Support for Fair Use with Project DReaM

Sun Microsystems Laboratories
Version 1.0 Rev A
April 2008

1. Introduction

"Fair use" is a U.S. legal doctrine permitting users to reproduce portions of copyrighted material for various limited-scope purposes. Examples of fair uses can include reproducing a few words in a long work, to reprinting copies of artwork as “thumbnails” on a web site. Fair use would seem to be a very different thing than digital rights management ("DRM") which typically encrypts content to prevent unauthorized usage. Thus, the whole idea of fair use within a DRM system seems somewhat contradictory; however, DReaM supports the application of the fair use principle. The DReaM system provides a means to mediate the needs of content copyright holders and of users of copyrighted content for conditions when fair use is asserted. A brief overview of fair use and some specific assumptions on its applicability to DReaM will be the groundwork for building a DRM architecture supporting fair use. The conditions of anonymous fair use are covered under Section 4. Sections 5 and 6 discuss tracking and auditing methods to uncover the anonymity under conditions of misappropriation. Section 7 details the DReaM architecture and the processes for anonymous fair use and the auditing of the process.

2. Overview

Fair use is a U.S. legal doctrine, codified in section 107 of the U.S. Copyright Act**,** that allows users to reproduce portions of copyrighted material for various limited-scope purposes including scholarship, review, and parody. While fair use applies only in the U.S., many other nations support similar concepts such as “fair dealing.” It is also intricate: fair use, after all, allows one to use someone else's copyrighted content without their permission and possibly even in ways to which they may object.

DReaM is a Sun Labs initiative to develop a Digital Rights Management solution based on open standards and royalty-free technologies[1]. DReaM seeks to address the interests of copyright holders as well as those of content users, and supports fair use in those jurisdictions where applicable.

3. Assumptions

For the U.S. jurisdiction, DReaM makes no attempt to enforce the legal rules that govern fair use, in part because what constitutes fair use is not easily defined and depends very much on the specific purpose and facts of the activity performed. Courts are ultimately the arbiters of what constitutes fair use. This lack of a clear line means that it is impossible to have an automatic program that will always correctly determine what activity falls under the "fair use" exception. Instead, if we are to properly support fair use, we must rely on the user's judgement. Of course the user takes some risk in asserting that her usage is "fair." If the copyright holder disagrees, a court may be called in to decide---and it may decide that the user is liable for copyright infringement. Further examples of activities that U.S. Courts have regarded as fair use can be found in many places.[2]

1 http://www.openmediacommons.org/collateral/DReaM-Overview.pdf
2 http://fairuse.stanford.edu/Copyright_and_Fair_Use_Overview/chapter9/9-c.html

From the point of view of the digital rights management system, the user is making an assertion that the usage is fair use, an assertion that the DRM system trusts. To dispute the user's assertion, the copyright holder relies on out-of-band systems for resolution, e.g., communications between the user and the copyright holder or legal action. The DRM system supports only the fair use assertion and the decryption of the content for that usage; the DRM system cannot judge the validity of the fair use assertion itself.

The fair use architecture is specifically targeted towards the U.S. jurisdiction where fair use applies. For other jurisdictions, parts of this architecture may apply; however, legal consultation should be obtained.

4. Anonymous Fair Use

Fair use is enshrined in law[3]. While there is not, to our knowledge, specific case law regarding anonymous fair use, there is case law regarding both fair use and anonymized speech. Specifically one need not approach the copyright owner to request permission before making a fair use while U..S. law permits anonymized speech and publication. Thus DReaM supports anonymous fair use, albeit with tracking capability.

In order to make an assertion of fair use while using DReaM, the user must have legal rights to the content, perhaps by purchasing it, by borrowing it from a library, or as a loan from a friend who has purchased it. Thus some form of credentials have been provided by the user to the Service Provider. Note that the credentials do not necessarily identify the user in any aspect since the DReaM specification permits the user to maintain a level of anonymity. It should be possible for the user to register themselves by providing a sufficient amount of credentials. For conditions where anonymity is not supported, e.g., in0 e-commerce transactions requiring identification or IP logging, additional measures must be taken.

