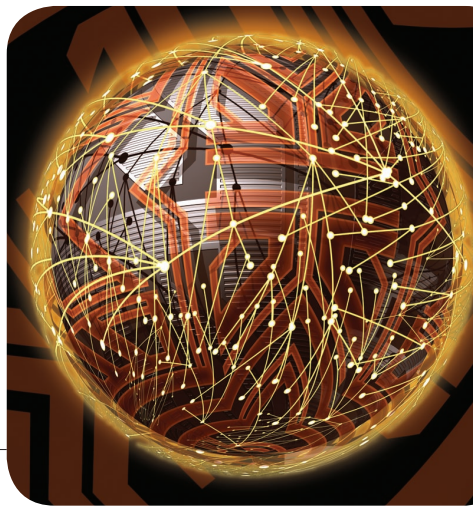
Usage Control Enforcement

Present and Future



For various reasons, both personal data and intellectual property must be protected. The authors explore the state of the art in usage control, which is about controlling the use of such data after it has been given away, and identify room for improvement.

Alexander Pretschner
Swiss Federal Institute of Technology

Manuel Hilty
AdNovum Informatik AG

Florian Schütz
RUAG Electronics

Christian Schaefer and Thomas Walter
DoCoMo Euro-Labs

Computer systems increasingly permeate our daily lives. Personal data is regularly collected when we use loyalty cards, shop online, consult doctors, use mobile phones, or file tax reports. The looming reality of ubiquitous computing will only increase the amount of personal data that's collected, which means more and more digital data is available about us, be it to public administrations, companies, or even publicly on the Internet. Similarly, there's a growing amount of intellectual property distributed digitally and used when watching DVDs, listening to MP3s, or reading eBooks. Companies also exchange intellectual property in the form of trade secrets when they collaborate in virtual organizations.

Various stakeholders are interested in ensuring that these kinds of data are used according to their intentions. For instance, individuals want to ensure that their privacy is respected and protected; companies strive to prevent the disclosure of trade secrets; and both data owners and providers are interested in collecting royalties when copyrighted content is consumed or distributed.

Usage control[1–3] is a research area that targets these problems—it's a generalization of access control that extends authority not only to who may access which data, but also to how the data may or may not be used or distributed afterward.

In this article, we present the state of the art as well as a possible future of control and signaling mechanisms that we call consumer-side enforcement mechanisms. This notion of "consumer" not only refers to the consumers as in "consumer electronics," but also to corporate and any other data recipients. We also present a taxonomy, which defines the properties and capabilities that enforcement mechanisms for usage control have and that they could theoretically have. Because most of the existing technology is used in the context of digital rights management (DRM), the mechanisms that we discuss also originate in this domain.

Background

We call data that's subject to usage control *ucdata*. The fundamental problem of distributed usage control is that data providers want to impose control on how data consumers' processing devices or information systems handle data. Unfortunately for the data providers, however, these machines are usually outside their scope of control or even visibility.

When data consumers request ucdata from a data provider, they might have to commit themselves to a usage control policy that reflects the data providers', data owners', or data subjects' interests. Dedicated mechanisms can give the data provider a limited amount of control over what the data consumer can do with the ucdata. For example, a consumer's media player might be configured in such a way that prevents a song from being played more often than allowed by a policy or in a way that prevents the song from being distributed. Along these lines, a control mechanism is the implementation of an algorithm, a dissemination format/medium, or a combination of both that prevents policy violations.[3] To do so, a control mechanism restricts how objects can be used

or enforces certain actions to be executed at a specific time.

However, there are situations in which imposing this kind of control on data consumers' processing devices or information systems isn't feasible, practical, or sensible. Companies, for example, are reluctant to let other parties control parts of their IT infrastructure. What they might rather be willing to do is provide evidence that their information processing indeed adheres to the stipulated policies. In these cases, deploying observation mechanisms is the more appropriate solution.[4,5] Observation mechanisms can best be explained with an analogy: a police officer can't prevent anybody from jaywalking—but he can fine the delinquents after catching them. Similarly, instead of granting control to the data provider, the data consumer could hence agree to inform the data provider about what happens in the consumer's systems (and of course, this information must be trustworthy). In case of policy violations, the provider can react accordingly. An observation mechanism then consists of two parts: a provider-side monitor that monitors the consumer's behavior and triggers penalties if necessary, and a consumer-side signaling mechanism that sends signals to the monitor to indicate what happens at the consumer's side.

