Verifiable Digital Object Identity System *

Alapan Arnab† & Andrew Hutchison
Data Networks Architectures Group
Department of Computer Science
University of Cape Town
Rondebosch, 7701
South Africa
{aarnab, hutch}@cs.uct.ac.za

ABSTRACT

Identification is a two part system comprising of a *token or label* (an identifier) that can be used to reference an entity and a *process* that can be used to create label-entity associations and verify that the reference and entity belong together. There are a number of identity systems for digital objects that provide identifiers (such as the Handle system, the DOI and URIs). However none of these systems provide verification services. The primary application for our proposed system is in a DRM system, where it is necessary to correctly match users' use licenses to the digital objects covered by the use licenses. In such a case, incorrect associations are effectively failures of the system, and could have wide ranging legal and economic impact, depending on the nature of the protected data.

In this paper we present an identity system for digital objects that support verification and the related details such as the identifier format, the verification process as well as a protocol to create identifiers for digital objects.

Categories and Subject Descriptors

K.6.5 [**Management Of Computing And Information Systems**]: Security and Protection; E.0 [**Data**]: General

*This work is partially supported through grants from the University of Cape Town (UCT) Council and the National Research Foundation (NRF) of South Africa. Any opinions, findings, and conclusions or recommendations expressed in this paper/report are those of the author(s) and do not necessarily reflect the views of UCT, the NRF or the trustees of the UCT Council.

†This is the author's version of the work. It is posted here by the permission of the ACM for your personal use.

DRM'06, October 30, 2006, Alexandria, Virginia, USA.

General Terms

Design, Security

Keywords

vdoi, data identity, identifier, identity verification, digital signatures, handle system

1. INTRODUCTION

In computer security, issues surrounding digital identity often revolve around user identity, in particular managing multiple user identities and various associated problems. A user's digital identity plays a very crucial role in determining which services and data can be accessed by that user and thus is a corner stone in the security building blocks of any system. However, what is often overlooked are the issues surrounding *data identity*, which also plays a crucial role as there is a need to ascertain that the user has the correct access permissions for a unit of digital data.

In RFC 2828, Internet Security Glossary, the process of *identification* (an act or process that presents an identifier to a system so that the system can recognize a system entity and distinguish it from other entities) and the closely related *authentication* (the process of verifying an identity claimed by or for a system entity) are well defined concepts, but the building blocks, *identity* and *identifiers*, have no clear definition [10]. Identity verification is defined as a process of "*presenting information to establish the truth of a claimed identity*".

Identity is a difficult concept to define; the Oxford English Dictionary (online version) and many other online dictionaries emphasize the idea of "sameness" or "oneness" to a set of characteristics, often to a group of people. The closest definition of identity, as applicable in a digital world is from "The American Heritage Dictionary of the English Language", which defines identity as "*the collective aspect of the set of characteristics by which a thing is definitively recognizable or known*" [2]. Identifier, however seems to be easier to define. In the DOI Handbook, an identifier is defined as:"
(1) A single unambiguous string or "label" that references an entity (e.g. ISBN 0-19-853737-9)
(2) A numbering scheme: a formal standard, an industry convention, or an arbitrary internal system providing con-

sistent syntax for generating a series of labels ... intention is establishing a one-to-one correspondence ..." [7].

Digital identifiers also need to be globally unique, i.e. the identifier references only one object in the world, regardless of the location of the object. However, an object can have more than one globally unique identifier. This allows for an object to be identified under different identifier schemes, including support for different languages etc.

Thus, an identification system for a unit of digital data or digital object[1](e.g. a file) can be considered to be composed of two parts – a set of labels that can be used to uniquely reference each digital object and a mechanism to verify the identity (i.e. given a set of labels and a digital object, it should be possible to refute the claim that the labels correspond to the digital object, if untrue).

Currently, there are a number of different identification schemes for digital objects. The identifier format for most many such schemes are based on the Universal Resource Identifier(URI), which defines a common standard for expressing identifier protocol and the label itself [10]. However, none of the schemes provide any verification support, and thus we consider them to be incomplete. In this paper we discuss a verifiable identification system for digital objects.

The primary application for our proposed system is in a Digital Rights Management (DRM) system, where there is crucial need to match the correct digital object, its use license and the user. However, this system can be applied to any digital objects including ordinary web-pages and thus could be a solution to check for the defacement of web-sites.

