Project Number:	**Contract Number: INFSO-ICT-*224287***
Project acronym:	**VITAL++**
Project Title:	**Embedding P2P Technology in Next Generation Networks: A New Communication Paradigm & Experimentation Infrastructure**
Title of Report	**Mechanisms for Security and Content Privacy**

Instrument:	STREP
Theme:	ICT-2-1.6
Report Due:	M21
Report Delivered:	M23
Lead Contractor for this deliverable:	WIT
Contributors to this deliverable:	Shane Dempsey (WIT), Kabir Ahmed (WIT), Stephen Garvey (WIT), Jens Fiedler (Fraunhofer), David Florez Rodriguez (TID), Apostolos P. Fournaris (UoP)
Estimated person Months:	9
Start date of project:	1st June 2008
Project duration	30 months
Revision:	Version 1.0
Dissemination Level:	PU - Public

This page intentionally blank

1 Table of Contents

2 List of Figures

3 Document History

The aim of this deliverable is to provide relevant information regarding the Peer-to-Peer client evaluation and market overview

Revision Month	Filename version	Summary of Changes
M7	V0.1	Initial Template & Table of Contents
M18	V0.2	Revision to Table of Contents and document structure
M19	V0.30	TID - IMS Security State of the Art
M20	V0.35	CS-SA description and MMI protocol description
M22	V0.4	Input from Fokus & UoP
M22	V0.5	Review and revision by RBB
M23	V0.55	Review and enhancements by WIT
M23	V1.0	Updated to v1.0 for external review.

4 Executive summary

In this document we describe in the context of P2P services and content of task 4.2, a number of scenarios pertaining to the applicability of security in the communication of the user with the server and its application peers. The state of the art in IMS and P2P security is reviewed. Security challenges of the Vital++ platform are considered and solutions are proposed.

Other important issues to be tackled are the use of a DRM concept for content protection/privacy and the user programmable privacy in the exposure of user-generated content in the network and its use by other community members.

5 Introduction

In this document we describe in the context of P2P-IMS services, specifically those being designed and implemented within Task 4.2, a number of scenarios pertaining to the applicability of security in the communication of the user with the server and its application peers. Other important issues to be tackled are the use of a DRM concept for content protection/privacy and the user programmable privacy in the exposure of user-generated content in the network and its use by other community members.

Prior to presenting the Vital++ solutions, we will review the state of the art in security mechanisms in IMS and Peer-to-peer networks. We will also describe generic mechanisms for content protection or Digital Rights Management as it is commonly known.

Scenarios describing the use of the Vital++ Content Security Sub-architecture are presented and its rights negotiation protocol "Mother May I" (MMI) is introduced and detailed.

6 State of the Art in IMS and P2P Security

6.1 IMS Security Overview

One of the basic concepts of IMS[1] is the separation between the Signal Plane and the Media Plane. Briefly, the media connections are directly established between the User Agents (UA), whereas the messages required to associate both UA will traverse a series of defined entities, P-CSCF, I-CSCF, AS… et al, as depicted in figure 1, with very precise responsibilities for that exchange of commands.

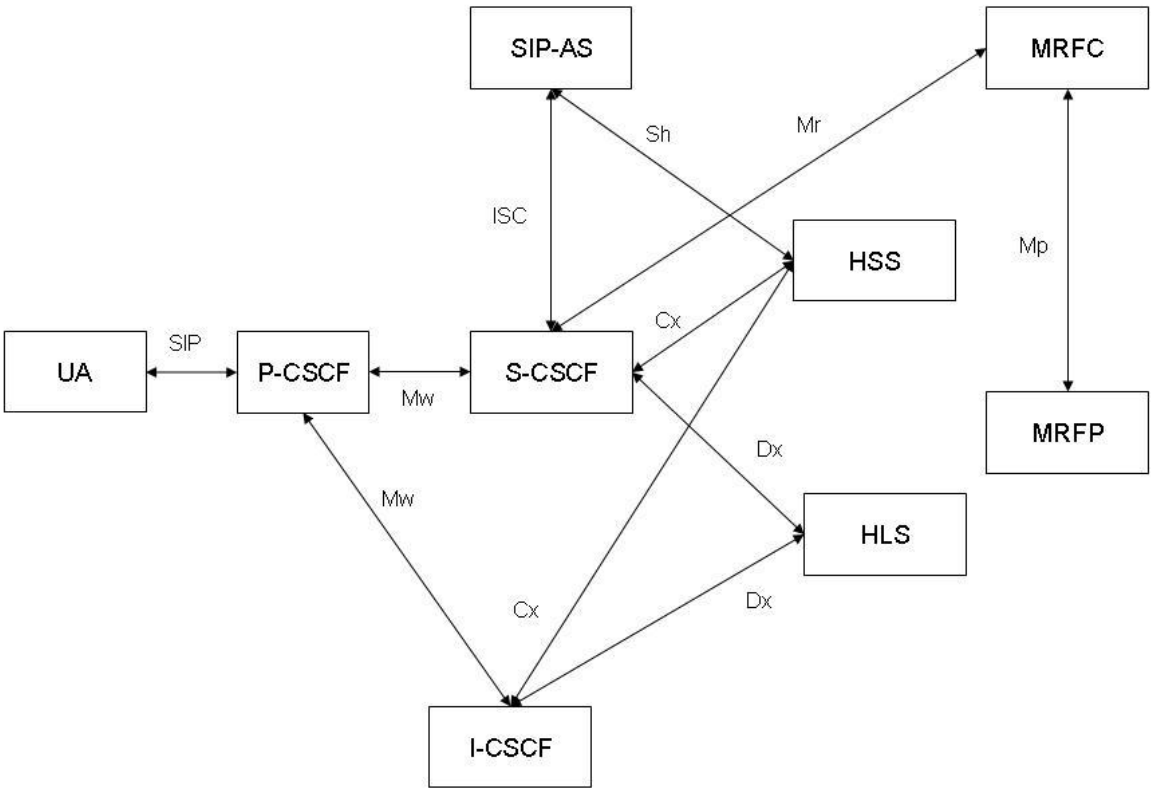

Figure 1 – Simplified depiction of an IMS System

[1] The 3G IP Multimedia System (IMS), Gonfalon Camarilla and Miguel A. García-Martin.

Therefore, in order to analyse security in the context of IMS it is necessary to deal with both areas separately because they will require different means and procedures.

6.1.1 Signal Plane Security

In order to provide security for the Signal Plane, IMS requires two IPsec[2] connections to be set up between the User Agent and the P-CSCF. Those two connections behave in different ways depending on the final transport protocol. If TCP is used, the response for a request is sent by the same connection through which it was received, being one connection allocated for the UA's requests and the other for the P-CSCF's, while in the case of UDP the response to a command received by one connection is sent by the other, being allocated one connection for the traffic from the P-CSCF to the UA, and the other one for the traffic in the opposite direction.

The establishment of an IPsec connection requires that both end-points exchange and agree a series of parameters[3]. This dialogue between the UA and the P-CSCF is carried out in the context of a standard SIP REGISTER procedure, which is depicted in the following figure.

[2] RFC 2401 Security Architecture for the Internet Protocol

[3] RFC 3329 Security Mechanism Agreement for the Session Initiation Protocol

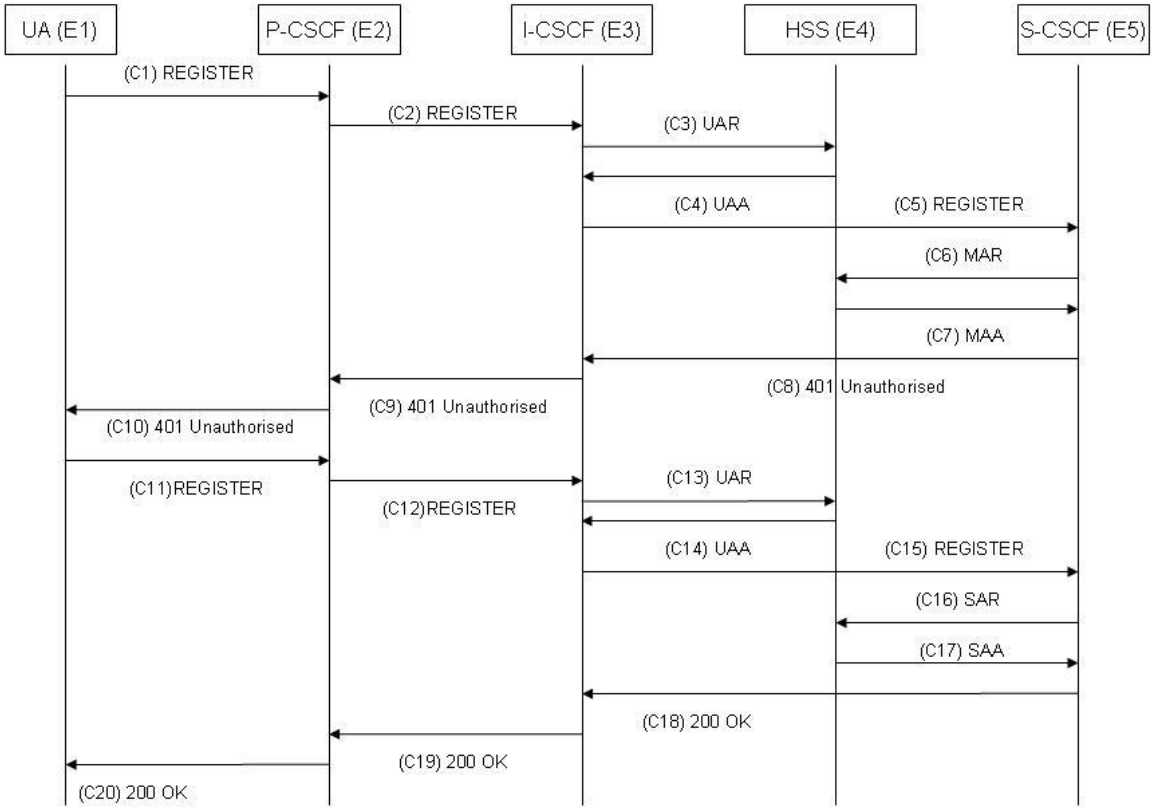

Figure 2 – SIP REGISTER

When the UA sends its initial REGISTER(C1), it contains a Security-Client Header Field, specifying the identifiers for the encryption mechanisms and algorithms to be used, together with the SPIs (Security Parameter Indexes) and ports for both IPsec connections.

Once the S-CSCF has obtained the required Authentication Vectors for this user by means of a DIAMETER MAR/MAA exchange, it sends a response 401 Unauthorised (C8), in order to prompt the UA to send its authentication data. This response contains a WWW-Authenticate header, one of its paramaters being a nonce that encrypts the Integrity Check (IK) and Session Check (CK) to be used in the IPsec connection, among a number of other parameters.

This response, 401 Unauthorised, sent by the S-CSCF also contains the IK and CK as independent headers, for the P-CSCF to be aware of the required parameters for the IPsec connections. That entity will not relay them in the 401 Unauthorised (C10) sent to the UA, in order to avoid them being sneaked by malicious agents. Those parameters travel, though, inside the nonce generated by the P-CSCF, that have been encrypted with the secret key this entity shares with the UA.

The P-CSCF includes a new Security-Server header in the 401 Unauthorised sent to the UA in order to make aware that entity of its supported encryption mechanisms and algorithms, along with the SPI (Security Parameter Index) and ports for both IPsec connections. It also sends, in the case that a number of encryption mechanisms are available, a priority number to hint at the preferred one.

Once the 401 (C10) has been received by the UA, both sides, UA and P-CSCF, have the required information to establish the IPSec connections and the next REGISTER (C11) and following SIP commands will be sent by using those secure channels. This first REGISTER on the secure channel contains a Security-Verify header that replicates the contents received in the Security-Server Header of the 401 Unauthorised (C10), so that the P-CSCF can be sure that the REGISTER comes from a legitimate source.

The different IMS entities (P-CSFC, I-CSCF, S-CSCF, AS, etc) inside a single IMS domain establish IPsec connections in order to exchange SIP messages, those secure channels being called the Zb interface.

If different IMS domains require the exchange of SIP messages between them, they will resort to the use of SEGs (SEcurity Gateways). Those SEGs will establish an IPsec connection using ESP (Encapsulated Security Payloads)[4] running in tunnel mode. Those connections are created and maintained by using IKE (Internet Key Exchange)[5] and are commonly known as the Za interface.

The entities inside an IMS domain consider their assigned SEGs as another entity and therefore will use the Zb interface to interact with them.

6.1.2 Media Plane Security

Regarding the Media Plane, since the Media is transmitted directly between the UAs, IMS does not make any assumption on the mechanism/algorithms to be used.

How both UAs are going to exchange their data is negotiated in the SDP (Session Description Protocol) attached as a body to the INVITE-OK-ACK exchange. This is commonly carried out by means of the m= lines of the SDP, where the protocols used by the UA and their assigned ports are specified.

Therefore, an IMS UA could use standard protocols like TLS[6] for data that do not have real-time requirements and SRTP[7] for those that have. For instance, a commercial SIP Softphone like Mercuro[8] supports SSL, TLS and IPsec.

[4] RFC 2406 IP Encapsulated Security Payload

[5] RFC 2409 Internet Key Exchange (IKE)

[6] RFC 5246 The Transport Layer Security (TLS) Version 1.2

[7] RFC 3711 The Secure Real Time Transport Protocol (SRTP)

[8] http://www.mercuro.net/

6.2 P2P Security Overview

Peer-to-peer (P2P) networks have become increasingly popular in recent years due to their distributed and dynamically scalable nature. Although this technology is not yet fully mature, a wide range of possible applications are under development by several enterprise vendors in order to take advantage of this extremely useful network tool. So, P2P networks, initially developed for file-sharing, are expected to be used in more sophisticated applications like Voice over IP or live video streaming and Video on Demand. Capitalizing this trend, researchers have defined structured and unstructured P2P networks providing a self-organizing substrate for large-scale P2P applications. In most such approaches, the main P2P problems are related to bandwidth management, context management and network scalability. However, due to the extending variety of P2P network applications P2P security has also been elevated into a serious issue.

Making a P2P network secure is a significant challenge. Since the P2P network was not originally designed to withstand an adversary attack, it can be easily compromised. A series of security issues need to be addressed if a P2P network is to be called secure. Among the most important issues is the protection of the P2P content against unauthorized use, the protection of the user's identity and privacy of the transmitted user messages from the rest of the P2P overlay environment. The first issue is addressed by employing techniques that involve encryption of the content data in such a way that only the legitimate content receiver during a content distribution operation can access the data. The second and third issue is addressed by employing appropriate techniques, methodologies and protocols that allow complete privacy to the user's node transactions and identity. This action is achieved through anonymizing a user's message exchanges and communications which could reveal sensitive user data or describe a user's habits, characteristics and preferences. In the following subsections, existing solutions on the above issues are highlighted and discussed.

6.2.1 Content Encryption

Among the main achievements of P2P technology is the free and unhindered distribution and management of specific content. However, content distribution can be subject to various attacks that aim to eavesdrop and obtain the content data without the consensus of their legitimate issuer. The problem of securing P2P content during its transmission inside the P2P overlay is very serious and is vital in systems where content is subject to copyright laws or is subject to specific billing policies. In such P2P systems content protection through proper encryption mechanisms is very important.

