheduleID	E1	O	0..1	Identieifer of the schedule that is protected. Used if service protection is program based. Only Schedule ID which is related to the GlobalService ID is allowed.	anyURI
KeyStartTim	E1	M	1	Provides the start time of the period for which the BSD/A requires a SEAK or SEK and/or PEAK or PEK	unsignedInt

e				for creating secured STKMs This field expressed as the first 32bits integer part of NTP timestamps.	
KeyEndTime	E1	M	1	Provides the end time of the period for which the BSD/A requires a SEAK or SEK and/or PEAK or PEK for creating secured STKMs. This field expressed as the first 32bits integer part of NTP timestamps.	unsignedInt

13.1.2.2.1.2. Key Request Response

After the reception of the Key_Request message, the BSM sends this message to the BSD/A for the delivery of SEAK/PEAK or SEK/PEK. In case a SEAK or SEK is used for Service Protection, the use of the SEAK or SEK is bound by its start and end-times. During the lifetime of the Service, the SEAK can be changed periodically. In case a PEAK is used for Service Protection, the PEAK is used throughout the total lifetime of the Program. If both PEAK and SEAK are used in parallel, then the TEK encrypted with the PEK and the PEK encrypted with the SEK SHALL be present in the STKM. When only the PEAK is provided, the STKM should only include the TEK encrypted with the PEK.

Name	Type	Category	Cardinality	Description	Data Type
KeyRequest Response	E			Response to the Key Request message Contains the following attributes: tag version messageID destination source status time Contains the following elements: GlobalServiceID GlobalContentID ScheduleID SPPType ServiceKey ProgramKey AccessCriteriaDescriptor ProtectionAfterReceptionFlag TerminalBindingFlag	
tag	A	M	1	Identifier for the message type	unsignedByt

					e
version	A	O	0..1	BCAST enabler version supported by this message	unsignedInt
message ID	A	M	1	Key Request Message ID	string
destination	A	M	1	BSD/A ID (Note: To be independent of the underlying network protocols, Destination is included in the message.)	string
source	A	M	1	BSM ID (Note: To be independent of the underlying network protocols, Source is included in the message.)	string
status	A	M	1	Indication of the reception status of Key Request Message. Global Status codes are used as specified in Appendix C.	unsignedByte
time	A	O	0..1	The time when this message is sent. This field contains the 32bits integer part of an NTP time stamp.	unsignedInt
GlobalService ID	E1	M	1	Identifier of the service to be encrypted	anyURI
GlobalContentID	E1	O	0..1	Identifier of the content that is protected. This field is mandatory if GlobalContent ID was provided in the key request message.	anyURI
ScheduleID	E1	O	0..1	Identifier of the schedule that is protected. This field is mandatory if schedule ID was provided in the key request message.	anyURI
SPPType	E1	M	1	This specifies the type of the Service protection profile used by the BSM. 0 if service protection profile == DRM Profile 1 if service protection profile == Smartcard Profile 2-127 reserved for future use 128-255 reserved for proprietary use	unsignedByte
ServiceKey	E1	O	0..N	It specifies the SEAK or SEK Contains the following attribute: keyIdentifier value Contains the following elements: ServiceKeyStart ServiceKeyEnd ServiceKeyMTKStart ServiceKeyMTKEnd	
keyIdentifier	A	M	1	Provides the identifier of the SEAK/SEK. The SEAK/SEK identifier is the same as the one provided to the terminal the LTKM and is included with the STKM	hexBinary

				generated by the BSDA The SEAK/SEK identifiers are as defined for each profile in this specification.	
value	A	M	1	This field contains the SEAK if SPP type == 0 This field contains the SEK if SPP type == 1	hexBinary
ServiceKeyStart	E2	M	1	Provides the start time of the period in which the SEAK or SEK provided can be used by the BSD/A in creating secured STKMs. This field expressed as the first 32bits integer part of NTP timestamps.	unsignedInt
ServiceKeyEnd	E2	M	1	Provides the end time of the period in which the SEAK or SEK provided can be used by the BSD/A in creating secured STKMs This field expressed as the first 32bits integer part of NTP timestamps.	unsignedInt
ServiceKeyMTKStart	E2	O	0..1	TEK ID start value for SEK/PEK validity This field is mandatory if SDPP type ==1	hexBinary
ServiceKeyMTKEnd	E2	O	0..1	TEK ID end value for SEK/PEK validity This field is mandatory if SPP type ==1	hexBinary
ProgramKey	E1	O	0..1	This field contains the PEAK if SPP_type == 0 and is only applicable to the DRM Profile. This field SHALL NOT be used for the Smartcard Profile. In the Smartcard Profile there is no service key / program key hierarchy available. For the Smartcard Profile the PEK is send using Service Kery fields as described above. Note: Either Service Key, Program Key or both SHALL be included for the DRM Profile. Contains attribute: - keyIdentifier - value	hexBinary
keyIdentifier	A	M	1	Provides the identifier of the PEAK/PEK. The PEAK/PEK identifier is the same as the one provided to the terminal with the LTKM.	hexBinary

value	A	M	1	This field contains the PEAK if SPP type == 0 This field contains the PEK if SPP type == 1	hexBinary
AccessCriteria Descriptor	E1	O	0..N	The Access Criteria Descriptor to be included in the STKM. Whenever access criteria are defined for a piece of Content, then these access criteria take precendence over the access criteria which where defined for the service to which the content item is related.	hexBinary
ProtectionAfterReception Flag	E1	M	1	2 bit field defining the required protection after the removal of the service protection, as specified paragraph 6.3.1	unsignedByte
TerminalBindingKey	E1	O	0..1	An element indicating that a terminal binding key must be used. It contains the following attributes: - keyIdentifier - value	hexBinary
keyIdentifier	A	M	1	Number identifying the Terminal Binding Key ID (TBK ID) that is needed to access the service. This element is only present if the TerminalBindingFlag is present.	hexBinary
value	A	M	1	The value of the terminal binding key	hexBinary

13.1.2.2.1.3. Key Delivery Message

This message is sent from the BSM to the BSD/A for the delivery of SEAK/PEAK or SEK/PEk without a request from the BSD/A. If the BSD/A receives this message, then the BSD/A replies to the BSM with Key Delivery Confirmation message. In case a SEK is used for Service Protection, the use of the SEAK or SEK is bound by its start and end-times. During the lifetime of the Service, the SEK can be changed periodically. In case a PEK is used for Service Protection, the PEK is used throughout the total lifetime of the program. If both PEAK and SEAK are used in parallel, then the TEK encrypted with the PEK and the PEK encrypted with the SEK SHALL be present in the STKM. When only the PEAK or PEK is provided, the STKM should only include the TEK encrypted with the PEK.

The message is the same as the Key Request Response message defined above in Section 13.1.2.2.1.2. The root element of the associated XML schema for this message SHALL have the name “KeyDelivery” instead of “KeyRequestResponse”. Status can be set to any value and SHALL be ignored by BSD/A

13.1.2.2.1.4. Key Delivery Confirmation

This message is sent from the BSD/A to the BSM to acknowledge the reception of Key Delivery Message.

Name	Type	Category	Cardinality	Description	Data Type
KeyDeliveryConfirmation	E			Confirmation to the Key Delivery Contains the following attributes: tag version messageID destination source status time	

				Contains the following elements: GlobalServiceID GlobalContentID ScheduleID	
tag	A	M	1	Identifier for the message type	unsignedByte
version	A	O	0..1	BCAST enabler version supported by this message	unsignedInt
messageID	A	M	1	Key Delivery Message ID	string
destination	A	M	1	BSM ID (Note: To be independent of the underlying network protocols, Destination is included in the message.)	string
source	A	M	1	BSD/A ID (Note: To be independent of the underlying network protocols, Source is included in the message.)	string
status	A	M	1	Indication of the reception status of Key Delivery Message. Global Status codes are used as specified in Appendix C.	unsignedByte
time	A	O	0..1	The time when this message is sent. This field contains the 32bits integer part of an NTP time stamp.	unsignedInt
GlobalServiceID	E1	M	1	Identifier of the service to be encrypted	anyURI
GlobalContentID	E1	O	0..1	Identifier of the content that is protected.. This field is mandatory if GlobalContent ID was provided in the key delivery message	anyURI
ScheduleID	E1	O	0..1	Identifier of the schedule that is protected. This field is mandatory if schedule ID was provided in the key delivery message.	anyURI

13.1.2.3 LTKM and Registration Key Material Delivery

This paragraph describes the delivery of the LTKM and/or Registration key material are from the BSM to the BSD/A over interface SP-4, for subsequent broadcast distribution. Note that delivery of LTKM and/or registration key material applies to the DRM Profile only.

13.1.2.3.1 Message Flows

Tags are defined in the following table to identify each message. There are two cases for delivery of LTKM material for broadcast distribution of LTKM's or Registration Key Material for subsequent broadcast distribution.

1. In the first case the BSD/A sends a Key_Request message to the BSM. the BSM then sends a Key_Request_Response message to the BSD/A. The Key Request Response should contain all LTKMs or registration key material that needs to be broadcast in the requested time period.

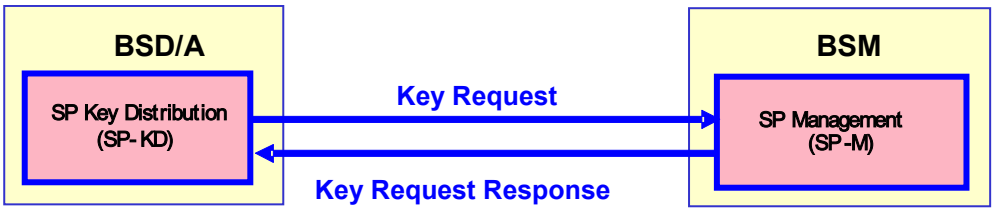


Figure 20 – Message Flow Between BSD/A and BSM for Delivery of LTKM and Registration Key Material

2. In the second case the BSM sends a Key_Delivery message to the BSD/A. The BSD/A then sends a Key_Delivery_Confirmation message to the BSM.

.

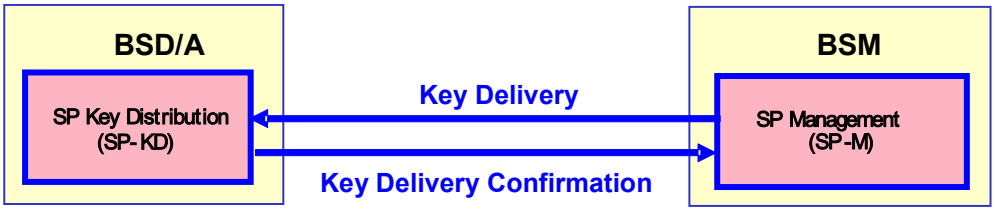


Figure 21 – Alternative Message Flow Between BSD/A and BSM for Delivery of LTKM and Registration Key Material

Tag	Message Type	Key
5	Key_Request	Long-Term Key Message
6	Key_Request_Response	Long-Term Key Message
7	Key_Delivery (same as Key_Request_Response)	Long-Term Key Message
8	Key_Delivery_Confirmation	Long-Term Key Message
9	Key_Request	Key Material for Registration
10	Key_Request_Response	Key Material for Registration
11	Key_Delivery (same as Key Request Response)	Key Material for Registration
12	Key_Delivery_Confirmation	Key Material for Registration

13.1.2.3.1.1. LTKM or Registration Key Material Request

This message is sent from the BSD/A to the BSM for the request for the delivery of LTKM or Registration Key material for broadcast distribution of LTKM's or Registration Key Material.

Name	Type	Category	Cardinality	Description	Data Type
LTKMorReg Request				Key Request Message Contains the following attributes: tag	

				version messageID destination source time Contains the following elements: GlobalServiceID GlobalContentID ScheduleID DistributionStart DistributionEnd	
tag	A	M	1	Identifier for the message type	unsignedByte
version	A	O	0..1	BCAST enabler version supported by this message	unsignedInt
messageID	A	M	1	This message ID	string
destination	A	M	1	BSM ID (Note: To be independent of the underlying network protocols, Destination is included in the message.)	string
sourceID	A	M	1	BSD/A ID (Note: To be independent of underlying network protocols, Source is included in the message.)	string
time	A	O	0..1	The time when this message is sent. This field contains the 32bits integer part of an NTP time stamp.	unsignedInt
GlobalServiceID	E1	M	1	Identifier of the target service	anyURI
GlobalContentID	E1	O	0..1	Identifier of the content that is protected. Only GlobalContent ID which is related to the GlobalService ID is allowed.	anyURI
ScheduleID	E1	O	0..1	Identifier of the schedule that is protected. Only Schedule ID which is related to the GlobalService ID is allowed.	anyURI
Distribution Start	E1	M	1	This field is mandatory if LTKM or Registration Key material is provided. Provides the start time of the period in which the LTKM or Registration Key material should be distributed by the BSD/A. This field expressed as the first 32bits integer part of NTP timestamps.	unsignedInt
Distribution End	E1	M	1	This field is mandatory if LTKM or Registration Key material is provided. Provides the end time of the period in which the LTKM or Registration Key material should be distributed by the BSD/A.	unsignedInt

				This field expressed as the first 32bits integer part of NTP timestamps.	

13.1.2.3.1.2. LTKM or Registration Key Material Request Response

After the reception of the Key Request Message, the BSM sends this message to the BSD/A for the delivery of LTKM or Registration Key material for broadcast distribution of LTKM's or Registration Key Material.