Functional usage control requirements

To elicit requirements in the area of usage control, we conducted several interviews with public administrations, military organizations, data protection officers, healthcare providers, and commercial organizations such as telecommunication providers, banks, credit-card companies, automotive original equipment manufacturers, insurance companies, and retailers. We conducted a further detailed study about usage control requirements in the area of mobile communication.[6] Requirements in the area of DRM, in particular to entertainment industries, are documented elsewhere,[7,8] so we incorporated those findings into our study. We also analyzed existing literature on privacy requirements.[9,10]

Here, we present the distilled results of our requirements elicitation study. When talking about usage control requirements in this article, we exclusively focus on so-called "obligations," which are requirements that govern data's future usage. We first look at the different types of data usage that can be controlled, and then analyze the obligations' typical structure.[11]

Actions and usage

We divide actions on data into two classes. *Usage* is concerned with operations on ucdata, whereas non-usage actions are all those that don't involve ucdata. One example for a non-usage action is making a payment for a previously received movie. Usage can be classified as follows: *black-box usage* is characterized by ucdata not being interpreted in any way (that is, it treats the data as meaningless sequences of bits). Black-box usage includes the management and distribution of ucdata. *White-box usage*, in contrast, interprets ucdata—rendering it, processing it (which includes data modification), or, in the case of programs, executing it.

Obligations

Obligations can take two different forms. Usage restrictions prohibit specific usages under given circumstances. Action requirements express mandatory actions that must be executed either unconditionally (not in direct connection with a usage) or after a specified usage has been performed. Both usage restrictions and action requirements can be narrowed down with conditions, which specify circumstances under which usage restrictions or action conditions apply. Conditions are concerned with time, cardinality, events that happened, purpose, and environment. Usage restrictions are statements of a form equivalent to "if (condition) then not (usage)." Examples include "document D must never be printed" and "movie M can be played only once" (note that the latter, albeit not having the exact structure shown earlier, is a usage restriction because it defines the circumstances under which the movie can be played and hence also the circumstances under which it must not be played).

Action requirements are statements of a form equivalent to "if (condition) then (action)." Examples for action requirements include "delete data D 20 days after reception" and "notify the data owner after each usage of data D." Clearly, action requirements and usage restrictions can look very similar—for example, "notify the data owner before each usage of data D" is a usage restriction because it prohibits use of D until a notification has been sent to the data owner.

Let's look at each type of condition in more detail. In the following examples, the parts of the usage restrictions or action requirements that are typeset in italics formulate the respective conditions:

- Time conditions are exemplified by requirements such as "file F must be deleted *within 20 days*," "F must *never* be distributed," and "movie stream M must *not* be viewed for *more than a total of 10 hours*." We haven't encountered examples in which an action's execution has no time limit, that is, conditions such as "action A must *eventually* be executed." Instead, a data provider sets a time limit like "action A must be executed *within the next six months*."
- Cardinality conditions refer to how often a usage occurs. Examples for cardinality conditions include,

"movie M may be played *only once*" or "play game G *at most twice* before it is paid."
- Event-defined conditions define situations with regard to event occurrence. Examples include, "*if the data provider revokes document D*, the document must not be used anymore," and "document D must not be further distributed *until the author officially releases D*."
- Purpose conditions refer to the purpose of use of ucdata. Data consumers often encounter requirements that limit the allowed purposes of use. For example, objects that are labeled "*for personal use only*" shouldn't be used in a business context.
- Environment conditions relate to the organizational, technical, and physical environment. Examples include compliance with the Sarbanes-Oxley Act, firewalls that are certified with respect to the Common Criteria, and the data consumer's geographical location.

These conditions can all be combined to describe more complex circumstances—for example, the requirement "movie M must *only* be viewed *at most three times* and *only within 30 days*" combines a cardinality condition and a time condition. Moreover, usage restrictions and action requirements can be combined to form more complex policy statements. The following obligation combines an action requirement with a time condition and a usage restriction with a purpose condition: "data set D needs to be *deleted within seven days* and must *exclusively be used for scientific purposes*."