Content encryption is based on well known cryptographic schemes like public key, symmetric key or stream ciphers, but it is not restricted only to those solutions. According to the type of content to be distributed in the P2P overlay, the encryption technique may vary and can include watermarking, lightweight encryption and steganography. Therefore, there is a wide variety of content encryption schemes that aim at specific types of P2P networks like file sharing, streaming data and/or publish/subscribe networks.

There exist encryption protocols sitting on top of a P2P network like QUIP[9] or Eventguard[10] which require the presence of special-purpose, server-like structures, called key server, for key management. In these protocols secure channels based on content related keys are constructed whenever a specific content item is to be distributed in the P2P network. The key servers are responsible for storing the content's key and upon request from authorized node users release those keys encrypted under the node's public key. Transmission of that key enables the communication of the requested content through secure channel based on symmetric key encryption. These content encryption protocols seem to be easily applicable to publish/subscribe P2P networks since the key servers can be associated with payment mechanisms for accounting and authorization management. A key server can separate subscribers from publishers. Each publisher can set the price for publishing his content and the key server provides the appropriate keys to both publisher and subscriber for this content. The problems arising from these methodologies are related to the computational complexity of encryption/decryption of every data packet reaching a node.

Other approaches in content encryption try to avoid the above problem by encrypting only parts of the shared content[11]. They use partial encryption algorithms applied to the content by breaking it into several fragments and then encrypting only some of them. As a result, the whole content cannot be retrieved until the encrypted fragments are decrypted efficiently. This technique minimizes the number of encryptions per content and thus greatly reduces the computational cost for securing some content data. Partial data encryption can be used in multimedia content management where the media data can be partitioned according to various aspects like encoding layers, coding parameters, object background or data blocks.

[9] Corman, A., Schachte, P. and Teague, V. (2007). QUIP: A Protocol For Securing Content in Peer-To-Peer Publish/Subscribe Overlay Networks. In Proc. Thirtieth Australasian Computer Science Conference (ACSC2007), Ballarat Australia. CRPIT, 62. Dobbie, G., Ed. ACS. 35-40.

[10] Srivatsa, M. & Liu, L., Securing publishsubscribe overlay services with eventguard, in `Proceedings of the 12th ACM conference on Computer and communications security' (2005)

[11] X. Liu and A. M. Eskicioglu, "Selective Encryption of Multimedia Content in Distribution Networks: Challenges and New Directions," Proc. 2nd IASTED Int. Conf. on Comm., Internet, and Info. Technol., pp. 527-533, Scottsdale, AZ, November 17-19, 2003.

A different approach on securing stored content in a P2P network is through secret sharing schemes. In that case, the content publisher encrypts the content data with a key K and splits the key into several fragments so that only a specific number of those fragments, called threshold, can lead to reconstruction of the key and the decryption of the content. Each server of the P2P network encrypts one of the key fragments along with the content's block.

6.2.2 Signalling Privacy & Obfuscation

Most P2P networks are designed to be open to the public so that anyone wishing to gain permission to the network can easily obtain it. This does not exclude P2P networks where authentication is required since it suffices that a new peer complies with the network's policies to obtain a unique identity and become part of that network. However, even an authenticated peer, considered a legitimate part of the network, cannot be excluded from being subject to or participating in eavesdropping or surveillance of other peers' behaviour, characteristics and content exchanges. Network surveillance is increasingly popular in private and governmental organizations and can be used for political censorship. ISP's try to limit or block existing P2P network functionalities in their subscriber overlays by monitoring their users' behaviour and blocking their access to the P2P network resources. Such actions, harming the peers' right to privacy, constitute a problem that is still not widely addressed in real P2P networks thus generating an urgent need for privacy enhancing systems that are both effective and practical.

Achieving a high level of privacy in a large and highly dynamic P2P network is a demanding task that is focused on achieving privacy on a peer's identity. Content encryption, as described in the previous subsection can be used to obfuscate the data exchanges of a peer, thus making it difficult to generate surveillance background of this peer's habits in the network However, this action is not enough. While serious efforts have been made to conceal data with content encryption techniques, the message exchanges in a P2P network are sufficient to reveal information about one or more users' activities. For example, using only the connection patterns gathered during a one-month period (comprising a stable population of 10,000 BitTorrent users), it has been found that communities of users can be extracted sharing interest in the same content. As a result, an efficient privacy-enhanced P2P network must be able to support, apart from content encryption, also signal and connection obfuscation. This approach leads to the solution of P2P networks supporting user anonymity.

Anonymity ensures that a user may use a resource or service without disclosing the user's identity. Thus, the crucial function of anonymity is to protect the user from information leakage concerning their personal, private and sensitive data like the user name, ID, IP address, during communication with others. Anonymity in P2P systems includes publishing anonymity, sending

anonymity and receiving anonymity. Publishing anonymity refers to the creation or publishing of some information content without revealing who created this material. This attribute is also called censorship resistance. Sender/receiver anonymity protects P2P content senders/receivers from being exposed to other entities during message delivery. Matching the above two requirements provides complete anonymity which is enhanced using encryption in message exchanges. Thus, the user's privacy in P2P systems is ensured by adopting cryptographic techniques like hash functions, random numbers, public key or symmetric key approaches and secret sharing schemes. Those techniques are employed to provide confidentiality, message integrity and anonymous delivery.

Existing P2P anonymity and privacy approaches are based on message transmission anonymity and can be divided into three basic categories:[9] unimessage based, split message-based and replicated message-based. In the unimessage based approach the message is sent as a whole while in the split message-based approach it is fragmented into various pieces and the receiver can collect the various pieces and recover the original message. In the replicated message-based approach, multiple copies of each message are generated and spread in the whole system.

6.2.2.1 Unimessage Based Approach

Most anonymity systems follow the unimessage based approach and they achieve their goal by encrypting P2P messages and assigning a single anonymous path for the message delivery. So, the main objective of this approach is to hide the message transmission path with in the P2P graph. A widely used version of this approach is the *Mix or Onion routing mechanism*. It assumes that a sender A wants to transmit a message *m* to the receiver B through a path that involves nodes N_i where *i* is an integer with a value between 0 to the number of hops to reach B. We symbolize this path as $A \rightarrow N_0 \rightarrow N_1 \rightarrow \ldots N_i \ldots \rightarrow B$. This path is hidden through the use of a layer–encrypted data structure following public key cryptographic principles. The objective of this methodology is that each node in the path only knows its successor IP address and has no knowledge about A's or B's IP address. The receiver node B can retrieve that message but not the IP address of the node that sent it (node A). The packet structure is organized as follows: The innermost layer includes the IP address (or original ID) of B and the original message encrypted using B's public key. Node A then wraps this layer by encrypting it using the public key of node B's predecessor. Along the reversed sequence of the node in the path, node A keeps wrapping the packet following the described layer-encrypted pattern until the node that is one hop away from A is reached (node N_0). The final packet to be transmitted is structured in layers using encryption with the public key of each node along the path from node A to node B. After completing the above operation, node A transmits the package to the next node in the path. During transmission, each intermediate

node decrypts the received package using its private key to receive an inner layer of the package and a successor's node IP address (or unique ID). Thus, the only information that this node can retrieve from the transmission is where the package needs to be further forwarded because the package itself is still encrypted (with the remaining nodes' keys in the path) and therefore useless. When the package reaches the receiver node B it will be decrypted using B's private key and the original message will be retrieved. Using the above described approach, the path remains hidden from all nodes, including B and all the intermediate nodes except A and as a result anonymity is achieved.

There is a wide variety of existing proposals using Onion routing mechanism and unimessage based approach in general and these proposals can be further divided into three categories: fundamental path based, probability based and mimic traffic enhanced approach.

Fundamental Path Based scheme directly employs Mix or Onion routing techniques for anonymity. It is the most widely used unimessage based anonymity approach and many researchers have proposed solutions that follow this scheme. The APFS[9] solution introduces a special node called bootstrapping node, as a coordinator of the anonymous path construction. This bootstrapping node provides a list of online nodes that are available to construct anonymous paths. Each node constructs an onion path pointing to another node, called tail node, which acts as an anonymous transferring agent. Indexing of the system is done by some nodes that are assigned server duties and post their tail nodes to the coordinator. Thus, the client nodes can upload their resource lists and requests to the server node through onion routing paths. This approach can work efficiently in file transfer-retrieval P2P systems.

A more advanced version of onion routing is used in the Tor[9] where instead of using a single layered encryption packet, a multilayer – incremental path construction methodology is followed. The initiator, sender node A, extends the path hop by hop and negotiates session keys with the intermediate nodes on the path. This makes the onion routing more reliable. Tor is able to support TCP based applications. Tor focuses on providing a resilient, usable service for low-latency, interactive Internet tasks such as Web browsing and SSH sessions. However, attempts to integrate and implement Tor into real P2P systems has significantly reduced global Tor performance to the detriment of non-P2P users. Other similar fundamental path based scheme solutions, all using mix or onion routing techniques, are the MorphMix[11] and GAP[12].

Probability based schemes employ probability principles to construct anonymous paths. This is done by choosing intermediate nodes on the paths using probability forwarding and thus strengthens user anonymity.

[12] SwarmScreen: Privacy Through Plausible Deniability in P2P Systems. Northwestern EECS Technical Report. March, 2009.

Characteristic representatives of this scheme are Crowds[13], Shortcut[14] and AP3[15].

Crowds[13] is an anonymous web transaction protocol that can only provide sender anonymity. Each intermediate node has the ability to choose its successor randomly and forward the message or deliver the message to its receiver itself. In order to secure the communication channel, symmetric key encryption is used. As a result, a series of key-related actions are necessary, like key generation, establishment and distribution, resulting in considerable overhead. To solve such problems, Shortcut was introduced[14]. This protocol uses onion routing. For anonymous reply, the sender node establishes an onion-based reply packet and encapsulated in the message query. This packet is an anonymous return path. At the end, the query is probabilistically forwarded in the system. Each node receiving the query either acts as a reply agent for the sender or forwards the query to another node using probabilities. A reply agent node adds his IP (or unique identity) to the query message and forwards it to a randomly chosen neighbour. From this point on, each node receiving the query message, either forwards it to a random node or floods the system based on some probability. In Shortcut, it has been found that the response time is small. AP3 is similar to Crowds but operates on top of the application layer.

Mimic Traffic Enhanced scheme is an improvement on the fundamental path-based approach that is achieved by introducing dummy traffic to the P2P system. These "mimic" signals can help mask the data flow in the P2P network so that an adversary cannot distinguish dummy traffic from real one. The Tarzan system[16] is a characteristic example of this methodology. While onion routing provides a small set of proximity nodes, each Tarzan node involves all other nodes in its proxy set. To achieve that, Tarzan uses gossip based protocol for proxy discovery. In Tarzan, mimic traffic is injected to the communication link to protect real data from eavesdropping. This is done when a node establishes k bidirectional links with k neighbour nodes and all those nodes maintain and balance the mimic traffic following a series of criteria in order to shape this traffic to a time invariant pattern. However, this technique introduces a significant amount of traffic overhead to the P2P system. An enhancement on the Tarzan approach is to generate each mimic packet by splitting the last real packet into two fragments and permuting them in a new

[13] Xiao RY. Survey on anonymity in unstructured peer-to-peer systems. Journal Of Computer Science And Technology 23(4): 660-671 July 2008

[14] Scarlata V, Levine B N, Shields C. Responder anonymity and anonymous peer-to-peer ¯le sharing. In Proc. the 9th Inter- national Conference of Network Protocol (ICNP), Riverside, CA, USA, 2001, pp.272{280

[15] DINGLEDINE, R., MATHEWSON, N., AND SYVERSON, P. Tor: The second-generation onion router. In Proc. of USENIX Security Symposium (2004), pp. 303–320.

[16] Rennhard, Plattner B. Introducing MorphMix: Peer-to-peer based anonymous Internet usage with collusion detection. In Proc. ACM Workshop on Privacy in the Electronic Society ,Washington DC, USA, 2002, pp.91-102.

package. This technique seems to improve the overall traffic overhead of the P2P overlay. In general, the mimic traffic schemes significantly improve the anonymity of users but introduce a very large amount of additional traffic.

Summarizing the above, the unimessage based approach can provide high level of anonymity since both the messages and the message delivery paths are encrypts and cannot be eavesdropped. However, there are some drawbacks to this approach. The sender node must obtain enough proxies in order to construct an anonymous path. Also, the anonymous paths are not always reliable and are very difficult to be maintained since by definition a P2P system is highly dynamic and nodes always enter and leave the overlay. Finally, the computational cost for the cryptographic operations in each node is significant due to constant encryption-decryption operations (public or private key cryptography) performed for one onion routing like operation

6.2.2.2 Split message based approach

In the split message based approach secret sharing schemes are used in order to achieve anonymity. A secret sharing scheme consists of a secret that is fragmented into a series of pieces called shares that are distributed to individual entities. An entity collects a number of such shares and when reaching a threshold of that number the secret can be recovered. This mechanism in P2P systems is employed in order to achieve publishing user anonymity and is used in order to split the message into different shares. Split message based anonymity systems are FreeHaven[17], SSMP[18] and RR[19].

FreeHaven provides censorship resistance for users and employs IDA (Information Dispersal Algorithm) for secret sharing. A single fragment of a package will not disclose the publisher ID and its content. The Mix technique is used for anonymous communication. In FreeHaven a community of servers is required. Those servers host and exchange fragments with others. The idea behind FreeHaven is that when a provider entity publishes some P2P content it splits it into fragments using IDA, sets a threshold for their reconstruction, marks the fragments with a unique ID, and uploads the shares to one of the servers. Servers publish the received shares and exchange some fragments with other servers. A requester issues a query containing the ID of the desired content to any server. The server floods the request to the system and receives the fragments replied from other responding servers. Anonymity paths are used in order to deliver the fragments. Through the Free Haven infrastructure a high level of reliability is provided due to the existence of

[17] Bennett K, Grotho C. GAP Practical anonymous net- working. In Proc. Privacy Enhancing Technologies Workshop, Germany, 2003, pp.141{160.

[18] Reiter M K, Rubin A D. Crowds: Anonymity for web transactions. ACM Transactions on Information and System Security, 1998, 1(1): 66-92.

[19] Xiao L, Xu Z, Zhang X. Low-cost and reliable mutual anonymity protocols in peer-to-peer networks. IEEE Transactions on Parallel and Distributed Systems (TPDS), 2003, 14(9): 829-840.

redundant fragments. However, both the fragments trading procedure and fragments storage result into significant overheads to a P2P system.

Mutual anonymity can be provided into P2P systems (unstructured usually) by the SSMP solution proposed in Han et al.[20]. In this protocol secret sharing is performed during message issuing (query issuing) and content transmission (downloading). So, when a query is generated by an issuer it is split using a secret sharing scheme into fragments that are transmitted to the issuer's neighbouring nodes. The fragment shares are then flooded into the P2P system. SSMP uses probabilistic flooding where each intermediate node either sends a share to a randomly chosen neighbour or broadcasts the share. Once a node receives enough shares (depending on the secret sharing scheme threshold) it recovers the query and transmits it to the system. The SSMP approach has a small computational overhead because only secret sharing is used and not public key encryption/decryption but it suffers from a large traffic overhead due to its flooding mechanism.