The paper is laid out as follows: in section 2 we discuss why verifiable identifiers are necessary in DRM systems and why current identity schemes for digital objects are unsatisfactory. This section is followed by a discussion on the requirements for an identity system for digital objects in section 3. We then detail our proposed system in 4, followed by a discussion of the security considerations for such a system in section 5, before concluding in section 6

1.1 Scope

The system presented here is aimed for single digital objects and not complex, heterogenous objects. For example, in a web page, the HTML file is a single object, addressable by our system. However, objects embedded by links in the HTML file (for example images) need separate identifiers. Verification of a group of identifiers is not handled in our paper but would be a trivial extension.

2. BACKGROUND

In [5], Gladney discussed the need for a strong identity system for digital libraries. Gladney argued that future users of digital libraries would need a strong assurance for the authenticity of the digital data, and discussed how a strong identity system for digital objects provides this assurance. However, the systems discussed by the author (most of which are discussed below in section 2.1) do not provide any verification service.

[1]In this paper, we would like to define a digital object as "a stream of logical contiguous bits stored as a single unit, typically in a file system on disk or magnetic tape" (adapted definition from Wikipedia) but not consider structures used in programming languages (although some of the concepts discussed could apply).

2.1 Current Digital Object Identity Systems

There are a number of different identity systems for digital objects, most of them independent of each other. However, most modern identity systems make use of the Universal Resource Identifier (URI) as the base for their identifier system. The URI system defines both the syntax for an identifier system and a grammar on how to interpret the identifiers [3] and enjoys almost universal support at application level. For this reason, our proposed identity system also bases its identifier format on the URI.

While URI provides a standard base for creating identifiers, identity systems for digital data also need to provide mechanisms for locating the digital data. Thus, using the identifier on a system (usually networked), the identity system locates the data through the use of resolution servers. Resolution of the identifier does not necessarily locate the data itself, but would usually locate metadata and a direct network address to the data. However resolution systems, such as the Universal Resource Locator (URL) used for web pages have a problem with persistence - digital data can be easily moved to different servers, web sites can be reorganised etc. - and thus do not provide an efficient mechanism as an identifier for digital objects. For this reason, most of the digital object identification systems have focused on allowing for the persistent identifiers – the location of the data can change without a change in the identifier.

Probably the most widely used persistent identity system for digital objects is the Digital Object Identifier (DOI) system [6], which is in turn based on the Handle system [7]. The Handle system is primarily an identifier resolution system similar to DNS. When it recieves a query, it looks up the identifer in its database and finds an appropriate server. The Handle server then returns the server address to the requestor. Often, the resolution is not directly to the data itself, but to a website that has a direct link to the data. For example, in the ACM digital library, the DOI system resolves to the abstract page for a given entry (usually papers from various journals and conference proceedings). The Handle system also provides distributed administration and resolution mechanisms allowing for greater availability. Rosenblatt has previously advocated the use of DOI for DRM systems [8], but DOI lacks verification support (as discussed in more detail section 2.2), and thus we feel it is unsuitable for DRM systems.

Other digital object identification systems include the ARK Persistent Identifier Scheme, which is a persistent identification scheme [6] and the Extensible Resource Identifier (XRI), which is a broad identity scheme aimed at combining multiple resource identification schemes [1]. Although the ARK Specifications recognises that identification is an association between an object and a label, and verification is required [6], the scheme has no verification support.

2.2 Problems with non-verification

The problem with non-verification in current identifier mechanisms for digital objects is clearly demonstrated in the DOI Handbook's numbering system – the DOI handbook always has the identifier **doi://10.1000/182**. Thus edition 1 (released February 2001) has the same identifier as the fourth edition (released April 2004). While this does allow one identifier to identify the latest version, it also means that the identification for earlier versions of the document are effectively lost. Furthermore, the DOI (or any other sim-

ilar system) does not have any mechanism to prove that a downloaded version of the document is the same as the document located through the resolution process. Thus, if the latest version is compromised (by a hacker, virus, disgruntled maintainer or even negligence), there is no mechanism for the user to know that the data is compromised.

The same problem exists in conventional web pages – there is no mechanism to inform the user that the page served by the server is in fact the intended page and not a defaced or outdated page. A verifiable identification system for digital objects would be able to overcome these problems.

2.3 DRM Systems

DRM systems aim to provide "*persistent access control*" for digital objects [9]. There are at least three identity systems involved in a typical DRM system – a system to identify users (both end users and rights holders), a system to identify digital objects and a system to identify use licenses[2]. One of the cornerstones in any DRM system is to ensure that the digital object and the use license correspond to each other; and should there be a mismatch, the user should not be allowed access to the digital object (there is also another authentication step involving the user and the use license).