Taxonomy of consumer-side enforcement mechanisms

We present our taxonomy of consumer-side enforcement mechanisms for usage control, which covers both control and signaling mechanisms. We'll see later in this section that control mechanisms essentially work in one of the following two ways: they can inhibit usage attempts (to enforce usage restrictions), and they can execute additional actions (to enforce action requirements), both under given conditions. A signaling mechanism is therefore a special instance of a control mechanism because it executes a specific type of action (the sending of signals) under given conditions. We'll see, however, that the signaling subset isn't widely used yet, leaving much potential for future exploitation.

Overview

Our taxonomy includes all criteria we deemed important for classifying consumer-side enforcement mechanisms. These criteria stem from studying existing mechanisms and from building conceptual models of consumer-side enforcement.[3] But some criteria aren't completely covered by existing mechanisms. These criteria will constitute one part of the future of control mechanisms that we'll discuss later.

We divided the taxonomy (see Figure 1) into three parts. The *applicability* part defines to which problems and under which circumstances a certain mechanism can be applied. This category includes the usage classes that can be controlled, supported usage control requirements, limitations on a mechanism's range of control, the supported data class, and the required infrastructure that defines the organizational and technical conditions under which the mechanism will work correctly.

The *implementation* part of the taxonomy defines how the mechanism is implemented. This includes the class of enforcement that the mechanism employs, the implementation layer (the hardware, media, operating system, or application), whether the mechanism's implementation is distributed among provider and consumer, its logging capabilities, and information about the licenses or policies the mechanism uses. The third category of criteria is that of *non-functional properties* that include cost as well as the performance, reliability, and privacy guarantees that a mechanism can provide. A discussion of these is outside this article's scope.

Applicability

The applicability category contains several classification criteria.

Usage class. This criterion defines the classes of usage to which a mechanism can be applied (process, render, execute, manage, distribute). We distinguish between three different levels of support for a given class of usage.

Some usage classes are *completely disabled*, meaning that a data consumer can't perform any usage of them. This is typically the case for white-box usages if the data is encrypted; no device can decrypt the data and perform a usage on it. Usages can also be *selectively enabled*, meaning that specific usages are possible, provided that the data consumer meets necessary requirements. This is, for example, the case if a device possesses the necessary cryptographic keys. Finally, some usages *aren't controlled at all*—that is, the consumer can perform them at will. Typically, these are black-box usages.

Supported requirements. This criterion covers whether the mechanism supports usage restrictions or action requirements, and which types of conditions it supports.

Limitations. This criterion defines when and under which conditions a mechanism releases data from its scope of control. For instance, when images are