Another solution in split message based approach is the Rumor Riding (RR)[21] protocol that employs a random walk scheme. RR offers mutual anonymity and does not need to construct anonymous paths based on public key cryptography but uses symmetric key cryptographic algorithms (AES) for encryption. As a result, the computational overhead of anonymity paths is greatly reduced. RR drives both the cipher and key of a message randomly walking in the P2P systems. The cipher and key are called cipher rumor and key rumor, respectively. RR allows each peer to adaptively determine the length of rumors to guarantee a pair of rumors to meet with a high probability. The peer that receives a pair of rumors can recover the original query. This peer then floods the query on behalf of the unknown initiating peer. Message receipt works in a similar way. RR has many benefits, including low computational overhead due to the use of symmetric key cryptography instead of public key cryptography, low traffic overhead since the rumor based scheme has fewer signals compared to the secret sharing scheme and high anonymity guarantee.

In general, the split message based approach offers very promising solutions in anonymity through the use of secret sharing or random walk schemes. However, it still suffers from some serious efficiency issues due to the flooding mechanisms and anonymous path construction.

6.2.2.3 Replicated Message Based Approach

This approach is based on the notion that if a message is transmitted in an extremely noisy environment it is highly unlikely that this message will be

[20] Han J, Liu Y, Xiao L, Xiao R, Ni L M. A mutual anonymous peer-to-peer protocol design. In Proc. the 19th International Parallel & Distributed Processing Symposium (IEEE IPDPS), Denver, CA, USA, 2005, p.68.1.

[21] Han J, Liu Y. Rumor riding: Anonymizing unstructured peer- to-peer systems. In Proc. IEEE International Conference on Network Protocols (ICNP), Santa Barbara, California, 2006, pp.22-31.

eavesdropped successfully. Thus, in replicated message based approach, broadcast and multicast are implemented in order to achieve anonymity. The content of a message is hidden by encrypting them with the receiver's public key so that only the receiver can read it. The message is then replicated in several copies and is transmitted in a broadcast or multicast way. This procedure is adopted in the P^5 system[22] where some participants form a broadcasting group and send fixed length packages in a fixed data rate. Following the replicated message based approach each message in the group is encrypted with the receiver's public key and broadcasted to all the group's participants. To disable trace analysis by potential attackers, P^5 introduces noise packets to keep a fixed transmitting rate for each user. P^5 also designs a clever hierarchical binary tree to partition all users into different broadcasting groups. Each node of the binary tree refers to a broadcasting group. Any message sent to a group is forwarded to all members of this group, all groups in its subtree, and all upstream groups tracing back to the root.

The practicality of the replicated message approach is questionable due to the high traffic overhead introduced by its broadcasting mechanism and the high computational overhead due to the constant encryptions-decryptions required for this approach.

6.2.2.4 Privacy in Structured P2P systems.

Achieving privacy specifically for structured P2P networks is not an easy task since the network topology by default can be known. The use of DHT, a widely accepted practice in structured P2P systems, enables an attacker to pre-calculate the location of the content to be published. This way information can be extracted about the user issuing or receiving that content. Additionally, the P2P network mapping through DHT enables the pinpointing of a particular node within the P2P overlay. Thus, anonymity is a challenging problem. The approaches described in the previous subsections are applied in structured P2P networks but they might not be adequate to provide the same high privacy and obfuscation level as in unstructured P2P networks. There are a series of solutions on structured network anonymity enhancement techniques that are based on the aforementioned approaches but they do not provide very optimistic results. So, privacy in structured P2P networks constitutes a still open research topic.

[22] Sherwood R, Bhattacharjee B, Srinivasan A. P5: A protocol for scalable anonymous communication. In Proc. IEEE Symposium on Security and Privacy, Oakland, California, USA, 2002, pp.58-70.

7 Vital++ Security Mechanisms

The Vital++ security mechanisms focus mainly on the communication between peers, as the security mechanisms for the client-to-IMS/NGN communication already exist. We will depict the relevant P2P-Security issues and how they are solved in the project. We will also give some examples, how the security mechanisms can be used to create higher level use cases and scenarios.

7.1 Security Challenges for the Vital++ Platform

The communication partners in the VITAL++ architecture consist of the set of clients and the IMS entry server, the P-CSCF. Thus, there are two main communication channels to be secured, which are the communication between a client and the IMS and the communication between the clients (P2P). The security challenges for the client-server communication have already been addressed by the 3GPP during the standardization process and are therefore not discussed here. For the security challenges between the clients, the following challenges do apply.

- **Sybil Attack**

A malicious Node can join an overlay and impersonate several different identities, i.e. he is present in the overlay with multiple IDs, also in terms of network-address and port number. Other peers may think that they are talking to different peers, but in fact talk to only one single node. This way, an attacker may gain control over a large part of the overlay. With an increasing number of identities, the probability for an honest node to contact the malicious node rises significantly. The malicious node's chances for further misbehaviour (e.g. on routing) also increase, based on the overlay's distribution algorithm and related routing functions. For a reputation based system, which determines the trustworthiness of a node by some kind of vote, such an attack can result in a catastrophe, affirming the malicious node's false identity.

- **Bootstrapping**

When a node enters an overlay, it does so by querying an already known node, which is in the overlay (boot node). This node will provide the new node with information on how to join the overlay (e.g. which other nodes are its new neighbours, etc.) The bootstrapping attack now simply assumes that the boot node is malicious and gives the new node only neighbours, which are already under its (the malicious node's) control. This makes the new node the only node in a private overlay, filled solely with malicious instances of the malicious node. The results are the same as for a Sybil attack with a huge number in malicious identities.

- **Information falsification**

Information falsification is a problem in a P2P system, where information or content is stored on foreign nodes (e.g. in a DHT). A writer node stores the information in a storage node, which might be malicious. Without security measures, a reader node which requests that information from the storage node might receive information which is different from the originally stored one as the storage node can modify the information arbitrarily. Therefore, sensitive information that must not be corrupted cannot be stored in a distributed way without additional security mechanisms.

- **Eclipse attack**

The purpose of an eclipse attack is to isolate a victim node from the overlay so that no more message routing from or to that node appears. In order to achieve this, the attacker impersonates all possible direct neighbours of the victim in the overlay (comparable to a directed sybil attack, s.a.)

The eclipse attack is only possible, if a malicious node has an influence on its own position in the overlay, i.e. if its neighbour nodes cannot verify and reject its request to join at that specific point in the overlay.

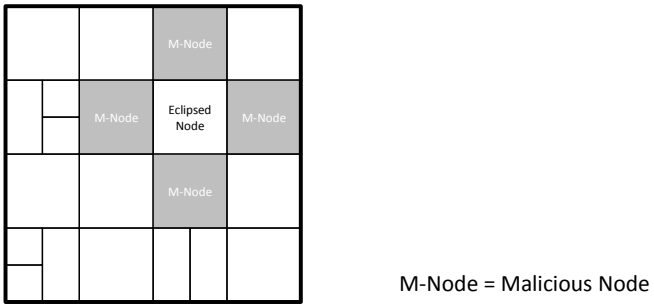

Figure 3 - Eclipse Attack in a 2D Content Addressable Network

The eclipse attack is different from the bootstrapping attack as it can occur while the victim node is already part of a healthy P2P overlay, whereas the bootstrapping attack must occur during the joining phase of the victim node.

The following table gives an overview over the relevant security challenges and how they are addressed in VITAL++.

Challenge	Solution
Sybil Attack	Overlays are created by a central instance and identities can be verified by using VITAL++ certificates and signature.

Bootstrapping Attack	Overlays are created by a central instance.
Information Falsification	VITAL++ digital certificates and signatures are applied.
Eclipse Attack	Overlays are created by a central instance so malicious nodes cannot claim arbitrary positions in an overlay.

7.2 Communications Channel Protection in P2P-IMS

7.2.1 Securing the Client - NGN/IMS Link

7.2.2 Securing the P2P Links

The P2P-Authentication sub-architecture works with certificates, i.e. digitally signed chunks of data, which describe an entity and its properties, e.g. identity and access rights. In the VITAL++ scope, three levels of certificates are distinguished, as shown in the following table.

Root Certificate	• Self-signed. • Pre-installed in every client and P2P-Authentication server module.
Server Certificate	• Signed by Root-CA. • Pre-installed in every P2P-Authentication server module. • Describes the identity of the server domain and its public key. • Acquired by each client during registration.
Client Certificate	• Signed by a P2P-Authentication server module on request. • Describes the identity of the client and its public key.

Table 1 - VITAL++ Certificate Types

Finally, each client is equipped with these three certificates, which allow it to perform all authenticity transactions and checks as explained in then following sub-sections.

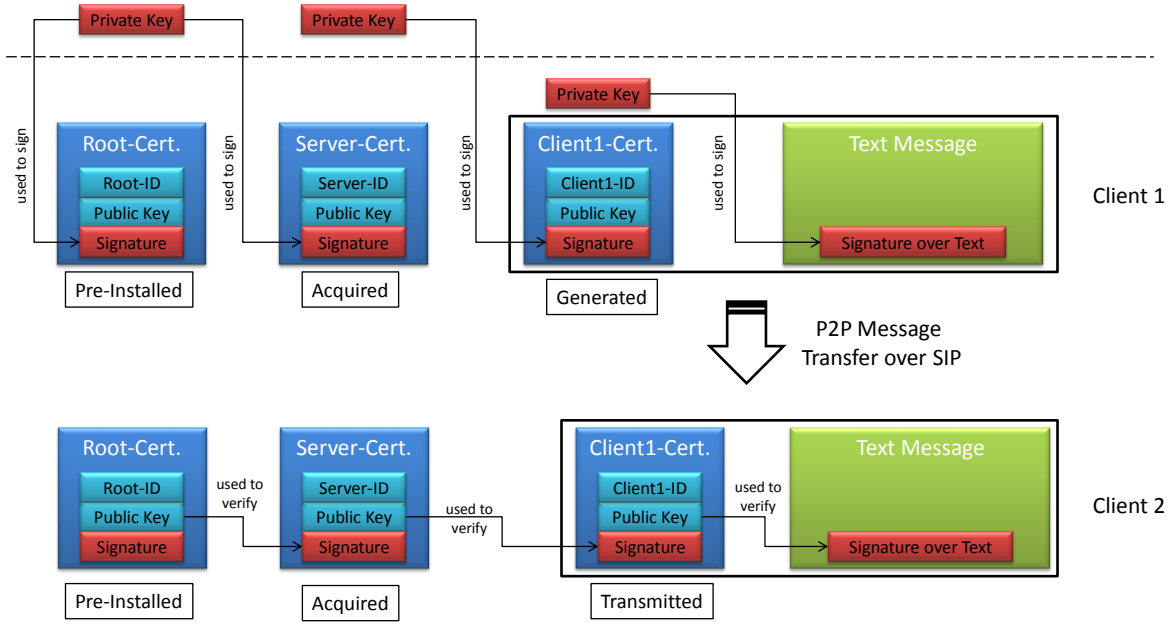

Figure 4 - Relation between Certificates and Messages

The relation between the certificates and their use in order to enable authentic message exchange is depicted in Figure 4.

The relevant P2P-Authentication transactions are described in detail in the deliverables 3.1 and 3.2. The following figure gives an overview over the transactions.

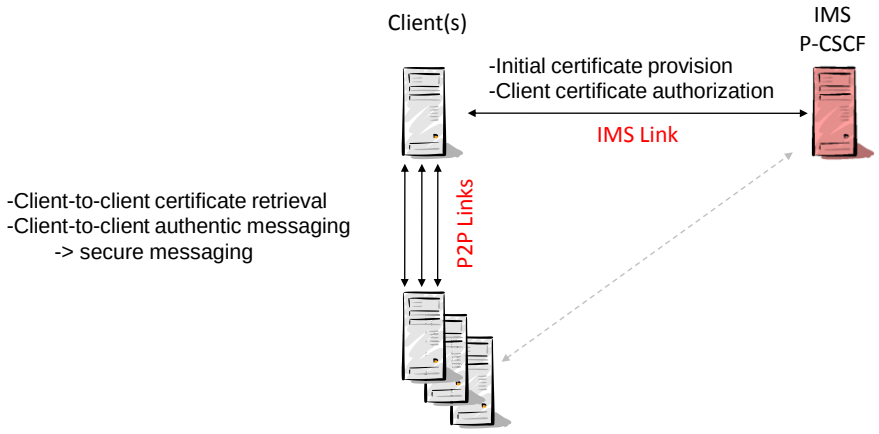

Figure 5 - VITAL++ P2P Authentication Transactions

By using these transactions, clients can send signed messages and can verify message signatures, which are received.

7.3 Usage Scenarios

In this chapter we introduce some scenarios, which do profit from the VITAL++ security mechanisms. The scenarios are just examples to highlight the usability of the mechanisms.

7.3.1 Secure messaging

In deliverable 3.2 we have depicted authentic message exchange. As these messages are still in plain text, they can be read and understood by intermediate nodes by applying the same mechanism as introduced in the deliverable 3.1, section 8.1.4.3 (Client certificate authorization).

In order to establish a common secret key for symmetric encryption of messages, the Diffie-Hellman algorithm for key-agreement (DH) can be applied. The originator of an encrypted communication creates the DH-Parameters g,p and A, which he signs with his private key before he sends them to the communication partner. Signing these parameters is necessary to avoid a man-in-the-middle attack on the DH key-agreement. The partner then computes the parameters B and K, and sends B back to the originator, also signed with his private key. The originator then completes his computation of K. So both entities have the same secret knowledge K, which can be used for further encryption of the message exchange. The Diffie-Hellman algorithm is explained in more detail in annex B.

After this step, both communication partners can exchange encrypted messages, which can additionally be signed in order to detect possible message falsification during transmission.

7.3.2 Authentic media distribution

The current model of content protection is based on a digital rights management system, implemented in the scope of the VITAL++ content protection sub-architecture. Content distribution is based on a P2P overlay, which means that peers receive content from other peers. All peers will use the same symmetric key to decrypt the protected media stream. This allows them to falsify content by simply exchanging frames and encrypting them with the same symmetric key before forwarding them downstream to other peers.

An approach to solve this problem would be the following algorithm:

1. When the media source publishes its stream, it marks it as “signed”, so that receivers know that they can expect signatures to use for authenticity checking. Note: receivers also know the public identity of the media source from the content lookup.

2. The source of the media stream signs either each media frame or computes a hash of each frame and signs this.
3. The source sends the signature along with the media frame downstream.
4. Each node that receives a media frame and a signature has to forward both downstream, according to the distribution schema for this overlay.
5. A node which receives a media frame will also receive a signature and can verify the authenticity of the media frame and decide to display it or not or generate a user alert.
6. A node which receives only the media frame but not the signature can detect a flaw in the distribution which allows the same rejecting reactions as in 5).