5. Tracking

Copyright holders want to ensure that content released for purposes of fair use is used only for that purpose. We have chosen to develop an architecture to support fair use in a way that enables the user to exercise his or her rights anonymously from the copyright holders but also provide to copyright holders a means to identify the user if they believe that content provided under fair use has been misappropriated. The holder can use a tracking technique for identifying such users. Tracking, in this context, is defined to be the following - the method of identifying the source of the "leakage" of content.

DReaM supports technical means for enabling this passive style of tracking content. One common class of tracking methods is based on watermarking. DReaM provides a means of offering watermarked content; however, watermarking itself is not currently part of the DReaM specification. In further sections, we will use watermarking as part of the tracking and auditing processes.

3 http://www.copyright.gov/title17/92chap1.html#107

There are different entities which may perform watermarking of the content. It could be applied by the servers of the service provider, the client device, or even by an intermediary subsystem. Watermarking the content at the client is complicated; it requires a trusted client as well as a watermarking system that is trusted by the service provider. There may also be additional resource constraints on the client which may furthermore hinder the process. Watermarking at an intermediary may relieve the resource burden from the clients or the service providers; however, the intermediary is entrusted with a considerable amount of information on the tracking process, something that may not be in the best interest of the user or the service provider. The service provider will most likely prefer to do the watermarking on the content since they will have more flexibility and control over the strength of the watermark as well as the choice of tracking technologies to track the content.

6. Auditing

The interests of the copyright holders are protected by service providers, parties who routinely store and deliver content with the assumption that there are contractually binding agreements between the copyright holders and service providers. Even though the copyright holders are responsible for protecting their copyrights, the service provider is responsible for protecting the copyrighted content itself. Thus, DReaM provides to service providers an auditing process that uncovers the identity of users associated with watermarked content while ensuring that the anonymity of users not related to the misappropriated content. Details are discussed in a further section.

7. DReaM Architecture and Fair Use

The DReaM architecture supports anonymous fair use through the Anonymizing Agent acting as the intermediator between the Client and the Service Provider to address the concerns of the users and copyright holders. The architecture has the capability of extending and adapting as new technologies and methodologies become available. The Anonymizing Agent is a third-party entity separate from the Client or Service Provider and operating independently with different interests.

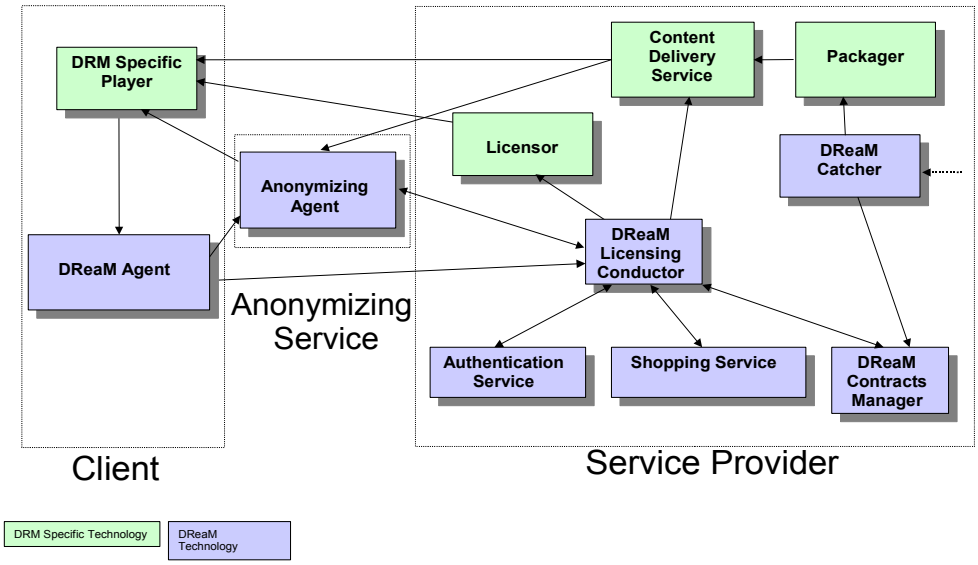

Figure 1 DReaM Architecture supporting Fair Use through the Anonymizing Agent.