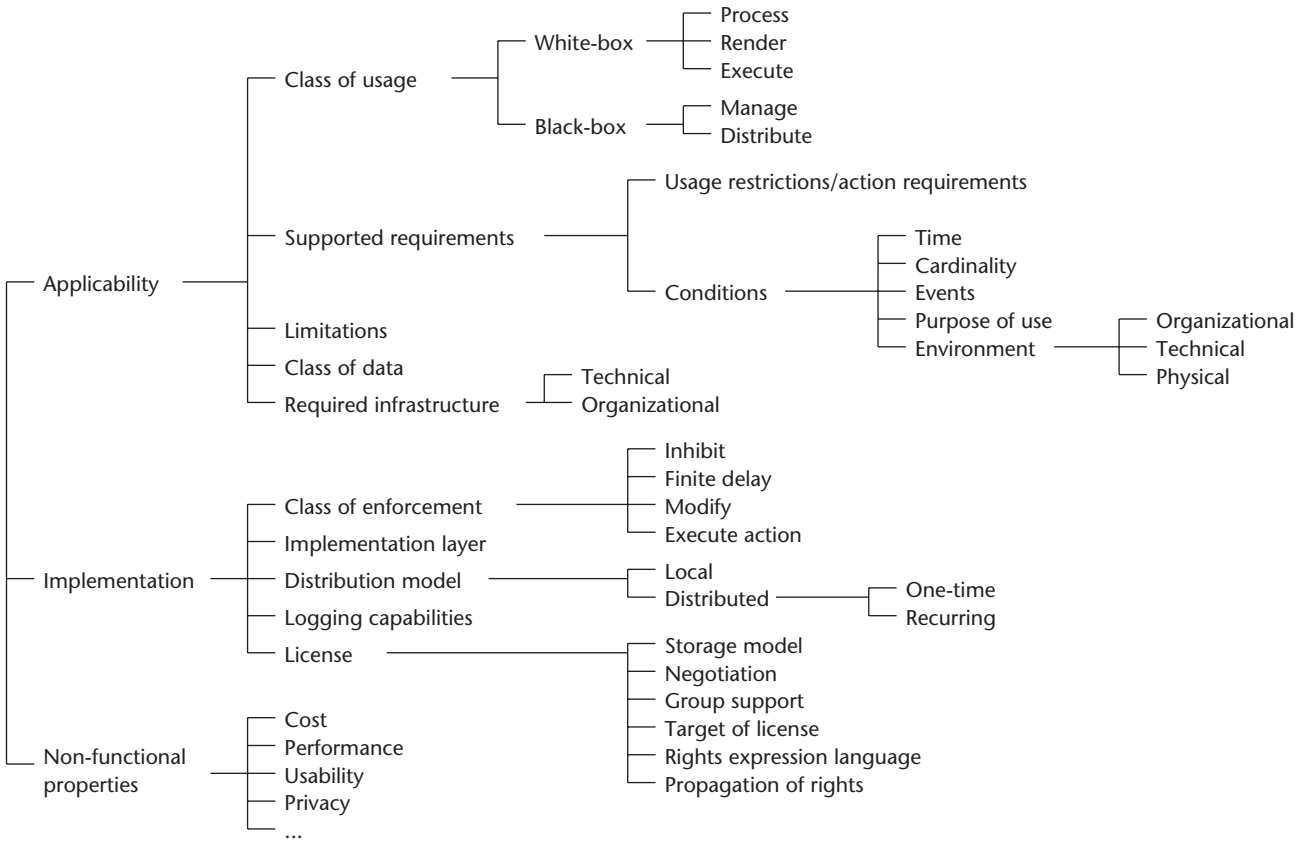


Figure 1. Taxonomy overview. The taxonomy of control mechanisms differentiates between applicability, defining to which problems and under which circumstances a certain mechanism can be applied, implementation defining how the mechanism is implemented, and non-functional properties such as costs and performance.

shown on a screen, control generally ends because the screen can be photographed. However, watermarking technology can enable further control. If data is modified in a way that lets a mechanism trace its origin, the respective mechanism is said to be traceable, and untraceable otherwise.

Data class. This criterion is concerned with the classes of data that a mechanism supports. Examples are "streaming media" or "files," and also "text," "video," "audio," or "executable" content.

Required infrastructure. This criterion supplies requirements on the infrastructure. These are divided into technical and organizational requirements. In terms of *technical restrictions*, the question of which class of media or devices is required for the mechanism to work is considered. In which technical environments would the mechanism fail? Operating systems, infrastructures such as regularly updated directories, and available hardware define possible instances of this category. For example, a mechanism that controls video output might rely on the availability of specific hardware. In terms of *organizational restrictions,* we consider the organizational measures that must be complied with for the mechanism to work as expected. If, for instance, certificates must be issued according to a specific standard, then the mechanism might not work properly if employees can create arbitrary certificates.

The required infrastructure differs from environment conditions in that the latter formulate usage control requirements, whereas the former define preconditions that must be satisfied if the control mechanism is to work properly.

Implementation

The implementation category contains several classification criteria as well.

Class of enforcement. To enforce usage control requirements, a control mechanism can do one of the following:[3]

- *Inhibition.* The mechanism can inhibit attempted usage.

- *Finite delay*. The mechanism can delay an attempted usage until a specified condition is met (for example, until a payment is confirmed). We only allow finite delays because the inhibition function covers infinite delays.
- *Modification*. The mechanism can modify the effect of a usage (for example, by downsampling an audio or video file's output) or replace a usage with a different one (such as open a file in read-only mode instead of editing mode).
- *Execution of actions*. The mechanism can execute a required action such as sending a notification to the data owner, triggering a micropayment, or deleting a document.

