The Technical and Legal Dangers of Code-Based Fair Use Enforcement

JOHN S. ERICKSON, MEMBER, IEEE, AND DEIRDRE K. MULLIGAN

Invited Paper

Digital rights management (DRM) mechanisms, built upon trusted computing platforms, promise to give content providers the ability to reliably and deterministically impose rules on end-user experiences with information resources ranging from literary works and scholarly publications to a vast array of entertainment content. These mechanisms are able to constrain the user's local interaction with content by ensuring that only a predefined range of content behaviors may be invoked, by only authorized agents in only authorized hardware, software, and network environments.

DRM represents just the first wave of a class of technologies that aspire to not only implement copyright-protecting usage controls on computing devices, but increasingly to take on the enforcement of a broader set of organizational and public policies. When technical mechanisms for policy enforcement are strengthened by laws and other governmental controls that stipulate their use—and penalize their avoidance or circumvention—end-user freedoms are at risk of being controlled at their most granular level exclusively by parties who write the policies and control their means of enforcement.

This paper focuses on policy enforcement in the specific context of content use. It reviews the concepts and architecture of policy specification and enforcement, citing examples from the special case of DRM, and provides a detailed discussion of how usage control policies are evaluated in DRM systems. Since the expression and interpretation of policies is only one "layer" of the general problem of persistent policy enforcement, we will consider the role that trusted computing systems will play in ensuring that computing agents interpret policies in reliable and deterministic ways. Finally, we will consider the challenges inherent in the construction of technical mechanisms that mimic social policies.

Keywords*—*Copyright law, digital rights management (DRM), fair use, policy enforcement, trusted computing, trusted systems.

I. Introduction

Digital rights management (DRM) mechanisms, built upon trusted computing platforms [1], promise to give content providers the ability to reliably and deterministically impose rules on end-user experiences with information resources ranging from literary works and scholarly publications to a vast array of entertainment content. These mechanisms are able to constrain the user's local interaction with content by ensuring that only a predefined range of content behaviors may be invoked, by only authorized agents in only authorized hardware, software, and network environments.

DRM represents just the first wave of a class of technologies that aspire to not only implement copyright-protecting usage controls on computing devices, but increasingly to take on the enforcement of a broader set of organizational and public policies. Today such mechanisms are being applied in areas ranging from corporate document security, to usage control and privacy mitigation for commercial content and services, to the protection of private data within the enterprise. When technical mechanisms for policy enforcement are strengthened by laws and other governmental controls that stipulate their use—and penalize their avoidance or circumvention—end-user freedoms will be at risk of being controlled at their most granular level exclusively by parties who write the policies and control their means of enforcement.

Whose rules should control the end user's experience? In particular, will technology-based policy enforcement cause the social policies and common practices that have traditionally influenced the copyright process to be replaced by rules privately constructed by content owners and system providers, and privately enforced by a collection of OSs and DRM mechanisms? Conversely, are there ways to apply these emerging architectures that might actually help protect the limitations on copyright owners' exclusive rights, and in particular preserve the flexible fair use doctrine? Will policy makers step in to ensure that DRM systems protect consumers' interest and reflect the balance of copyright law [24]? And given that real-life policy regimes are often dependent upon the context of use and the intentions of users, can code-based policy-enforcement architectures *ever* be adequate enforcers of social policies (such as the U.S. fair use statute) across a borderless Internet?

Manuscript received September 3, 2003; revised December 15, 2003.

J. S. Erickson is with the Digital Media Systems Program, Hewlett-Packard Laboratories, Norwich, VT USA 05055 (e-mail: john.erickson@hp.com).

D. K. Mulligan is with the Samuelson Law, Technology and Public Policy Clinic, Boalt School of Law, Berkeley, CA USA 94720-7200 (e-mail: dmulligan@law.berkeley.edu).

Digital Object Identifier 10.1109/JPROC.2004.827365

This paper focuses on policy enforcement in the specific context of content use. It reviews the concepts and architecture of policy specification and enforcement, citing examples from the special case of DRM. It provides a detailed discussion of how usage control policies are evaluated in DRM systems, especially in the case of emerging rights expression languages (RELs). And since the expression and interpretation of policies is only one "layer" of the general problem of persistent policy enforcement, we will consider the role that trusted computing systems will play in ensuring that computing agents interpret policies in reliable and deterministic ways.

Finally, we will consider the challenges inherent in the construction of technical mechanisms that mimic social policies. These challenges often present themselves under the following circumstances.

- The social policy at issue is not reducible to code [25]; implementations of policy-capable architectures fall short, due to factors ranging from policy languages that are not expressive enough to the inability to obtain or measure the attributes necessary to adequately evaluate a policy and render a decision.
- The act of automating enforcement, regardless of whether the policy is reducible to code, alters the practical effect of the policy [26].
- Policy-enforcement implementations are too weak to deterministically enforce the policies in the face of security threats.
- Parties employ technical measures to constrain the ability of individuals to effectively exercise their legal rights.

II. Controlling Information Use: An Introduction to Policy Enforcement

The design of technical solutions for controlling the use of content must begin with the definition and codification of high-level policies into explicit, machine-interpretable (but also human-readable) expressions or specifications, accompanied by a compilation of such expressions into formats that can be interpreted by a set of low-level system components assembled to enforce such policies. This section provides insight into the general problem of policy expression, including a consideration of the specific problem of usage control exemplified by DRM. By the end of this section, the reader should appreciate the factors that must be considered when creating a technology-based policy-enforcement regime, and in particular the challenge of enforcing policies across a wide range of users and platforms.

A. Understanding Policy Expression

Intuitively, we may think of a policy as a rule that specifies how some entity should behave when presented with a specific set of circumstances. Ponder [2] provides us with a somewhat more useful definition, from the world of policy-based distributed systems management: a persistent declarative specification, derived from management goals, of a rule defining choices in the behavior of a system.

- *Persistent*, in the sense that "one-off" commands to perform actions are not policies. Policies should be relatively static in comparison with the state of the system. Effective governments and organizations do not write laws and policies on the fly; similarly, policies governing the behavior of distributed systems remain fixed.
- *Declarative*, in the sense that policies define choices in behavior in terms of the conditions under which predefined operations or actions can be invoked, rather than changing the functionality of the actual operations themselves. Policies specify what behavior is desired, not how the behavior will be achieved and maintained.
- *Derived from management goals* means that policies are derived from "high-level" or "abstract" policies like laws, business goals, service-level agreements, or trust relationships. In this paper, we focus on the derivation of technically enforceable policies from social policy, especially copyright law.

When we refer to policies in a technical sense, we mean a set of explicit, machine-actionable and (usually) human-readable expressions; this is arguably the preferred situation, but does not strictly need to be the case. For example, policies that have been hard-coded into software or hardware may start with the characteristics listed above; the problem is that the inherent inflexibility of hard-coded policies limits their ability to keep pace with changes to both the high-level goals that led to their creation and the low-level implementation.

In this discussion, we will consider two classes of policies: authorizations and obligations.

- *Authorization* policies define what activities the subject of the policy can and cannot perform on a set of target objects; these are essentially access control policies intended to protect resources from unauthorized use. Authorization policies are typically what we think of as the purview of DRM systems.
- *Obligation* policies define what actions the subject of the policy must or must not perform on a set of target objects when a particular event occurs. Obligation policies are interesting, in that they give us the tools for implementing policies calling for user warnings and alerts, monitoring and logging of actions, etc.