Name	Type	Category	Cardinality	Description	Data Type
LTKMorRegRequestResponse				Key Request Response Contains the following attributes: tag version messageID destination source status time Contains the following elements: GlobalServiceID GlobalContentID ScheduleID Data DistributionStart DistributionEnd	
tag	A	M	1	Identifier for the message type	unsignedByte
version	A	O	0..1	BCAST enabler version supported by this message	unsignedInt
messageID	A	M	1	Key Request Message ID	string
destination	A	M	1	BSD/A ID (Note: To be independent of the underlying network protocols, Destination is included in the message.)	string
source	A	M	1	BSM ID (Note: To be independent of underlying network protocols, Source is included in the message.)	string
status	A	M	1	Indication of the reception status of Key Request Message. Global Status codes are used as specified in Appendix C.	unsignedByte
time	A	O	0..1	The time when this message is sent. This field contains the 32bits integer part of an NTP time stamp.	unsignedInt

GlobalServiceID	E1	M	1	Identifier of the target service	anyURI
GlobalContentID	E1	O	0..1	Identifier of the content that is protected. This field is mandatory if GlobalContent ID was provided in the Key request message.	anyURI
ScheduleID	E1	O	0..1	Identifier of the schedule that is protected. This field is mandatory if schedule ID was provided in the Key request message.	anyURI
Data	E1	O	0..N	LTKM material for broadcast distribution of LTKM's or Registration Key Material for subsequent broadcast distribution. For LTKM material a single data element carries a single BCRO. For Registration Key material a single data element carries a single message of either one of the messages below: - device_registration_response(), - update_ri_certificate_msg(), - update_drmtime_msg(), - update_contact_number_msg(), - re_register_msg(), - token_delivery_response(), - domain_registration_response(), - domain_update_response(), - join_domain_msg(), - leave_domain_msg(), as specified in section 7 of [XBS DRM extensions-v1.0]	hexBinary
Distribution Start	E1	O	0..1	This field is mandatory if LTKM or Registration Key material is provided. Provides the start time of the period in which the LTKM or Registration Key material should be distributed by the BSD/A. This field expressed as the first 32bits integer part of NTP timestamps.	unsignedInt
Distribution End	E1	O	0..1	This field is mandatory if LTKM or Registration Key material is provided. Provides the end time of the period in which the LTKM or Registration Key material should be distributed by the BSD/A. This field expressed as the first 32bits integer part of NTP timestamps.	unsignedInt

13.1.2.3.1.3. LTKM or Registration Key Material Delivery

This message is sent from the BSM to the BSD/A for the delivery of LTKM material for broadcast distribution of LTKM's or Registration Key Material without a request from the BSD/A. If the BSD/A receives this message, then the BSD/A replies to the BSM with Key_Delivery_Confirmation message.

This message is the same as the LTKM or Registration Key Material Request Response message defined above in Section 13.1.2.3.1.2. The root element of the associated XML schema for this message SHALL have the name

"LTKMorRegDelivery" instead of "LTKMorRegRequestResponse". Status can be set to any value and SHALL be ignored by BSD/A.

13.1.2.3.1.4. LTKM or Registration Key Material Key Delivery Confirmation

This message is sent from the BSD/A to the BSM to acknowledge the receipt of the LTKM or Registration Key Material Delivery message.

This message is the same as the Key Delivery Confirmation message defined above in Section 13.1.2.2.1.4.

13.1.2.4 STKM Delivery

This paragraph describes the delivery of STKM's from the BSM to the BSD/A or from the BSD/A to the BSM over interface SP-4. The STKM delivered from the BSM to the BSD/A can be sent to Terminal using broadcast channel. The STKM delivered from the BSD/A to the BSM can be sent to Terminal using interaction channel.

13.1.2.4.1 Message Flows from BSM to BSD/A

Tags are defined in the following table to identify a type of each message. There are two cases for delivery of STKM to the BSD/A when STKM generation is done by BSM.

1. The first case consists of the STKM Request message by the BSDA and the Response with the Delivery of the STKM data by the BSM, i.e. BSD/A initiated STKM request.

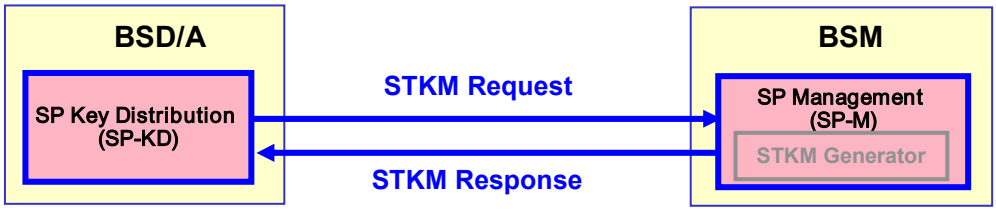


Figure 22 – Message Flow Between BSD/A and BSM for Delivery STKMs

2. The second case is BSM initiated. In this case the BSM requests a set of TEK's from the BSDA which it will use during a specific time period to encrypt the service or program. In response, the BSDA delivers the TEKs and the associated security protocol parameters. With this data, the BSM can send an STKM delivery message to the BSD/A. The BSD/A confirms this delivery message.

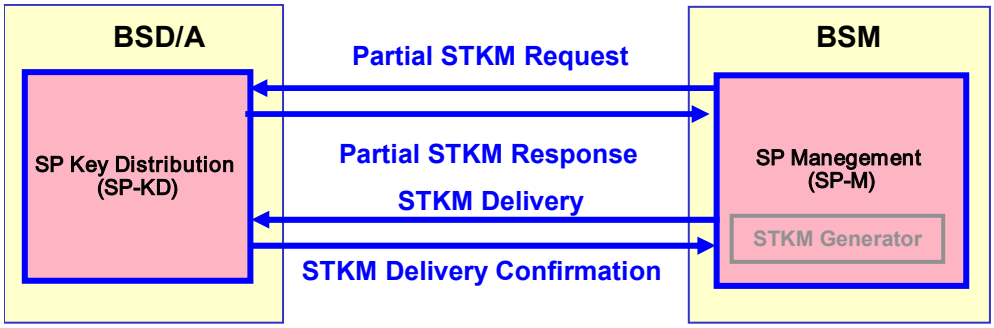


Figure 23 – Alternative Message Flow Between BSD/A and BSM for Delivery STKMs

Services can be shared between multiple BSM's, where each BSM uses it's own SEK and it's own SEK validity time. Furthermore the validity of the SEK is coupled to the TEK ID for anti-replaySo when the TEK ID is wrapped around, the TEK has to be encrypted with the new SEK. . E.g. in the case of protection at the RTSP layer with the Smartcard Profile the amount of TEK ID's are limited to 16 bits. Therefore wrap around time and wrap around indicators are included as attributes

of each TEK. These TEK wrap around times cannot be chosen randomly, but should be coordinated with the subscription periods of the BSM, e.g. at the end of each month or week

Tag	Message Type	Key
23	STKM_Request	TEK
24	STKM_Response	STKM
25	Partial_STKM_Request	Partial STKM including TEK
26	Partial_STKM_Response	Partial STKM including TEK
27	STKM_Delivery (same as STKM_Response)	STKM
28	STKM_Delivery_Confirmation	STKM

13.1.2.4.1.1. STKM Request

This message is sent from the BSD/A to the BSM for the acquisition of the Short Term Key Messages.

Name	Type	Category	Cardinality	Description	Data Type
STKMRequest				Request message for STKMs Contains the following attributes tag version messageID destination source time Contains the following elements: GlobalServiceID GlobalContentID ScheduleID SPPType KeyMaterial	
tag	A	M	1	Identifier for the message type	unsignedByte
version	A	O	0..1	BCAST enabler version supported by this message	unsignedInt
messageID	A	M	1	This message ID	string
destination	A	M	1	BSM ID (Note: To be independent of the underlying network protocols, Destination is included in the	string

				message.)	
source	A	M	1	BSD/A ID (Note: To be independent of the underlying network protocols, Source is included in the message.)	string
time	A	O	0..1	The time when this message is sent. This field contains the 32bits integer part of an NTP time stamp.	unsignedInt
GlobalServiceID	E1	M	1	Identifier of the service to be encrypted	anyURI
GlobalContentID	E1	O	0..1	Identifier of the content that is protected. Used if service protection is program based. Only GlobalContent ID which is related to the GlobalService ID is allowed.	anyURI
ScheduleID	E1	O	0..1	Identifier of the schedule that is protected. Only Schedule ID which is related to the GlobalService ID is allowed.	anyURI
SPPType	E1	M	1	This specifies the type of the Service protection profile used by the BSM. 0 if service protection profile == DRM Profile 1 if service protection profile == Smartcard Profile 2-127 reserved for future use 128-255 reserved for proprietary use	unsignedByte
KeyMaterial	E1	M	1..N	The key material used to encrypt the service or program Contains the following attributes: masterKey masterSalt type traffic_authentication_flag validityTime cryptoPeriod wrapAroundTime wraAaroundInidcator Contains the following element TrafficProtectionProtocolParameters NextTrafficKey	
masterKey	A	M	1	The master key used for traffic and content encryption	hexBinary
masterSalt	A	M	1	The master Salt used for traffic and content encryption	hexBinary
type	A	M	1	The traffic protection protocol used. This attribute can have the following values, as specified in the STKM in Section 7.2. Allowed values are: 0 if traffic_protection_protocol ==	unsignedByte

				TKM_ALGO_IPSEC 1 if traffic_protection_protocol == TKM_ALGO_SRTP 2 if traffic_protection_protocol == TKM_ALGO_AUENCRYP 3 if traffic_protection_protocol == TKM_ALGO_DCF 4-127 reserved for future use 128-255 reserved for proprietary use	
traffic_authentication_flag	A	M	1	True if the traffic_authentication_flag in the STKM should be set to TKM_FLAG_TRUE (authentication will be used). False otherwise.	boolean
validityTime	A	M	1	NTP time when the traffic encryption key is used to encrypt the service or program. This value indicates to the BSM which Service Key it needs to use to encrypt the traffic encryption key. This field expressed as the first 32bits integer part of NTP timestamps.	unsignedInt
cryptoPeriod	A	M	1	The crypto period used for service protection. The Validity-Time of the next TEK SHOULD be 1 crypto period later than the Validity Time of this TEK.	unsignedInt
wrapAroundTime	A	M	1	This indicates the wrap around time of the TEK sequence to which this TEK belongs. After the TEK wrap around time the TEK key indicator, master key index or security parameter index is reset. This field is used to indicate that the first TEK after the Wrap around time SHOULD be encrypted with a new SEK. This field expressed as the first 32bits integer part of NTP timestamps.	unsignedInt
wrapAroundIindicator	A	M	1	This field is set to “true” for the first TEK after the Wrap around time has passed. It is used to indicate that this and subsequent TEKs SHOULD be encrypted with a new SEK.	boolean
TrafficProtectionProtocolParameters	E2	M	1	This specifies the data related to the traffic protection protocol for the STKM, as defined in the STKM in Section 5.5. Contains the following elements: SPI MKI KeyIndicator KeyIdentifier	

				Note the following: • SPI is mandatory in case 'type' of 'KeyMaterial' is 0 • MKI is mandatory in case 'type' of 'KeyMaterial' is 1 • KeyIndicator is mandatory in case 'type' of 'KeyMaterial' is 2 • KeyIdentifier is mandatory in case 'type' of 'KeyMaterial' is 3 • SynchronisationSource is mandatory in case 'type' of 'KeyMaterial' is 1 and is optional in other cases This constraint is expressed by using the <choice> element in XML Schema	
SPI	E3	O	0..1	Security Parameter Index. Contains the following attributes: spi nextSpi	
spi	A	M	1	security_parameter_index	unsignedInt
nextSpi	A	M	1	next_security_parameter_index	unsignedInt
MKI	E3	O	0..1	Master Key Index Contains the following attributes: mkilength mki mediaFlows	
mkiLength	A	M	1	master_key_index_length	unsignedInt
mki	A	M	1	master_key_index	unsignedInt
mediaFlows	A	M	1	number_of_media_flows	int
KeyIndicator	E3	O	0..1	Key Indicator Contains the following attributes: keyIndicatorLength keyIndicator	
keyIndicator Length	A	M	1	key_indicator_length This field is mandatory in case 'Type' of 'Key material' is 2	unsignedInt
keyIndicator	A	M	1	key_indicator	unsignedInt

KeyIdentifier	E3	O	0..1	Key Identifier Contains the following attributes: keyIdentifierLength keyIdentifier	
keyIdentifierLength	A	M	1	key_identifier_length	unsignedInt
keyIdentifier	A	M	1	The key identifier	hexBinary
NextTrafficKey	E2	O	0..1	Flag for indication of the next traffic key in an STKM. If true, the STKM SHALL also include the next encrypted traffic key. Note that this field is only relevant to DRM profile STKMs. Next traffic key has the following attributes, only if Next traffic key is set to "true": masterKey masterSalt	boolean
masterKey	A	O	0..1	The next master key used to encrypt the service or program. This field is mandatory if next traffic key is "true"	hexBinary
masterSalt	A	O	0..1	The next Master Salt used to encrypt the service or program. This field is mandatory if next traffic key is "true"	hexBinary

13.1.2.4.1.2. STKM Response

After the reception of the STKM Request message, the BSM sends this message to the BSD/A for the delivery of STKM.

Name	Type	Category	Cardinality	Description	Data Type
STKMResponse	E			This message is the response to the STKM Request message. Contains the following attributes tag version messageID destination source status time Contains the following elements: GlobalServiceID	

				GlobalContentID ScheduleID STKM	
tag	A	M	1	Identifier for the message type	unsignedByte
version	A	O	0..1	BCAST enabler version supported by this message	unsignedInt
messageID	A	M	1	Key Request Message ID	string
destination	A	M	1	BSD/A ID (Note: To be independent of the underlying network protocols, Destination is included in the message.)	string
source	A	M	1	BSM ID (Note: To be independent of the underlying network protocols, Source is included in the message.)	string
status	A	M	1	Indication of the reception status of STKM Request Message. Global Status codes are used as specified in Appendix C.	unsignedByte
time	A	O	0..1	The time when this message is sent. This field contains the 32bits integer part of an NTP time stamp.	unsignedInt
GlobalServiceID	E1	M	1	Identifier of the service to be encrypted	anyURI
GlobalContentID	E1	O	0..1	Identifier of the content that is protected. This field is mandatory if GlobalContent ID was provided in the STKM request message.	anyURI
ScheduleID	E1	O	0..1	Identifier of the schedule that is protected. This field is mandatory if schedule ID was provided in the STKM request message.	anyURI
STKM	E1	M	1..N	The STKM STKM has the following attribute: - validityTime	hexBinary
validityTime	A	M	1	This validityTime attribute is used to associate the stkm with the TEK. The validityTime of the STKM SHALL be the same as the validityTime of the TEK to which this stkm is associated. This field is expressed as the first 32bits integer part of NTP timestamps.	unsignedInt

13.1.2.4.1.3. Partial STKM Request Message

The Partial STKM Request message is used by the BSM to request a set of TEK's from the BSD/A to deliver a set of TEK's to be used for the encryption of the service or program. The set of TEK's to be delivered are indicated by a start time and an end time in the request message.