In principle, inhibition and execution of actions are sufficient to enforce usage control requirements. The other two classes increase convenience: users aren't allowed to perform a given usage, but the mechanism "helps" them find an alternative. The difference between usage control requirements and the classes of enforcement is that the latter are used to implement the former—the former describe what is desired, and the latter describe how to enforce that desire.

Implementation layer. A control mechanism can be implemented in hardware (including media), software, or a combination of both. Software resides at the application, runtime system, operating system, or CPU level.

Distribution model. A control mechanism can either be local or distributed. Local control mechanisms never need a connection to a server; one such example is CD copy protection. Distributed mechanisms are divided into those that need a one-time connection and those that require recurring connections. Mechanisms in the former category need to connect only once to fetch a license.

The latter category, to which all signaling mechanisms belong, covers the cases in which the client and the server communicate more often. A further distinction is between the client providing one-way feedback to the server about how its data is used and the server being involved in the enforcement by, for example, counting usages. A combination of both is also possible.

Logging capabilities. A control mechanism can have logging capabilities, which can be used for enforcement purposes (for example, a mechanism can log usages to the end of enforcing cardinality constraints). Logging might also be required for compliance reasons, such as for being compliant to a standard like ISO 17779.

License. A license is an agreement between a data consumer and a data provider. It contains usage requirements that the consumer has committed to and defines what the consumer may and may not do with specific data. Important aspects of license are the following.

- *Storage model*. The license for given content can be stored together with the content, in either a local or remote repository.
- *Negotiation*. The license's exact content can be defined by the data provider, proposed by the data consumer, or negotiated in bidirectional communication.
- *Group support*. This category captures whether the mechanism's licenses support grouping the subjects or devices.
- *Target of license*. A license can restrict different targets. For example, a license could state whether a particular device is allowed to perform a given usage, or it might authorize a particular user to use any device he or she owns. The latter requires some form of user authentication. Other licenses might state what will happen with a data object regardless of the device or user. Thus, a license's target could be an object, a device, a user, or any combination thereof.
- *Rights expression language (REL)*. What REL is used to specify licenses? Existing languages such as the Open Digital Rights Language (ODRL; odrl.net/1.1/ODRL-11.pdf) or the eXtensible rights Markup Language (XrML)[12] might be supported fully or partially by a control mechanism.
- *Propagation of rights*. The propagation of rights addresses two aspects, one of which is delegation. Does the mechanism support the delegation of rights to other subjects? If yes, are the rights delegated as a whole or modified before? The other aspect is processing. Are rights/licenses modified after processing operations? For instance, what is the resulting license when several data items are aggregated, such as in a statistical application?

For space reasons, we don't discuss the category of non-functional properties in this article.

State of the art: Existing control mechanism

We now turn our attention to a few existing control mechanisms and analyze how they fit into our taxonomy. In a survey of existing control mechanisms,[13] we looked at 28 different solutions. We covered significant technologies for control mechanisms, provided we could get access to information in the literature or on the Web.

Selected mechanisms

To illustrate how we can apply the classification cri-

Table 1. Mechanism overview.

	OMA	ADOBE	WMDRM	HDCP
APPLICABILITY				
Controlled usages	-white box, +render, +manage,* +distribute*	-white box, +process, +render	-white box, +process, +render	-white box, -black box, +render
Conditions	time, cardinality, event-defined†	time, cardinality, event-defined†	time, cardinality, event-defined†	—
Limitations	no control after rendering	little control after rendering (watermarking)	no control after rendering	no control after rendering
IMPLEMENTATION				
Class of enforcement	inhibition, execution of actions	inhibition	inhibition	inhibition, modification
Layer of embedding	not specified	SW, optionally HW	SW	HW
Negotiation	automated, data provider	automated, data provider	automated, data provider	automated, device
Rights language	subset of Open Rights Digital Language	Portable Document Rights Language	subset of eXtensible rights Markup Language	—

*Both very limited.