Receivers now just need to obtain the certificate of the media origin. It would be fatal, if every receiver would query the media source for the certificate, so it must be obtained in a different way.

As every node in the overlay should have the relevant certificate of the source, it can always forward it downstream to other receiving nodes. This way, a receiver can query any of its feeding nodes for the source certificate. The receiver can check the integrity of the received certificate by applying the already explained mechanisms for certificate checking along with the knowledge of the public identity of the source from the original content lookup procedure.

7.3.3 Secure DHT

One of the most important P2P mechanisms is a distributed hash-table (DHT), which allows the storage of nearly arbitrary chunks of data under the assignment of a unique key. The basic operations for a DHT are join, store and get. The join operation is out of scope here, as we assume an already operational overlay. For the store operation, the storing node does not know if the information it is going to store is really related to the writer. Also, for the get operation, the reading node does not know whether the received information is unchanged/genuine. Figure 6 shows these relations.

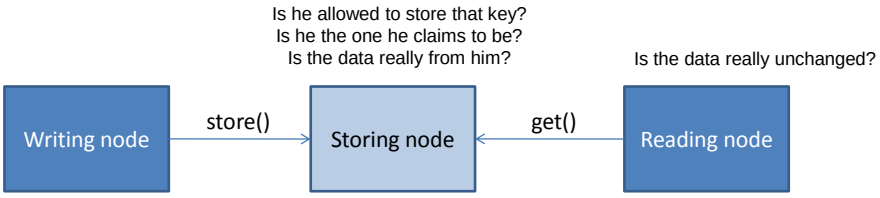

Figure 6 - DHT Security issues

Whenever a node performs a store operation, it will store not only the data chunk, but also a signature over the data chunk and a sequence number. The sequence number can be used by the storing node to detect a replay attack.

The storing node must also store its own certificate in the DHT with the same mechanism to allow an offline-verification of its signature.

When a reading node now retrieves the data chunk, he can verify the integrity of the data by checking the signature.

This way, data stored in a DHT can be secured against falsification by either writing or storing node. Such a DHT can be used e.g. to store SIP contacts or other data objects, which shall be accessible during the absence of the writing node (e.g. playlists or profiles).

8 State of the Art in Content Privacy & Protection

In this chapter, we describe the state of the art in technologies and specifications for Content Protection within the 3GPP and ETSI defined IP Multimedia Subsystem. By content protection we primarily refer to mechanisms for encrypting content where appropriate for the purpose of protecting Vital++ user privacy and also guarding the legal rights of the copyright holders and publishers of the content.

8.1 Content Protection & DRM

Digital Rights Management (DRM) is a system to protect digital assets and control the distribution and usage of those digital assets. This is achieved by the design and deployment of content access-control technologies that can be used by hardware manufacturers, publishers, copyright holders and individuals to impose limitations on the usage of digital content and devices.

8.1.1 DRM Standards

There are a few competing standards and specifications available when implementing a system for digital rights management. Rather than following these specifications absolutely, we have attempted to create a system which is compatible.

8.1.1.1 Open Mobile Alliance

The Open Mobile Alliance[23] DRM specification version 2.0[24] was approved in 2006. Many of the major European wireless carriers announced OMA-compliant content services during that year. OMA DRM 2.0 is supported by major security industry players such as RSA and phone/device vendors such as Nokia, Samsung, Philips and Sony Ericsson.

Each participating device in OMA DRM 2.0 has an individual DRM PKI certificate with a public key, and the corresponding private key. Each Rights Object (RO) is individually protected for one receiving device by encrypting it with the device public key. The RO in turn contains the key that is used to decrypt the media object. Delivery of Rights Objects requires a registration with the Rights Issuer (RI, the entity distributing Rights Objects). During this registration, the device certificate is usually validated against a device blacklist by means of an Online Certificate Status Protocol (OCSP) verification. Thus, devices known to be hacked can be excluded once they try to register with an RI and receive

[23] Open Mobile Alliance (OMA), http://www.openmobilealliance.org/

[24] OMA Digital Rights Management v2.0, http://www.openmobilealliance.org/Technical/release_program/drm_v2_0.aspx

new ROs for content access. Rights are expressed using the XML-based Open Digital Rights Language (ODRL) specification.[25]

8.1.1.2 MPEG Rights Expression Language

The competing standard is the MPEG-21 Rights Expression Language (MPEG REL). This includes the Rights Data Dictionary (RDD), and Intellectual Property Management Protocol (IPMP). ISO officially ratified the MPEG REL in 2004 and other standards bodies decided to adapt it to their needs. The MPEG-21 REL is an XML-based declarative language that declares rights and conditions for authorized distribution and use of any content, resources, or services. It defines the syntax and semantics of a machine interpretable language that can be used to specify rights unambiguously. To support guaranteed end-to-end interoperability, consistency, and reliability between different systems and services, the MPEG-21 REL has the following features:

- richness and extensibility in declaring rights, conditions and obligations;
- ease and persistence in identifying and associating these with digital contents;
- and flexibility in supporting multiple usage/business models.

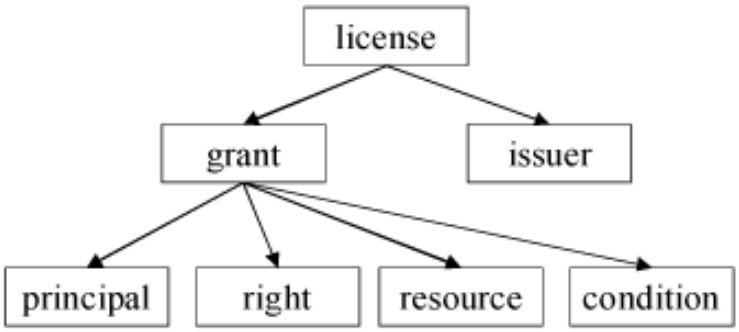

Figure 7 - MPEG Rights Expression Language Data Model

The MPEG-REL data model is shown in Figure 7.[26] A license consists of one or more grants and issuers; a grant contains four basic entities, including principal, right, resource, and condition; an issuer identifies the one who digitally signs the license, and provides details about the issuance.

In a grant, a principal is the identification of a party to whom the right is granted. A right is the "verb" that the granted principals can exercise against some resources under certain conditions, such as play, print, and adapt. A resource is the "object" to which the principals can be granted a right. A resource can be a digital work, a service, or a piece of information that can be owned by a principal. A piece of content or service, together with its metadata

[25] The ODRL Initiative, An Open Rights Language for the Digital Commons, http://odrl.net/

[26] ISO/IEC 21000-5 FCD Part 5: Rights Expression Language, ISO/IEC JTC 1/SC 29/WG 11/N5349, December 2002, Japan.

is referred to as a "Digital Item" and is described using an XML "Digital Item Declaration".

A condition specifies the terms, conditions, and obligations under which the rights can be exercised. For instance, a condition can be the time interval within which the right can be exercised, or it can be a state of holding the prerequisite right that has been issued by a trusty entity. MPEG-REL can be extended with new rights (for a type of resource or application) and new licensing conditions.

8.1.1.3 Open Media Commons

The Open Media Commons is an industrial consortium led by SUN Microsystems to develop an open-source and royalty-free set of media distribution specifications. This is a fundamental difference between the outputs of the Open Media Commons and the MPEG21-REL, which must be licensed from the MPEG consortium for commercial use. DReaM is the Digital Rights Management specification of the Open Media Commons.

The DReaM architecture[27] supports the separation of rights management system components, which is the systematic decoupling of authentication, licensing, rights management and protection technologies. This disintermediation enables the choice and selection of these technologies independent of each other without compromising the integrity of the solution. Key elements of disintermediation in the DReaM architecture include:

- Separation of rights management from the content protection systems. Usage rights are defined in a separate license management system that would be facilitated by DReaM. This allows for the unmodified use of players and DRM clients already installed on devices without inheriting their limitations and shortcomings. As a result, a wide variety of usage models that are not supported by today's system suppliers can be supported in a DReaM disintermediated solution.
- Separation of identity and authentication services from individual hardware devices. Instead of merely authenticating the device on which content can be viewed, identity can be bound to a smart card (e.g., Java Cardtm) for personalization in DRM systems. This allows us to bind content rights to individuals (or roles) instead of devices.

DReaM's strength is that it is designed to work with other DRM technologies such as MPEG-REL or OMA DRM. It acts as an interoperability layer with a straightforward yet comprehensive text-based license negotiation protocol called "Mother May I" (MMI).

[27] Fernando G. [et al.], Project DReaM – An Architectural Overview, SUN Research Labs, September 2005.

DReaM's sister project is the Open Media Stack which is currently active. OMS is attempting to develop a complete media solution, including video, audio, transport, control, and content security.[28]

8.1.2 Super-distribution

Super-distribution is an approach to distributing digital products such as software, videos, and recorded music in which the products are made publicly available and distributed in encrypted form instead of being sold in retail outlets or online shops.[29]

Super-distribution deliberately avoids the difficult problem of copy protection, super-distributed content may be freely copied. Indeed, copying is encouraged as the content owner retains control over the ability to use and modify the content. In order to access the content, the licensor must obtain a decryption key. This makes it an efficient mechanism for content distribution as there are no major economic or technological restrictions on who may publish data.[30] It is compatible with many licensing technologies as the only requirement is support for key distribution.

It is the most widely supported Digital Rights Management mechanism as the Open Mobile Alliance has used the technology in its aforementioned DRM solutions. These have been incorporated into over 280 different mobile phones.

In general a Super-distribution system has the following components:

- A cryptographic wrapper for digital products that cannot be removed and remains in place whenever the product is copied.
- A digital rights management system for tracking usage of the product and assuring that any usage of the product or access to its code conforms to the terms set by the product's owner.
- An arrangement for secure payments from the product's users to its owner.

However, it is conceivable that a super-distribution system could be used to make content available free of charge to some or all users, providing they agree to certain licensing terms. An example of this would be "fair use", described below.

---

[28] Open Media Commons, Open Media Video Stack, Specifications, http://www.openmediacommons.org/collateral/OMS-video-specs.html

[29] Ryoichi M. [et al.], "Superdistribution: The Concept and the Architecture". Transactions of The Institute of Electronics, Information, and Communication Engineers, vol. E73 #7, July 1990.

[30] Ryoichi M [et al.], "Superdistribution: An Electronic Infrastructure for the Economy of the Future". Transactions of the Information Processing Society of Japan, vol.38 #7 July 1997.

8.1.3 Fair Use

Fair use is a US legal doctrine, which permits users to reproduce portions of copyrighted material for various limited scope purposes. It is codified in section 107 of the US Copyright Act. Common examples of fair use include reproducing sections in a long work for educational use or using thumbnail images in a website. While the term "fair use" applies specifically in US law it has analogues in other countries in the concept of "fair dealing" providing exceptions to copyright restrictions for purposes such as education, parody or news broadcast. Countries that recognise the principle of "fair dealing" include Australia, the UK, Ireland and Canada. Germany, like other nations that are signatories to the TRIPS agreement[31] on intellectual property rights, makes similar allowances for fair use[32]. A complete description of fair use and its legal underpinnings is obviously beyond the scope of this deliverable.

The idea of "Fair Use" within a DRM system appears to be counter-intuitive but it has obvious benefits in that it promotes the distribution of content while not diminishing the legal rights of the owners of the intellectual property. It is in principle a very powerful concept when combined with super-distribution. The Open Media Commons suggested mechanisms[33] for the incorporation of Fair Use into a DRM system using super-distribution, which we have adopted. These are:

- Users must assert fair use during the licensing process. Categories of fair-use may be provided (e.g. educational use) but the assertion is left to the user's own judgement.
- The user must use valid user credentials but these may be anonymised to the content provider by a trusted third party. For example, this may be a network operator or 3rd party DRM system provider.
- Credentials may be provided by a variety of AAA mechanisms and providers. For example, a network operator may provide IMS-authenticated credentials. Web-based identity management technologies promoting Single Sign On (SSO) may also be used. Examples include Open-ID and Liberty Alliance.
- Users' audited fair use licensing requests may be made available to a content provider in the event of a subsequent dispute regarding content usage.

[31] World Trade Organisation, Agreement on Trade-Related Aspects of Intellectual Property Rights (TRIPS), http://www.wto.org/english/tratop_e/trips_e/t_agm0_e.htm

[32] Postel, H. "The Fair Use Doctrine in the US American Copyright ACT and Similar Regulations in the German Law", Chicago Kent Journal of Intellectual Property, 2006.

[33] SUN Microsystems Laboratories, Support for Fair Use with Project DReaM (v 1.0 revA), April 2008.

A digital rights management system may make provision for content consumers asserting "fair use" but cannot enforce it since what constitutes fair use is not easily defined and is ultimately a matter for courts to decide. This makes it impossible to automate a system to determine if an activity is indeed fair use.

9 Vital++Content Privacy & Protection Mechanisms

9.1 Content Protection Challenges for Vital++

The Vital++ Content Security (CS) architecture has been designed to enable content providers to control the distribution of their content using a Digital Rights Management technology. Vital++'s DRM system took its requirements from RBB, the consortium's content provider, and related to real-world business requirements. The requirements include:

- Conditional Access to streaming content
- Encryption of file-based and streamed content where required
- Allow flexible rights expression
- Integrate with accounting
- Respect for privacy and consumer rights
- Assertion of fair-use for purposes such as backup, education, etc.
- Identity-based conditional access (providing a better alternative to Geo-IP blocking)

9.2 Modular Architecture

The Vital++ Content Protection sub-system has been designed with modularity in mind. Functionality is divided into a set of discrete components and orchestrated using an open source workflow engine. Components may be replaced for reasons such as scalability and interoperability (e.g. with a different accounting system) with minimal or no changes to the codebase. We have adopted the "loosely-coupled" Service Oriented Architecture (SOA) approach

9.2.1 Identity-based Conditional Access

By focussing on techniques for fair use and DRM, the architecture actually enables content distribution scenarios, which would otherwise be difficult. One example is identity-based conditional access. RBB, like many broadcasters, are state funded to provide content free of charge to citizens of the state. However, issues arise where citizens are travelling but still want to avail of their right to access the content, which is funded through their taxes and/or license fees. Currently, many Internet users are frustrated in attempting to access such streamed content when they travel because of the process known as Geo-IP blocking. Here, a black-list of public IP addresses is maintained based on the network class assignments of the Internet Assigned Number

Authority. (IANA)[34]. This is a crude mechanism that is obviously problematic when content is accessed by travelling users, potentially using mobile devices.

The Content Protection Subsystem contains a conditional access system that evaluates licensing requests against business rules set by the content provider. These rules can be set across multiple parameters including:

- User Identity (user-based access)
- Group Membership of the User (group-based access)
- Time (i.e. content may only be licensed for a certain timeframe)
- Device capabilities and media encoding
- Accounting rules such as pre-paid versus post-paid billing.

Identity based conditional access enables Vital++ to leverage the strong authentication, and potentially message encryption, of IMS in identifying whether a network subscriber is entitled to access content. For example, A Vodafone Germany subscriber could be allowed to access RBB content anywhere in the world providing they authenticate using their IMS.