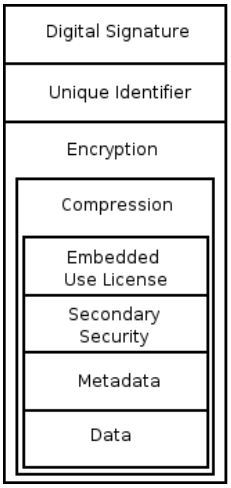

Figure 1: A layered view of DRM Packages

In figure 1 we show a DRM data package as a set of layers as we see it (as far as we know, this exact approach is not used by any current DRM system, but most systems would make use of something similar). Currently, to ensure integrity and thereby establish correspondence, DRM packages and use licenses are signed and a valid signature establishes the integrity of the respective digital objects. But, the digital signature only provides for the integrity of the package and does not actually provide any mechanism to verify that the identifier has any relation to the data, because the digital signature is for the integrity of the entire package.

But the identifier is related to the data and not to the package. In a flexible packaging system, it is necessary to allow for this type of flexibility where packages with the same data require different encryption, compression or secondary security features. However, a use license should not necessarily be restricted to a specific package but rather to the data contained in the package.

The above problem is not crucial in media DRM solutions, where there is typically only one DRM package for each digital object. However, in a broader context, for example enterprise DRM systems, there could be multiple versions of the data and simple user error (or identifier policy) could assign one identifier for multiple versions of the package. This could have serious security implications as it is possible that a user could have access rights to one version of the data but not other versions of the data - but the use license will not have any mechanism to correct this.

[2]Use licenses identify the rights allowed by the rights holder to the end user and are very similar in concept to access control lists in some Unix based file systems.

3. REQUIREMENTS

In section 2 we discussed why we feel current identification schemes are inadequate for digital objects. In this section we discuss the requirements for a digital object identification system, which we have divided into three types: core requirements, optional requirements and security requirements.

3.1 Core Requirements

3.1.1 Globally Unique Identifier Scheme

As discussed in the ARK specifications, naming of digital objects is a political issue and not a technological issue [6]. Names are also common and often share a one to many relationship with objects. Since the intended application of the identity system is global, the identifier scheme needs to create globally unique identifiers. While an identifier should only resolve to one object, an object can have more than one identifier.

3.1.2 Persistent Identifiers

Even if an object is totally removed from public access, there is a high probability that the object was replicated. Due to the global nature of the Internet, the identity of the digital object must not be tied to the location of the object. Once an identifier is issued to an object, the identifier should not be reassigned to another object [1].

3.1.3 Versioning Support

Because an identifier can refer to only one object, there is a need to provide versioning support. Current digital object identification schemes often leave versioning as an optional component, but verification will require versioning.

3.1.4 Wild Card Query Support

Identifiers should support wild card queries from applications.

3.1.5 Identity Verification Service

Given an identifier and a digital object, it should be possible to examine the truth of the association between the two entities.

3.1.6 Federation of Verification

Any server that is involved in the proposed scheme must be able to verify any identifier from the system. If the server does not have the necessary information, it should be able

to forward the request to the appropriate server, and return the information to the requestor.

3.1.7 Federated Resolution

The DOI Handbook describes identifiers using the Handle service as "actionable identifiers" [7]. This means that given an identifier, the server can forward the user to either a service that gives information about the data, including a direct download of the data or a direct link to the data. Thus, resolution can be defined as a process that links an identifier to information about the identifier [7].

Any server that is involved in the proposed scheme must be able to provide resolution service for identifiers; although, like federation of verification, they could choose to forward the request to the appropriate server, and return the information to the requestor.

3.1.8 Encryption Independence

It should be assumed that verification and resolution requests will not be signed or encrypted and thus the system should not require the use of any encryption service. However it could make the use of encryption an optional service.

3.2 Optional Requirements

3.2.1 Internationalisation

The roman character set is not the only character set in use around the world. For this reason, it would be useful to allow for internationalisation of identifiers and other services.

3.3 Security Requirements

3.3.1 Chain of Trust

There is a need to establish a chain of trust, as verification is useless if the process itself is not trusted.