† Only limited support in the form of revocation capabilities.

teria we presented in the previous section and to give examples for the state of the art, we describe four of the surveyed technologies. We selected these particular technologies because they show a variety of different approaches, illustrate some important points, and are relevant in practice. Three of them—Adobe LiveCycle Rights Management, Windows Media DRM, and High-Bandwidth Digital Content Protection (HDCP)—are implementations of mechanisms. The Open Mobile Alliance (OMA) DRM is a specification of a mechanism. Table 1 gives an overview of the findings according to our taxonomy.

In Table 1, in the row labeled "controlled usages," a "-" stands for usage classes that are completely disabled. A "+" stands for usage classes that can be selectively enabled. Note that the "+" overrides the "-"; that is, if we label white-box usages as completely disabled, we can selectively enable rendering. Usage classes that aren't mentioned in the text aren't controlled.

OMA. The OMA has issued specifications for DRM on mobile platforms, including mobile phones and handheld devices (www.openmobilealliance.org/release_program/drm_v2_0.html). These specifications define the mechanisms that different companies can implement to ensure interoperability between their DRM solutions.

OMA DRM uses a subset of ODRL (with a few additions) as an REL. The classes of usage that it can control are rendering, management, and distribution, although the latter two can be controlled only in limited form. The OMA REL lets us restrict "export" permissions, which are defined as allowing the export of DRM content and corresponding rights objects to target systems other than the OMA DRM system. We view this as a way to control content management. Furthermore, OMA contains the "forward lock" option, which is a weak mechanism that prevents content from being forwarded—a (limited) form of controlling distribution.

OMA also supports super-distribution by encrypting content and issuing the rights objects with the necessary keys separately. Super-distribution means that the content can be freely distributed, but authorized users and devices are the only ones who can use the content. Various types of data are supported, not just audiovisual data. Furthermore, OMA DRM supports multiple devices owned by one user.

Adobe LiveCycle Rights Management. Adobe LiveCycle Rights Management[4] is a solution for integrating rights management into distributed business processes (and application servers). In terms of control mechanisms, policies stored at the server side are used to control usages including opening (viewing), modifying, and printing documents. Adobe LiveCycle Management lets data consumers use documents in an offline manner, in which case usage might be restricted for a specified amount of time.

Adobe uses its proprietary Portable Document Rights Language (PDRL; www.adobe.com/devnet/livecycle/policyserver/articles/pdrl.pdf) as an REL. In PDRL, license and policy are separated. A license creates the binding between a protected document and a policy by containing signed references to both document and policy. The policy contains the exact usage rules. PDRL contains a quite specific and limited set of rights but is designed such that it can be extended in the future.

Most current control mechanisms target specific application areas and are concerned with DRM of multimedia data.

Windows Media DRM (WMDRM). WMDRM[14] is a DRM system for the Windows Media platform. Originally designed to support secure delivery of media content to a PC or device over insecure networks, it has evolved to support a wider variety of usage requirements. Still, it's tailored toward digital audio and video content in the Windows Media format, both as files and streams. WMDRM controls processing and rendering operations on digital media.

WMDRM distributes the content in encrypted form and supports super-distribution. The corresponding decryption key is distributed within a valid license, which is bound to a client device. WMDRM uses XrML[12] as an REL, which supports a wide variety of conditions, including time conditions and cardinality.

High Bandwidth Digital Content Protection (HDCP). HDCP[14] defines a DRM protection scheme for streaming encrypted audiovisual data between a transmitter and a receiver. Applicable policies, which are set by the content provider, can prevent content from being stored, recorded, or displayed on non-HDCP compliant devices. Alternatively, the content's quality can be reduced if displayed on non-HDCP compliant devices.

General observations

We can observe several significant points when looking at existing control mechanisms. Here, we concentrate on the mechanisms' applicability and implementation.

Applicability. Most current control mechanisms target specific application areas and are concerned with DRM of multimedia data, although the selection we outlined contains exceptions. Therefore, most mechanisms exclusively support audiovisual data. Usually, mechanisms can inhibit white-box usages because encryption protects the data. In many cases, a mechanism can selectively enable only rendering actions because only rendering devices can read the data. Black-box usages can be controlled in a few cases, for example, if the data is stored on protected hardware.