In Figure 1, the Client represents the application and/or device through which the user interacts to receive services and content from the Service Provider. The Service Provider offers services and content enabled by different components in its subsystem such as managing contracts, delivering content, etc. For a user to obtain content for fair use, the Client communicates through the Anonymizing Agent to maintain anonymity from the Service Provider. Details on the individual components within the Client and Service Provider can be found in[4]. More details on the process will be discussed further in the following sections.

As you will see from the Architecture, the Anonymization Service learns a great deal about the user: her identity, the content she is choosing to view under fair use, etc. This potentially puts a great deal of power in the Anonymization Service, which must therefore clearly be one that can be trusted. The circumstances under which the Anonymizing Agent provides any personally identifiable information about the user to anyone must be clearly stated in a description of terms of use that the user can inspect before asserting fair use. Privacy protections for the user are a combination of technical, business, and legal protections; in this document we specify only the technical ones.

7.1 User Registration

First, the user must have an account with the Anonymizing Agent before invoking the fair use process. The account may be created at the Anonymizing Agent through a webpage or by the Client application. The Anonymizing Agent must validate the user through the Service Provider in order to create an account. It is important to note, the user's identity will be revealed to the Service Provider only at this time, exclusive of audits. The identification created with the Anonymizing Agent should be different from

4 http://www.openmediacommons.org/collateral/DReaM-Overview.pdf

the account with the Service Provider. Also, the ability to register at any Anonymizing Agent or at several, must be permitted.

7.2 Identifying Content

The Anonymizing Agent must be made aware that the user has rights to the content (has purchased it, borrowed it from a library, etc.) By registering the entire content list of a user's rights at the Anonymizing Agent, privacy will be lost to the Anonymizing Agent. At most, only the titles of fair use asserted content should be available to the Anonymizing Agent. This can be accomplished by associating a Content Identifier Token (CIT) with each content title within a user's playlist (a list of content titles for which the user has usage rights) to identify the content. The user has sole control over the playlist. Thus any information provided from the playlist to an outside entity can only be carried out with the user's permission. The CIT shall contain the following fields: UserId, ContentId, ServiceProvider, and Signature. The user or Client must not be identifiable from the content name itself. There is no question that the contents of the CIT is easily guessable; however, as shown in the following process, the authenticity can be protected. To assure non-repudiation of the CIT, it is recommended that the Client and Service Provider follow trusted computing practices such as using the Java Card™ to protect their private keys. The creation of such a token occurs when the user obtains rights on a content. The following steps are outlined:

1) The Service Provider generates the CIT with the information as described above.
2) The CIT is signed by the Service Provider to ensure the authenticity of the information and the signature is appended as part of the token. The resulting token may be encrypted with the Client's public key to protect the privacy of the user.
3) The CIT shall always be available to the Client and may be downloadable and stored on the Client in the playlist or online. The Client and user must be able to retrieve these tokens anonymously at anytime.

This process supports models where the playlist can reside on the server and/or the Client as long as the Client can anonymously retrieve these tokens. The Client or User must not be identifiable by the Service Provider from any fair use assertions and content without the auditing information from the Anonymizing Agent. Details on how the CIT is used is discussed further in the following sections.

7.3 Fair Use Assertion

Once the user has been registered, the user may assert fair use on any of the owned content. To start the fair use assertion process, the Client must present to the user a registration form listing the fair use purpose as shown in the example below. This form may be retrieved from the Anonymizing Agent or already pre-exist on the Client.