Constraints may be specified to limit the applicability of both authorization and obligation policies, based upon time or the values of attributes of the objects to which the policies refer.

For purposes of discussion, a rough sketch of the composition of a policy (based upon [2]) is illustrated in Fig. 1. Arguably, most policies within the realm of distributed systems management, including usage control for content, may be mapped onto this general model.

- The *mode* of the policy distinguishes between authorizations (positive or negative) and obligations (positive or negative).
- The *trigger* specifies the event that a positive obligation policy applies to. It can specify a wide range of events, from a security violation that an administrator

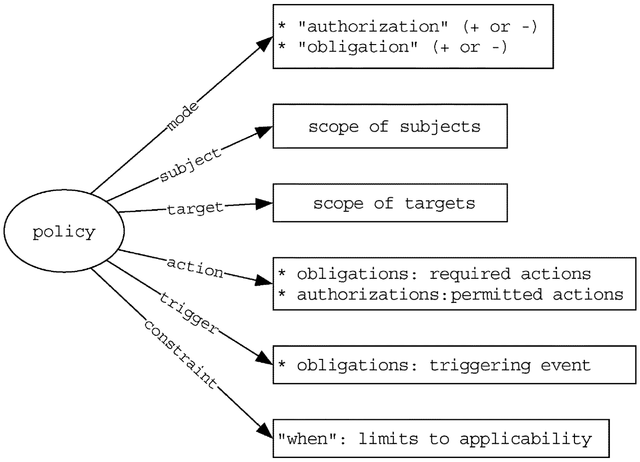
policy
mode
subject
target
action
trigger
constraint
* "authorization" (+ or -)
* "obligation" (+ or -)
scope of subjects
scope of targets
* obligations: required actions
* authorizations:permitted actions
* obligations: triggering event
"when": limits to applicability

Fig. 1. General structure of a policy.

or agent should respond to, to operations against objects that a provider wishes to log, to changing system environmental conditions that should be monitored.

- The *subject* of a policy specifies the scope of human or automated managers and agents to which the policies apply and which interpret obligation policies.
- The *target* of a policy specifies the scope of objects on which actions are to be performed.
- The *actions* specify what must be performed for obligations and what is permitted for authorizations. Actions are specified to the granularity of method invocations; as we will discuss later, the enforcement of authorization policies requires that an ability to intercept method invocations be built into the code responsible for "handling" the content.
- The *constraint* limits the applicability of a policy, e.g., to a particular period, or making it valid after a particular date (i.e., implementing an *embargo*). In addition, the constraint could be based on attribute values of the subject or target objects. Constraints must be evaluated every time an obligation policy is triggered or authorization policy is checked to determine whether the policy still applies, since attribute values may change.

This section has only touched upon authorization and obligation policies. Powerful policy specification frameworks such as Ponder also provide for the consistent specification of information filtering, delegation, and "refrain" policies.[1]

B. DRM: Policy Enforcement for Controlling Content Use

DRM technologies provide content originators with a range of control over how their information resources may be used. The control exercised may have little or no relation to "rights" as defined by copyright law [24]. Rather than implementing "rights," they may be more accurately viewed as supplanting copyright with a private system of control [27]. Just as there are many ways to deploy information in today's digital environment, there are many potential control points where policies may be enforced, not all of which offer the same granularity of control or flexibility. A major challenge for DRM since its emergence in the mid-1990s has been to gain adoption for system components that introduce these control points, either as extensions to popular applications and/or OSs, or as stand-alone systems themselves.

Park *et al.* [3] outline a general taxonomy for the *controlled dissemination of information* that we may overlay on the architecture of policy enforcement. From Fig. 2, inspired by such an analysis, we see that various DRM mechanisms may implement a range of policies, from very limited and coarse-grained usage control to flexible capabilities that maintain originator control over very specific uses of resources over time.

Using Fig. 2 as a guide, the steps for establishing usage policies for resources and enforcing them with a DRM mechanism may be broken down as follows.

1) A set of possible "uses" or actions against the controlled resource must be defined. In today's software environment, these uses are typically defined by the inner workings of specific applications, such as *view*, *print* or *copy*. These implementation-level actions must then be bound to policy-level terminology, either directly or through some contextual filter. The ability to intercept actions must be built into the application and/or system in order for policies to affect use.
2) Policy enforcement is accomplished by a combination of system elements that together implement the controls as specified by the policies. A policy-enforcement point (PEP) may be thought of as an intermediary situated between user applications (viewers, rendering tools, printer drivers, Web services) and policy-setting authorities; the job of the PEP is to *permit* or *deny* the requested use based upon an evaluation of applicable policies.
3) For a given controllable action, there will be some set of *applicable policies* that govern access to that action. From the previous section, we see that these policies may specify *conditions* that must be met prior to the action being allowed, or *obligations* that must be satisfied as the resource is used; typically these conditions will require the presence of a particular authenticated credential and/or environmental attribute. The applicable policy will be determined by the *context* of the attempted use, which may be implied by the nature of the attributes specified by the policy.
4) In some implementations, the policies may be fixed or built into the PEP, or they may be embedded within or otherwise *attached to* the content as it is distributed. Either of these cases is limiting and inflexible; in the first, the originator cannot change the policies once the policy interpreter has been deployed; in the second, the policy cannot be changed once the content has been deployed.

A third case is more interesting, in which the policies are managed externally and separately (in time and space) from

[1] *Refrain policies* define the actions that subjects must not perform on target objects even though they may be technically capable of performing the action. They are used for situations where negative authorization policies are inappropriate because the target objects cannot be trusted to enforce such policies.

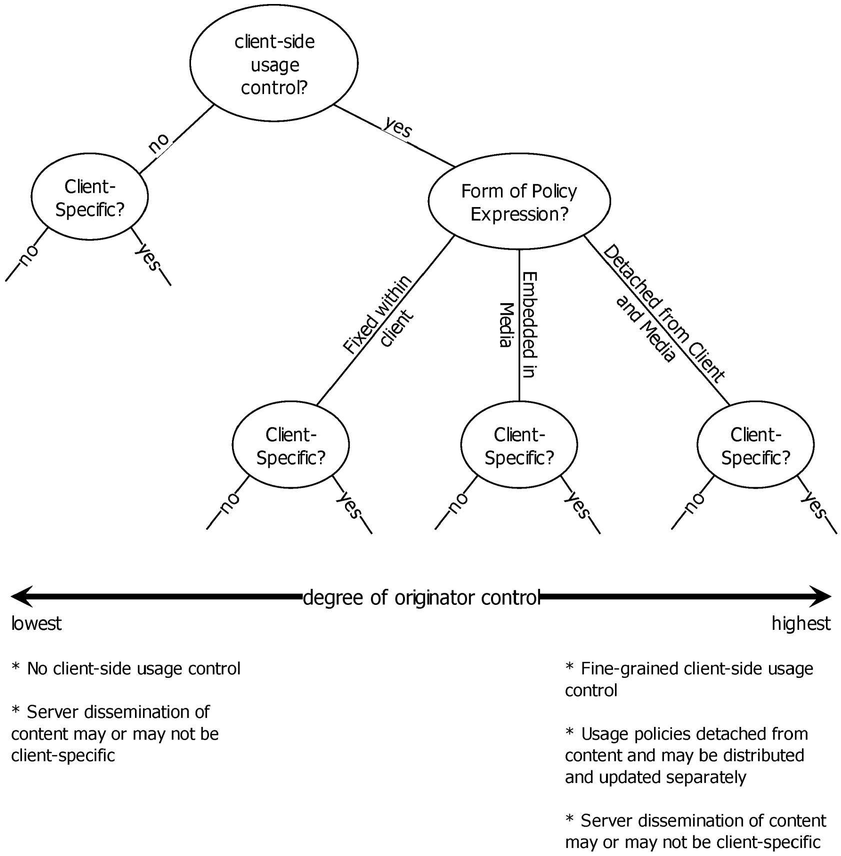
client-side usage control?
no
yes
Client-Specific?
no
yes
Form of Policy Expression?
Fixed within client
Embedded in Media
Detached from Client and Media
Client-Specific?
no
yes
Client-Specific?
no
yes
Client-Specific?
no
yes
degree of originator control
lowest
highest
* No client-side usage control
* Server dissemination of content may or may not be client-specific
* Fine-grained client-side usage control
* Usage policies detached from content and may be distributed and updated separately
* Server dissemination of content may or may not be client-specific