3.3.2 Privacy and Data Confidentiality

It should be possible to assign an identifier of a digital object without revealing neither the details of the owner of the digital object nor the actual object itself. The later is particularly important, not only from a data confidentiality point of view, but also for practical reasons (if data sizes are too big, then communication time and cost will make the system unpractical).

3.3.3 Access Control

There needs to be limited access to the records of a resolution and verification service.

3.3.4 Data Integrity

If records are replicated to other servers (for caching), there is a strong need to ensure that data integrity is preserved.

4. VERIFIABLE DIGITAL OBJECT IDENTITY SYSTEM

There is a simple solution to allow verification – with every mapping between the identifier and the object, include a digital signature of the digital object. Thus, resolving an identifier would not only locate the digital object, but would also verify whether the object is the intended object. A digital signature would also allow for verification of the object either offline (by including the signature as part of the identifier tag) or online through a related web-service. In the remainder of this paper we discuss the Verifiable Digital Object Identity (VDOI) System, which we believe is a better identity system for digital objects.

The VDOI system can be broken into four components:

1. Identifier Format
2. Identity Verification
3. Identifier Resolution
4. Management of Identifiers

4.1 Basic Setup

Like DOI, the VDOI is also an extension of the Handle service [7]. By basing the system on the Handle service, the identifier format and the resolution process will follow the protocols of the Handle service. This also provides persistence of identifiers. The VDOI system will also have a web service frontend that will handle the verification and management functions. SOAP will be the basis of the communication protocol for the web service functions.

4.2 Identifier Resolution

The VDOI System will based on the the Handle service, and thus will follow the protocols defined in RFC 3650 [12]. This should also mean that a Handle server should be able to resolve a VDOI identifier and vice versa.

Resolution should be possible by any VDOI server and can be handled in two ways. Firstly, the servers could keep a store of all possible identifiers and corresponding resolution addresses; but this could be a lot of data to store and thus be impractical. However, there should be a few "root" servers that could handle such storage in case of failures of original servers. The second approach would be to forward the resolution request to the appropriate server and then return the response back to the requestor. In this scenario, the server could also cache frequent requests for faster access.

4.3 Identifier Format

As explained earlier, VDOI is based on the Handle system and thus the identifier format is also based on the Handle service. It should be possible to extend this format to cater for internationalisation through the use of Internationalized Resource Identifiers (IRIs) as detailed in RFC 3987 [4]. However, the current scheme presented in this paper, unfortunately does not conform to the IRI specifications. The proposed format is detailed below:

vdoi://dir_id.reg_server/object_class/id/version

The *vdoi* tag represents the service identifier. The *dir_id* and *reg_server* are part of the Handle service protocol for identifiers and this identifier is thus compatible for resolution with any Handle service. Verification support can however only be provided by the VDOI service. The registration server is responsible for the allocation of the actual identifier. The directory identifier represents the server that allocated the identity of the registration server (for example, 10 represents the DOI foundation). The directory identifier is handled by the Handle system.

The *id* is generated by the registration server and can be in any alpha-numeric scheme desire. It is left up to the registration server to make sure that the id is unique. Combining the unique id with the rest of the identifier guarantees global uniqueness. Like the *id*, the version scheme does not have a prescribed format. A suggestion is to use **MajorVersion.SubVersion.MinorVersion** format. Thus two objects of different identifiers may have the same id, but by using different version numbers have globally unique identifiers. It is recommended that objects with different versions keep the existing identifiers (or maybe change identifiers at major versions). The major advantage is the flexibility in licensing, as licenses could therefore implement a wildcard scheme for access to objects.

The *object_class* tag is a feature aimed to ease administration of identifiers. We propose to use a set of integers to denote these classes, and thus a standardised mapping could be useful. This approach also increases the number of identifiers that can be used by the registration server. Identifiers must be case insensitive. Although the handle system does prescribe the use of case sensitive identifiers, the DOI foundation have commented on the complexity of such a system [7].

4.4 Identifier Verification

When an identifier is issued to a digital object, a signed hash of the digital object is stored along with the resolution and supplementary information (identifier of owner of digital identifier for example). The signed hash will be used for the verification service provided by the VDOI server.

For verification, the user (or service) submits the identifier of the digital object in question as well as the hash of the digital object (taken by the user or service) to the VDOI. The server will then verify the submitted information in relation to the information stored in its own database, and return either a message confirming validity or invalidity to the requestor. Thus this process removes the user's knowledge of the true identity of the hash if the object does not match the identifier reducing the chance of a successful attack in attaching a false identifier to a digital object.