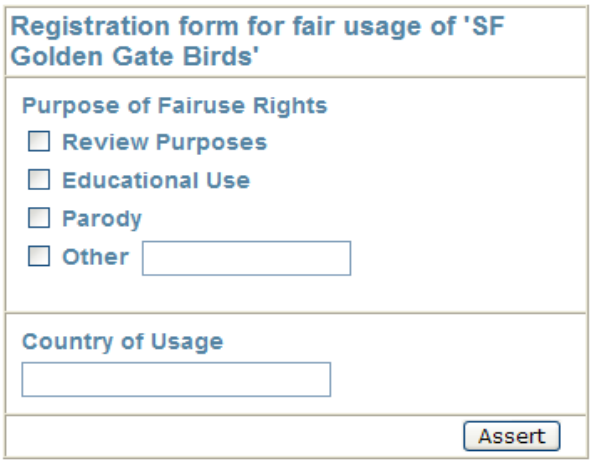

Figure 2 Example Fair Use Registration Form

Note that in Figure 2, such a form must include an “other” purpose since it is impossible to fully anticipate all fair uses. Additionally, from the perspective of the copyright holder, the form educates the user that fair usages are for certain purposes.

After the user completes the fair use registration form, the Client shall initiate the communication with the Anonymizing Agent to receive the content purposed for fair use by the Service Provider. Figure 3 depicts the overall flow of messages between the Client, Anonymizing Agent, and Service Provider.