Fig. 2. Spectrum of originator-based usage control provided by DRM.

both the policy-enforcing client and the deployed content. This third model best characterizes most modern, distributed DRM technologies, and is examined more closely in the next section. Both the embedded and external policy sets are typically expressed using some form of REL.

Some combination of the embedded and external policy models is likely; this might occur if the originator wishes to attach certain "default" or generic policies to the deployed resource, which a recipient may subsequently augment through a separate transaction or a provider may revoke, for security or other reasons [4]. Furthermore, policies may be written such that they can be applied to broadly defined *groups* of resources and/or principals, perhaps relating to *roles* within an institution, and, therefore, might be issued well in advance of use. In Section III, we will discuss how this may be used to approximate "fair use" in DRM systems under certain circumstances.

C. A Generalized DRM Reference Model

The evolution of DRM technology has included an increasing adoption of accepted security practice and an application of standardized protocols, especially those built upon XML [5]. Technology providers have been pressured to incorporate open, nonproprietary standards in their implementations, to accommodate those stakeholders who require features such as cross-platform, cross-organizational authentication and authorization (or *privilege management*). Included within the latter would be mechanisms for interoperable policy expression.

One important, emergent standard is the Security Assertion Markup Language (SAML) [6], an XML-based framework that standardizes the exchange of security information. SAML information is expressed in the form of *assertions* about *subjects*, where a subject is an entity (either human or computer) that has an identity within some security domain.[2]

We know of no commercial DRM mechanism that has been built upon the SAML framework and its related standards, although most proprietary architectures may be found to be special cases of the conceptual architecture or *domain model* upon which SAML is based. In light of this, we can derive a generalized DRM reference model based upon the SAML pattern to illustrate how users may be granted and may exercise specific "use rights" to objects under originator control. Particular implementations of this model would incorporate a mix of standardized and/or proprietary infrastructure for identification, metadata, authentication, and cryptography, and would typically lump together the individual functional blocks shown in Figs. 3 and 4.

[2] An example of a subject might be a person, identified by his or her e-mail address in a particular Internet domain—although, for reasons of privacy, this use of a personal identifier might not be appropriate for an actual DRM application. The SAML notion of *subject* is consistent with our previous review of the structure of policies.

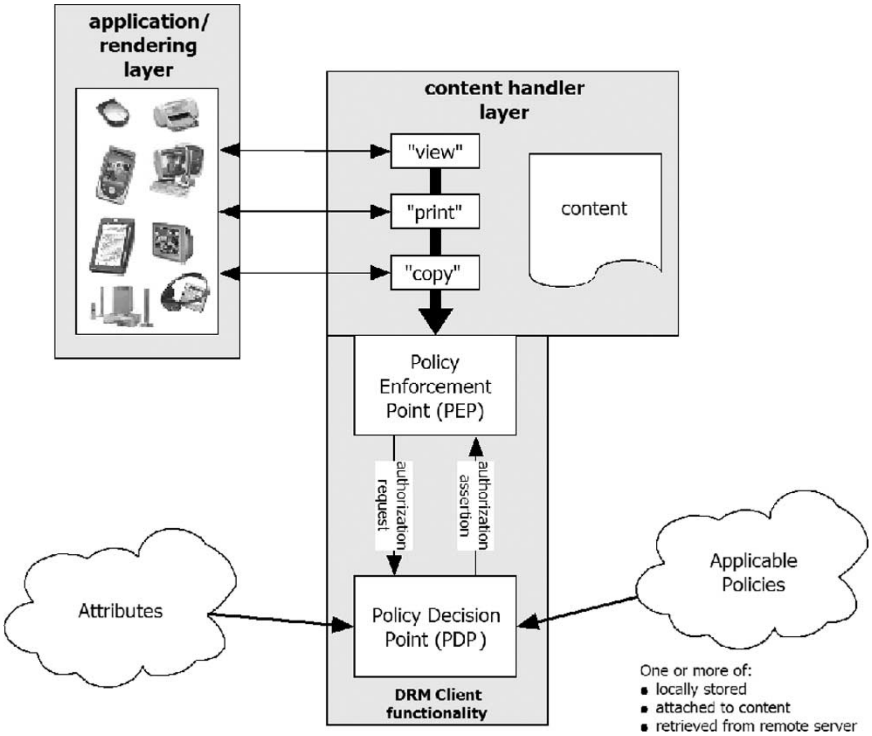

Fig. 3. Schematic of fine-grained, policy-based usage control (user side).