9.2.2 Integration with Accounting

The CP subsystem integrates with a standardised IMS accounting system using the Rf and Ro diameter interfaces required for post-paid and pre-paid billing, respectively. Additionally, a Vital++ accounting and billing system is provided. This permits a content provider to register a charging scheme for a particular item or collection of content. A charging scheme may use charging data such as a cost/byte, a direct item cost and incentivisation schemes based on being a good "netizen" of the Vital++ overlay by contributing bandwidth and storage to the overlay. The accounting system is described in more detail in D4.1.

9.2.3 Flexible rights expression

By flexible rights expression we mean a mechanism that is rich enough to permit a content provider to describe rights using a range of parameters including:

- Group and user content access rights;
- Content expiry scheduling; (e.g. RBB mandate that content is only available for 7 days after publishing)
- Different rules for roaming users (some content may not be available when roaming)
- Rules determining the quality (codec and bit-rate) of licensed content

[34] Internet Assigned Numbers Authority – IANA, http://www.iana.org

To achieve these goals the CP subsystem uses the rules engine from the Open Source Java Drools[35] project. This is capable of processing rules and data-types of effectively arbitrary complexity. A side-effect is that the Content Protection rules are decoupled from the implementation of the CP subsystem and may be created and tested using graphical tools. Drools logic is processed using a workflow system, which we also use within the CPS. Rights Expression is described in more detail in #9.3 of this document.

The described rules are defined using the Drools rules specification rather than MPEG-REL or ODRL but we have adopted a similar data model. The reason for implementing a generic rules engine, rather than one specialised to evaluate MPEG-REL for instance, is that it is suitable for developing arbitrarily complex rules while not precluding us from targeting a rules specification such as MPEG-REL or ODRL in the future.

9.2.4 Description of functions

The CPS is integrated within the IMS network as shown below.

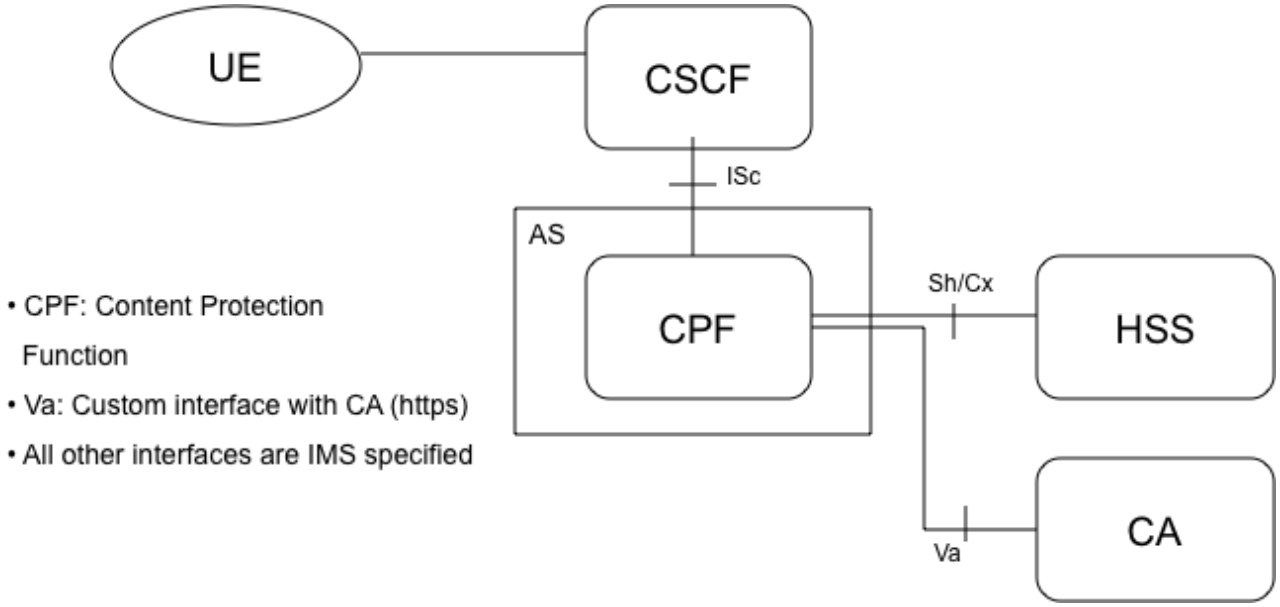

Figure 8 - CPS integration within the IMS

The User Equipment (UE) here represents a Vital++ node. The node accesses the functions of the Content Protection Function, the logic of the Content Protection Subsystem, using the IMS ISc interface. The ISc is a SIP protocol connection that is used when the S-CSCF loads a trigger point corresponding to the message that has been presented to it. In our case the message will be matched based on a known "service identifier" e.g. content-protection@<vital-domain> and the Vital++ SIP header that is added to all Vital++ messages.

[35] Drools Business Logic Integration Platform, http://jboss.org/drools/

The process of licensing a piece of content follows a Request/Response model and uses the SIP Instant Messaging conversation mechanism defined by the 3GPP. By re-using an existing mechanism we rely on the existing IMS authentication and message security model. This is explained in further detail in D4.1.

The Content Protection Function (CPF) is deployed within a standard IMS application server corresponding to the Java Community Process's JSR 289[36] specification. This is the latest specification for Sip Servlets. The CP subsystem runs successfully in the open source "SailFin"[37] servlet container.

The CPF may additionally use the HSS to verify a subscriber's credentials using the Sh (profile information) and Cx interface. However, this is a non-standard use of the Cx interface and has not been implemented within Vital++.

The content protection system uses Public Key Infrastructure (PKI) to mutually authenticate content provider and content consumer. The CPS acts as a trusted intermediary meaning that the content consumer and provider do not have to interact directly in the licensing process. The mutual authentication means that the content consumer can be confident that the licensed content is being licensed by the correct provider and hasn't been maliciously tampered with. The content provider benefits from IMS and PKI-based identity management being used to verify the identity of the consumer making it difficult for the licensing party to impersonate another user. A Certificate Authority (CA) is used to associate public-private key pairs with IMS identities.

9.2.4.1 VITAL++ DRM Interface Component

The design and implementation follows the format of the "Mother May I" (MMI) protocol specified by the Open Media Commons (OMC) initiative[38]. Our modified protocol specification is described in greater detail in D4.3. It is based on sending text-based attribute value pairs and is therefore inherently extensible. For Vital++ we have re-implemented core components of the OMC DReaM entirely using Java and using standardised components for workflow management and rules processing.

9.2.5 CP Subsystem Constituent Components

Required CPS components are shown in purple. Optional nodes that may be aggregated and/or provided by third parties are shown in green.

[36] JSR SIP Servlet v1.1, http://jcp.org/en/jsr/detail?id=289

[37] Glasfish SailFin, http://wiki.glassfish.java.net/Wiki.jsp?page=SailFin

[38] Open Media Commons initiative, http://openmediacommons.org/

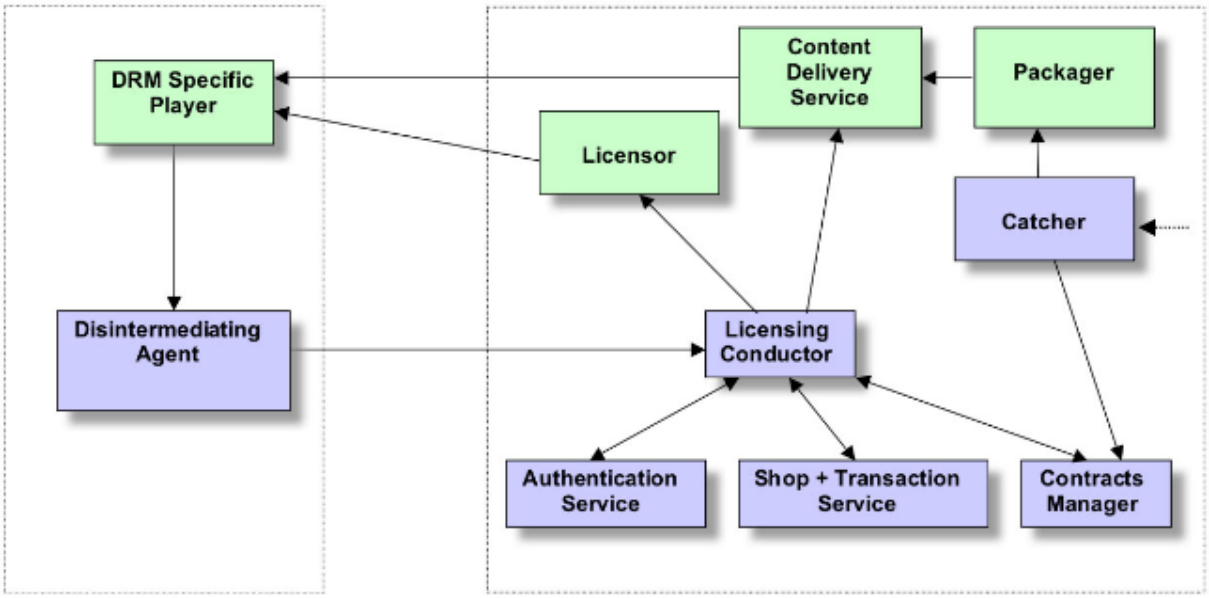

Figure 9 - CPS model is based on Open Media Commons

Client - DRM Specific Player

The DRM Specific player is a client-side player application that has DRM specific support for handling protected content and licenses. This is the Vital++ client, enhanced with the CPS client library.

Client - Disintermediating Agent

This refers to the client side library implementing the Request-side of the licensing protocol supported by the CPS (see 9.2.6). It dis-intermediates in that it supports a flexible licensing model (MMI) that can be customised to fit content protection and rights management solutions from different vendors. The server-side counterpart of the client disintermediating agent is the licensing conductor.

Licensor

The licensor is tightly bound to the DRM specific content protection technology. In Vital++ we have implemented a content protection system suitable for both live streaming and file sharing.

Licensing Conductor

The Licensing Conductor plays the role of managing the licensing processes involved in the CP Subsystem solution. It has interfaces to the CP Subsystem Client, Shopping and Transaction Service, Authentication Service, Contracts Manager and the Licensor. It performs the necessary e-commerce transactions

and authentication of the user. It instructs the Licensor to generate the license for a given user for specific content. The licensing conductor is implemented using a java workflow engine and may therefore be customised based on future requirements.

Contracts Manager

The Contracts Manager stores business rules associated with content, as well as user rights. This component has interfaces to the Licensor via the Licensing Conductor. The Licensor generates a license for a given piece of content based on the business rules and user rights that are available in the Contracts Manager.

Authentication Service

The authentication service is where subscribers, users and devices are cleared for access to services and content. Authentication functions in Vital++ are provided by the IMS Core and augmented by the CA for the purposes of mutually authenticating content provider and consumer.

Shop and Transaction Service (Accounting Services)

The workflow functions of shopping and transacting purchases includes everything from collecting payments from buyers to paying sellers and making sure that everyone is appropriately compensated in a secure manner. This component will be used to ensure that Ro and Rf charging are applied correctly for consumed media.

Content Delivery Service

In Vital++ the Content Delivery Service uses the P2P overlay. A usage example is further described in D4.2. The overlay distributes protected content while keys are distributed using IMS signalling. A "super distribution" mechanism is used to ensure that all Vital++ distribution paradigms (live streaming, file sharing, media-on-demand) can be supported.

Packager

The packaging process involves combining content data/files with associated metadata and creating logical packages that include the defined business rules.

Catcher

The Catcher performs content ingestion. It receives content and associated business rules from the content supplier. The content, which is unprotected at

this stage, is passed to the Packager. The business rules associated with the content are passed to the Contracts Manager.

9.2.6 Disintermediation

One of the key features of the CP Subsystem architecture is the ability to accommodate the inclusion of specific rights management and conditional access components from third parties while avoiding the need to incorporate all their back-end components. This is an important feature in any commercial Vital++ system. A mobile operator may, for example, have deployed an Open Mobile Alliance DRM system that is currently supported by many handsets from vendors such as Sony Ericsson and Nokia. The CPS will permit the incorporation of such a system so long as the required components are implemented.

The disintermediation system enables multiple instances of these components to exist in a Content Protection or Conditional Access System.

- The content protection-specific components of CP Subsystem include: player, licensor and packager.
- Components that are not content protection-specific include: a disintermediating agent, conductor, catcher, licensing conductor, contracts manager, authentication service, shop and transaction system, and content delivery service.

The process of disintermediation happens as follows:

1. Client requests a license
2. Front-end service redirects client to a client disintermediation agent
3. Disintermediating agent contacts Conductor (back-end service)
4. Conductor contacts back-end services for authentication and rights verification
5. Conductor signals front-end service with instructions to deliver license to client
6. Front-end service delivers license

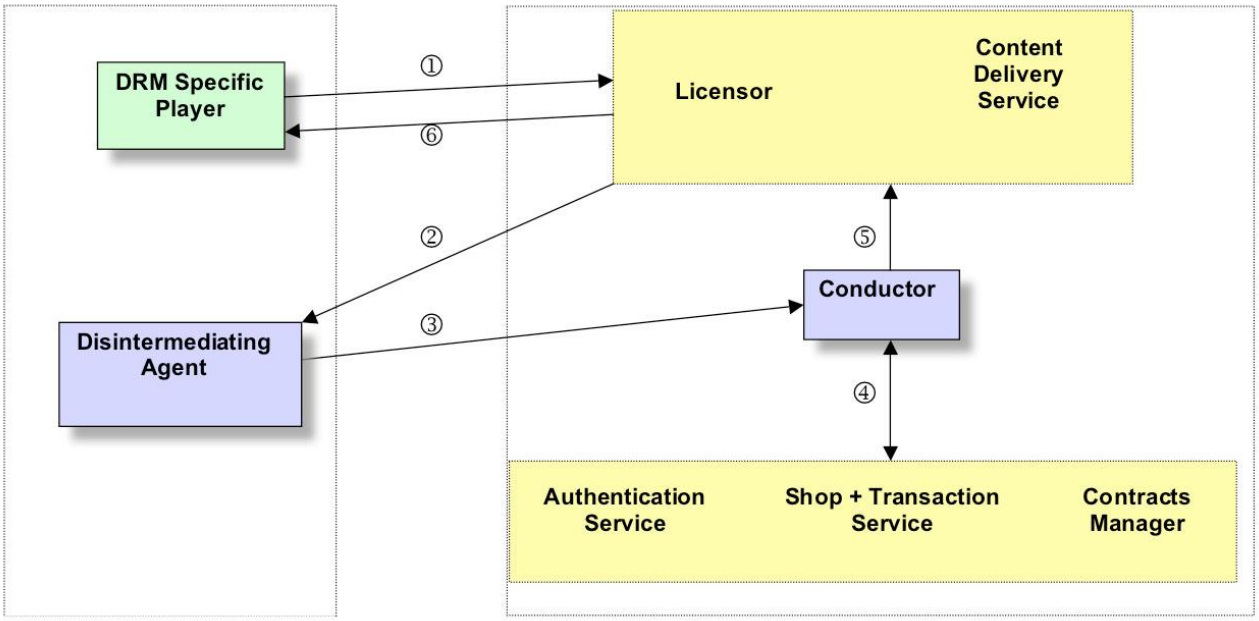

Figure 10 - Disintermediation process

9.2.7 Message Exchange and Transactions

In this section we give an overview of the MMI profile, which is used for Vital++. As described earlier, MMI defines a request-response protocol for licensing content based on the work of SUN's DReaM project. The Vital++ MMI profile is described in more detail in D4.3. Additionally, we include message sequence diagrams for content publishing and content licensing using the CP subsystem.