Name	Type	Category	Cardinality	Description	Data Type
PartialSTK				Partial STKM Request Message	

MRequest				Contains the following attributes tag version messageID destination source time Contains the following elements: GlobalServiceID GlobalContentID ScheduleID TEKStartTime TEKEndTime	
tag	A	M	1	Identifier for the message type	unsignedByte
version	A	O	0..1	BCAST enabler version supported by this message	unsignedInt
messageID	A	M	1	This message ID	string
destination	A	M	1	BSD/A ID (Note: To be independent of the underlying network protocols, Destination is included in the message.)	string
source	A	M	1	BSM ID (Note: To be independent of the underlying network protocols, Source is included in the message.)	string
time	A	O	0..1	The time when this message is sent. This field contains the 32bits integer part of an NTP time stamp.	unsignedInt
GlobalServiceID	E1	M	1	Identifier of the service to be encrypted	anyURI
GlobalContentID	E1	O	0..1	Identifier of the content that is protected. Used if service protection is program based. Only GlobalContent ID which is related to the GlobalService ID is allowed.	anyURI
ScheduleID	E1	O	0..1	Identifier of the schedule that is protected. Only Schedule ID which are related to the GlobalService ID is allowed.	anyURI
TEKStartTime	E1	M	1	This is the start time of the TEKs that are used for the encryption of the service or program. This field expressed as the first 32bits integer part of NTP timestamps.	unsignedInt
TEKEndTime	E1	M	1	This is the end time of the TEK that are used for the encryption of the service or program.	unsignedInt

				This field expressed as the first 32bits integer part of NTP timestamps..	

13.1.2.4.1.4. Partial STKM Response Message

The Partial STKM Response message is used by the BSD/A to deliver the TEK's and the associated traffic protection protocol parameters to the BSM.

Name	Type	Category	Cardinality	Description	Data Type
PartialSTKMResponse	E			Partial STKM Response Message Contains the following attributes tag version messageID destination source status time Contains the following elements: GlobalServiceID GlobalContentID ScheduleID SPPType KeyMaterial	
tag	A	M	1	Identifier for the message type	unsignedByte
version	A	O	0..1	BCAST enabler version supported by this message	unsignedInt
messageID	A	M	1	This message ID	string
destination	A	M	1	BSM ID (Note: To be independent of the underlying network protocols, Destination is included in the message.)	string
source	A	M	1	BSD/A ID (Note: To be independent of the underlying network protocols, Source is included in the message.)	string
status	A	M	1	Indication of the reception status of TEK Request Message. Global Status codes are used as specified in Appendix C.	unsignedByte
time	A	O	0..1	The time when this message is sent. This field contains the 32bits integer part of an NTP time stamp.	unsignedInt
GlobalServiceID	E1	M	1	Identifier of the service to be encrypted	anyURI

GlobalContentID	E1	O	0..1	Identifier of the content that is protected. Used if service protection is program based. This field is mandatory if Global Content ID was provided in the TEK request message.	anyURI
ScheduleID	E1	O	0..1	Identifier of the schedule that is protected. This field is mandatory if schedule ID was provided in the TEK request message.	anyURI
SPPType	E1	M	1	This specifies the type of the Service protection profile used by the BSM. 0 if service protection profile == DRM Profile 1 if service protection profile == Smartcard Profile 2-127 reserved for future use 128-255 reserved for proprietary use	unsignedByte
KeyMaterial	E1	M	1..N	The key material used to encrypt the service or program KeyMaterial has the following attributes: - masterKey - masterSalt - type - traffic_authentication_flag -validityTime - cryptoPeriod - wrapAroundTime - wrapAroundInidcator KeyMaterial contains the following elements - TrafficProtectionProtocolParameters - NextTrafficKey	
masterKey	A	M	1	The master key used for traffic and content encryption	hexBinary
masterSalt	A	M	1	The master Salt used for traffic and content encryption	hexBinary
type	A	M	1	The traffic protection protocol used. This attribute can have the following values, as specified in the STKM in Section 7.2: - 0 if traffic_protection_protocol == TKM_ALGO_IPSEC - 1 if traffic_protection_protocol == TKM_ALGO_SRTP - 2 if traffic_protection_protocol == TKM_ALGO_AUENCRYP - 3 if traffic_protection_protocol == TKM_ALGO_DCF	unsignedByte

				4-127 reserved for future use 128-255 reserved for proprietary use	
traffic_authentication_flag	A	M	1	True if the traffic_authentication_flag in the STKM should be set to TKM_FLAG_TRUE (authentication will be used). False otherwise.	boolean
validityTime	A	M	1	NTP time when the traffic encryption key is used to encrypt the service or program. This value indicates to the BSM which Service Key it needs to use to encrypt the traffic encryption key. This field expressed as the first 32bits integer part of NTP timestamps. The NTP value SHALL be bound by the start and end-times as indicated in the TEK request message.	unsignedInt
cryptoPeriod	A	M	1	The crypto period used for service protection. The Validity-Time of the next TEK SHOULD be 1 crypto period later than the Validity Time of this TEK.	unsignedInt
wrapAroundTime	A	M	1	This indicates the wrap around time of the TEK sequence to which this TEK belongs. After the TEK wrap around time the TEK key indicator, master key index or security parameter index is reset. This field is used to indicate that the first TEK after the Wrap around time SHOULD be encrypted with a new SEK. This field expressed as the first 32bits integer part of NTP timestamps.	unsignedInt
wrapAroundIndicator	A	M	1	This field is set to "true" for the first TEK after the Wrap around time has passed. It is used to indicate that this and subsequent TEKs SHOULD be encrypted with a new SEK.	boolean
TrafficProtectionProtocolParameters	E2	M	1	This specifies the data related to the traffic protection protocol for the STKM, as defined in the STKM in Section 5.5. Contains the following elements: SPI MKI KeyIndicator KeyIdentifier Note the following: • SPI is mandatory in case 'type' of 'KeyMaterial' is 0 • MKI is mandatory in case 'type' of	

				'KeyMaterial' is 1 • KeyIndicator is mandatory in case 'type' of 'KeyMaterial' is 2 • KeyIdentifier is mandatory in case 'type' of 'KeyMaterial' is 3	
SPI	E3	O	0..1	Security Parameter Index. Contains the following attributes: spi nextSpi	
spi	A	M	1	security_parameter_index	unsignedInt
nextSpi	A	M	1	next_security_parameter_index	unsignedInt
MKI	E3	O	0..1	Master Key Index Contains the following attributes: mkilength mki mediaFlows	
mkiLength	A	M	1	master_key_index_length	
mki	A	M	1	master_key_index	unsignedInt
mediaFlows	A	M	1	number_of_media_flows	int
KeyIndicator	E3	O	0..1	Key Indicator Contains the following attributes: keyIndicatorLength keyIndicator	
keyIndicator Length	A	M	1	key_indicator_length	unsignedInt
keyIndicator	A	M	1	key_indicator	unsignedInt
KeyIdentifier	E3	O	0..1	Key Identifier Contains the following attributes: keyIdentifierLength keyIdentifier	
keyIdentifier Length	A	M	1	key_identifier_length	unsignedInt
keyIdentifier	A	M	1	key_identifier	hexBinary

NextTraffic Key	E2	O	0..1	Flag for indication of the next traffic key in an STKM. If "true", the STKM SHALL also include the next encrypted traffic key. Note that this field is only relevant to DRM profile STKMs. Next traffic key has the following attribute, only if Next traffic key is set to "true": - masterKey - masterSalt	boolean
nextTrafficKey	A	O	0..1	The next traffic encryption key used to encrypt the service or program. This field is mandatory if next traffic key is "TRUE"	hexBinary
masterKey	A	O	0..1	The next master key used to encrypt the service or program. This field is mandatory if NextTrafficKey is "true"	hexBinary
masterSalt	A	O	0..1	The next Master Salt used to encrypt the service or program. This field is mandatory if NextTrafficKey is "true"	hexBinary

13.1.2.4.1.5. STKM Delivery

This message is used by the BSM to deliver the STKM to the BSDA.

This message is the same as the STKM Response message defined above in Section 13.1.2.4.1.2. The root element of the associated XML schema for this message SHALL have the name "STKMDelivery" instead of "STKMResponse". Status can be set to any value and SHALL be ignored by BSD/A.

13.1.2.4.1.6. STKM Delivery Confirmation

This message is used by the BSD/A to confirm the reception of the STKM delivery message.

This message is the same as the Key Delivery Confirmation message defined above in Section 13.1.2.2.1.4. The root element of the associated XML schema for this message SHALL have the name "STKMDeliveryConfirmation" instead of "KeyDeliveryConfirmation".

13.1.2.4.2 Message Flows from BSD/A to BSM

Tags are defined in the following table to identify a type of each message. There are two cases for delivery of STKM to the BSD/A when STKM generation is done by the BSD/A.

1. The first case consists of the STKM Request message by the BSM and the Response with the Delivery of the STKM data by the BSD/A, i.e. BSM initiated STKM request.

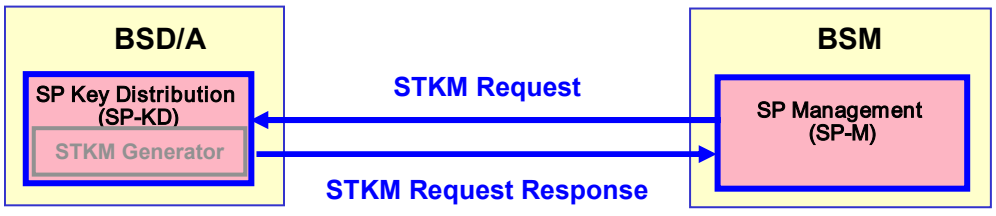


Figure 24 – Message Flow Between BSM and BSD/A for Delivery STKMs

2. The second case is BSD/A initiated. In this case the BSD/A sends an STKM delivery message to the BSM. The BSM confirms this delivery message.

Figure 25 – Alternative Message Flow Between BSM and BSD/A for Delivery STKMs

Tag	Message Type	Key
29	STKM_Request	STKM
30	STKM_Response	STKM
31	STKM_Delivery (same as STKM_Response)	STKM
32	STKM_Delivery_Confirmation	STKM

13.1.2.4.2.1. STKM Request

This message is send from the BSM to the BSD/A for the acquisition of the STKM. This message is the same as the Key Request message defined in Section 13.1.2.3.1.1. The root element of the associated XML schema for this message SHALL have the name "STKMRequest" instead of "KeyRequest".

13.1.2.4.2.2. STKM Request Response

This message is used by the BSD/A to delivery the STKM to the BSM. This message is the same as the STKM Response message defined above in Section 13.1.2.4.1.2.

13.1.2.4.2.3. STKM Delivery

This message is used by the BSD/A to deliver the STKM to the BSM.

This message is the same as the STKM Response message defined above in Section 13.1.2.4.1.2. The root element of the associated XML schema for this message SHALL have the name "STKMDelivery" instead of "STKMResponse". Status can be set to any value and SHALL be ignored by BSD/A.

13.1.2.4.2.4. STKM Delivery Confirmation

This message is used by the BSM to confirm the reception of the STKM delivery message.

This message is the same as the Key Delivery Confirmation message defined above in Section 13.1.2.2.1.4. The root element of the associated XML schema for this message SHALL have the name "STKMDeliveryConfirmation" instead of "KeyDeliveryConfirmation".

13.2 Interface CP-4

Interface CP-4 has three functions:

1) To deliver Service and Program key material from SP-M in the BSM to the SP-KD in the BSD/A for the service protection.

2) To deliver the LTKM and/or Registration key material from SP-M in the BSM to SP-KD in the BSD/A, for subsequent broadcast distribution of these data..

3) To deliver of the STKM's from the BSM to the BSD/A.

A BSM that support service and/or content protection SHALL support Interface CP-4. A BSD/A that support service and/or content protection SHALL support Interface CP-4.

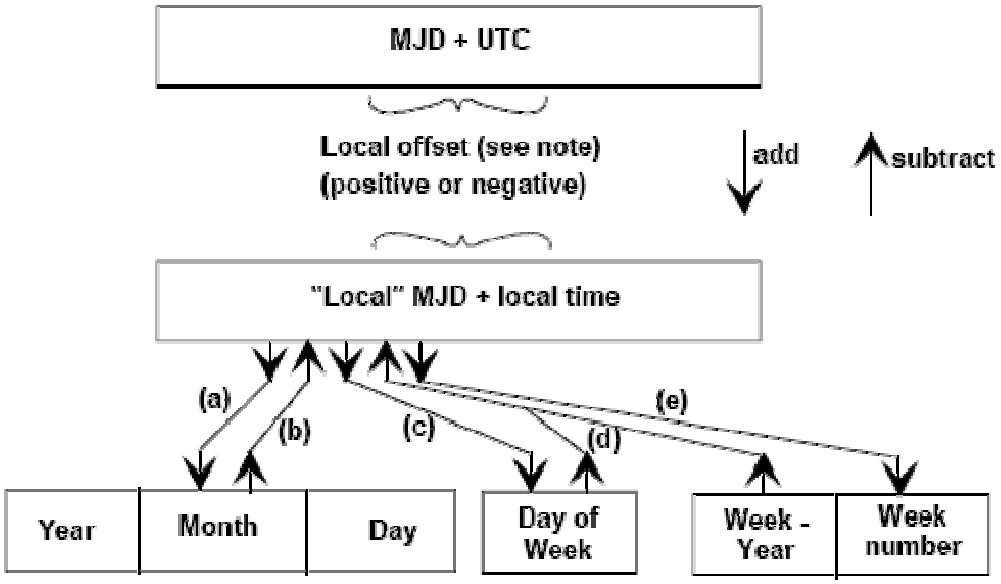
The message flows for interface CP-4 are the same as the message flows for SP-4. Therefore the same protocol stack and messages as described for SP-4 SHALL be used for CP-4. The technical difference between SP-4 and CP-4 is the value of the "Protection After Reception Flag" element in the "Service and Program Key material delivery" messages. In case content protection needs to be applied, the BSM sets the "Protection After Reception Flag" to value 0x00, 0x01 or 0x02 as defined in Section 5.5.1. Setting the "Protection After Reception Flag" to 0x03 is applied in case of service protection. This is valid for both streams and files.