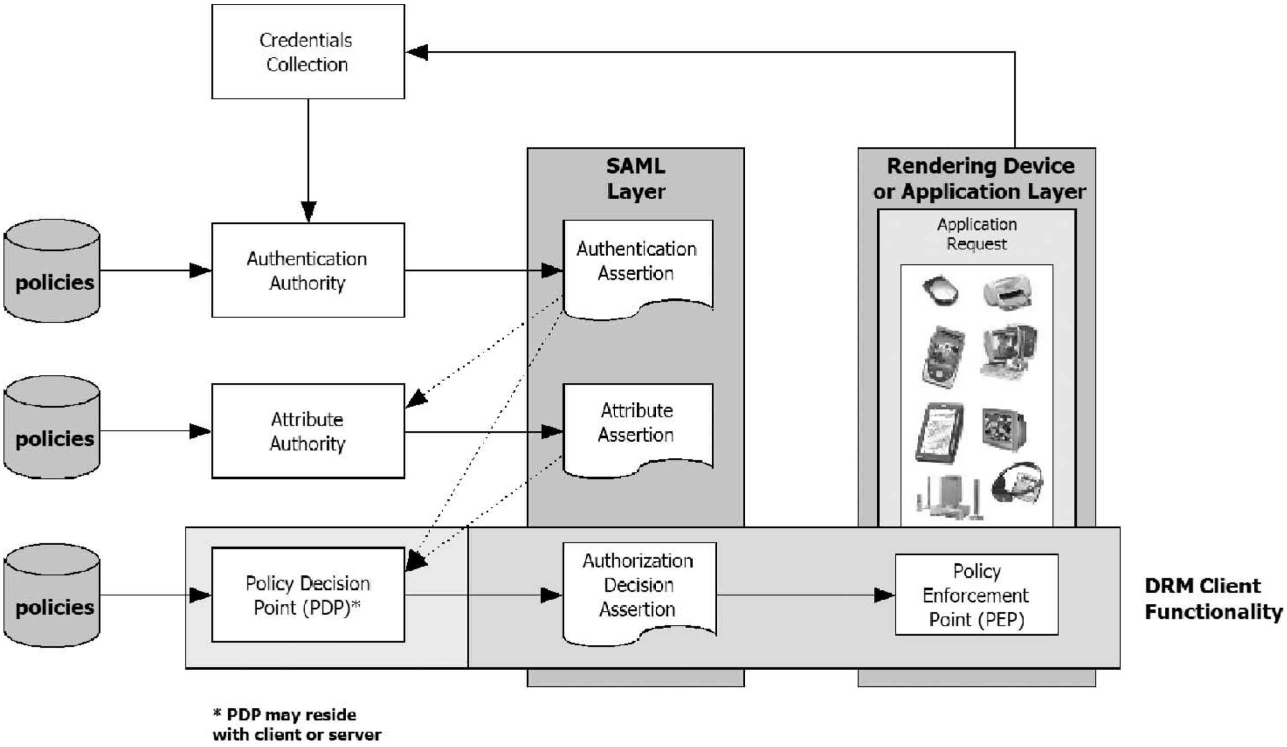

Fig. 4. A SAML-based DRM reference model.

The SAML domain model isolates authentication and authorization functions to a greater extent than today's DRM systems are likely to, but in so doing it highlights the common functional elements present in each of those systems. Also, the SAML model describes the data flow framework but not necessarily the *sequence of operations* followed by these systems.

The following outlines a typical DRM process flow (derived from [21]) that may be layered over this DRM reference model.

1) *Not Shown*: The user obtains a manifestation of a resource by some means, perhaps through file-transfer or streaming protocols, or as a result of a direct request to a file server or through peer-to-peer (P2P) file sharing, e-mail, or direct media transfer (i.e., on removable media). If the resource is retrieved from a remote service based upon a request originated by the user, it may be cryptographically "individualized" to the user's environment.[3]
2) The user attempts some action against the content in some way; the PEP within the context of the rendering application or device determines that the requested action requires authorization. The PEP requires an *authorization decision*, the assertion for which may be contained within an external data package, received from a remote *policy decision point (PDP)* (such as a remote server) that interprets policies, or may be a signal from a local PDP component. If the PEP cannot find a suitable assertion, it passes a request to a PDP, which must locate and evaluate applicable policies.

   Contemporary DRM architectures embed PEPs within rendering applications or within implementations of kernel-level *content handler* application program interfaces (APIs) that provide a repository-like access model for content across applications.[4] Both the application-hooking and repository approaches depend upon external PDPs implemented as system-level extensions. The general relationship of these system components is depicted in Fig. 3.
3) If the PDP cannot locate applicable policies within the user's environment or available policies have expired, attributes of the user's request, including the usage context, are packaged in a message and sent to an authorization authority (a *license server* in the industry vernacular) by the DRM client functionality.
4) The license server must verify the submitted client attributes against an identity or attribute database. Such a database is indicated in Fig. 4 at the point where authentication assertions are transferred from an authenticating authority. Examples might include the license server itself or some third party, perhaps within a single-sign-on infrastructure such as Liberty Alliance [7], Shibboleth [8], or Microsoft .NET Passport [9].
5) The license server determines the applicable policies, if any, for this resource.
6) Policies being evaluated by any of the *authentication authority*, *authorization authority*, or PDP may require some evidence (in the form of an attribute assertion) that a financial transaction has taken place. Such a transaction would take place at this point, if it has not previously.
7) The contents of the license package are assembled. In the vernacular of DRM, this package would include the *rights specification*, various identifiers or attributes, revocation information, cryptographic keys to the content, all individualized to the content, and the context of use. In SAML terms, the rights specification would include either usage policies to be processed by a local PDP or (if already evaluated by a remote PDP) an authorization decision, suitably encoded for the client.
8) The license is securely packaged (including authentication information for the package, such as a digital signature) and transferred to the client.
9) The DRM client authenticates the policies it receives, evaluates the applicable policy (if not done upstream), decrypts the content, and issues an authorization to the viewing component for the particular requested action.
10) The content is rendered or otherwise used, as requested.

The interactions between the DRM client functionality and the license server are carried out using some (typically proprietary) rights messaging or transaction protocol; the "payload" of the messages that make up such a protocol are composed using the vocabulary defined by policy expression language, specialized to the PDP and PEP. It is clear that the ability to fully express both rights *requests* and rights *grants* (or permissions) should be elemental to any rights messaging protocol.

Again, various specializations on this model are possible, but generally most commercial DRM systems will be found to follow this pattern. Note that early-generation approaches to DRM were characterized by a simple passing of cleared content to the rendering application following decryption by the DRM client, without any rigorous authentication of the receiving application and without the benefits of protected execution. From a security standpoint, this approach is risky and is, therefore, the motivation for DRM mechanisms based upon authenticated code and trusted execution and which provide kernel-level support for handling unencrypted content.

From a policy specification standpoint, a disadvantage of previous DRM architectures was that their simplistic content handling model provided no ability to intercept actions

[3] This is the "client specific" aspect noted in the Park taxonomy. In most cases, one cannot expect to control the use of a resource simply by issuing policies; it will also be necessary to secure the content (using encryption) to ensure that the resource can only be actioned in the protected environment. How this encryption is applied will determine how resistant the system is to compromise; at one extreme a single or limited set of keys might be used to encrypt all copies of the information, enabling a single physical copy to be shared with an unlimited set of authorized users. At the other extreme, the system might individualize the secured content to the recipient (or some attribute of the recipient's environment, such as a CPU serial number). The encryption technique chosen by the originator is, thus, an implied form of policy, based upon whether the content is retrieved from an external repository or through simple file transfer (e.g., P2P).

[4] By "repository-like access model," we refer to the uniform, stable, extensible interface models that digital object repository architectures such as FEDORA [22] or CNRI [23] expose to applications. An advantage of handling content in this fashion is that an object's behaviors are application independent, meaning that the policies that control those behaviors will be application-independent as well. OS-based uniform content handlers behave much like local digital object repositories, providing additional capabilities including containment.

and, therefore, no basis for fine-grained policy expression and enforcement.[5] In the next section, we discuss how certain policy languages, specialized for DRM, do provide this ability.

D. Rights Expression: Policy Languages for DRM

So far in this paper, we have mapped DRM onto two frameworks that typically are not applied by the industry; first, onto a general model for the policy-based management of distributed objects (Ponder); second, onto an open framework for security assertions (SAML). These steps were not necessary to provide an understanding of DRM *per se*, but were taken to examine DRM as a specific example of the general policy-enforcement problem. It is essential to view DRM systems in the context of the higher level policies they attempt to proxy, if we are to fully appreciate the difficulties inherent in mapping even more expansive policy regimes onto policy-enforcing architectures. By exposing the root problems of automated policy enforcement within the narrow scope of DRM, we hope to show the challenges—and, perhaps, futility—inherent in code-based public policy enforcement on a grander scale.