Any server should be able to provide verification for a VDOI identifier. VDOI servers could approach this in two ways. Firstly, the servers could keep a store of all possible identifiers and corresponding hashes; but this could be a lot of data to store and thus be impractical. However, there should be a few "root" servers that could handle such storage in case of failures of original servers. The second approach would be to forward the verification request to the appropriate server and then return the response back to the requestor. In this scenario, the server could also cache frequent requests for faster access.

A self verification scheme can also be supported if the digital object is wrapped in an envelope with the identifier and the digital signature from the VDOI. However this scheme does allow for the possibility of attaching a false identifier to a digital object if both objects have the same hash. With the use of a strong and secure hashing algorithm however, the chances of successfully creating such an attack can be substantially lowered.

4.5 Management of Identifiers

Management of identifiers fall into two categories – the registration of a new identifier and the maintenance of the identifier. Because the identifiers are persistent, there is no need to delete an identifier once an association has been made. Maintenance of identifiers include updates of resolution addresses and updates to meta-data of the registration such as owner of the data object, copyright information and search terms.

Updates of meta-data and resolution addresses imply that other servers that had the information will also need to be updated. However, it is highly likely that updates will not be regular, thus synchronising servers at regular intervals should be sufficient.

In the remainder of this section, we present a protocol for registering a new identifier. All communication must take place using signed SOAP messages and through an established encrypted tunnel (like an SSL session). It is assumed that the server has access to the user's public key.

```
V   => VDOI Server
R   => Requestor

Rid  => Requestor Identifier
Vid  => VDOI Server Identifier

Oc  => Object's class
Ov  => Object's version
Os  => Object's digital signature
Oid => Object's identifier
Ooid=> Object's old identifier (optional)
Or  => Object's resolution address

ad             => Additional Data (Optional)
t1, t2, t3, t4 => Timestamps
n1, n2, n3     => Nonces

R->V: Rid, Oc, Ov, t1, n1, Ooid
V->R: Vid, Rid, Oid, t2, n1, n2
R->V: Rid, Oid, Os,  Or, t3, n2, n3, ad
V->R: Vid, Rid, Oid, t4, n3
```

In the first step, Ooid refers to an existing identifier if this is a registration for a new version. Only the server that issued the existing identifier can issue a new version. The nonces are used to maintain linkages between communications, while the timestamps maintain freshness of messages. Timestamps are also useful in detecting dead connections should a requestor not follow through with the protocol. Additional information in step 3 could be information required by the registration server.

In step 2, an identifier is set aside for a set amount of time. The time interval allows the requestor to add the identifier to the object (for example in the title of document). Step 4 serves as a confirmation of registration for the object. An example of the XML messages (step 3) is shown in figure 2.

Once the server acquires the object's digital signature, it extracts the hash and signs it. The registration server never gets a copy of the digital object, just the metadata, the hash and the respective identifier. Thus the VDOI system can be used for sensitive data on an open network like the Internet. This promotes data privacy and security especially as data itself does not have to leave the control of the owner to receive an identifier.

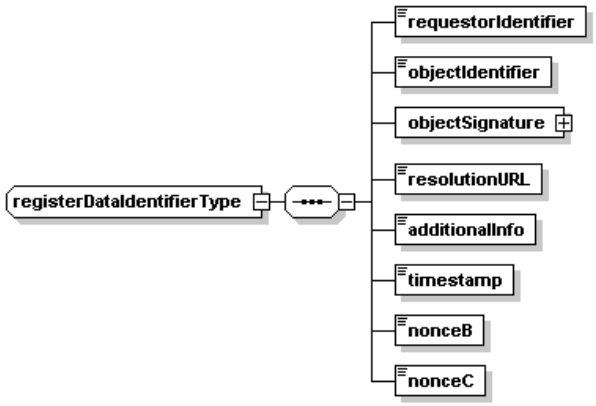

Figure 2: XML schema for *registerDataIdentifierType* type for the VDOI system

4.6 Chain of Trust

In the system, the user is always trusted to provide the correct data. The original registration server is also trusted not to tamper with the data (but any cached copies can be seen to be untrusted).

It is the registration server's responsibility to maintain and secure the records. Some of this functionality is provided by the Handle service framework. In an implementation, the registration and verification service is a business opportunity and thus there will be economic incentive for the service to maintain and secure the records.