9.2.7.1 Licensing Workflow

As described previously the licensing conductor uses the Java Business Process Management (JBPM) open source workflow management engine to describe the licensing process. This is an inherently flexible approach as we have sought to decompose the licensing functions into modular blocks corresponding to the different states of the licensing process. The diagram below shows how request

handling, rules processing and accounting are integrated within a single workflow.

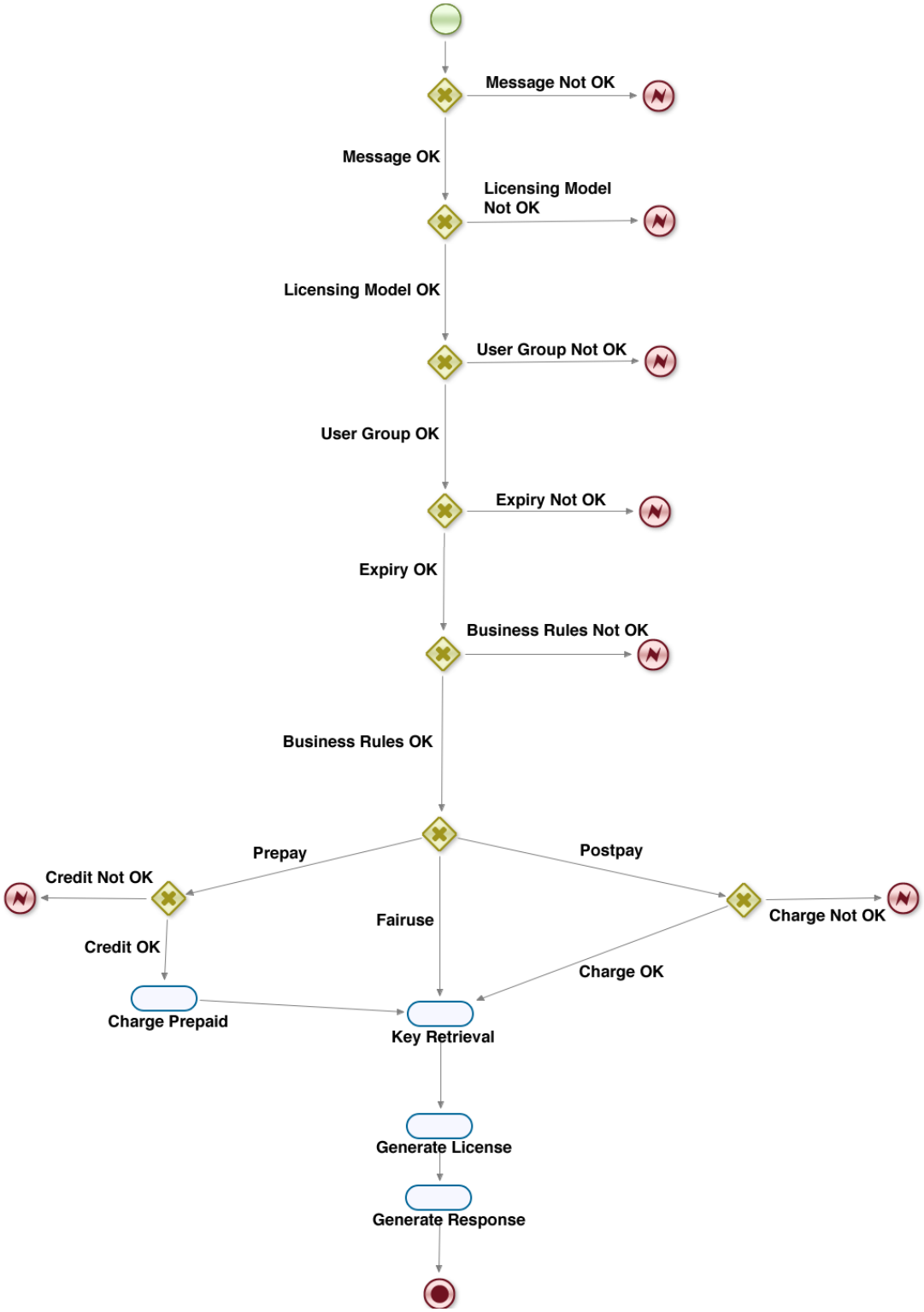

Figure 11 - JBPM Licensing Workflow

The licensing workflow is designed with three basic types of content consumer in mind:

- Prepaid customers
- Postpaid customers
- Customers exercising fair-use

The workflow is triggered by an incoming MMI request and is parsed and checked to see if it is well-formed. If the request is well-formed, the licensing model is checked for validity. The user's group is also checked for validity. Afterwards, the requested content is checked for expiry. If the requested content has not expired, the workflow makes a decision as to what kind of content consumer is requesting the content - either prepaid, postpaid, or a user exercising fair-use.

A prepaid customer has their account checked to see if they have enough credit available. If so, they are charged for the content - if not, an error message is generated. A postpaid customer is charged for the content, with an error message created and sent if the charge was unsuccessful. A user exercising fair-use is not charged and proceeds to key retrieval. Next, the workflow retrieves the key for the given content. A license is then generated, using the client's signature, the content provider's signature and the content key itself. With license generation complete, the MMI response is generated and sent back to the client, ending the workflow.

The JBPM graphical workflow design tool may be used to modify the workflow without the need to re-implement or recompile any java code.

9.2.7.2 CI/CS interactions

Elements of the workflow are also re-used by the Content Indexing subsystem. The CI and CP subsystems interact during the content discovery and licensing phases. The CI requests information from the CP subsystem about:

- The licensing model available for a particular content item (e.g. fair use, pre-paid, etc.)
- Whether the content is available to the user or user’s group?
- Whether the content has been successfully licensed by the user. Once content has been licensed the CI can make overlay details available to the licensed user. A user already in possession of a valid unexpired license for a content item does not have to re-license it.

9.3 Rights Expression

Rights expression refers to the mechanism by which the owner(s) of the intellectual and distribution rights for a piece of digital content can specify those rights in a form intelligible to the content protection system.

Vital++ adapts the "Mother May I" (MMI) mechanism specified in SUN's DReaM.

DReaM-MMI

Dream-MMI is an alternative method for expressing and controlling rights using the Mother-May-I paradigm. This paradigm provides a mechanism to request and obtain rights, release unused rights and for the aggregation of requests and releases. Messages are currently transported using HTTP/HTTPS however the system is designed is such a way as to facilitate the use of alternative transport mechanisms such as SIP based messages or Diameter based messages. The current version of the MMI specification provides three basic profiles for content: Video/Music, Documents and Applications

The MMI protocol uses a <*major*>.<*minor*> numbering scheme similar to HTTP RFC2616. Each MMI message must include the version number and messages must be compliant with that version. There are two types of MMI messages: **MMIRequest** and **MMIResponse**. The MMIRequest message can be further classified into **MMIRightsRequest** and **MMIRightsRelease**.

MMIRightsRequest messages are used to request rights associated with a set of specific content or service. Each rights request contains MMIRightsRequestElements that can be granted or denied and if denied additional hints are provided. The MMIRightsRelease messages are used to surrender any previously granted usage rights for specific content or service. The response to these messages is either a success or failure including an appropriate error code. A full description of the request message and request elements can be found in the appendix A.

MMIRightsResponse is the message, which is used by the Licensor to respond to each MMI Rights Request from a client. If the request contains several request elements then each request element is replied to. The request is either granted or denied or an error is reported. A hint can also be sent in the response that gives the client an additional guideline for future requests. A full description of the Response Message can be found in Appendix A.

The server responds to a MMIRightsRequest with a MMIRightsResponse, which contains a Status field that specifies the status of the request. In case of errors in the original request, the Status field will contain one or more codes explaining the reasons for the failure. If there were no errors in the original

request, the server must send a "RequestOK" status code in the Status field. Note that even if the Status reports a "RequestOK" message, individual MMIRightsResponseElements may be denied. Also, note that specific RightsCodes cannot be reported in the Status field. RightsCodes can only be reported in the RightsErrorStatus field in the individual MMIRightsResponseElements. Servers that do not want to report specific error messages to the client due to security considerations, may opt to return only non-specific error codes such as Identity Error, Device Error, RightsElementError, RightsParseError etc. A list of the status codes can be found in the Appendix A.

In the original DReaM protocol specification the Open Media Commons group specified that *"HTTPS should be used to achieve confidentiality and integrity protection"*. For Vital++ HTTPS may be used in addition to SIPS mechanisms provided by IMS using Transport-layer (TLS) or network-layer (IPSec) security mechanisms. The MMI messages are simply sent in the body of SIP messages , in the same manner as Session Description Protocol (SDP)[39]

Transportation of requests can be made over HTTP GET, POST or SIP MESSAGE. The fields of the request follow the Java Properties style naming convention. For example:

```
MMIVersion
MMIMessageType
Identity.AuthServiceId
Device.LocationId
Device.DeviceId
Rights.ProfileId
Rights.ReqElem.Id
Rights.<Rights.ReqElem.Id>.ContentId
Rights.<Rights.ReqElem.Id>.ServiceId
Rights.<Rights.ReqElem.Id>.VerbId
Rights.<Rights.ReqElem.Id>.<VerbId>.Verb
Rights.<Rights.ReqElem.Id>.<VerbId>.Count
Rights.<Rights.ReqElem.Id>.<VerbId>.Duration
Rights.<Rights.ReqElem.Id>.<VerbId>.<VerbSpecArgs>
Signature.SigAlg
Signature.Signature
```

The following is an example of the an MMI Request using a HTTP GET:

```
http://www.greatcontent.org/myService?MMIVersion=1.0\
```

[39] IETF, 7SDP: Session Description Protocol, http://www.ietf.org/rfc/rfc2327.txt

```
&MMIMessageType=MMIRightsRequest\
&Identity.AuthServiceId=www.myAuthService.org\
&Device.DeviceId=123456abc\
&Rights.ProfileId=org.omc.dream.profiles.media\
&Rights.ReqElem.Id=23\
&Rights.23.ContentId=113%2C114%2C115\
&Rights.23.VerbId=1\
&Rights.23.1.Verb=SimplePlay\
&Rights.23.1.Count=1\
&Rights.23.VerbId=2\
&Rights.23.2.Verb=Record\
&Rights.23.2.Count=1\
&Rights.23.2.Target=123456abc
```

The following would be the MMI Response sent as a result of the above MMI Request:

```
HTTP/1.1 OK
Content-type: text/plain
Content-length: nnnn
MMIVersion=1.0
Status=RequestOK
Response.ReqElemId=23
Response.23.Notification=granted
Response.23.Hint.HintIndexNum=1
Response.23.Hint.1.Label=CanDo
Response.23.Hint.1.ContentId=113,114,115
Response.23.Hint.1.VerbId=1
Response.23.Hint.1.1.Verb=SimplePlay
Response.23.Hint.1.1.Count=29
Response.23.Keys=OD6Ox9svtSgFJ+iXkZ
ReqHash.HashAlg=http://www.w3.org/2001/10/xml-exc-c14n#
ReqHash.RequestHash=jAxX0LfgwutvEdJb748IU4L+8obXPXfqTZ
ResponseId=1003
Signature.SigAlg=http://www.w3.org/2001/10/xml-exc-c15n#
Signature.Signature=OWqP5Gqm8A1+/2b5gNzF4L4L
```

As mentioned earlier, a limited set of profiles have been currently defined for the MMI specification. These profiles will address DRM requirements for:

- Multimedia (Video, music) content
- Documents (Files)
- Applications

If additional profiles are required they can be created using the Profile Extension Framework.

A profile consists of:

- Profile Identifier: A profile is identified by a namespace.
- A brief description of what the profile is intended for.
- Base Profile: Service providers and content producers can extend profiles to meet their specific requirements.
- Verbs allowed by a profile, and the relevant arguments for each verb.

Details of the profiles and the verbs and arguments supported by them can be found in the appendix A.

9.4 Integration with Accounting Subsystem

The Accounting subsystem is described in more detail in D4.1. It consists of elements including:

- A Charging Gateway Function (CGF) – An IMS charging gateway for storing usage data
- A Billing & Rating Function (BRF) – A flexible and highly scalable accounting system based on spreadsheet worksheets. The BRF has a webservice interface. It receives usage data in XML format and responds with an XML rating document. The rating document may be transformed into a customer bill for service and network usage.
- A Charging Control Function (CCF) – A rules-based charging decision function that evaluates whether a service can be provided to a particular user based on their charging profile and that of the service. E.g., the service may require "post-pay" and the user account may be "pre-pay" only. The CCF is implemented using JBPM workflows.

The BRF within the Accounting system associates a charging worksheet with a service or content provider (e.g. RBB) and the service or content being provided. The worksheet describes additional logic for special tariffs to incentivise good behaviour on the overlay (serving content), group and requesting content "bundles". For the purposes of Vital++ experimentation and demonstrations, the rating worksheets are stored in a common XML database shared between the content provider and the accounting provider. In practice the content provider may outsource Rating and Accounting functionality.

Throughout the Vital++ system common identifiers are used to promote consistency and simplify integration between the various systems. These include:

- A Content Identifier used primarily in the Content Indexing subsystem
- A Service Provider Identifier used by the Accounting Subsystem
- SIP URI's as user identifiers

9.5 Usage Scenarios for Content Protection

Here, we describe the business rules for content protection and show sequence diagrams for the licensing and publishing process.

9.5.1 Business Rules

In discussions with RBB a selection of content protection business rules have been elicited. They are evaluated when a user requests a license for a content item(s) using the MMI protocol defined in Annex A. The result of the evaluation determines whether the content can or cannot be licensed.

Eligibility Rules

- Category of Content Provider
  - Professional Providers can publish more content to a super-peer;
- Account Type
  - Credit balance required for pre-pay content licensing;
  - Some content could be made "post-pay" only;
- Time
  - Content is only available for 7 days from the date of release;
- Location/Network
  - Designated content must be available free to national subscribers regardless of what country they are in. This decision is made based on the subscriber's network provider;
  - Export-restrictions for some content.

Rule Type	Condition	Result
Content Provider Category	Category==’professional’	Allow publishing to super-peer
	Category==’prosumer’	Don’t allow publishing to super-peer
Subscriber Account Type	Account_type==’prepay’	Account balance must be in credit AND Account(balance) >= Content(cost) Can’t access content where Content(Account_type)==’postpay’
	Account_type==’postpay’	Can Access Content where Content (Account_type)==’postpay’
Subscriber	Status==’active’	Can Access OR publish content, depending on evaluation of other

Account Status		rules
	Status==’inactive’	Can’t access OR publish content
Date/Time	Date()<= Content(publish_date)+ time_restriction	Content can be licensed so long as ‘time restriction’ days/hours/mins hasn’t elapsed. E.g. 7 day rule mentioned above
	Date()>= Content(publish_date)+ time_restriction	Content cannot be licensed.
Location	Subscriber (location) != ContentProvider(location) AND for-all (content(blacklist)!=location)	Some content cannot be licensed from blacklisted locations.
Network	ContentProvider(whitelist) contains network provider(id)	Content can be licensed to a subscriber accessing through ‘network provider’ without considering their location. However, other rules should still be considered.
	!(ContentProvider(whitelist) contains NetworkProvider(id))	Check location rules. If Subscriber(location_!= ContentProvider(location) then content cannot be licensed.