Policy languages for DRM—RELs—differ somewhat in how they express relations between the essential entities that compose a policy. For example, the DRM-oriented language XrML 2.1 [10] models rights as highly specific *grants* that define the relationship between a *principal*, a *right*, a *resource*, and a *condition*.[6] Multiple grants may be bundled together in a *license*. There may be several grants from a given *issuer* within a license, as well as several sets of grants, from different issuers. XrML policies evaluate to *deny*; *permit*; or *indeterminate*.

Rights within XrML are specific *verbs* that map to actions within an application domain, defined through an XML schema known as a *content extension*. The definition of particular rights may be shared among applications within a domain; for example, the MPEG-21 working group is defining a set of verbs that will be common to a variety of multimedia applications that implement the MPEG-21 specification [11].

A more indirect approach is taken by the policy language XACML [12], which assumes a highly distributed environment in which all policies, attributes, and decisions may be remotely sourced.[7] One notable difference between XACML and XrML is that a *permit* result may be accompanied by an optional specification of *obligations* that must be fulfilled by the enforcement point prior to allowing access.

XACML also differs in that it introduces the notion of *request context* as a way of "insulating" between language abstractions and the more specific application-domain attributes. This feature apparently helps bridge the gap between specific code implementations and contexts and abstract notions of rights, as expressed in the policies.

E. Trusted Policy Enforcement

The expression and interpretation of policies comprise only the highest levels of a policy-enforcement regime; policy writers must also have some assurance that computing systems will interpret issued policies in reliable and deterministic ways. Issuers must establish trust within this distributed system, at least as it concerns their policies and the resources they have applied them to. *Trusted computing platforms* provide one basis for this [1], [13], [14].

A trusted system must undergo a process of *authenticated boot*, whereby only authenticated components that are part of the chosen, certified profile are loaded by an authenticating boot loader. These components will have been previously tested and signed by some appropriate authority; any component that is a candidate for loading will be required to match the signature that has been stored within the profile prior to loading. The profile generally acts like a "signature" for the configuration.

Components within trusted systems will check the authenticity of components they must interact with. In the future, components may refuse to interoperate with components that they do not trust; the key is to realize that this "trust" is inherently *relative*, and the decision about whether to accept a specific certification is an individual one that must be based upon the needs and wishes of the particular application developer or domain administrator. Applied to policy enforcement, this means that any two given policy domains will not necessarily trust each other, even if both have been implemented with accepted trusted system principles.

In this paper, we are only assuming *relative trust* for a particular context; for our purposes, the issuer will not be concerned about whether the target system can enforce all policies, only the policies that she issued and cares about. We do not assume some pervasive level of control over an entire system, nor do we assume a level of control that requires all actions on a system to be "cleared" through some authority. We require only deterministic and reliable behavior, accompanied by protected execution, from the particular "stack" of system elements that a policy is relevant to: from the implementation of the action, to the system components the action subscribes to, down to the hardware on which the code executes.

How will such trust be established? Typically, next-generation trusted computing environments will be populated by certified, named configurations, which will be characterized by the equivalent of a cryptographically signed registry. Components that must access and interact with remote systems will require those systems to produce

[5] Arguably, this is only a disadvantage from the perspective of the policy specifier. This simplistic approach to DRM has the advantage that the user may attempt a wide variety of unanticipated actions against the content, a flexibility that might not be possible if finer-grained control is exerted.

[6] The XrML authorization algorithm considers (up to) eight parameters: a *principal*; a *right*; a *resource/target* (optional); the *time interval* of intended use; a set of (potentially) relevant *licenses*; a set of "root" *Grants* that should also be considered; a (possibly empty) set of other *contextual information*; and a set of previously traversed *Grants*.

[7] Some actor or *principal* attempts some action against a resource. The domain-specific *request context* of the request is constructed, based upon attributes of the *principal*, the *resource*, and the *environment*. The request context is used as the basis for determining the *applicable policy*; the policy is evaluated, and the *decision* is returned to the enforcement point. XACML policy decisions are similar to XrML, giving *deny*, *permit*, *indeterminate*, and *not-applicable*.

certificates that can be used to demonstrate the authenticity of their configurations. Attributes of these configurations will typically be the basis of usage policies for resources deployed to these systems.

The expansion and contraction of a system's "umbrella of trust" must be highly dynamic to match changes in the real world; in particular, the revocation and exclusion of components must be an integral part of interactions between components within the infrastructure. An important design concept for any policy-enforcing infrastructure must, therefore, be the ability for it to "heal" itself, constantly working to mitigate damage that may be caused by the global propagation of rogue components and other security compromises.

III. Challenge of "Coding" Copyright Law

Earlier we identified several challenges inherent in the construction of technical mechanisms that mimic social policies (see Section I). We now explore those in the context of DRM.

A. Underdeveloped Policy Languages and Missing Attributes

It is clear that only those policies that can be reliably reduced to *yes/no* decisions will be successfully automated. Access control policies that fit within narrow application domains, such as the handling of confidential documents within corporations, may be suited to automated policy enforcement, but policies that are subject to many exemptions or that are based upon conditions that may be indeterminate or external will be difficult or impossible to automate [15], [25], [26].

As we and others have noted, copyright law is difficult (if not impossible) to reduce to code. Some rights are clearly articulated. Even those clearly articulated rights are subject to myriad exceptions, the applicability of which depends upon a variety of factors including role, intent, and purpose of use, just to name a few. A primary defense to a claim of infringement, *fair use*, as discussed in more detail below, is by nature fuzzy.

To date, RELs provide no semantic support for users, or anyone other than the copyright holder, to express rights. Nor do they create opportunities for users to engage in uses that they believe should be considered "fair." The DRM systems provide little opportunity to consider attributes necessary or desirable to yield correct outcomes outside the established semantics.

B. Automating Enforcement Alters the Policy

Regardless of whether policies can be reduced to code, there are additional policy questions posed by the nature of machine-enforced policies [26].

Even in situations where there is general agreement about the scope and application of a rule and the rule is subject to coding—clearly not the case with copyright—efforts to automate the enforcement of that rule may be problematic.[8] For example, individuals must obey traffic signals. In particular, they must stop at red lights. Whether a light is red is easily ascertainable and is subject to a simple yes/no construct. However, drivers sometimes run red lights for reasons that many may agree are necessary or desirable—for example, to expedite arrival at the hospital when carrying a sick individual, or to avoid another traffic accident. Therefore, while there may be wide-scale agreement that the rule should generally be followed and the rule is capable of being coded, there may still be good reasons to avoid machine-automated enforcement that would eliminate the ability of individuals to break the law.

In the area of copyright law, the evolution of the doctrine of "fair use" is tightly bound to the practice of after-the-fact adjudication. Technical innovation has provided individuals with new opportunities to interact with content. Where copyright holders believed those uses infringed on their exclusive rights, the courts were presented with the opportunity to determine whether or not the new uses were protected under the fair use doctrine or prohibited. Efforts to limit the ability to take actions with content that have not been previously supported by the courts as legal will stifle this evolution and risks freezing fair use at a particular historic moment.

Similarly, a side effect of the historic inability to monitor and enforce rules for all uses of copyrighted works has been a peace of sorts between copyright policy and individual privacy. But with the increasing ability to monitor and/or control all uses of a copyrighted work based upon the identity of the user, even for those uses occurring exclusively within the home, DRM systems put privacy and copyright enforcement on a collision course.