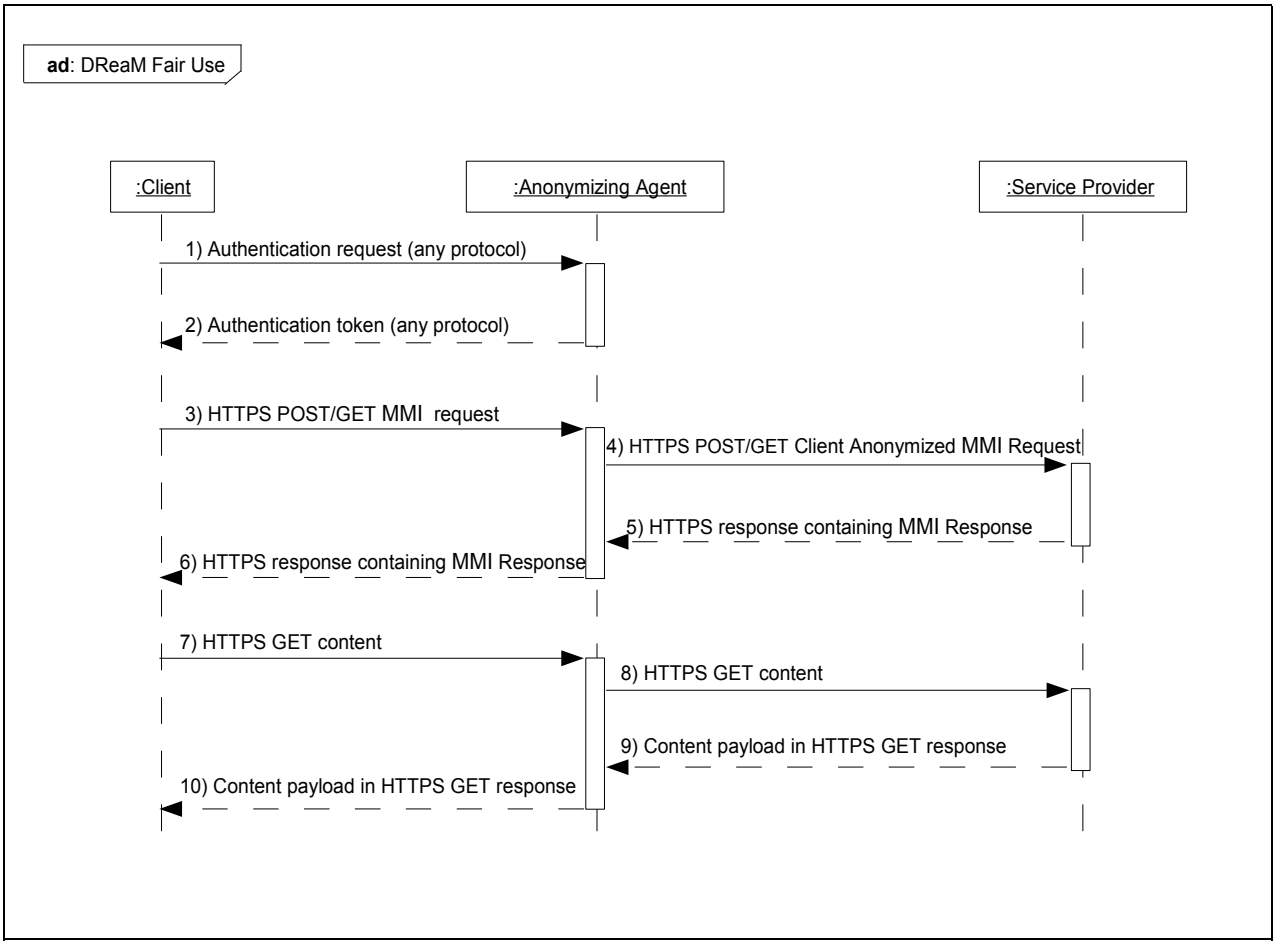

Figure 3 DReaM-MMI Message flow

7.3.1 Authentication

The Authentication process may be implemented in any manner and are not considered normative in this architecture. However, there normative condition is that the authentication process must not result in revealing to the Service Provider the User's intent of asserting fair use.

Step 1. The Client contacts the Anonymizing Agent to authenticate the user to start the assertion process. The authentication process may be unique to the Anonymizing Agent, federated by another party. The Client may already have received an authentication token from the Anonymizing Agent, thus continue on to Step 3.

Step 2. The Client shall receive from the Anonymizing Agent an authentication token as part of the response. This token shall be used by the Anonymizing Agent to identify the User as the source of the fair use request.

7.3.2 DReaM-MMI Communications

The DReaM-MMI Fair Use protocol follows a more stringent set of requirements thus will result in a subset of a generic DReaM-MMI protocol. The normative form of this exchange is presented in this section including the description. For details on the syntax of the DReaM-MMI protocol for fair use, please refer to Appendix A.

Step 3. Client

```
message = createRequestMsg(authToken, CIT, Reason, Jurisdiction)
sendToAnonAgent(message)
```

The Client sends a fair use MMI message of type "MMIRightsRequest" and profile "org.omc.dream.profiles.fairuse" to the Anonymizing Agent. The message must contain the AuthTkn, ContentId, Verb with the value of AssertFairUse, and non-null values for the verb specific arguments Reason and Jurisdiction. The AuthTkn shall have the authentication token as its value. The ContentId shall contain the CIT. The constructed MMI Message is sent to the Anonymizing Agent as either a HTTPS POST or GET as DReaM-MMI supports both.

Step 4. Anonymizing Agent

```
serviceprovider_pubkey = getPubKey(message->CIT->ServiceProvider)
client_pubkey = getPubKey(message->AuthTkn)
anonDeviceSegment = getDeviceSegment(message->CIT->ServiceProvider,
        message>DeviceSegment)
if
  isAuthentic(message, client_pubkey) == TRUE &&
  isAuthentic(message->CIT, serviceprovider_pubkey) == TRUE &&
  isAccountHolder(client, message->CIT->UserId, message->CIT->ServiceProvider) == TRUE)
then
  newMessage = copy(message)
  credToken = genUniqCredTkn()
  newMessage = rewriteAuthTkn(newMessage, credToken)
  newMessage = rewriteContentId(newMessage, message->CIT->ContentId)
  if(containsDeviceSegment(newMessage) == TRUE)
        newMessage = rewriteDeviceSegment(newMessage, anonDeviceSegment)
  newMessage = signMsg(newMessage, anonagent_privkey)
  storeInAuditDatabase(message, credToken, CIT)
  sentMsg(newMessage, message->CIT->ServiceProvider)
else
  sendErrorResponse()
endif
```

The Anonymizing Agent acts as a relay to keep the User and Client anonymous. The Anonymizing Agent verifies the authenticity of the message, rewriting portions of the MMI Message that may reveal the identity of the User and Client to the Service Provider, and forwards the new MMI Message to the Service Provider. The Anonymizing Agent verifies the authenticity of the MMI Message source through the