The Drools Expert[40] rules engine is used to process business logic encoded in text-based rules. Drools is an open source rules processing engine that is popular among Java Enterprise systems developers as a means to describe, deploy and execute business rules using formal textual statements, decoupled from an applications codebase. The content provider registers licensing rules with the Content Security subsystem. These rules can be parameterized and hence associated with individual users, user groups, content types, network context (e.g. user location) and billing scenarios (e.g. pre-pay, post-pay).

For example: the following is true if the subscriber has a prepay account and their account balance is sufficient to afford the content item.

```
Subscriber(Account_type == “prepay”) &&
Subscriber(account_balance)>=Content(cost_estimate)
```

[40] Drools Expert – Jboss Community, http://www.jboss.org/drools/drools-expert.html

The use of Drools rules to describe a content provider's business rules and JBPM workflow to describe the licensing process means that the business logic is decoupled from the application in a Service Oriented Architecture manner.

The resulting implementation requires minimal knowledge of IMS to modify the licensing workflow and no knowledge of IMS to create business rules.

The diagram below shows how request handling, rules processing and accounting are integrated within a single workflow.

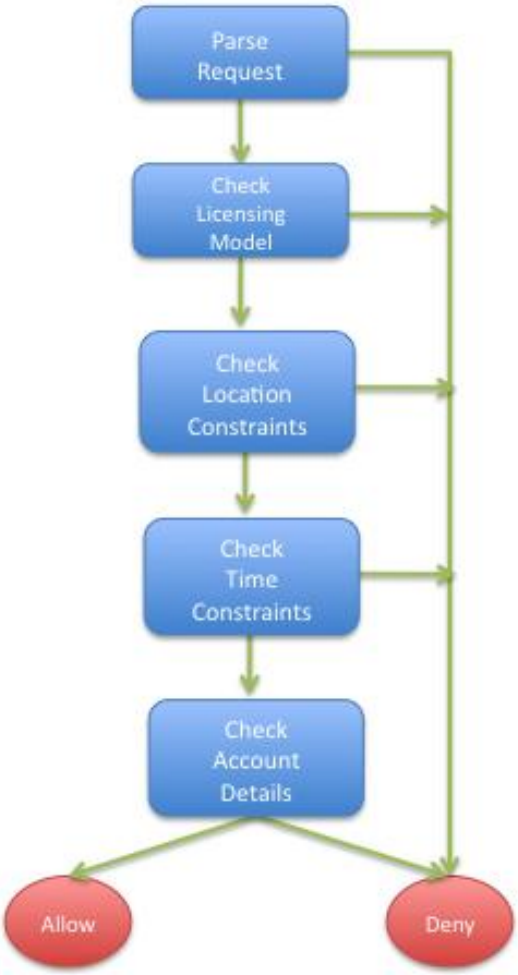

Figure 12 - Licensing Workflow

9.5.2 Content Publishing

The following sequence diagram shows how content publishing is achieved, demonstrating the interactions between a platform user's Vital++ client, CI and CP subsystems.

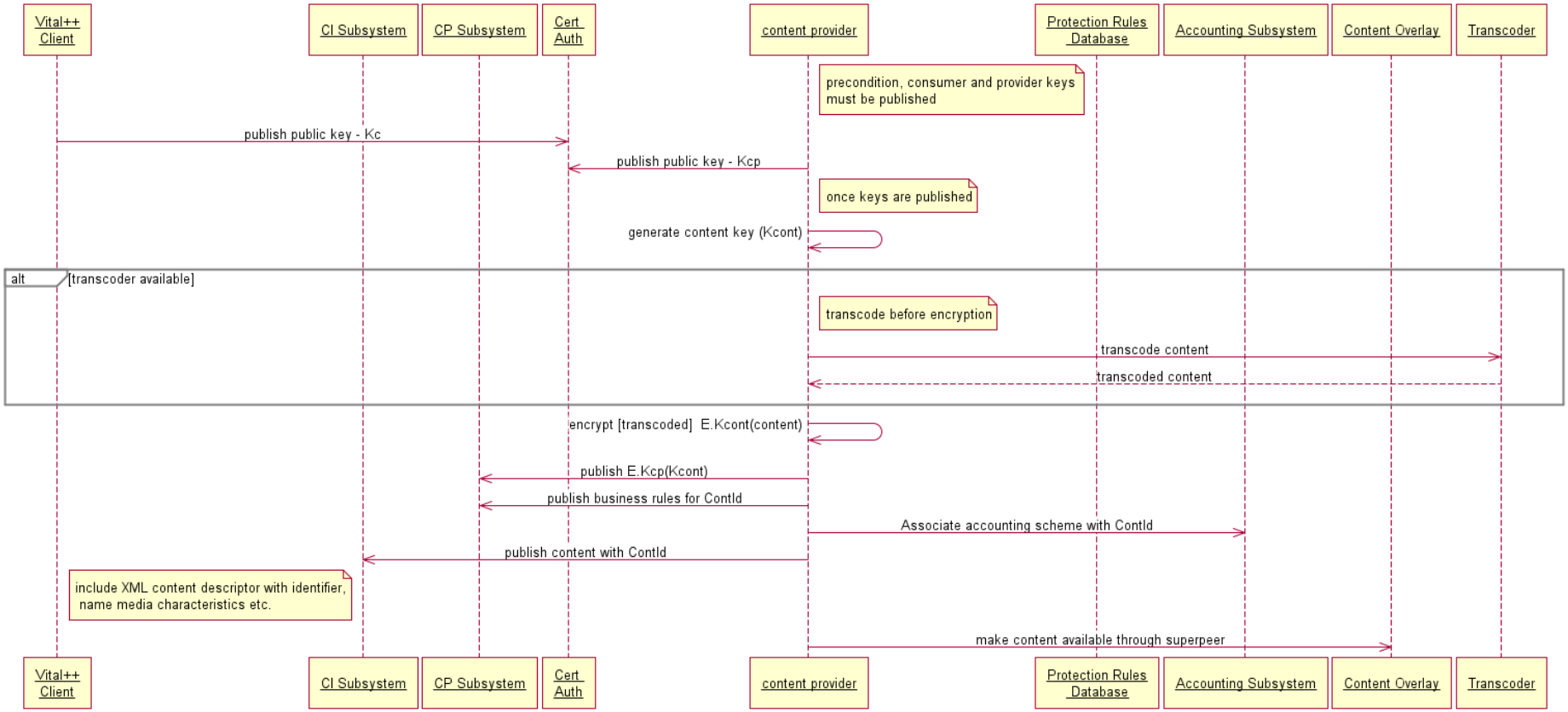

Figure 13 - Content Publishing Sequence Diagram

The steps are as follows:

1. The subscriber generates a public & private key pair using their Vital++ client and publishes their public key to a Vital++ certificate authority.
2. The content provider does the same.
3. For each piece of content to be encrypted, a symmetric key is generated. This is encrypted using the private key of the content provider and distributed to the CP subsystem.
4. Content is transcoded prior to encryption, as part of the content publishing process
5. The content is encrypted within the content provider's "super peer". In practice for a live content stream the SRTP symmetric encryption mechanism is used. (See RFC3711[41]).
6. The Content Provider associates accounting and licensing rules with that content identifier.
7. The Content Provider notifies the CI of the new content.
8. The content is now accessible via the Vital++ content overlay. The Vital++ overlay structure is described further in D3.1.

9.5.3 Content Licensing

The following sequence diagram shows how published content is licensed, detailing interactions between a user's Vital++ client, the CP and CI subsystems.

[41] IETF, RFC 3711 – Secure Real Time Transport Protocol, http://www.ietf.org/rfc/rfc3711.txt

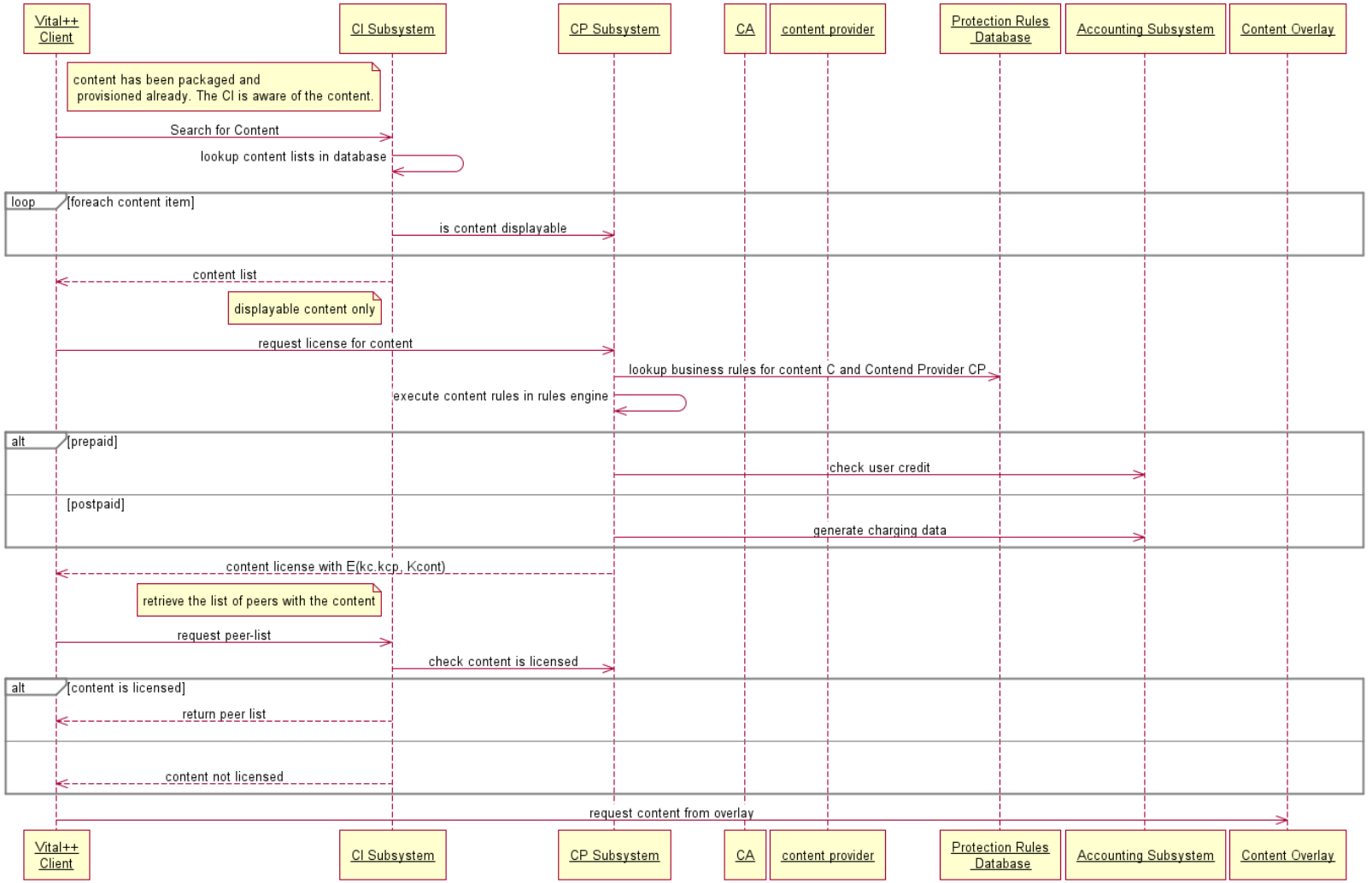

Figure 14 - Content Licensing Sequence Diagram

The steps are as follows:

1. The subscriber searches for, or discovers, content using their Vital++ client, which communicates directly with the Content Indexing Subsystem. This is described in more detail in D3.1.
2. The CI asks the Content Protection (CP) subsystem whether the requested content is displayable to that particular subscriber. Some content may not be available e.g. (post-pay only). The CP responds with an approved list of content.
3. Once the client has downloaded the content list matching their search they will license a content item using a SIP message with a body containing the MMI protocol (described later in this document)
4. The CP processes the license request based on the content provider's business rules.
5. The accounting system is used to calculate the cost to the customer and check the user credit where appropriate. The Accounting Subsystem is described in more detail in D4.1.
6. If the CP determines that the subscriber can successfully license the content it distributes the symmetric key required to unlock the content to the client. The CP distributes this key in an encrypted form, encrypted using the private key of the content provider and the public key of the subscriber.
7. The client may then request the list of peers required to join the content overlay from the CI
8. The CI checks that the content has been successfully licensed before making the peer list available.