5. SECURITY CONSIDERATIONS

Verification of identity is a security service, and thus there is a need to ensure the integrity of the service. However, the service provided is essentially a public one, and thus accessibility of the service also needs to be taken into account. The identification system provides the following classes of services:

1. Resolution of identifiers
2. Verification of identifiers
3. Management of identifiers
4. Management of the system (administration etc.)

Resolution and verification are free public services that must be able to function anonymously. Registration and management of identifiers (for example the change of resolution address) could be a paid service and thus may have restricted access. Management of the service is a private service and thus must have restricted access. For the rest of this section, we examine the services against the 5 security services identified in ITU's X.800 specifications as well as availability, which is not explicitly mentioned in X.800 [11].

5.1 Authentication and Access Control

Authentication and access control are services which are controlled by the administrators of the servers. These services are only required to authenticate administrators and for the management of identifiers; and thus restricted to server management.

5.2 Data Confidentiality

This system provides a security service, and thus it is paramount that the data is stored in a secure environment. The use of a secure tunnel for management provide data confidentiality at the transport level. Because the system does not require the actual data being registered, confidentiality of the original data is assured.

5.3 Data Integrity

This system provides a security service, and thus it is paramount that the data stored are correct. The hash of the object is signed by the registration service and this provides for a check of data integrity for the verification. The use of signed hashes of all the stored data could help with data integrity for the remainder of the records. The use of signed messages provide data integrity for all communication.

5.4 Non Repudiation

The use of signed SOAP messages for all communication provide non repudiation for all communication. Non repudiation of requests would depend on the requestor (user) management systems deployed.

The hashes are signed by the registration service and thus provides for non repudiation.

5.5 Availability

Availability is of critical importance for persistent identifiers. We propose the use of multiple root servers that hold copies of all data, and the use of a distributed architecture should allow for a higher tolerance of denial of service attacks or high load of requests.

6. CONCLUSION

In this paper we have presented an identity system for digital objects, which includes the verification of identity. As far as we are aware, this is the first digital object identity system that supports verification of identifiers. Although primarily geared towards the use in a DRM system, the system can be used for other purposes through the use of a generic web service interface. The system we have presented could be used as a mechanism to prevent defacement of web sites; although the approach will not work for highly dynamic web pages.

As part of the system, we have presented the requirements of a digital object identity system, the design of the system including a protocol for registering identifiers. We have also explored the security considerations for such a system.

7. ACKNOWLEDGEMENTS

The authors would like to thank the anonymous reviewers for their comments and suggestions.

8. REFERENCES

[1] Extensible Resource Identifier (XRI) general syntax and resolution specification, 2004.

[2] American Heritage Dictionaries, Ed. *The American Heritage Dictionary of the English Language*, fourth ed. Houghton Mifflin Company, 2000.

[3] Berners-Lee, T., Fielding, R., and Masinter, L. RFC 3986 – Uniform Resource Identifier (URI): Generic Syntax, 2005. URL: http://www.faqs.org/rfcs/rfc3986.html.

[4] Duerst, M., and Suignard, M. Internationalized Resource Identifiers (IRIs), 2005. URL: http://www.faqs.org/rfcs/rfc3987.html.

[5] Gladney, H. M. Trustworrthy 100-year digital objects: Evidence after every witness is dead. *ACM Transactions on Information Systems 22*, 3 (2004), 406 – 436.

[6] Kunze, J., and Rodgers, R. The ARK persistent identifier scheme, 2005. URL: http://www.ietf.org/internet-draft/draft-kunze-ark-10.txt.

[7] Paskin, N. *The DOI Handbook*, 4.0.0 ed. International DOI Foundation, 2004.

[8] Rosenblatt, B. Solving the dilema of copyright protection online. *The Journal of Electronic Publishing 3*, 2 (1997). URL: http://www.press.umich.edu/jep/03-02/doi.html.

[9] Rosenblatt, B., and Dykstra, G. Integrating content management with digital rights management - imperatives and opportunities for digital content lifecycles. White paper, Giantsteps Media Technology Strategies, 2003. URL: http://www.giantstepsmts.com/drm-cm_white_paper.htm.

[10] Shirey, R. RFC 2828 – Internet security glossary, 2000. URL: http://www.faqs.org/rfcs/rfc2828.html.

[11] Stallings, W. *Network Security Essentials – Applications and Standards*, international second ed. Prentice Hall, 2003.

[12] Sun, S., Lannom, L., and Boesch, B. RFC 3650 – Handle System Overview, 2003. URL: http://www.faqs.org/rfcs/rfc3650.html.