Files can be protected using DCF. DCF files can be offered by the BSA or Content Creator. In this case the file is transported as any other file to the terminal. In case service ort content protection is required for a file, the file has to be protected using DCF by the BSD/A. Therefore, a BSD/A that supports service and/or content protection SHALL support DCF protection of files. Note, that also DCF files, which were protection by the BSA or Content Creator can be additionally encapsulated in another DCF for additional service or content protection, i.e. DCF inside a DCF. The usage rights of the 'outer' DCF file is dictated by the "Protection After Reception Flag". The usage rights of the 'inner' DCF is dictated by the Rights Objects that the device has for the file, that were provided by the BSA or Content Creator.

14. Conversion between Time and Date Conventions

(Note: this text is modified from [ETSI EN 300 468 V1.6.1] for BCAST purposes. In particular, the version in this text maintains byte alignment everywhere.)

The types of conversion which may be required are summarized in Figure 26.



NOTE: Offsets are positive for Longitudes East of Greenwich and negative for Longitudes West of Greenwich.

Figure 26 – Conversion routes between Modified Julian Date (MJD) and Co-ordinated Universal Time (UTC)

The conversion between MJD + UTC and the "local" MJD + local time is simply a matter of adding or subtracting the local offset. This process may, of course, involve a "carry" or "borrow" from the UTC affecting the MJD. The other five conversion routes shown on the diagram are detailed in the formulas below:

Symbols used:

- D: Day of month from 1 to 31
- Int: Integer part, ignoring remainder
- K, L ,M', W, Y': Intermediate variables
- M: Month from January (= 1) to December (= 12)
- MJD: Modified Julian Date
- MN: Week number according to ISO 2015 [21]
- mod 7: Remainder (0-6) after dividing integer by 7
- UTC: Universal Time, Co-ordinated
- WD: Day of week from Monday (= 1) to Sunday (= 7)
- WY: "Week number" Year from 1900
- X: Multiplication

- Y: Year from 1900 (e.g. for 2003, Y = 103)

To find Y, M, D from MJD

- Y' = int [(MJD - 15 078,2) / 365,25]
- M' = int { [MJD - 14 956,1 - int (Y' × 365,25)] / 30,6001 }
- D = MJD - 14 956 - int (Y' × 365,25) - int (M' × 30,6001)
- If M' = 14 or M' = 15, then K = 1; else K = 0
- Y = Y' + K
- M = M' - 1 - K × 12

To find MJD from Y, M, D

- If M = 1 or M = 2, then L = 1; else L = 0
- MJD = 14 956 + D + int [(Y - L) × 365,25] + int [(M + 1 + L × 12) × 30,6001]

To find WD from MJD

- WD = [(MJD + 2) mod 7] + 1

To find MJD from WY, WN, WD

- MJD = 15 012 + WD + 7 × { WN + int [(WY × 1 461 / 28) + 0,41] }

To find WY, WN from MJD

- W = int [(MJD / 7) - 2 144,64]
- WY = int [(W × 28 / 1 461) - 0,0079]
- WN = W - int [(WY × 1 461 / 28) + 0,41]
- EXAMPLE: MJD = 45 218 W = 4 315
- Y = (19)82 WY = (19)82
- M = 9 (September) N = 36
- D = 6 WD = 1 (Monday)

NOTE: These formulas are applicable between the inclusive dates 1900 March 1 to 2100 February 28.

14.1 Local Time Offset

This 16-bit field contains the current offset time from UTC in the range between –12 hours and +13 hours at the area which is indicated by the combination of country_code and country_region_id in advance. These 16 bits are coded as 4 digits in 4-bit BCD in the order hour tens, hour, minute tens, and minutes.

The positive or negative offset from the UTC is indicated with the 1 bit local_time_offset_polarity. If this bit is set to "0" the polarity is positive and the local time is advanced to UTC. (Usually east direction from Greenwich). If this bit is set to "1" the polarity is negative and the local time is behind UTC. Please note that the local_time_offset_polarity is represented by the first bit of the first nibble representing the hour tens field. The first nibble of the local_time_offset is therefore encoded as follows:

Table 59: Local Time Offset Coding

local_time_offset_polarity	offset hour tens	first nibble
0 (i.e. "+")	0	0000
0 (i.e. "+")	1	0001
1 (i.e. "-")	0	1000
1 (i.e. "-")	1	1001

15. Interfacing to underlying BDSes

15.1 BCMCS

Interfacing to underlying BCMCS BDS SHALL be as specified in the BCMCS adaptation specifications [BCAST10-BCMCS-Adaptation].

15.2 MBMS

Interfacing to underlying MBMS BDS SHALL be as specified in the MBMS adaptation specifications [BCAST10-MBMS-Adaptation].

15.3 IPDC over DVB-H

Interfacing to underlying DVB BDS SHALL be as specified in the DVB adaptation specifications [BCAST10-DVBH-IPDC-Adaptation].

16. Broadcast Roaming – Roaming at Service Provider Level (Informative)

Section 5.7 "Broadcast Roaming" in [BCAST10-Services] describes roaming across different Service Providers. BCAST provides messages that allow the Terminal to acquire relevant roaming rules, allowing it to discover available services based on service provider roaming agreements.

This chapter provides additional information relating to Service Protection and Content Protection aspects that must also be taken into account when considering broadcast roaming (across different service providers) when content / services are also protected.

16.1 Broadcast Roaming –DRM Profile

A terminal can have access to content / services protected using the DRM Profile provided:

1. BCAST service provisioning and/or roaming message exchange with the visited Service Provider is successful (see [BCAST10-Services]).
2. Registration with the Rights Issuer has been completed (see Section 5.3). This means the Rights Issuer has to authorise delivery of GROs the terminal, i.e. the Terminal and Rights Issuer must mutually accept each other's certificates.

Note that Step 1 can require the appropriate broadcast/ service roaming agreements to be in place between home and visited Service Provider.

16.2 Broadcast Roaming – Smartcard Profile

A terminal can have access to content / services protected using the Smartcard Profile provided:

1. BCAST service provisioning and/ or roaming message exchange with the visited Service Provider is successful (see [BCAST10-Services]).
2. Subscriber Key Establishment has been completed (see Section 6.5), allowing SMK/SRK to be derived. For the (U)SIM Smartcard Profile this means the visited network BSM must be able to obtain the SMK/SRK from the home network BSM with which GBA bootstrapping is accomplished. If no agreement exists between home and visited BSM, LTKMs will not be able to be delivered to the Smartcard / Terminal, preventing access to protected / content available through the visited BSM. In the case of the (R-)UIM/CSIM Smartcard Profile, the Visited BSM relies on the Home BSM to perform authentication of the subscriber. Upon successful authentication, the Home BSM provides the SMK (i.e. the TK) to the Visited BSM. Subsequently, LTKMs can be delivered from the Visited BSM to the BCAST Terminal, assuming the existence of roaming agreement between the Visited and Home Service Providers.

Note that Step 1 can require the appropriate broadcast/ service roaming agreements to be in place between home and visited Service Provider.

Appendix A. Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
n/a	n/a	No prior version –or- No previous version within OMA
OMA-xxyyz-V1_0-20021001-A	01 Oct 2002	Initial document to address the basic starting point Ref TP Doc# OMA-TP-2002-1234-xxyyzForApproval
OMA-xxyyz-V1_1-20030405-A	05 Apr 2003	description of changed Ref TP Doc# OMA-TP-2003-0321-xxyyzV1_1forApproval

A.2 Draft/Candidate Version 1.0 History

Document Identifier	Date	Sections	Description
Draft Versions OMA-TS_BCAST_SvcCntProtection-V1_0	30 Dec 2004	n/a	Initial Draft
	28 Feb 2005	Previous 5.1.1; 5.4.1; Previous Appendix C.4	Inserted agreed CRs: OMA-BCAST-2004-0170R02; OMA-BCAST-2004-0172R03; OMA-BCAST-2005-0004R02
	11 Mar 2005	9.1; 9.2; 6.4; 5.4.3; Previous 5.1.2.3.2	* Inserted agreed CRs: OMA-BCAST-2005-0102R01; OMA-BCAST-2005-0103R01; OMA-BCAST-2005-0104R02; OMA-BCAST-2005-0105R02; OMA-BCAST-2005-0107R01 * Added descriptive headers to changed sections * Updated document number based on comments from Process Chair (Dwight Smith)
	16 Mar 2005	Previous 5.1.1	Replaced previous text with a reference to relevant sections in the BCAST AD
	21 Apr 2005	2.1; 5.4.2; 9.1; 9.2; Previous 5.1.2.3.1; Previous 5.1.2.4; 5.3; 16	* Inserted agreed CRs: OMA-BCAST-2005-0169R01; OMA-BCAST-2005-0172R01; OMA-BCAST-2005-0176 * Added references to the OMA DRM v2.0 Extensions for Broadcast Support document * Added text in Normative References section
	03 Jun 2005	5.4.3; 5.5; 9.1, 9.2	* Inserted agreed CRs: OMA-BCAST-2005-0216R03; OMA-BCAST-2005-0217R01; OMA-BCAST-2005-0219; OMA-BCAST-2005-0220R02 * Change MKI field size, in 9.2, from 32 bits to 72 bits as per BCAST-169R01 * TS document now includes hyper-links references in the history table.
	23 Jun 2005	4.1; 4.4; Previous 5.1.2.1; Previous 5.1.2.1.1; Previous 5.1.2.1.2; 5.4.1; 9.2; Previous 5.1.2.4; 2.1	* Inserted agreed CRs: OMA-BCAST-2005-0122R02; OMA-BCAST-2005-0218R02; OMA-BCAST-2005-0221R02; OMA-BCAST-2005-0248R02; OMA-BCAST-2005-0264R01 * Added text in Normative Reference section
	30 Jun 2005	9.2	* Inserted agreed CR: OMA-BCAST-2005-0267R01 * Added reference to OMA-BCAST-2005-0264R01 in Draft History section * Added text in Normative Reference section
	06 Jul 2005	4.3; Previous 5.1.1.2.1	* Moved Figure 3 from Section 5.1.1.2.1 to Section 4.3, as requested by email from David Castleford on 04 July 2005. * Deleted Section 5.1.1.2.1 because it becomes empty.
	15 Aug 2005	5.4.3; Previous	* Changed name from "Key Management Layer" to "Authentication Layer" as per resolution to BAC BCAST Action Item 041.

Document Identifier	Date	Sections	Description
		5.1.2.3.3; 2.1; 3.4; 5.5; 5.5.2; Previous 5.1.2.4.5; 14; 5.5.2	* Changed name from "Rights Management Layer" to "Long Term Key Delivery Layer" as per resolution to BAC BCAST Action Item 041. * Changed name from "Key Delivery Layer" to "Short Term Key Delivery Layer" as per resolution to BAC BCAST Action Item 041. * Changed name from "Service Encryption Layer" to "Content Layer" as per resolution to BAC BCAST Action Item 041. * Changed name from "key stream layer" to "Short Term Key Delivery Layer" assuming that "key stream layer" is same as "Key Delivery Layer". * Inserted agreed CR: OMA-BCAST-2005-0231 * Inserted agreed CR: OMA-BCAST-2005-0232 * Added text in Normative References section
	29 Aug 2005	4.1; 5.4.1; 5.4.2; 5.5; 9.1; 9.2; 3.2	* Where appropriate, changed term from "rights object" to "long-term key message" as per resolution to BAC BCAST Action Item 055. * Inserted agreed CR: OMA-BCAST-2005-0371 * Inserted agreed CR: OMA-DLDRM-2005-0268R01 * Inserted agreed CR: OMA-BCAST-2005-0348R03 * Inserted agreed CR: OMA-BCAST-2005-0346R02
	30 Sep 2005	9.1; 4.1;	* Inserted agreed CR: OMA-BCAST-2005-0430R02 * Inserted agreed CR: OMA-BCAST-2005-0438R01 * Inserted agreed CR: OMA-BCAST-2005-0459R01
	20 Oct 2005	9.1; 2.1; Previous 5.1.1; 13;	* Inserted agreed CR: OMA-BCAST-2005-0512R01 * Inserted agreed CR: OMA-BCAST-2005-0558 * Inserted agreed CR: OMA-BCAST-2005-0431R01 * Inserted agreed CR: OMA-BCAST-2005-0465R02
	23 Nov 2005	4.3; 5.5; 6.6.1; Appendix B	* Inserted additional changes made by agreed CR: OMA-BCAST-2005-0558, which were not inserted in earlier version by mistake. * Inserted agreed CR: OMA-BCAST-2005-0566R02 * Inserted agreed CR: OMA-BCAST-2005-0528R01 * Inserted agreed CR: OMA-BCAST-2005-0466R03 * Created SCR tables
	06 Dec 2005	14.1; 14.2; 14.3; 14.4; Previous 5.1.2.3; Previous 5.1.2.4; Appendix B	* Inserted agreed CR: OMA-BCAST-2005-0624R01 * Inserted agreed CR: OMA-BCAST-2005-0625R01 * Inserted agreed CR: OMA-BCAST-2005-0626R01 * Inserted agreed CR: OMA-BCAST-2005-0627R01 * Inserted agreed CR: OMA-BCAST-2005-0656R01 * Removed Sections 5.1.2.3 and 5.1.2.4 (addressing and registration) as per agreed CR OMA-BCAST-2005-0656R01 * Added entries to SCR tables to reflect new agreed CRs
	03 Jan 2006	3.2; 5.5;; 17.1; 6.5.2; 6.8; Appendix B	* Inserted agreed CR: OMA-BCAST-2005-0723R01 * Inserted agreed CR: OMA-BCAST-2005-0673R01 * Inserted agreed CR: OMA-BCAST-2005-0672R01 * Inserted agreed CR: OMA-BCAST-2005-0730 * Inserted agreed CR: OMA-BCAST-2005-0667 * Added entries to SCR tables to reflect new agreed CRs
	25 Jan 2006	2.1; 4.3; 6.4.1; 6.4.1.2	* Inserted agreed CR: OMA-BCAST-2005-0664R04 * Inserted agreed CR: OMA-BCAST-2005-0687R01 * Edited entries to SCR tables to reflect new agreed CRs
	16 Feb 2006	6.5.2; 3.2; 2.1;7.3;6.4.1.1;5.5; 4.3;6.5.2;13	* Inserted agreed CR: OMA-BCAST-2005-0761 * Inserted agreed CR: OMA-BCAST-2006-0095 * Inserted agreed CR: OMA-BCAST-2006-0072R02 * Inserted agreed CR: OMA-BCAST-2006-0021R02 * Inserted agreed CR: OMA-BCAST-2006-0045 * Inserted agreed CR: OMA-BCAST-2006-0051 * Inserted reference to RFC 2045 * Inserted withdrawn CR: OMA-BCAST-2006-0047R01 (see below) * Inserted agreed CR: OMA-BCAST-2005-0653R02