9.5.4 Fair Use Scenario

Effectively, a fair use requests is just another kind of licensing request. This scenario is shown in greater detail using internal components within the CP Subsystem including the Licensing Conductor, Contracts Manager and Licensor described in #9.2.5

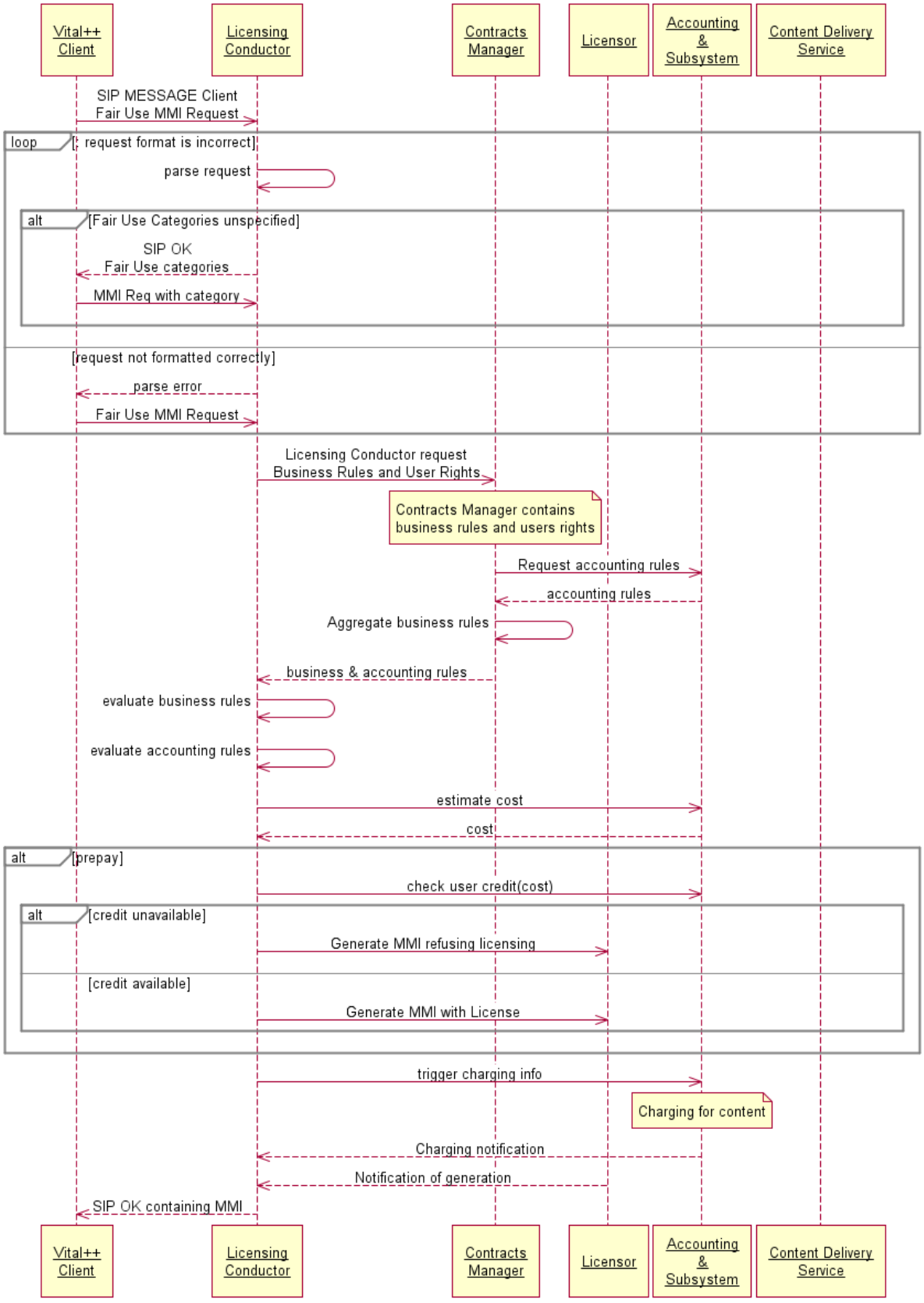

Figure 15 - Fair Use scenario

The steps are as follows:

1. The subscriber issues a fair use request to the CP subsystem using a SIP MESSAGE. The licensing conductor is the first point of contact in the CP subsystem.
2. The licensing conductor checks to see that the request is formatted correctly and contains the category of fair use assertion, e.g. educational, parody, news, etc. If not, a response is generated with the suitable fair use categories. The subscriber responds with the correct fair use category.
3. Upon receipt of a correctly formatted request the licensing conductor requests the business rules from the contracts manager database.
4. The contacts manager aggregates accounting and business rules. The resulting drools rules are sent to the licensing conductor
5. The licensing conductor evaluates the business and accounting rules to determine if the client is entitled to license the content
6. The licensing conductor requests the accounting subsystem to estimate the subscriber's cost for licensing the content. This may not be an exact calculation as an accounting scheme may encompass incentives or tariffs depending on the amount of traffic the client has relayed/requested in the past.
7. In the event of a pre-pay scenario the subscriber's credit must be checked to determine they can afford the content. If they can afford the content, an MMI response containing a license is generated, otherwise a license refusal is generated.
8. Charging information is triggered. This effectively closes the charging transaction. The charge may be nil if the content is free OR if a license cannot be granted.
9. A SIP OK Response is generated with the MMI response to the licensing request and sent to the subscriber's vital++ client.

Annex A – Vital++ Rights Negotiation Protocol Structure

This annex describes technical details of the MMI rights negotiation protocol used within Vital++. It is a modification of the DReaM MMI protocol.

Protocol version

MMI Version

```
MMIVersion = 1*DIGIT "." 1*DIGIT
```

MMI Messages

There are two types of MMI messages: MMIRequest and MMIResponse. MMIRequest is further categorized
into MMIRightsRequest and MMRightsRelease.

```
MMIMessage = MMIVersion (MMIRequest | MMIResponse)
```

Table 2: MMI Message Format

MMI Request Messages

```
MMIRequest = MMIMessageType IdentitySegment [DeviceSegment] RightsSegment
[SignatureSegment]
```

Table 3: MMI Request Message Format

Field	Description
MMIMessageType	Indicates if the message is requesting rights or surrendering previously requested rights.
IdentitySegment	Is used to identify and authorize users and/or their role(s). The IdentitySegment contains all the identity information associated with a user. If a user is already authenticated, it could contain the authentication token (AuthTkn) assigned to an authenticated user in a given session to avoid re-authenticating the user for every MMIRequest.

DeviceSegment	Contains all the information about the device or devices with which the user is associated and is requesting rights for. In the simplest case, this will contain information about the device or the desktop from which the user is requesting information. However, this could contain information about a set of devices which the user is associated with and is requesting rights for. Also, these devices can be tethered or associated with the current device which may or may not connect directly to the network. The DeviceSegment is optional but may be required by certain rights management systems.
RightsSegment	Encapsulates the actual rights that are either being requested or released and associated information. The actions or verbs for which rights are requested (or released) are contained in the RightsSegment. The definitions of the verbs are specified in the profiles. ProfileId, which is used to identify profiles, is also contained in the RightsSegment. The RightsSegment contains one ProfileId and one or more MMIRightsRequestElements grouped together for transaction optimization purposes.
SignatureSegment	Encapsulates the Signature and associated algorithm information for verifying the Signature. Signature enables the authentication, auditing, verifying message integrity for non repudiation. It is optional but it may be mandated by the specific service provider. Although this is optional, it is RECOMMENDED that a Signature mechanism be used especially when the transport is not secure (like http).

Table 4: MMIRequestMessage Elements

IdentitySegment = AuthServiceID [AuthTkn]

Table 5: MMIRequestMessage IdentitySegment Format

Field	Description
AuthServiceId	An opaque identifier (URI) that specifies what security service is used for authenticating and/or authorizing the user. This might specify a service provided directly by a standalone server, or some federated third party.
AuthTkn	Is a token assigned by the authentication service (AuthServiceId) when a user is authenticated. After negotiating for this token, it is used in all subsequent requests in a given session (till the authentication token expires or till the user explicitly terminates the session). The AuthTkn MUST be base64 encoded. It is assumed that a single user ID and one or more role IDs are associated to each user during the registration process. The authentication process may or may not use the same identifiers in order to protect user privacy. However, it is expected that the license servers can derive user and role IDs associated with the user from the authentication token through appropriately negotiating with the authentication

	service.

Table 6: MMIRequestMessage IdentitySegment Elements

DeviceSegment = [LocationId] [#DeviceId]

Table 7: MMIRequestMessage DeviceSegment Format

Field	Description
LocationId	Identifies the geographical location where the user and/or the device(s) are physically located. It is assumed that the LocationId is assigned by the rights management system during the registration process. For example, this may be used to convey the DVD region codes.
DeviceId	Uniquely identifies a given device. This DeviceId is also assigned by the rights management system during the registration process.

Table 8: MMIRequestMessage DeviceSegment Elements

Field	Description
ProfileId	Uniquely identifies a profile defined in this MMI Specification or a later revision. The identified profile defines the verbs that can be used in this MMIRequest message.
MMIRightsRequestElement	Represents an atomic rights request element. All the content (or service) and the requested rights information (action and associated terms) is contained within this element.

Table 9: MMIRequsestMessage RightsSegment Elements

Field	Description
ReqElemId	Identifies this request element and has to be unique for this authentication session.
ContentId*	Uniquely identifies the content resource or resources for which rights are requested.
ServiceId*	Uniquely identifies the service for which rights are requested. This is an optional field.
VerbElement	This is an element that encapsulates a single Verb (defined in following table) along with arguments for which rights are requested. Multiple Verbs may be associated with a single RequestElement by encapsulating each Verb within a separate VerbElement. The VerbElement may contain additional information such as the duration for which the rights to perform the

	action(verb) is requested, the number of times the action may be performed, and other profile and verb specific fields.
* Each MMIRequestElement may contain multiple ContentIds and ServiceIds, or just any one of the two depending on the information that the service provider requires.	

Table 10: MMIRequestMessage MMIRightsRequestElement Elements

Field	Description
VerbElementId	This is an identifier used to distinguish among VerbElements within a MMIRightsRequestElement. It must be unique within a MMIRightsRequestElement.
Verb	This is an identifier that represents the action for which rights are requested. These verbs are profile specific and are defined in section 6
Count	This is the number of times the rights for these verbs are requested (if applicable). For instance, if the verb is Record and the Count is 3, rights are requested for recording the content 3 times.
Period	Marks the absolute start and end time within which the content rights should be used. The Period is as defined in [RFC2445] Sec 4.3.9 Period of time
Duration	This indicates the duration, as defined in [RFC2445] Sec 4.3.6 Duration, in which the content can be consumed.
VerbSpecificArgs	Other fields that are relevant to the Verbs. The specific fields that are applicable to a particular verb depend on the profile and are detailed in Section 6.

Table 11: MMIRightsRequestElement VerbElement Elements

Field	Description
SigAlg	URI identifying the signature algorithm used to sign the message. A conforming implementation must support those identifiers and algorithms defined by the XML Signature specification[XMLSigAlg] to be able to verify messages.
Signature	Digitally signs the entire MMIMessage (including SigAlg field), with the given signature algorithm, using the private key associated with the sender's public key. Signature **MUST** be base64 encoded.

Table 12: MMIRequestMessageSignatureSegment Elements

MM Rights Response

Field	Description
HashAlg	URI identifying the hashing algorithm used to hash the message. A conforming implementation must support those identifiers and algorithms defined by the XML Signature specification[XMLSigAlg].
RequestHash	Digital hash of the MMIRightsRequest that this MMIRightsResponse corresponds to.

Table 13: MMIRightsResponse RequestHashSegment Elements

Field	Description
ReqElemId	The same identifier used in the MMI Rights Request Element that this response is associated with.
Notification	Three values allowed: • "granted" indicates that the request is granted • "denied" indicates the request is denied • "error" indicates that there is an error in the MMIRightsRequestElement referred to by ReqElemId. Information about the specific error is available in RightsErrorStatus
Keys	Used to encapsulate content protection keys. Typically, the content symmetric encryption keys are asymmetrically encrypted for the client. Keys MUST be base64 encoded.
RightsErrorStatus	A RightsCode that is defined in the MMI Status Codes appendix. Note further syntactical rules for RightsErrorStatus. RightsErrorStatus is mandatory if there was an error in processing the MMIRightsRequestElement referred to by ReqElemId. It can be present only when notification="error". Also, when RightsErrorStatus is present, Hint MUST not be present.
Hint	An optional hint provided by the server as a guideline for future requests.

Table 14: MMIRightsResponse MMIRightsRequestElement Elements

Field	Description
HintIndexNum	The index number associated with this Hint. This has to be unique within the RightsResponseElement.

ContentId	Id of content requested.
VerbElements	VerbElements are as specified in the MMIRequestElement
RightsErrorStatus	A RightsCode that is defined in the MMI Status Codes appendix. Note further syntactical rules for RightsErrorStatus. RightsErrorStatus is mandatory if there was an error in processing the MMIRightsRequestElement referred to by ReqElemId. It can be present only when notification="error". Also, when RightsErrorStatus is present, Hint MUST not be present.
Label	Two values supported: • "CanDo" is used to indicate the rights the user is entitled to • "CannotDo" is used to indicate the reason (VerbElements) that caused the denial of the request. Hint with Label "CannotDo" MUST be used only when the request is denied. It should be noted that when a request is granted the terms of the grant are as specified in the request and not what the Hint indicates. In such cases, Hint is purely used to refine further requests. A user may release the granted rights and re-request using the arguments in the Hint.

Table 15: MMIRightsResponse Hint/Label Elements

MMI Error Handling

MMI Servers should respond to MMI Protocol errors with a HTTP or SIP '200 OK' response and send an MMIRightsResponse as part of the HTTP response body. The MMIRightsResponse contains a status code indicating the nature of the error.

If the HTTP or SIP connection is broken before a client receives a response for its MMI request, the client can assume that its request was not processed by the server and no rights were granted or released. The client can choose to repeat the request.

On the server side, if the HTTP or SIP connection broke in the process of sending out the response, the server should roll-back the rights granted, as if the MMI request was never made.

NOTE: In the case of proxies, there is possibly a race condition where the server assumes everything went well and doesn't roll back the granted rights, but the client hasn't received the server's MMI response. In such a case, the client can repeat the MMIRequest with the same ReqElemId and the server should repeat the previous response and not treat it as a new MMIRequest.

MMI Status Codes

StatusCode	**Description**
GeneralCode	

UnsupportedProtocolVersion	MMIVersion is not supported by the server
InvalidSignature	Signature sent in request is invalid
InternalServerError	Server is not able fulfil a request because of some error on the server side
ParseError	An unspecified parse error in the request. This can indicate any parsing error that cannot otherwise be qualified as one of the errors below.
IdentityError	There was a general, unspecified error in the IdentitySegment. If the server wants to be more specific, one or more of the IdentityCodes can be sent instead.
DeviceError	There was a general, unspecified error in the device segment. If the server wants to be more specific, one or more of the DeviceCodes can be sent instead.
RightsElementError	There was a general, unspecified error in one of the MMIRightsRequestElements. If the server wants to be more specific, one or more of the RightsCodes can be sent instead.
RequestOK	No errors in the request.
IdentityCode	
UnknownUser	UserId does not match any user on the server
UnknownRole	RoleId does not match any roles at the server
AuthServiceIDError	AuthServiceID invalid.
AuthTokenInvalid	AuthTkn is found to be invalid
DeviceCode	
UnregisteredDevice	DeviceId does not match any of the registered devices
LocationIdNotSupported	Provided LocationId is not supported by the server
RightsCode	
UnsupportedProfile	The ProfileId is not supported by the server
ContentNotFound	Requested content cannot be found
RightsParseError	An unspecified parse error in the MMIRightsRequestElement.
InvalidRightsDuration	Invalid Rights Duration requested.

InvalidRightsCount	Invalid Rights Count requested.
VerbIncorrectNumArguments	Number of arguments for the Verb is incorrect
VerbArgumentSyntaxError	Syntax error in one of the arguments for the verb.

MMI Profiles

Media Profile

Verb	**Verb Description**	**Verb-specific Argument**	**Argument Description**
SimplePlay	Allows the user to play the content without forwarding or reversing it.	- None -	
ForwardPlay	Allows the user to forward content	-None-	

Annex B – Diffie-Hellman key agreement

For reference, the Diffie-Hellman key agreement process is depicted in the following table as sequence of operations.

Step	Server	Client
1	Chooses a prime p and an integer g.	
2	computes $A=g^a \bmod p$	
3	sends A, g and p to the client	
4	computes $K_s=A^b \bmod p$	computes $B=g^b \bmod p$
5		computes $K_c=B^a \bmod p$

$K_c=K_s$, because

$$K_s = A^b \bmod p = (g^a \bmod p)^b \bmod p$$
$$= g^{ab} \bmod p$$
$$= (g^b \bmod p)^a \bmod p = B^a \bmod p = K_c$$

Thus, both entities have the same key material and can use it to create a common symmetric key, which can then be used e.g. for AES encryption.

- End of document -