In the context of red lights, rather than creating a technology that would physically prevent individuals from running red lights, we have chosen to use technology to identify and fine those who run red lights. This improves and rationalizes enforcement, without limiting human judgment in a manner that could prevent desirable if facially illegal behavior. Where individuals are ticketed based on machine enforcement, they are able to protest the ticket. This ensures that technology neither impedes socially desirable outcomes nor penalizes individuals who seek them. By conflating policy automation with machine-enforced prevention, technology can upset the balance experienced in the application of law.[9]

Within the context of copyright, we must examine the risks of automating enforcement even in areas the limited where it may be possible to agree on the language necessary to do so. There are some transactions between copyright holders

[8] Using H. L. Hart's description such rules exist where there is a "core of certainty" and a "penumbra of doubt" [32]. The authors use this example to illustrate the limits of code, not to align themselves with Hart's concept of law as largely rule based, as opposed to the interpretive model of law set forth by R. Dworkin [33].

[9] Another example is the enforcement of speed limits; automated limiters are possible (or at least *feasible*), but warning systems are arguably more desirable.

and users that may benefit from the reduced transaction costs that technical mechanisms can provide, such as automating requests to use a work that an individual believes is within the copyright holder's exclusive rights. On the other hand, a significant portion of the uses of copyrighted works are unregulated, and those that are may be subject to numerous exceptions based on context, traits of the user, and the user's intentions (to name a few).

The reductive nature of technology has produced DRM systems that prevent many legitimate, noninfringing uses of copyrighted works by individuals [34]. The curtailment of users' rights is matched by an expansion of copyright holders' control over both purchase and postpurchase interactions with copyrighted works. The expansion of copyright holders control over the use of copyrighted works interferes with not only "fair use" of works, but importantly with individuals' expectations of personal use and privacy that are supported by copyright and privacy laws and based on widely shared norms. Given these risks, concern about the design and widespread adoption of DRM systems (especially on trusted platforms) is appropriate.

C. Focusing on Fair Use[10]

The U.S. Copyright Act (17 USC) imposes limits on the exclusive rights[11] granted to originators of creative works. The limits are enumerated in fifteen separate sections of the Copyright Act; librarians and others concerned with the impact of DRM on copyright policy frequently point to the balancing test of fair use (17 U.S.C. § 107) to illustrate the risks of strict DRM systems.

Section 107 states that *the fair use of a copyrighted work, including such use by reproduction in copies or phonorecords or by any other means specified by that section, for purposes such as criticism, comment, news reporting, teaching (including multiple copies for classroom use), scholarship, or research, is not an infringement of copyright*. The section then lists four *nonexclusive* factors that courts must balance in determining whether a particular use is fair:

- the purpose and character of the use;
- the nature of the copyrighted work;
- the amount and substantiality of the portion used in relation to the copyrighted work as a whole;
- the effect of the use upon the potential market for or value of the copyrighted work.

Note that the fair use exemption presents broad factors rather than specific rules. A fair use determination, therefore, requires a consideration of facts on a case-by-case basis. Clearly, then, in the case of fair use, there can be no explicit set of rules that can be implemented and automatically evaluated by computing systems. In addition, a fair use is by definition *unauthorized* and, therefore, does not require any interaction with or compensation to the copyright holder. Indeed, many legitimate fair uses, which typically include criticism, commentary, news reporting, teaching and scholarship, and research, might well conflict with the interests of the copyright holder.

D. Technical Architectures for Approximating Fair Use

The preceding discussion has explored a fundamental truth of policy-enforcing systems: they are designed to *permit* or *deny* requests based upon the evaluation of a set of authenticated assertions. To accommodate even some approximation of actions that may be protected by fair use, authorities must somehow preauthorize a set of yet-unspecified actions that the user may invoke for yet-undefined purposes, which together will provide users with "space" for fair use.

Such a *fair use space* would define the trusted boundary that the originator and user agree fair use will lay within. This space would need to be narrow enough to alleviate the copyright holder's concerns about global compromise of their controlled material, but broad enough to give users enough room for their legitimate, "unauthorized" use. As we have seen, many factors may be considered in the evaluation of a policy and, thus, may contribute to the definition of a fair use space, from the authenticated role, to the specific requested action, to environmental constraints that may be written into the policy.

In [16], Burk and Cohen consider three principles that DRM systems might follow to better accommodate fair use.

- *Coding for fair use*: in which the policy-enforcement regime—the "code"—has been designed to approximate fair use norms. In the context of our current discussion, this could be equivalent to either making the policy-enforcement regime less rigid, such as building useful actions into the application (or content handler) that are not under strict originator control; alternatively, a set of preauthorized policies for such uses could be provided, either with the content or the policy-enforcement agent. This approach might accommodate the "80%" case, in which typical "fair uses" are reasonably anticipated; it also might capture some of the spontaneity that is generally characteristic of fair use claims. It has the weakness that it does not accommodate unanticipated uses, unless the policy-enforcement regime is extremely loose.[12]
- *Key access for fair use*: in which an external decision maker is introduced into the authorization process. The suggested approach, which might be thought of as a usage rights escrow model, has the advantage of injecting human judgment into the flow and can accommodate uses that might be contrary to the interests of the rights holder/originator, if the decision maker is an independent third party. This approach, however, requires the user to actively seek permission for the

[10] See also [15] for the challenges posed by fair use and other limitations.

[11] The *rights bundle* includes the rights to produce copies, distribute copies, prepare derivative works, and render public performances.

[12] Ironically, early-generation DRM systems that provided only content encryption, loosely coupled from the rendering application—regarded by the industry as "weak" in part because they do not provide for fine-grained, policy-based usage control—are a trivial example of this sort of loose policy regime. Once the content is in the clear, it may be used in a wide variety of unanticipated ways.

desired use; this preauthorization of the proposed use changes the fair use dynamic. As Burk and Cohen emphasize, a purely transaction-oriented model is costly, restricts spontaneity, and (potentially) compromises anonymity.

The inclusion of alternative, specialized authorization authorities in the system, whether human or automated, accommodates nontraditional bases for authorization decisions: in particular, this approach allows users' explicit *intentions* to be considered. A rich dialog similar to early online permissions systems [17] would be required to profile the intended use;[13] the attributes collected in such a profile would lead to an authorization decision, either by a human or automated interpreter.

- *Mixed fair use infrastructure*: in which these two approaches are combined to capture a greater scope of fair use. This approach is attractive, in that it follows a fundamental principle of computer architecture: *make the common case fast, and make the less common (or infrequent) case possible*.[14] By this dictum, accommodations for common (or "default") cases of fair use should be "built into" policy-enforcement architectures at the lowest level, reducing the transactional cost and preserving the spontaneity and (presumably) the anonymity characteristic of traditional fair use. This could be accomplished by preauthorizing certain uses within content handlers, either *de facto* by not requiring authorization or by preauthorizing certain actions. Authorizations for unanticipated or infrequent uses would be accommodated through the third-party system.

The proposed infrastructure, an interpretation of which we have diagrammed elsewhere [18], is attractive in that it is consistent with existing policy-enforcement architectures.

Several concerns still remain, however.