Document Identifier	Date	Sections	Description
			* Inserted agreed CR: OMA-BCAST-2005-0752 * Inserted agreed CR: OMA-BCAST-2005-704R01 * Inserted noted CR: OMA-BCAST-2005-0738R01 (see below) * Inserted agreed CR: OMA-BCAST-2005-0753R02 * Inserted agreed CR: OMA-BCAST-2006-0029R01 * Inserted agreed CR: OMA-BCAST-2005-0766R01 * Inserted agreed CR: OMA-BCAST-2005-0731 * Inserted agreed CR: OMA-BCAST-2005-0691R02 * Inserted agreed CR: OMA-BCAST-2006-0052R01 * Inserted agreed CR: OMA-BCAST-2006-0108R01
	24 Feb 2006	7.5.1; 5.5	* Removed CR OMA-BCAST-2006-0047R01 (which was withdrawn) and inserted agreed CR OMA-BCAST-2006-0047 * Removed CR OMA-BCAST-2005-0738R01 (which was noted) and inserted agreed CR OMA-BCAST-2005-0738R01 * Inserted TKM_ALGO_DCF specification in Section 5.5. The specification was introduced in agreed CR OMA-BCAST-2005-0466R03 but was deleted inadvertently by agreed CR OMA-BCAST-2005-0656R01
	03 Mar 2006	6.6.1;6.6.2; 10.2.1; 10.2.2; 10.2.3; 10.2.4; 2.1;?;6.5.2.1; 5.7.2	* Inserted agreed CR: OMA-BCAST-2006-0033R01 * Inserted agreed CR: OMA-BCAST-2006-0088R04 * Inserted Normative Reference to RFC 3237 * Inserted agreed CR: OMA-BCAST-2006-0145 * Inserted agreed CR: OMA-BCAST-2006-0146 * Inserted agreed CR: OMA-BCAST-2006-0150
	15 Mar 2006	2.1; 3.2; 14.1.1.1; 14.2.1.1; 14.3.1.1; 14.4.1.1; 17.1;9.2; 5.3.1;5.5; 6.5.2.1; 7.8; 4.2; 4.3; 4.5.1; 12; 4.1; 9.4; 6.3;6.4; 11; Previous 5.5; 10.2; old Appendix D; 14.1.2.1.2; 14.2.2.1.2; 14.2.2.1.4; 14.3.2.1.2; 14.3.2.1.4; 14.4.2.1.2; 14.4.2.1.4; Appendix B	* Inserted agreed CR: OMA-BCAST-2006-0224R01 * Inserted agreed CR: OMA-BCAST-2006-0223 * Sorted Definitions in alphabetical order * Inserted agreed CR: OMA-BCAST-2006-0202 * Inserted agreed CR: OMA-BCAST-2006-0201 * Inserted agreed CR: OMA-BCAST-2006-0203 * Inserted agreed CR: OMA-BCAST-2006-0204 * Inserted agreed CR: OMA-BCAST-2006-0049R03 * Inserted agreed CR: OMA-BCAST-2006-0239R01 * Inserted agreed CR: OMA-BCAST-2006-0213 * Inserted agreed CR: OMA-BCAST-2006-0209R01 * Inserted agreed CR: OMA-BCAST-2006-0166R03 * Inserted agreed CR: OMA-BCAST-2006-0207R01 * Inserted agreed CR: OMA-BCAST-2006-0158 * Inserted agreed CR: OMA-BCAST-2006-0164R02 * Inserted agreed CR: OMA-BCAST-2006-0206R01 * Inserted agreed CR: OMA-BCAST-2006-0165R01 * Inserted agreed CR: OMA-BCAST-2005-0722R05 * Inserted agreed CR: OMA-BCAST-2006-0089R04 * Inserted agreed CR: OMA-BCAST-2006-0126R02 * Inserted agreed CR: OMA-BCAST-2006-0211R01 * Inserted agreed CR: OMA-BCAST-2006-0255 * Re-created the SCR tables to reflect agreed CRs
	24 Mar 2006	4.5	* In response to authors of agreed CR OMA-BCAST-2006-0089R04, moved previous Section 5.5 to new Section 4.5.1
	12 Apr 2006	Entire Document	* The current document is the Agreed CR: OMA-BCAST-2006-0337R02
	03 Oct 2006	Entire Document	* Inserted the following Agreed CRs: OMA-BCAST-2006-0760, OMA-BCAST-2006-0748R02, OMA-BCAST-2006-0506R01, OMA-BCAST-2006-0190R07, OMA-BCAST-2005-0527R01, OMA-BCAST-2006-0733, OMA-BCAST-2006-0701, OMA-BCAST-2006-0680, OMA-BCAST-2006-0660R01, OMA-BCAST-2006-0665, OMA-BCAST-2006-0667, OMA-BCAST-2006-0668R02, OMA-BCAST-

Document Identifier	Date	Sections	Description
			2006-0674, OMA-BCAST-2006-0655, OMA-BCAST-2006-0596R02, OMA-BCAST-2006-0591, OMA-BCAST-2006-0398R04, OMA-BCAST-2006-0634R02, OMA-BCAST-2006-0616R01, OMA-BCAST-2006-0663R02, OMA-BCAST-2006-0636, OMA-BCAST-2006-0618R01, OMA-BCAST-2006-0654R01, OMA-BCAST-2006-0480R02, OMA-BCAST-2006-0505R01, OMA-BCAST-2006-0648, OMA-BCAST-2006-0407R01, OMA-BCAST-2006-0408R01, OMA-BCAST-2006-0409, OMA-BCAST-2006-0410R02, OMA-BCAST-2006-0477, OMA-BCAST-2006-0478R01, OMA-BCAST-2006-0479, OMA-BCAST-2006-0396R03, OMA-BCAST-2006-0291R01, OMA-BCAST-2006-0290R01, OMA-BCAST-2006-0343R01, OMA-BCAST-2006-0289, OMA-BCAST-2006-0292, OMA-BCAST-2006-0264R04 * Inserted the resolutions for the Closed comments: SC-New-0184, SC-New-0202, SC-New-0203, SC-New-0208, SC-New-0126, SC-New-0245, SC-New-0237, SC-New-0180, SC-New-0009, SC-New-0388, SC-New-0225, SC-New-0077, SC-New-0305, SC-New-0297, SC-New-0194, SC-New-0197, SC-New-0209, SC-New-0240, SC-New-0304, SC-New-0277, SC-New-0279, SC-New-0035, SC-New-0078, SC-New-0348, SC-New-0080, SC-New-0081, SC-New-0039, SC-New-0082, SC-New-0083, SC-New-0084, SC-New-0312, SC-New-0085, SC-New-0087, SC-New-0088, SC-New-0089, SC-New-0316, SC-New-0049, SC-New-0092, SC-New-0096, SC-New-0100, SC-New-0343, SC-New-0284, SC-New-0063, SC-New-0117, SC-New-0121, SC-New-0068, SC-New-0127, SC-New-0093, SC-New-0094, SC-New-0289, SC-New-0342, SC-New-0129, SC-New-0131, SC-New-0133, SC-New-0248, SC-New-0292, Sc-New-0250, SC-New-0137, SC-New-0140, SC-New-0294, SC-New-0095, SC-New-0222, SC-new-0145, SC-New-0151, SC-New-0223, SC-New-0213, SC-New-0314
	04 Oct 2006	6.7.1; 6	* Restored STKM part of Section 6.7.1, based on discussions with David Castleford (Orange). The part was deleted based on the Editor's misunderstanding of CRs BCAST-2006-0562R03, BCAST-2006-0680, and BCAST-2006-701. * Re-numbered section headings in Section 6, as required by resolution to Closed comment SC-New-0288. * Inserted the resolutions for the Closed comments: SC-New-0290, SC-New-0293
	10 Oct 2006	1.1; 6.7	* Modified the heading numbers in Section 1.1 and Section 6.7. * Inserted the following Agreed CRs: OMA-BCAST-2006-0470, OMA-BCAST-2006-0615, OMA-BCAST-2006-0316R03
	10 Nov 2006	9.2; 5.5; 6.7; 7; 5.6.2.1; 6.8.2.1; 4.1; 4.5.1; 6.7.2; 11; 10.1.3; 6.12;	* Inserted the following Agreed CRs: OMA-BCAST-2006-0671R02, OMA-BCAST-2006-0823R01, OMA-BCAST-2006-0609R02, OMA-BCAST-2006-0867R01, OMA-BCAST-2006-0883R01, OMA-BCAST-2006-0898R01, OMA-BCAST-2006-0595R02 * Inserted the resolutions for the Closed comments: SC-New-0134, SC-New-0002, SC-New-0350, SC-New-0362, SC-New-0334, SC-New-0310, SC-New-0285, SC-New-0130, SC-New-0288, SC-New-0136, SC-New-0138, SC-New-0139, SC-New-0272, SC-New-0141, SC-New-0142, SC-New-0144, SC-New-0115 *Removed Section 14 as specified by Agreed resolution to comment SC-New-0197
	07 Dec 2006	4.5; 15.1; 15.2; 15.3; 11.4; 4.1; 6.11.2.1; 12.4; 2.1; 13.1; 13.1.1; 11.3; 4.3; 4.2; 12.2; 12.3; 6; former Section 7; 4.5.1; 3.3; 11.3; 12; 15.3; 3.1; 4.3.1; 5.3; 5.5.1; 5.5.2; 9.1; 6.7.1.1;	* Inserted the following Agreed CRs: OMA-BCAST-2006-0918, OMA-BCAST-2006-0904, OMA-BCAST-2006-0935, OMA-BCAST-2006-0998, OMA-BCAST-2006-0939, OMA-BCAST-2006-0833R07, OMA-BCAST-2006-0890R01, OMA-BCAST-2006-0944R01, OMA-BCAST-2006-1007, OMA-BCAST-2006-1006R02, OMA-BCAST-2006-0694R05, OMA-BCAST-2006-0955R05, OMA-BCAST-2006-1009R03, OMA-BCAST-2006-0642, OMA-BCAST-2006-0688, OMA-BCAST-2006-0699R01, OMA-BCAST-2006-1013, OMA-BCAST-2006-0936R01, OMA-BCAST-2006-0957 * Inserted the resolutions for the Closed comments: SC-old-007, SC-New-0298, SC-New-0195, SC-New-0394, SC-New-0232, SC-New-0276, SC-New-0351, SC-New-0309, SC-New-0311, SC-New-0352, SC-New-0281, SC-New-0282, SC-New-0353, SC-New-0355, SC-New-0356, SC-New-0091, SC-New-0357, SC-New-0097, SC-New-0098, SC-New-0099, SC-New-0317, SC-New-0360, SC-New-0363, SC-New-0101, SC-New-0103, SC-New-0319, SC-New-0366, SC-New-0321, SC-New-0062, SC-New-0335, SC-New-0375, SC-New-0337, SC-New-