`SignatureSegment`. It then verifies the signature of the CIT in the ContentId field to have originated from the Service Provider. The Anonymizing Agent inspects the content of the CIT to determine if the user has rights to the content. The values of the user name and service provider pair must match an account that the user has already registered as covered earlier in section 7.1. Once this has been verified, it can be deduced that the user has rights to the content since the authenticity and origin of the CIT have been confirmed, and the requesting Client has an account at the Service Provider with the UserId as specified in the CIT. The Anonymizing Agent generates a unique credential token to associate the User with the information in the MMI message. This token must not be sufficient for the Service Provider to identify the User directly. The identifier, the MMI message, and their association shall be stored in the database of the Anonymizing Agent for auditing purposes. The Anonymizing Agent takes the original MMI Message request from the Client and rewrites portions including the `ContentId`, `IdentitySegment`, and `SignatureSegment`. The `AuthTkn` of the `IdentitySegment` is replaced with the credentials token, and the Signature of the `SignatureSegment` is replaced with the signature of the newly rewritten MMI Message by the Anonymizing Agent. The `ContentId` is replaced with the ContentId value from the CIT. Other optional portions of the MMI Message such as the `DeviceSegment` may reveal the identity of the user; thus, these portions must be replaced with an alternatively anonymized identifier which is supported by the Service Provider. Then the Anonymizing Agent forwards the rewritten message to the Service Provider. All network communication between the Anonymizing Agent and the Service Provider must be done using HTTPS protocol.

Step 5. Service Provider

```
if
  isAuthentic(newMessage, anonagent_pubkey) == TRUE
then
  randURI = genRandURI()
  if
        enableWatermark ==  TRUE
  then
        [watermark, watermarked_content] = watermarkContent(original_content)
        storeInAuditDatabase(message, randURI, watermark)
  else
        storeInAuditDatabase(message, randURI)
  endif
  notification = "granted"
  response = createResponseMsg()
  response = insertNotification(response, notification)
  response = insertKey(response, randURI)
else
  notification = "denied"
  response = createResponseMsg()
  response = insertNotification(response, notification)
endif
response = signMsg(response, serviceprovider_privkey)
```

```
respondWithMsg(response)
```

Upon obtaining the MMI request, the Service Provider checks the authenticity of the message through the SignatureSegment. As watermark is optional, the Service Provider may generate a unique watermark identifier associated with a watermark. The non-encrypted content may then be watermarked. The location of the content shall be assigned to a randomly generated URI. The Service Provider stores the MMI message, watermark identifier, content location URI, and their association in the database for auditing purposes. The Service Provider constructs the MMI Message response with the URI as part EMM stored in the Keys field and with the Notification field with “granted”. The Notification must always, exclusive of errors, result in "granted" since the DReaM system does not make legal decisions to grant or reject a fair use requests based on the Reason and Jurisdiction. After signing this message, the Service Provider completes the HTTPS request from the Anonymizing Agent by responding with the MMI response.

Step 6. Anonymizing Agent

```
if
   isAuthentic(response, serviceprovider_pubkey) == TRUE &&
   response->RequestHash == hash(newMessage)  ### see Step 4 above for newMessage
then
   if
            notification == "granted"
   then
            forwardingURI = generateForwardingURI(response->Keys)
            newResponse = rewriteKeys(response, forwardingURI)
   else
            newResponse = response
   endif
else
   newResponse = createErrorResponseMsg()
endif
newResponse = rewriteRequestHash(newResponse, hash(message)) ### see Step 3 for message
newResponse = signMsg(newResponse, anonagent_privkey)
respondWithMsg(newResponse)
```

The Anonymizing Agent checks the authenticity of the MMI Message response through the SignatureSegment and then verifies the association of MMI Message response to the request through the ReqHashSegment. Once both are confirmed, the Anonymizing Agent generates an anonymizing URI and inserts the URI into the EMM. The anonymizing URI forwards --- not redirects --- any requests to the URI provided by the Service Provider in the previous step. This URI is used by the Client to access the content from the Service Provider through the Anonymizing Agent. This proxy role ensures that a level of anonymity is maintained between the user and the Service Provider. It is important to note that the content must not be cached by the Anonymizing Agent. The MMI response received from the Service Provider is modified in the following manner: Keys contains the new EMM, regenerated value for RequestHash, and then re-signed by the Anonymizing Agent. The MMI message is sent to the Client as

a HTTPS response to the request in Step 3.