- It still represents only an approximation of fair use. The bulk of the model requires some form of authorization (either preauthorized or via some "fair use" transaction) prior to use.
- The discussion largely equates "use" with application functions, while fair use considerations tend to be heavily weighed toward *user intent*. This problem is mitigated to a degree if the architecture is limited to only coarse-grained policy enforcement, and/or accommodates *intention-based* authorization requests.
- The most favorable aspects of the model require certain compromises on the part of content originators that may prove unrealistic. First, approaches to coding for fair use may either expose their content to excessive risk (in their view, due to a loosening of the policy-enforcement regime) or the preauthorized policies might prove too difficult to efficiently specify. Second, as discussed earlier, virtually all DRM mechanisms utilize encryption for transport security, ensuring that their content is handled by a trusted agent; some systems individualize this encryption to the client. Thus, in addition to transferring the appropriate authorization attributes to the client, the authorizing third party must transfer an appropriate key or cause one to be made available. In either case, the third party is dependent upon the packager, under originator control, making this key available.
- The key escrow compromise, while providing a novel way to allow for unanticipated and unauthorized uses, comes at the price of *anonymity*. As others have noted, *fair information practice principles*, particularly collection limitation, disaggregation of identifying and transactional data, and data destruction, should inform the design and implementation of all aspects of DRM [30]. This compromise solution exposes an individual's intellectual and cultural interests and activities to others. Research suggests that such exposure may chill access to information [31]. Divulging personal information prior to using a legally acquired copyrighted work within the confines of the home is at odds with current consumer expectations [15], [28], [29].

E. *"Where Do We Challenge the Code?"*[15]

Emerging technical architectures for policy enforcement give content providers the ability to maintain tighter controls over the use of their works than copyright statutes allow, with no avenue of immediate recourse for the individual user [15], [24]. Traditional content deployment mechanisms have always provided for a narrow set of preferred uses[16] while being sufficiently loose so as to not preclude a wide set of *unanticipated* uses, as well as controlled actions with noninfringing intent. Until now, content owners relied upon the legal system to enforce rules that the deployment and usage "architectures" could not enforce. Typically, the unauthorized use is claimed to be copyright infringement, and the defense is fair use or some other exemption. A court determines if this is indeed the case; regardless, the course-grained architecture allowed the action to proceed *technically*, with the legalities determined later. This inherently favors the individual user.

We have seen that trusted systems may change this dynamic; as generally conceived, they may remove the "healthy ambiguity" present in previous enforcement regimes, and in particular remove the opportunity for unanticipated (and, therefore, unauthorized) actions. In this conventional view of the trusted system, the *code* becomes the ultimate arbiter;

[13] A simple approach would allow the user to specify their intentions in free text, which would be the basis for a human authorization decision in an otherwise automated system.

[14] In their classic text *Computer Architecture: A Quantitative Approach*, Patterson and Hennessey argue that "the most important and pervasive principle in computer design" is to *make the common case fast*: when making a design tradeoff, favor the frequent case over the infrequent case; the performance gains and cost savings due to optimizing for certain occurrences are greater if those occurrences are frequent.

[15] See [27].

[16] A least in the mind of the content originator.

there are typically no technical provisions that allow the individual to disagree with the system's determination, regardless of whether the user may *legally* be in the right. In this sense the system would be broken from a copyright perspective: the system may protect the creator's copyright while upsetting the balance of copyright law by taking away users rights and the ability of new "rights" to emerge through the organic legal process.

Policy-enforcement systems that accommodate variable use requests from individuals could provide those users with a limited ability to "challenge the code," in the sense that they could request authorizations for controlled actions for reasons other than purchase. In extreme cases, these requests might even deliver to individuals technical capabilities not previously installed in content-handling components.

IV. Trusted Content Handling and the Future of DRM

We will soon witness the introduction of DRM mechanisms that take advantage of trusted computing features in different ways. As we have discussed, one approach will expose a protected, uniform content handler interface to applications, creating a uniform point for usage policies to be enforced for a variety of actions in a way that is independent of the underlying media format [19]. Such an approach would also provide a uniform place to apply individualized content security, where media files in a variety of formats could be secured to the individual user and/or platform, as well as protected handling within memory. Widespread adoption of this model means that policy enforcement would cease to be provided by distinct applications (such as with today's various DRM systems), but rather would become tightly integrated with the system and application layers.

We can expect the policy expression mechanisms for these systems to become increasingly standardized and to be extended to include Web services as well; the implications will be that policy management for entertainment content, enterprise documents and distributed services will converge to one model. We should expect emerging standards for use rights expression (e.g., XrML from OASIS and MPEG; XACML from OASIS; ODRL from OMA [20]) to become well established, and their disparate domains (general purpose computing, services and mobile devices) will be pressured to interoperate and possibly converge. Such convergence will be driven by the need for a more user-centric usage model for content as it is distributed through many channels, to a variety of end-user devices.

Finally, consumers will continue to demand flexibility in using and experiencing commercial content in alternative ways (sound tracks for personal videos, etc). An optimistic prediction suggests that the entertainment industries will realize the opportunities inherent in this model and will offer a compromise, and may even distribute certain content in formats that accommodate derivative use—perhaps at added cost.

V. Conclusion

The coming adoption of trusted computing principles within end-user systems promises to increase the commercial practicality of DRM technologies. Trusted computing platforms and the migration of DRM components into the OS is likely to make controlled, conditional access to content a more attractive alternative to information providers and an increasingly common part of the end-user experience.

Our purpose in this paper has been to consider where the world of policy-enforcing trusted systems is headed. We have considered how the emergence of trusted computing environments will affect our use of information, and we have addressed a few of the problems that technologists will face as they attempt to "implement" public policy, including and especially laws, within computing systems. We have focused on the problem of using these emerging policy-enforcing architectures to enforce copyright restrictions, and especially on the challenge of ensuring that fair use and related limitations of copyright law remain accessible to users of information.

References

[1] Trusted Computing Group (formerly TCPA) Main Specification v1.1b [Online]. Available: http://shorl.com/hokestobogyte

[2] N. C. Damianou, "A policy framework for management of distributed systems," Ph.D. dissertation, Imperial College, London, U.K., 2002. [Online] Available: http://shorl.com/bamubrypedrypre.

[3] J. Park, R. Sandhu, and J. Schifalacqua, "Security architecture for controlled digital information dissemination," presented at the Annu. Computer Security Applications Conf. (ACSAC), New Orleans, LA, 2000.

[4] J. S. Erickson, "A copyright management system for networked interactive multimedia," presented at the Conf. Dartmouth Institute for Advanced Graduate Studies: Electronic Publishing and the Information Superhighway, Boston, MA, 1995.

[5] W3C recommendation: Extensible markup language (XML) 1.0, T. Bray, J. Paoli, C. M. Sperberg-McQueen, and E. Maler. (2000, Oct. 6). [Online]. Available: http://shorl.com/hedrosyprefany

[6] Assertions and protocol for the OASIS Security Assertion Markup Language (SAML) (2002, Nov. 5). [Online]. Available: http://shorl.com/dastugritepifry

[7] Identity systems and Liberty specification version 1.1: Interoperability, Liberty Alliance Project. (2003, Feb. 14). [Online]. Available: http://shorl.com/hiverujahibru

[8] Shibboleth architecture v5, M. Erdos and S. Cantor. (2002, May 13). [Online]. Available: http://shorl.com/hofrebabrugiba

[9] Microsoft .NET Passport review guide, Microsoft Corp. (2003, June). [Online]. Available: http://shorl.com/jajemopastajy

[10] eXtensible rights Markup Language (XrML) Core 2.1 specification, ContentGuard, Inc.. (2002, May 20). [Online]. Available: http://shorl.com/fekonypyfysu

[11] MPEG-21 Information technology multimedia framework—Part 5: Rights expression language [Online]. Available: http://shorl.com/gadrevemerasy

[12] OASIS eXtensible Access Control Markup Language (XACML) Committee specification 1.0 (2002, Oct. 8). [Online]. Available: http://shorl.com/hygidrymefriro

[13] P. England and M. Peinado, "Authenticated operation of open computing devices," in *Proc. ACISP 2002*, pp. 346–361.