Document Identifier	Date	Sections	Description
		5.5; 6.8.1.3	0125, SC-New-0286, SC-New-0339, SC-New-0075, SC-New-0341, SC-New-0381, SC-New-0380
	18 Dec 2006	13.1; 4.3.1; 4.3.2; 13.1.2.2; 13.1.2.3; 13.1.2.4; old Appendix D; 10.1.2; 10.1.3; B.2; 6.11.1; 2.1; 6.6.1; 5.5; 6.7.2; 6.6(various)6 .7(various) 3.2; 5.8; 4.4; 4.4.1; 4.4.2; 3.3; 2.1; 4.5.1; 7.3; 9.1	* Inserted the heading "13.1.2 BCAST Specific Interface" as was required by BCAST-2006-0833R07 * Replace "serviceBaseCID" with "BaseCID" for compatibility with the SG specification, as requested by KPN in Washington DC meetings. * Replace "program_flag" with "program_flag" for consistency. * Replace "ICRO" with "RO" as is required by resolution to comment SC-New-0077. * Inserted the following Agreed CRs: OMA-BCAST-2006-0878R01, OMA-BCAST-2006-0880R01, OMA-BCAST-2006-0190R09, OMA-BCAST-2006-0969R01, OMA-BCAST-2006-0932R01, OMA-BCAST-2006-0807R04, OMA-BCAST-2006-0902R01, OMA-BCAST-2006-0976R03, OMA-BCAST-2006-0994R01, OMA-BCAST-2006-0941R02, OMA-BCAST-2006-0938R02, OMA-BCAST-2006-0942R03, OMA-BCAST-2006-1100, OMA-BCAST-2006-0950R02, OMA-BCAST-2006-0903, OMA-BCAST-2006-0956R03 * Inserted the resolutions for the Closed comments: SC-New-0113, SC-New-0320, SC-New-0175, SC-New-0283, SC-New-0017, SC-New-0303, SC-New-0307, SC-New-0347, SC-New-0302
	10 Jan 2007	10.1	* Inserted Agreed CR: OMA-BCAST-2006-1055R01
	22 Jan 2007	6.2; 6.5.2; 6.6; 6.6.2; 6.7; 6.8.1.2; 6.9.1; 13.1.1.3.1; 6.6.4; 6.7; 6.7.2	* Inserted the resolutions for the Tentatively Closed comments: SCP-0004, SCP-0006, SCP-0009, SCP-0010, SCP-0013, SCP-0014, SCP-0015, SCP-0016, SCP-0017, SCP-0018, SCP-0019, SCP-0020, SCP-0024, SCP-0025, SCP-0026, SCP-0027, SCP-0028, SCP-0029, SCP-0030, SCP-0031, SCP-0032, SCP-0034, SCP-0037, SCP-0038, SCP-0039, SCP-0040, SCP-0043, SCP-0052, SCP-0061, SCP-0073 * Replaced [3GPP2 S.S0083-A] with [3GPP2 S.S0083], as was suggested in the interim OMA BAC BCAST meeting in Singapore * Inserted the following Tentatively Agreed CRs: OMA-BCAST-2007-0024R01, OMA-BCAST-2007-0046R01
	04 Feb 2007	6.10; 6.6.4; 6.7.2; 6.7.3; 6.8.1.2; 6.8.1.3; 6.8.2.2; 6.9.1; Appendix C	* Inserted the resolutions for the Tentatively Closed comments: SCP-0033, SCP-0046, SCP-0047, SCP-0048, SCP-0049, SCP-0050, SCP-0051, SCP-0053, SCP-0054, SCP-0055, SCP-0059, SCP-0060, SCP-0062, SCP-0063, SCP-0064, SCP-0204 * Inserted the following Tentatively Agreed CRs: OMA-BCAST-2007-0036
	27 Feb 2007	3.2; 6.6.4; 6.7.2; 11.1; 5.5.1; 7.3; 5.5; 4.5.2; 6.6; 6.6.2.2; 6.6.5; 6.7; 13.1.2.2.1.2 13.1.2.3.1.1 13.1.2.4.1.1 13.1.2.4.1.4 old Appendix D; 6.11; 6.12; 9.3; 4.5; 10; 5.3; 4.1; 6.2; 6.5.1; 6.6; 6.8.1.3; 8.1; 9.1; 9.2; 4.3; 10.1.5; 11.3.2; 3.3; 5.4.3; 3.4; 5.5.2; 5.6; 5.6.2.1; 5.8; 14; 6.6.2; 6.6.3; 6.7.1; 6.7.3; 6.8.2.1; 6.11.2.3; 6.11.2.4;	* Inserted normative reference to [3GPP TS 31.103 v6] * Inserted the following Tentatively Agreed CRs: OMA-BCAST-2007-0082R02, OMA-BCAST-2007-0083R01, OMA-BCAST-2007-0114, OMA-BCAST-2007-0106R01, OMA-BCAST-2007-0175, OMA-BCAST-2007-0177, OMA-BCAST-2007-0178, OMA-BCAST-2007-0182, OMA-BCAST-2007-0183, OMA-BCAST-2007-0184, OMA-BCAST-2007-0185, OMA-BCAST-2007-0199R02, OMA-BCAST-2007-0030R01, OMA-BCAST-2007-0173R01 * Inserted the resolutions for the Tentatively Closed comments: SCP-0067, SCP-0069, SCP-0074, SCP-0078, SCP-0079, SCP-0142, SCP-0080, SCP-0140, SCP-0082, SCP-0087, SCP-0092, SCP-0093, SCP-0094, SCP-0095, SCP-0096, SCP-0097, SCP-0098, SCP-0099, SCP-0100, SCP-0101, SCP-0102, SCP-0103, SCP-0104, SCP-0106, SCP-0110, SCP-0113, SCP-0116, SCP-0117, SCP-0241, SCP-0248, SCP-0118, SCP-0119, SCP-0120, SCP-0122, SCP-0123, SCP-0127, SCP-0128, SCP-0130, SCP-0131, SCP-0132, SCP-0133, SCP-0134, SCP-0135, SCP-0136, SCP-0137, SCP-0138, SCP-0139, SCP-0141, SCP-0143, SCP-0144, SCP-0145, SCP-0146, SCP-0147, SCP-0148, SCP-0149, SCP-0150, SCP-0152, SCP-0153, SCP-0154, SCP-0155, SCP-0202, SCP-0156, SCP-0157, SCP-0158, SCP-0161, SCP-0163, SCP-0164, SCP-0165, SCP-0166, SCP-0222, SCP-0167, SCP-0168, SCP-0169, SCP-0173, SCP-0175, SCP-0177, SCP-0178, SCP-0180, SCP-0181, SCP-0182, SCP-0183, SCP-0184, SCP-0185, SCP-0186, SCP-0187, SCP-0188, SCP-0189, SCP-0190, SCP-0191, SCP-0192, SCP-0193, SCP-0194, SCP-0195, SCP-0197, SCP-0198, SCP-0199, SCP-0200, SCP-0201, SCP-0205, SCP-0206, SCP-0207, SCP-0209, SCP-0212, SCP-0214, SCP-0215, SCP-0217, SCP-0220, SCP-0223, SCP-0224, SCP-0226, SCP-0232, SCP-0234, SCP-0245, SCP-0246, SCP-0268, SCP-0269, SCP-0270 (whole sentence was removed by another

Document Identifier	Date	Sections	Description
		6.9.1; 7.1; 8; 8.1; 8.2; 8.3; 8.3.2; 12; 13.1.1.3.1; 4.2; 4.3.1; 13.1.2.2.1.1 6.8.1.1; 11; 5.4.1; 5.7; 6.1; 6.11.1	inserted change), SCP-0279, SCP-0284, SCP-0286, SCP-0287, SCP-0288, SCP-0289, SCP-0290, SCP-0291, SCP-0292, SCP-0293, SCP-0294, SCP-0295, SCP-0296, SCP-0297, SCP-0298, SCP-0299, SCP-0300, SCP-0301, SCP-0302, SCP-0303, SCP-0304, SCP-0305, SCP-0306, SCP-0307, SCP-0310, SCP-0311, SCP-0312, SCP-0313, SCP-0314, SCP-0315, SCP-0316, SCP-0317, SCP-0318, SCP-0319, SCP-0320, SCP-0321, SCP-0322, SCP-0323, SCP-0324, SCP-0325, SCP-0326, SCP-0327, SCP-0328, SCP-0330, SCP-0331
	27 Mar 2007	2.2; 9.3.1; previous 9.3.1.2; 9.3.2; 3.3; 6.11.1; 10.2.2; 10.2.2.1; 6.6.3; 3.25.4.1; 5.5.2; 4.5; 4.5.1; 4.5.2; 6.2; 6.3.1; 6.3.2; 4.2; 4.3.1; 6.7; previous 6.7.1.4; 6.7.3; 6.8.1.3; 6.8.2.1; 11.2; 11.3.2; 11.3.3; 13.1.2.2.1.23 .2; 6.6; 6.6.2; 6.7.1; 6.7.1.1; 6.8; 6.8.1.2; 6.8.1.3; 4.3; 6.4; 6.5; 6.5.1; 6.5.2; 6.10.1; 6.10.1.1; 6.10.1.2; 6.11.1; 10.4; 6.11.2; 6.11.2.1; 6.11.2.2; 6.11.2.3; 6.11.2.4; 6.11.2.5; 6.11.2.6; 11.5; 16; 6.6.4; 6.7.2; 5.6.2.1; 6.8.2.2; 9.4; 8.1; 8.2; 8.3; 8.3.1; 8.3.2; 6.8.1.1; 9.1; 5.5.1; 4.1; 4.4; 6.10.3; 5.4.2; 5.4.3; removed Appendix D; 2.1; 13.1.2.1; 13.1.2.2.1.1 13.1.2.2.1.2 13.1.2.2.1.4 13.1.2.3.1.1 13.1.2.3.1.2 13.1.2.4.1.3	* Corrected the cardinality of “keyIdentifier” in Section "13.1.2.4.1.1 STKM Request" from zero to one as in Tentatively Closed comment SCP-0216. * Inserted the following Tentatively Agreed CRs: OMA-BCAST-2007-0075R01, OMA-BCAST-2007-0179R01, OMA-BCAST-2007-0240R02, OMA-BCAST-2007-0242R02, OMA-BCAST-2007-0299R01, OMA-BCAST-2007-0300R03, OMA-BCAST-2007-0306R01, OMA-BCAST-2007-0314, OMA-BCAST-2007-0201, OMA-BCAST-2007-0347, OMA-BCAST-2007-0348R01, OMA-BCAST-2007-0349R01, OMA-BCAST-2007-0181R01, OMA-BCAST-2007-0293R01, OMA-BCAST-2007-0294, OMA-BCAST-2007-0303, OMA-BCAST-2007-0038R02, OMA-BCAST-2007-0326R01, OMA-BCAST-2007-0383, OMA-BCAST-2007-0117R03, OMA-BCAST-2007-0376R01, OMA-BCAST-2007-0232, OMA-BCAST-2007-0229R02, OMA-BCAST-2007-0323R01, OMA-BCAST-2007-0322R01, OMA-BCAST-2007-0381R01, OMA-BCAST-2007-0389 * Inserted the resolutions for the Tentatively Closed comments: SCP-0236, SCP-0329, SCP-0285, SCP-0332, SCP-0308, SCP-0275, SCP-0398, SCP-0344, SCP-0126, SCP-0129, SCP-0151, SCP-0309, SCP-0333, SCP-0334, SCP-0335, SCP-0336, SCP-0338, SCP-0340, SCP-0348, SCP-0357, SCP-0362, SCP-0385, SCP-0390, SCP-0391, SCP-0396, SCP-0397, , SCP-0399, SCP-0400, SCP-0407, SCP-0408, SCP-0409, SCP-0412, SCP-0445, SCP-0451, SCP-0452, SCP-0453, SCP-0454

Document Identifier	Date	Sections	Description
		13.1.2.4.1.4 6.6.2.2; 6.6.3; 6.11; 5.8; 12; 6.1; 9.2; 12.1; 12.2; 12.4; 10.2; 5.5; 7.1	
	04 Apr 2007	6.8.1.2; 6.6.2.2; 6.9; 6.9.1; 13.1; 6.6.2; 6.6.4; 6.10.1; 4.5; 6.6; 5.5.1; 10.2.3; B.1; 13.1.1.3.1 13.1.1.2; 6.6.2.2; Appendix D; 7.3	* Replaced "SEK ID" with "SEK/PEK ID" throughout document as specified by Action Item BCAST-2007-A111 * Replaced the term "smartcard" with the term "Smartcard" where appropriate * Modified SCR Table B.1 in response to CR OMA-BCAST-2007-0424 * Inserted the following Tentatively Agreed CRs: OMA-BCAST-2007-0416R01, OMA-BCAST-2007-0423, OMA-BCAST-2007-0424, OMA-BCAST-2007-0419, OMA-BCAST-2007-0425R01, OMA-BCAST-2007-0372R01 * Inserted the resolutions for the Tentatively Closed comments: SCP-0085, SCP-0218, SCP-0349, SCP-0358, SCP-0361, SCP-0377, SCP-0404, SCP-0405, SCP-0417
	13 Apr 2007	6.6; 6.7.2; 6.5.2; 13.1.2.2.1.1 13.1.2.2.1.2 13.1.2.2.1.4 13.1.2.3.1.1 13.1.2.3.1.2 13.1.2.4.1.1 13.1.2.4.1.2 13.1.2.4.1.3 13.1.2.4.1.4 4.5.2; 6.9.1; 6.6.4; 6.8.1.2; 13.1; 13.1.1; 6.6.2; Appendix E; 6.7.1; 6.7.2; 2.1; 6.7.2.1	* Inserted the following Tentatively Agreed CRs: OMA-BCAST-2007-0306R04, OMA-BCAST-2007-0446, OMA-BCAST-2007-0447, OMA-BCAST-2007-0324R07 (except Change 5) * Inserted the resolutions for the Tentatively Closed comments: SCP-0046, SCP-0047, SCP-0249, SCP-0423, SCP-0450
	19 Apr 2007	10.1.2.1; 6.6.2.2; 3.2; 6.1; 6.4; 2.1; 6.7.1; 6.7.2; 6.7.2.2; E.1.3; 2.2; 7.1; 6.6.4	* Changed "Mandatory" to "Optional" in Table 52 as per the minutes for OMA Frankfurt meetings (April 2007) * Updated version numbers of 3GPP and 3GPP2 references * Inserted the following Tentatively Agreed CRs: OMA-BCAST-2007-0046R03, OMA-BCAST-2007-0328R03, OMA-BCAST-2006-0584R08, OMA-BCAST-2007-0324R07 (Change 5)
	27 Apr 2007	5.8; B.1; B.2	* Changed the sentence "The parameter baseCID is announced in the Service Fragment of the SG" to "The parameter baseCID is announced in the Service fragment and Content fragment of the SG" * Inserted the following Tentatively Agreed CRs: OMA-BCAST-2007-0523, OMA-BCAST-2007-0193R01
	04 May 2007	All	Cleanup in preparation for Approval as Candidate
Candidate Version OMA-TS_BCAST_SvcCntProtection-V1_0	29 May 2007	n/a	Status changed to Candidate by TP TP ref# OMA-TP-2007-0129R01-INP_BCAST_V1_0_ERP_for_Candidate_approval

Appendix B. Static Conformance Requirements (Normative)

The notation used in this appendix is specified in [IOPPROC].