If any MMI Requests result in an error from a forged message, invalid request, etc., an MMI response with the appropriate status codes1 shall be relayed back to the Client and further progress in the process must be halted.

7.3.3 Content Retrieval

Because DReaM is designed for many different media types, each having their unique retrieval/delivery mechanisms, normatives for retrieving content are not defined in the DReaM architecture. However, this section outlines one manner which content may be retrieved.

Step 7. The Client verifies the message through the `SignatureSegment` and `ReqHashSegment`. In receiving the `Notification` of "granted," the Client retrieves the content using the URI, which is provided in the `Keys` field of the MMI Response. and which refers to a location on the Anonymizing Agent. (The retrieval goes through the Anonymizing Agent to protect the identity of the user and/or Client from the Service Provider.)

Step 8. The Anonymizing Agent determines from its database the corresponding URI and forwards the request to the Service Provider.

Step 9. The Service Provider responds to Anonymizing Agent with the content as the payload of its response.

Step 10. The Anonymizing Agent relays the content from the Service Provider to the Client.

After completion of Step 10, the Client has received the content for fair use. The content may have been watermarked based on the policies of the Service Provider. The user must understand that a fair use grant by the DReaM system does not guarantee that the provided reason is covered under fair use.

7.4 Auditing

If a service provider discovers their content provided under fair use to be misappropriated, they may follow appropriate legal action against the user. A copyright owner or Service Provider seeking to determine the source of misappropriated content may appeal to the Anonymizing Agent to reveal the identity of the user associated with the content. The circumstances under which the Anonymizing Agent provides user information to the service provider must be clearly stated in a description of terms of use that the user can inspect before asserting fair use.

Step 1. The Service Provider discovers watermarked content being distributed in ways that the Service Providers believes violates copyright, providing evidence for this (the "evidence" must fulfill the requirements of the terms of service agreed to by the user). The Service Provider requests the id associated with the discovered content.

Step 2. Upon confirming the requirements of disclosure, the Anonymizing Agent notifies the user that the Anonymizing Agent will disclose the user's identity to the Service Provider.

Step 3. The Anonymizing Agent provides to the Service Provider the identification of the user associated with the request.

The conditions under which the Anonymizing Agent reveals the identity of the user to the service provider, e.g., simple request based on the contract, a subpoena, etc., must be publicly available. Additionally, requests to the Anonymizing Agent and resultant information provided must itself be auditable to prevent misuse of authority. It is also important to note that the Client and Service Providers must not have any ownership of the Anonymizing Agent to avoid conflict of interests.

7.5 Fair Use DReaM-MMI Messages

Below are sample DReaM-MMI Messages following Steps 3-6 of Figure 3.

Request Client -> Anonymizing Agent: Client constructs the MMI Message to request the content "ggbirdsvideo" for fair use and sends it to the Anonymizing Agent

```
MMIVersion=1.0
MMIMessageType=MMIRightsRequest
Identity.AuthServiceId=www.myCA.org
Identity.AuthTkn=48js9/U9dYc72SCxkjGq80fZ
Rights.ProfileId=org.omc.dream.profiles.fairuse
Rights.ReqElem.Id=6
Rights.6.ContentId=VXNlcklkOkFuYWtpbg0KU2VydmljZVByb3ZpZGVyOmh0dHA6Ly93d3cu
bXlTUC5jb20NCkNvbnRlbnRJZDpnZ2JpcmRzdmlkZW8NClNpZ25hdHVyZTp6UFFmeWlJcjhpYWx1RmtVV
0VSc2lVRVFxNGtqbDlKVWU
Rights.6.ServiceId=www.mySP.org
Rights.6.VerbId=1
Rights.6.1.Verb=AssertFairUse
Rights.6.1.Reason=Criticism,Teaching
Rights.6.1.Jurisdiction=UnitedStates
Signature.SigAlg=http://www.w3.org/2001/10/xml-exc-c15n#
Signature.Signature=UY8vlw8CG56+ntP4U90q
```