[14] P. England, B. Lampson, M. Peinado, and B. Willman, "A trusted open platform," *IEEE Computer*, vol. 36, pp. 55–62, July 2003.

[15] Supporting limits on copyright exclusivity in a rights expression language standard, D. Mulligan, A. Burstein, and J. Erickson. (2002, Aug. 13). [Online]. Available: http://shorl.com/gogystatadeli

[16] D. L. Burk and J. E. Cohen, "Fair use infrastructure for copyright management systems," *Harvard J. Law Technol.*, vol. 15, p. 41, 2001.

[17] J. S. Erickson, "Tools and services for Web-based rights management," presented at the WWW8 Workshop W7: Managing Intellectual Content on the Web: Use of the Digital Object Identifier (DOI), Toronto, ON, Canada, 1999.

[18] ——, "Fair use, DRM, and trusted computing," *Commun. ACM (Special Issue on Digital Rights Management and Fair Use by Design)*, vol. 46, no. 4, pp. 34–39, Apr. 2003.

[19] J. Manferdelli. (2001, July) New challenges in embedded security: Digital rights management. Consortium for Efficient Embedded Security (CEES), Boston, MA. [Online]. Available: http://shorl.com/fadrygrisibragry

[20] Open Digital Rights Language (ODRL) Version 1.1, R. Iannella. (2002, Sept. 19). [Online]. Available: http://shorl.com/dygrunumitygu

[21] B. Rosenblatt *et al.*, *Digital Rights Management: Business and Technology*. New York: Wiley, 2001.

[22] S. Payette and T. Staples, "The Mellon Fedora project: Digital library architecture meets XML and Web services," presented at the 6th Eur. Conf. Research and Advanced Technology for Digital Libraries (ECDL), Rome, Italy, 2002.

[23] C. Blanchi and J. Petrone. (2001, Dec.) Distributed interoperable metadata registry. *D-Lib Mag.* [Online]. Available: http://shorl.com/gifopilybupy

[24] P. Samuelson, "DRM {and, or, vs.} the law," *Commun. ACM (Special Issue on Digital Rights Management and Fair Use by Design)*, vol. 46, pp. 41–45, Apr. 2003.

[25] E. W. Felten, A skeptical view of DRM and fair use, in Commun. ACM, vol. 46, no. 4, pp. 57–59, Apr. 2003.

[26] F. von Lohmann. (2002) Fair use and digital rights management: preliminary thoughts on the (irreconcilable?) tension between them. *Conf. Computers, Freedom, and Privacy* [Online]. Available: http://www.cfp2002.org/program/fairuse.shtml

[27] L. Lessig, *Code and Other Laws of Cyberspace*. New York: Basic, 2000, p. 135.

[28] J. E. Cohen, "DRM and privacy," *Commun. ACM*, vol. 46, no. 4, pp. 47–49, Apr. 2003.

[29] ——, "A right to read anonymously: A closer look at 'copyright management' in cyberspace," *Connecticut Law Rev.*, vol. 28, pp. 981–1039, Summer 1996.

[30] J. Feigenbaum, M. Freedman, T. Sander, and A. Shostack, "Privacy engineering for digital rights management systems," in *Proc. 2001 ACM Workshop on Security and Privacy in Digital Rights Management*, pp. 76–105.

[31] *American Civil Liberties Union v. Reno*, 31 F. Supp. 2d 473, 485–487 (E.D. Pa. 1999).

[32] H. L. A. Hart, *The Concept of Law*. Oxford, U.K.: Clarendon Univ. Press, 1961, ch. 7.

[33] R. Dworkin, *Law's Empire*. Cambridge, MA: Harvard Univ. Press, 1986.

[34] D. Mulligan, J. Han, and A. Burstein, "How DRM-based content delivery systems disrupt expectations of 'personal use'," presented at the ACM Workshop On Digital Rights Management 2003, Washington, DC.

John S. Erickson (Member, IEEE) received the B.S.E.E. degree from Rensselaer Polytechnic Institute, Troy, NY, in 1984, the M.Eng.EE from Cornell University, Ithaca, NY, in 1989, and the Ph.D. degree in engineering sciences from Dartmouth College, Hanover, NH, in 1997.

From 1984 to 1992, he was a Systems Architect and Project Leader for Digital Equipment Corporation. In 1995, he cofounded NetRights, LLC, to commercialize his research in technologies for copyright management and administration. NetRights was sold in 1997 to Digimarc (DMRC), a leading provider of digital image watermarking technologies. From 1997 to 1999, he was Vice President of Technology Strategy for Yankee Rights Management. He has spent many years focusing on the unique social, legal, and technical problems that arise when managing and disseminating information in the digital, networked environment. He was also the architect of Copyright Direct, the first real-time, Internet-based service to fully automate the complex copyright permissions process for a variety of media types. His current research at Hewlett-Packard Laboratories, Norwich, VT, touches on areas related to the policy-based management and preservation of digital information based on highly distributed, heterogeneous digital object repositories. He has been an active participant in various international metadata and rights management standardization efforts, and has contributed to a number of government panels. He is a frequent speaker and author on the topics of rights metadata; policy-enforcing digital object architectures; DRM technologies and standardization; and digital information objects, identifiers and services. He was awarded a U.S. patent in 1998 for rights management technologies and services that originated in his Ph.D. research at Dartmouth College; a number of related patents are pending.

Dr. Erickson currently serves on the Editorial Board of *IEEE Security & Privacy*.

Deirdre K. Mulligan received the B.A. degree in architecture and art history from Smith College, Northampton, MA, in 1988 and the J.D. degree from Georgetown University Law Center, Washington, DC, in 1994.

She was previously with the Center for Democracy and Technology, where she worked to advance privacy, free speech and other democratic values on the Internet. In 2001 she joined the faculty of the Boalt School of Law, Berkeley, CA, as Acting Clinical Professor and Director of the Samuelson Law, Technology, and Public Policy Clinic. She is the coauthor of "Privacy in the Digital Age: Work in Progress," *Nova Law Review*, vol 23, no. 2 (Winter 1999). With the Center for Democracy and Technology, she issued a report titled "Square Pegs and Round Holes: Applying the Campaign Finance Law to the Internet—Risks to Free Expression and Democratic Values" (Oct. 1999). She also prepared the report to the Federal Trade Commission of the Ad-Hoc Working Group on Unsolicited Commercial Email (July 1998).

Ms. Mulligan serves on the California Internet Political Practices Commission, which was created, as a result of the rapidly expanding role of the Internet in politics, to examine issues posed by political activity on the Internet in relation to the goals of the Political Reform Act of 1974 and to recommend necessary legislative changes. In addition, she serves on the National Academy of Science Committee on Authentication Technologies and their Privacy Implications to assess emerging approaches to authentication in computing and communications systems, focusing on the implications of authentication technologies for privacy.