Note: BCAST adaptation specifications, such as [BCAST10-BCMCS-Adaptation], [BCAST10-DVBH-IPDC-Adaptation], and [BCAST10-MBMS-Adaptation], in which it is specified how the BCAST 1.0 enabler is implemented over a specific BDS (Broadcast Distribution System), may overrule or adapt requirements from this SCR or provide additional requirements.

B.1 SCR for Clients

Item	Function	Reference	Status	Requirement
BCAST-SPCP-C-001	Support for Service Protection	4.1	O	
BCAST-SPCP-C-002	Support for Content Protection	4.1	O	
BCAST-TerminalCapability-C-003	Terminal with cellular radio interface and with Smartcard supporting Service Protection	4.1	O	BCAST-SPCP-C-005
BCAST-TerminalCapability-C-004	Terminal without cellular radio interface or without Smartcard supporting Service Protection	4.1	O	BCAST-SPCP-C-006
BCAST-SPCP-C-005	Support for Smartcard Profile for Service Protection	4.1	O	BCAST-ContentLayer-C-007 AND BCAST-STKM_SC-C-010 AND AND BCAST-SDP-C-014 AND BCAST-LTKM_SC-C-015 AND BCAST-KeyManagement-C-016
BCAST-SPCP-C-006	Support for DRM Profile for Service Protection	4.1	O	BCAST-ContentLayer-C-007 AND BCAST-STKM_DRM-C-011 AND BCAST-LTKM_DRM-C-013 AND BCAST-SDP-C-014
BCAST-ContentLayer-C-007	Processing of Content Layer encryption - SRTP	9.2	O	BCAST-SRTPsignal-C-030
BCAST-ContentLayer-C-008	Processing of Content Layer encryption - IPsec	9.1	O	
BCAST-ContentLayer-C-009	Processing of Content Layer encryption - ISMACRYP	9.3	O	
BCAST-STKM_SC-C-010	STKM for Smartcard Profile	6.7	O	
BCAST-STKM_DRM-C-011	STKM for DRM Profile	5.5	O	
BCAST-SC_Binding-C-012	Binding of STKM to Terminal with Smartcard	12	O	
BCAST-LTKM_DRM-C-013	LTKM for DRM Profile	5.4	O	
BCAST-SDP-C-014	Protection Signaling in SDP	10.1	O	
BCAST-LTKM_SC-C-015	LTKM for Smartcard Profile	6.6	O	
BCAST-KeyManagement-C-016	Service Access for Terminal with Smartcard	6.10	O	
BCAST-Recording-C-017	Recording	8	O	
BCAST-CPFilesDRM-C-018	Content Protection for Downloading Files in Terminals with no Smartcard	5.6.2.2	O	

Item	Function	Reference	Status	Requirement
BCAST-CP_RTP_DRM-C-019	Content Protection for Streaming in Terminals with no Smartcard	5.6.1.2	O	BCAST-CP_Form-C-023
BCAST-CPFilesSC-C-020	Content Protection for Downloading Files in Terminals with Smartcard	6.8.2.2	O	BCAST-ClientID-C-027
BCAST-CP_RTP_SC-C-021	Content Protection for Streaming in Terminals with Smartcard	6.8.1	O	BCAST-CP_Form-C-023 AND BCAST-ClientID-C-027 AND BCAST-SAC-C-028
BCAST-CP_Recorded_SC-C-022	Content Protection of Recorded Material in Terminals with Smartcard	6.9.1	O	BCAST-ClientID-C-027 AND BCAST-SC_Binding-C-012
BCAST-CP_Form-C-023	Content Protection Format	9.3.1	O	
BCAST-Domains-C-024	Broadcast Domains	5.3	O	
BCAST-MeteringDRM-C-025	Usage Metering for DRM Profile	5.9	O	
BCAST-MeteringSC-C-026	Usage Metering for Smartcard Profile	6.6.3	O	
BCAST-ClientID-C-027	BCAST Client ID	6.11	O	
BCAST-SAC-C-028	Use of SAC	6.8.1.3	O	
BCAST-ContentLayer-C-029	Processing of Content Layer authentication - ISMACRYP	9.3	O	BCAST-SRTPsignal-C-030
BCAST-SRTPsignal-C-030	SDP Signalling of SRTP	10.4	O	

B.2 SCR for BSD/A

Item	Function	Reference	Status	Requirement
BCAST-BSDASPCP-S-001	Support Backend interface for Service Protection	13.1	O	BCAST-BSDASPCP-S-002 OR BCAST-BSDASPCP-S-003
BCAST-BSDASPCP-S-002	Support BCAST specific interface for SP-4	13.1.2	O	
BCAST-BSDASPCP-S-003	Support SP-4 by the adaptation of DVB Simulcrypt Head-end interfaces	13.1.1	O	
BCAST-BSDASPCP-S-004	Support Backend interface for Content Protection	13.2	O	BCAST-BSDASPCP-S-005
BCAST-BSDASPCP-S-005	Support CP-4 by BCAST specific interface	13.2	O	
BCAST-BSDASPCP-S-006	Support Service Protection	4	O	BCAST-BSDASPCP-S-007 OR BCAST-BSDASPCP-S-012
BCAST-BSDASPCP-S-007	Support DRM profile for Service Protection	5	O	BCAST-BSDASPCP-S-008 AND BCAST-BSDASPCP-S-009 AND

Item	Function	Reference	Status	Requirement
				BCAST-BSDASPCP-S-010 AND BCAST-BSDASPCP-S-011
BCAST-BSDASPCP-S-008	Support delivery of STKM for DRM profile	5.5	O	
BCAST-BSDASPCP-S-009	Support the encryption for Service Protection of Stream for DRM Profile	5.6.1.1	O	BCAST-BSDASPCP-S-028 OR (BCAST-BSDASPCP-S-029 AND BCAST-BSDASPCP-S-038) OR (BCAST-BSDASPCP-S-030 AND BCAST-BSDASPCP-S-037) OR BCAST-BSDASPCP-S-031 OR (BCAST-BSDASPCP-S-032 AND BCAST-BSDASPCP-S-037)
BCAST-BSDASPCP-S-010	Support the encryption for Service Protection of File for DRM Profile	5.6.2.1	O	BCAST-BSDASPCP-S-028 OR BCAST-BSDASPCP-S-033
BCAST-BSDASPCP-S-011	Support SDP signalling for Service Protection of DRM Profile	5.8	O	BCAST-BSDASPCP-S-034
BCAST-BSDASPCP-S-012	Support Smartcard Profile for Service protection	6	O	BCAST-BSDASPCP-S-013 AND BCAST-BSDASPCP-S-014 AND BCAST-BSDASPCP-S-015 AND BCAST-BSDASPCP-S-016
BCAST-BSDASPCP-S-013	Support delivery of STKM for Smartcard profile	6.7.2	O	
BCAST-BSDASPCP-S-014	Support the encryption for Service Protection of Stream for Smartcard Profile	6.7.3	O	BCAST-BSDASPCP-S-028 OR (BCAST-BSDASPCP-S-029 AND BCAST-BSDASPCP-S-038) OR (BCAST-BSDASPCP-S-030 AND BCAST-BSDASPCP-S-037)
BCAST-BSDASPCP-S-015	Support the encryption for Service Protection of File for Smartcard Profile	6.8.2.1	O	BCAST-BSDASPCP-S-028 OR BCAST-BSDASPCP-S-033
BCAST-BSDASPCP-S-016	Support SDP signalling for Service Protection of Smartcard Profile	6.10.1.2	O	BCAST-BSDASPCP-S-034
BCAST-BSDASPCP-S-017	Support Content Protection	4	O	BCAST-BSDASPCP-S-018 OR BCAST-BSDASPCP-S-023
BCAST-BSDASPCP-S-018	Support DRM profile for content protection	5	O	BCAST-BSDASPCP-S-019 AND BCAST-BSDASPCP-S-020 AND BCAST-BSDASPCP-S-021 AND BCAST-BSDASPCP-S-022
BCAST-BSDASPCP-S-019	Support delivery of STKM for DRM profile	5.5	O	
BCAST-BSDASPCP-S-020	Support the encryption for Content Protection of	5.6.1.2	O	BCAST-BSDASPCP-S-028 OR (BCAST-BSDASPCP-S-029 AND BCAST-BSDASPCP-S-038) OR

Item	Function	Reference	Status	Requirement
	Stream for DRM Profile			(BCAST-BSDASPCP-S-030 AND BCAST-BSDASPCP-S-037) OR (BCAST-BSDASPCP-S-032 AND BCAST-BSDASPCP-S-037)
BCAST-BSDASPCP-S-021	Support the encryption for Content Protection of File for DRM Profile	5.6.2.2	O	BCAST-BSDASPCP-S-028 OR BCAST-BSDASPCP-S-031 OR BCAST-BSDASPCP-S-033
BCAST-BSDASPCP-S-022	Support SDP signalling for Content Protection of DRM Profile	5.8	O	BCAST-BSDASPCP-S-034
BCAST-BSDASPCP-S-023	Support Smartcard Profile for content protection	6	O	BCAST-BSDASPCP-S-024 AND BCAST-BSDASPCP-S-025 AND BCAST-BSDASPCP-S-026 AND BCAST-BSDASPCP-S-027
BCAST-BSDASPCP-S-024	Support delivery of STKM for Smartcard profile	6.7.2	O	
BCAST-BSDASPCP-S-025	Support the encryption for Content Protection of Stream for Smartcard Profile	6.7.3	O	BCAST-BSDASPCP-S-028 OR (BCAST-BSDASPCP-S-029 AND BCAST-BSDASPCP-S-038) OR (BCAST-BSDASPCP-S-030 AND BCAST-BSDASPCP-S-037)
BCAST-BSDASPCP-S-026	Support the encryption for Content Protection of File for Smartcard Profile	6.8.2.2	O	BCAST-BSDASPCP-S-028 OR BCAST-BSDASPCP-S-033
BCAST-BSDASPCP-S-027	Support SDP signalling for Content Protection of Smartcard Profile	6.10.1.2	O	BCAST-BSDASPCP-S-034
BCAST-BSDASPCP-S-028	Support IPSEC	9.1	O	
BCAST-BSDASPCP-S-029	Support SRTP	9.2	O	
BCAST-BSDASPCP-S-030	Support ISMACryp	9.3.1	O	
BCAST-BSDASPCP-S-031	Support PDCF	9.4	O	
BCAST-BSDASPCP-S-032	Support ISMACryp with PDCF	9.3.2	O	
BCAST-BSDASPCP-S-033	Support DCF	9.4	O	
BCAST-BSDASPCP-S-034	Support SDP signalling for protection	10.1	O	BCAST-BSDASPCP-S-035 AND BCAST-BSDASPCP-S-036

Item	Function	Reference	Status	Requirement
BCAST-BSDASPCP-S-035	Support SDP signalling for STKM	10.1.2 and 10.1.3	O	
BCAST-BSDASPCP-S-036	Support SDP signalling for LTKM	10.1.4	O	
BCAST-BSDASPCP-S-037	Support SDP signalling for ISMACryp	10.2	O	
BCAST-BSDASPCP-S-038	Support SDP signalling for SRTP	10.4	O	
BCAST-BSDASPCP-S-039	Support for STKM generation	13.1.2.4	O	
BCAST-BSDASPCP-S-040	Support the common attribute of STKM	7	O	
BCAST-BSDASPCP-S-041	Support for the operation for recording	8	O	
BCAST-BSDASPCP-S-042	Support for sharing a protected data stream for the different operators using both DRM and Smartcard profile	11	O	BCAST-BSDASPCP-S-043 AND BCAST-BSDASPCP-S-044 AND (BCAST-BSDASPCP-S-045 OR BCAST-BCAST-BSDASPCP-S-046)
BCAST-BSDASPCP-S-043	Support mapping for mapping of encryption and authentication keys	11.1	O	
BCAST-BSDASPCP-S-044	Support mapping for mapping between Key IDs for Smartcard profile and Key IDs for DRM profile	11.2	O	
BCAST-BSDASPCP-S-045	Support sharing SRTP Protected data Stream	11.3	O	
BCAST-BSDASPCP-S-046	Support sharing ISMACryp Protected data Stream	11.4	O	

B.3 SCR for BSM

Item	Function	Reference	Status	Requirement
BCAST-BSMSPCP-S-001	Support Backend interface for Service Protection	13.1	O	BCAST-BSMSPCP-S-002 OR BCAST-BSMSPCP-S-003
BCAST-BSMSPCP-S-002	Support BCAST specific interface for SP-4	13.1.2	O	
BCAST-BSMSPCP-S-	Support SP-4 by the	13.1.1	O	

Item	Function	Reference	Status	Requirement
003	adaptation of DVB Simulcrypt Head-end interfaces			
BCAST-BSMSPCP-S-004	Support Backend interface for Content Protection	13.2	O	BCAST-BSMSPCP-S-005
BCAST-BSMSPCP-S-005	Support CP-4 by BCAST specific interface	13.2	O	
BCAST-BSMSPCP-S-006	Support Service Protection	4	O	BCAST-BSMSPCP-S-007 OR BCAST-BSMSPCP-S-008
BCAST-BSMSPCP-S-007	Support DRM profile for Service Protection	5	O	BCAST-BSMSPCP-S-009 AND BCAST-BSMSPCP-S-010 AND BCAST-BSMSPCP-S-011 AND BCAST-BSMSPCP-S-016
BCAST-BSMSPCP-S-008	Support Smartcard profile for Service Protection	6	O	BCAST-BSMSPCP-S-012 AND BCAST-BSMSPCP-S-013 AND BCAST-BSMSPCP-S-014 AND BCAST-BSMSPCP-S-015 AND BCAST-BSMSPCP-S-016
BCAST-BSMSPCP-S-009	Support registration for DRM Profile	5.3	O	
BCAST-BSMSPCP-S-010	Support LTKM generation for DRM Profile for Service Protection	5.4	O	
BCAST-BSMSPCP-S-011	Support STKM generation for DRM Profile for Service Protection	5.5	O	
BCAST-BSMSPCP-S-012	Support Subscriber Key Establishment for Smartcard Profile	6.5	O	
BCAST-BSMSPCP-S-013	Support LTKM generation for Smartcard Profile for Service Protection	6.6	O	
BCAST-BSMSPCP-S-014	Support STKM generation for Smartcard Profile for Service Protection	6.7	O	
BCAST-BSMSPCP-S-015	Support for BCAST Client ID for Smartcard Profile	6.11	O	
BCAST-BSMSPCP-S-016	Support the common attribute of STKM	7	O	
BCAST-BSMSPCP-S-017	Support Content Protection	4	O	(BCAST-BSMSPCP-S-018 OR BCAST-BSMSPCP-S-019) AND BCAST-BSMSPCP-S-016