Request Anonymizing Agent -> Service Provider: Anonymizing Agent rewrites the `IdentitySegment` and `SignatureSegment` and relays it to the Service Provider

```
MMIVersion=1.0
MMIMessageType=MMIRightsRequest
Identity.AuthServiceId=www.mySP.org
Identity.AuthTkn=gjA9sd+E01Wjs89
Rights.ProfileId=org.omc.dream.profiles.fairuse
Rights.ReqElem.Id=6
Rights.6.ContentId=ggbirdsvideo
Rights.6.VerbId=1
Rights.6.1.Verb=AssertFairUse
Rights.6.1.Reason=Criticism,Teaching
```

```
Rights.6.1.Jurisdiction=UnitedStates
Signature.SigAlg=http://www.w3.org/2001/10/xml-exc-c15n#
Signature.Signature=8YR29pwv8B1/9RuIU
```

Response Service Provider -> Anonymizing Agent: Service Provider sends a response with the URI https://www.mySP.org/29Mc6E4NT9dk2gQfe of the requested content for fair use. The value of `Keys` is BASE64 encoded

```
MMIVersion=1.0
Status=RequestOK
Response.ReqElemId=6
Response.6.Notification=granted
Response.6.Keys=aHR0cHM6Ly93d3cubXlTUC5vcmcvMjlNYzZFNE5UOWRrMmdRZmU=
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
```

Response Anonymizing Agent -> Client: Anonymizing Agent rewrites the location of the content in `Keys` field as well as replacing the `ReqHashSegment` and `SignatureSegment`.

```
MMIVersion=1.0
Status=RequestOK
Response.ReqElemId=6
Response.6.Notification=granted
Response.6.Keys=aHR0cHM6Ly93d3cubXlDQS5vcmcvWTdFOWF5V2ZpMkFL
ReqHash.HashAlg=http://www.w3.org/2001/10/xml-exc-c14n#
ReqHash.RequestHash=gDe3XW2rnpW2yRmO92E1op+3E1axekPR3mkQW
ResponseId=1003
Signature.SigAlg=http://www.w3.org/2001/10/xml-exc-c15n#
Signature.Signature=IPX2eZoeg+09Q1Lej2j5NDhi07K
```

The Client accesses the content at https://www.myCA.org/Y7E9ayWfi2AK which forwards to the https://www.mySP.org/29Mc6E4NT9dk2gQfe which is the content itself.

8. Summary

In the U.S., the fair use doctrine permits users to apply fair use of copyrighted content as fair use is an exception to a copyright holder's exclusive rights. DReaM allows the user to anonymously assert fair use in a DRM environment while providing the service provider a means to track abuses. With the introduction of the Anonymizing Agent, a level of anonymity is ensured for the user. However, this anonymity can be removed through the auditing process if the content is discovered to be in violation of fair use rights. Content tracking methods provide the auditing process with the source of "leakage."

9. Acknowledgements

Many thanks to Pam Samuelson for various insights and, in particular, for organizing a workshop to

discuss fair use within a DRM system. That discussion was extremely useful in preparing this document. The authors would also like to thank Hal Abelson, Ben Adida, Larry Lessig, and Mike Linksvayer for help. Their help does not imply any Creative Commons endorsement of this document.

A. Appendix

```
DreaM-MMI Protocol for Fair Use

FairUseMMIMessage = MMIVersion (MMIRequest | MMIResponse)
```

Under fair use conditions, an MMI message contains a version and may be either a request or a response

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

MMIMessageType = "MMIRightsRequest"

IdentitySegment = AuthServiceID AuthTkn

DeviceSegment = [LocationID]

RightsSegment = ProfileId 1*MMIRightsRequestElement

ProfileId = "org.omc.dream.profiles.fairuse"

MMIRightsRequestElement = ReqElemId ContentId VerbElement

VerbElement = VerbElemId Verb Reason Jurisdiction

Verb = "AssertFairUse"

MMIResponse = 1*Status 1*MMIRightsResponseElement RequestHashSegment ResponseId
SignatureSegment

MMIRightsResponseElement = ReqElemId Notification [1#Hint] Keys
[1#RightsErrorStatus]
```