Item	Function	Reference	Status	Requirement
BCAST-BSMSPCP-S-018	Support DRM Profile for Content Protection	5	O	BCAST-BSMSPCP-S-020 AND BCAST-BSMSPCP-S-021 AND BCAST-BSMSPCP-S-022
BCAST-BSMSPCP-S-019	Support Smartcard Profile for Content Protection	6	O	BCAST-BSMSPCP-S-023 AND BCAST-BSMSPCP-S-024 AND BCAST-BSMSPCP-S-025
BCAST-BSMSPCP-S-020	Support registration for DRM Profile	5.3	O	
BCAST-BSMSPCP-S-021	Support LTKM generation for DRM Profile for Content Protection	5.4	O	
BCAST-BSMSPCP-S-022	Support STKM generation for DRM Profile for Content Protection	5.5	O	
BCAST-BSMSPCP-S-023	Support Subscriber Key Establishment for Smartcard Profile	6.5	O	
BCAST-BSMSPCP-S-024	Support LTKM generation for Smartcard Profile for Content Protection	6.6	O	
BCAST-BSMSPCP-S-025	Support STKM generation for Smartcard Profile for Content Protection	6.7	O	
BCAST-BSMSPCP-S-026	Support recording for DRM Profile for Content Protection	5.7 and 8	O	
BCAST-BSMSPCP-S-027	Support recording for Smartcard Profile for Content Protection	6.9 and 8	O	
BCAST-BSMSPCP-S-028	Usage Metering for DRM Profile	5.9	O	
BCAST-BSMSPCP-S-029	Support TBK for Smartcard Profile for Content Protection	12	O	

Appendix C. Global Status Codes

Table 60: Global Status Codes

Code	Status
000	**Success** The request was processed successfully.
001	**Device Authentication Failed** This code indicates that the BSM was unable to authenticate the device, which may be due to the fact that the device is not registered with the BSM. In this case, the user may contact the BSM, and establish a contract, or get the credentials in place that are used for authentication.
002	**User Authentication Failed** This code indicates that the BSM was unable to authenticate the user, which may be due to the fact that the device is not registered with the BSM. In this case, the user may contact the BSM, and establish a contract, or get the credentials in place that are used for authentication.
003	**Purchase Item Unknown** This code indicates that the requested service item is unknown. This can happen e.g. if the device has a cached service guide with old information. In this case, the user may re-acquire the service guide.
004	**Device Authorization Failed** This code indicates that the device is not authorized to get Long-Term Key Messages from the RI, e.g. because the device certificate was revoked. In this case, the user may contact the BSM operator.
005	**User Authorization Failed** This code indicates that the user is not authorized to get Long-Term Key Messages from the RI, e.g. because the device certificate was revoked. In this case, the user may contact the BSM operator.
006	**Device Not Registered** This code indicates that the device is not registered with the RI that is used for the transaction. When this code is sent, the response message includes a registration trigger that allows the device to register. In this case, the device may automatically perform the registration, and, if the registration is successful, re-initiate the original transaction.
007	**Server Error** This code indicates that there was a server error, such as a problem connecting to a remote back-end system. In such a case, the transaction may succeed if it is re-initiated later.
008	**Mal-formed Message Error** This code indicates that there has been a device malfunction, such as a mal-formed XML request. In such a case, the transaction may or may not (e.g. if there is an interoperability problem) succeed if it is re-initiated later. Note: This code can also be used between network entities.

009	**Charging Error** This code indicates that the charging step failed (e.g. agreed credit limit reached, account blocked). The user may in such a case contact the BSM operator. Note: This code can also be used between network entities.
010	**No Subscription** This code indicates that there has never been a subscription for this service item, or that the subscription for this item has terminated. The user may in such a case issue a service request for a new subscription.
011	**Operation not Permitted** This code indicates that the operation that the device attempted to perform is not permitted under the contract between BSM and user. The user may in this case contact BSM operator and change the contract. Note: This code can also be used between network entities.
012	**Unsupported version** This code indicates that the version number specified in the request message is not supported by the network. In this case, the user may contact the BSM operator. Note: This code can also be used between network entities.
013	**Illegal Device** This code indicates that the device requesting services is not acceptable to the BSM. E.g. Blacklisted. In this case, the user may contact the BSM operator.
014	**Service Area not Allowed** This code indicates that the device is not allowed services in the requested area due to subscription limits In this case, the user may contact the BSM operator or subscribe to the applicable service.
015	**Requested Service Unavailable** This code indicates that the requested service is unavailable due to transmission problems. In this case, the request may re-initiated at a later time. Note: This code can also be used between network entities.
016	**Request already Processed** This code indicates that an identical request has been previously processed. In this case, the user or the entity may check to see if the request had already been processed (i.e. received an LTK), if not retry the request.
017	**Information Element Non-existent** This code indicates that the message includes information elements not recognized because the information element identifier is not defined or it is defined but not implemented by the entity receiving the message. In this case related entities should contact each other.
018	**Unspecified** This code indicates that an error has occurred which cannot be identified. In this case related entities should contact each other.

019	**Process Delayed** Due to heavy load, request is in the cue, waiting to be processed. In this case the user or entity should wait for the transaction to complete. Note: This code can also be used between network entities.
020	**Generation Failure** This code indicates that the request information (message) could not be generated. In this case the user or entity should retry later.
021	**Information Invalid** This code indicates that the information given is invalid and cannot be used by the system. In this case the request should be rechecked and sent again.
022	**Invalid Request** This code indicates that the requesting key materials and messages (e.g., LTKM) are not valid and can not be fulfilled. In this case the request should be rechecked and sent again.
023	**Wrong Destination** This code indicates that the destination of the message is not the intended one. In this case the request should be rechecked and sent again.
024	**Delivery of Wrong Key Information** This code indicates that the delivered key information and messages (e.g., LTKM) are invalid. In this case the request should be rechecked and sent again.
025	**Service Provider ID Unknown** This code indicates a confliction when the Visited Service or Home Provider requests a message to the Home Service or Visited Provider.
026	**Service Provider BSM_ID Unknown** This code indicates a confliction when the Visited Service or Home Provider BSM requests a message to the Home Service or Visited Provider BSM.
027 ~ 127	Reserved for future use
128 ~ 255	Reserved for proprietary use

The informative Table 61 below proposes example values from the Global Status Codes in the table above for the transaction messages that require the use of Global Status Codes. The values shown below are for guidance purposes and the full range of values of the Global Status Codes are applicable to all messages if deemed required.

Table 61: Cross Reference Table (Informative)

TS-BCAST_SvcCntProtection	13.1.2.2.1.2 Key Request Response	000, 007, 008, 011, 012, 015, 016, 017, 018, 019, 020, 021, 022, 023
	13.1.2.2.1.4 Key Delivery Confirmation	000, 007, 008, 011, 012, 015, 016, 017, 018, 019, 020, 021, 022, 023
	13.1.2.3.1.2 Key Request Response	000, 007, 008, 011, 012, 015, 016, 017, 018, 019, 020, 021, 022, 023

	13.1.2.3.1.4 Key Delivery Confirmation	000, 007, 008, 011, 012, 015, 016, 017, 018, 019, 020, 021, 022, 023
	13.1.2.4.1.2 STKM Response	000, 007, 008, 011, 012, 015, 016, 017, 018, 019, 020, 021, 022, 023
	13.1.2.4.1.4 Partial STKM response message	000, 007, 008, 011, 012, 015, 016, 017, 018, 019, 020, 021, 022, 023
	13.1.2.4.1.6 STKM Delivery Confirmation	000, 007, 008, 011, 012, 015, 016, 017, 018, 019, 020, 021, 022, 023

Appendix D. Protected Outputs (Informative)

For some Protection_after_Reception values, rendering is allowed over appropriately protected output links. The definition of which protected link technologies are allowed is typically a deployment or trust authority issue and outside the scope of this specification.

Examples for such links could include, but are not restricted to, DTCP (DTCP-IP, DTCP-BT-Audio, DTCP-BT-Video, DTCP-USB, DTCP-1394), HDCP, BT-AD2P, BT-HFP.

Appendix E. Additional BCAST Response Parameter/Data for the MBMS Security Context (Normative)

Additional Parameters and Data are defined for BCAST to the MBMS security context response. This parameters and data are used in case of failure in the processing of MTK Generation Mode or MSK Update Mode for

- security_policy_extension
- parental control
- location based restriction

E.1 MTK Generation Mode

E.1.1 OMA BCAST operation response: security_policy_extension operation

Byte(s)	Description	Coding	Length
1	MBMS operation response Data Object tag ('53')	As defined in TS 31.101 [11] for BER-TLV data object	1
2 to 1+A bytes (A ≤ 4)	MBMS operation response Data Object length (L)	As defined in TS 31.101 [11] for BER-TLV data object	A
A+2	OMA BCAST operation response tag = 'DF'	Code to be defined by 3GPP CT6	1
(A+2)+1	Security_policy_extension operation response Data Object tag ('80')		1
(A+2)+2 to ((A+2)+1)+B (B ≤ 4)	Security_policy_extension operation response Data Object length (L)		B
((A+2)+ 1)+B +1	Status code		1

Status code coding:

- 0x01: lack of credit in the user purse
- 0x02: lack of credit in the service purse
- 0x03: Play_back counter invalid or equal to zero

E.1.2 OMA BCAST operation response: Parental control operation

Byte(s)	Description	Coding	Length
1	MBMS operation response Data Object tag ('53')	As defined in TS 31.101 [11] for BER-TLV data object	1
2 to 1+A bytes (A ≤ 4)	MBMS operation response Data Object length (L)	As defined in TS 31.101 [11] for BER-TLV data object	A
A+2	OMA BCAST operation response tag ('DF')	Code to be defined by 3GPP CT6	1
(A+2) +1	Parental control operation response Data Object tag ('81')		1
((A+2) +2) to ((A+2) +1)+B bytes (B ≤ 4)	Parental control operation response Data Object length (L)		B
((A+2) +1)+B +1	Status code		1
((A+2) +1)+B +2	Key reference of the second application PIN defined for the parental control		1
((A+2) +1)+B +3	Rating_type for the current program		1
((A+2) +1)+B +4	Level_granted value for the current rating_type		1

Status code coding:

- 0x00: User not authorized
- 0x01: PINCODE required

- 0x02: PINCODE not initialized

Key reference of the second application PIN coding:

The key reference are defined in the [ETSI TS 102.221]. The PIN used for the parental control SHALL be a second application PIN.

The values of the key references are '81' to '88'

Rating_Type for the current program coding:

This field indicates the rating_type of the current program. See Table 31 in Section 7.1 for the coding of this field.

Level_granted value for the current rating_type coding:

This field indicates the level_granted value for the rating_type of the current program. See Table 31 in Section 7.1 (rating_value column) for the coding of this field.

E.1.3 OMA BCAST operation response: Location based restriction operation

Byte(s)	Description	Coding	Length
1	MBMS operation response Data Object tag ('53')	As defined in TS 31.101 [11] for BER-TLV data object	1
2 to 1+A bytes (A ≤ 4)	MBMS operation response Data Object length (L)	As defined in TS 31.101 [11] for BER-TLV data object	A
A+2	OMA BCAST operation response tag ('DF')	Code to be defined by 3GPP CT6	1
(A+2) +1	Location based restriction operation response Data Object tag ('82')		1
((A+2) +2) to ((A+2) +1)+B bytes (B ≤ 4)	Location based restriction operation response Data Object length (L)		B
((A+2) +1)+B +1	Status code		1

Status code coding:

- 0x00: Blackout
- 0x01: need specific permissions

E.2 MSK Update Mode

E.2.1 OMA BCAST operation response: security_policy_extension operation

Byte(s)	Description	Coding	Length
1	MBMS operation response Data Object tag ('53')	As defined in TS 31.101 [11] for BER-TLV data object	1
2 to 1+A bytes (A ≤ 4)	MBMS operation response Data Object length (L)	As defined in TS 31.101 [11] for BER-TLV data object	A
A+2	"OMA_BCAST operation" tag = 'DF'		1
(A+2)+1	Security_policy_extension operation response Data Object tag ('80')		1
(A+2)+2 to ((A+2)+1)+B (B ≤ 4)	Security_policy_extension operation response Data Object length (L)		B
((A+2)+1)+B +1	Status code		1
((A+2)+1)+B +2	Number of rating_types (n)		1
	For (i=0; i < n; i++) {		
((A+2)+1)+B +3	Rating_type		1
((A+2)+1)+B +4	Level_granted		1
	}		

((A+2)+1)+B +3 +(2*n) to ((A+2)+1)+B +3 +(2*n) +L- (2+2*n)	MIKEY message (see note 1)		L-(2+2*n)
NOTE 1: Parameter present if a MIKEY verification message is returned.			

Status code coding:

- 0x00: success
- 0x01: lack of credit in the user purse
- 0x02: lack of credit in the service purse

Number_of_rating_types

This field indicates the number of rating_types received in the current LTKM

Rating_type:

This field indicates the rating_types stored in the Smartcard after the update caused by the received LTKM. See Table 31 in Section 7.1 for the coding of this field.

Level_granted:

This field indicates the level_granted value stored in the Smartcard after the update caused by the received LTKM and associated to the rating_type above. See Table 31 in Section 7.1 (rating_value column) for the coding of this field.