Action requirements aren't widely supported. In terms of conditions for usage restrictions, support for temporal and cardinality conditions is widespread. Event-defined conditions are supported in very few cases, mostly in the form of revocation capabilities. Purpose conditions aren't considered by any examined mechanism, and environment conditions aren't supported either.

In most cases, control is lost once the data is rendered. Watermarks can provide a limited form of auditing and might therefore decrease the probability of policy violations, a result of the intended deterrence. However, this is suitable only for detecting illegal distribution of data in hindsight. It remains to be seen whether there will be irremovable watermarks that are robust with regard to rendering operations.

Implementation. Most of today's control mechanisms are implemented in software and located at the application layer. This gives rise to serious security concerns regarding how to protect cryptographic keys and the decrypted content. Often, the solution to this problem is security by obscurity coupled with update mechanisms: the code is obfuscated and no documentation of the mechanism's inner details is available. Once an attacker has figured out how the mechanism works and breaks it, the mechanism's manufacturer updates it quickly so that a new attempt to break it is required. However, this is a risky strategy; conceptual weaknesses might not be easy to fix, and the manufacturer must ensure that old versions aren't being used anymore.

Most mechanisms enforce by inhibition; that is, they either allow a usage or completely refuse it. HDCP is the rare example of a mechanism that can also enforce by modification. The method of delaying a usage until a precondition is met was not encountered at all in the mechanisms that we surveyed, and action execution was supported only in very few cases. As we mentioned earlier in this article, many mechanisms today use the concept of super-distribution.

The gap

In this article, we've shown which features are present in today's existing control mechanisms. We now pin down the features that are necessary for control mechanisms in DRM applications as well as in the areas of privacy, compliance, and enterprise computing. We'll also look at whether these features could or should be implemented in future mechanisms.

Supported usage requirements

To cover application areas other than DRM, the

set of supported usages must further extend beyond processing and rendering of multimedia content. Additional white-box usages can be covered by encrypting the data and creating dedicated devices that perform the desired usage, possess the necessary keys, and are trusted to satisfy the associated conditions. Of course, the existing control limitations and the problem of protecting secrets remain and must be tackled separately.

Supporting black-box usage is even more difficult because it requires full control over access to the data, regardless of whether it's encrypted. This is possible on dedicated hardware devices such as DVD players but difficult on today's personal computers (and one problem here is that an application that has read one protected piece of ucdata must, in principle, not be allowed to perform any future storage or communication activities). However, the widespread use of super-distribution reduces the need for controlling black-box usage, at least in the area of DRM.

Action requirements and event-defined conditions should be straightforward to support and are necessary to express requirements such as those for payment and notification, which often occur in practice. Notification requirements are particularly important in privacy protection contexts.[10] Purpose conditions and organizational environment conditions, which are both important for privacy and compliance, are difficult to enforce because they refer to nontechnical aspects that often aren't visible to mechanisms. It remains to be seen how convincingly mechanisms can support these types of conditions. Observation mechanisms are a possible solution to this problem because a monitor can also take into account the results of audits, which we'll discuss later in the article.

Control limitations

The fact that rendering bounds the scope of control is hard to overcome because rendered data is useful only if it's no longer encrypted. The only known way to keep a certain amount of control in such cases is by using watermarks, and it isn't clear how more control should be achieved except for improved watermarking schemes. In a limited way, it's possible to protect the rendered output—for example, some screens aren't easily photographed, and electromagnetic emission can be reduced or contained.

Implementation layer

In most software-based solutions, protecting the cryptographic keys and the decrypted content are serious challenges. An increased availability of hardware-based protection such as smartcards and trusted platform modules could be a solution to this problem. For instance, the protection of cryptographic keys can be embedded in these hardware components.[15,16] Moreover, there's currently an increasing interest in monitoring the execution of programs at runtime—at the level of the CPU, via system call interposition, at the level of the runtime system (such as Java VMs), or at the level of application and service wrappers—thus providing different granularities of controlling the flow of information.

Classes of enforcement

Enforcement by inhibition, as done in most cases today, is sufficient for preventing undesired usages. Together with mechanisms that execute actions, we can guarantee adherence to all policies. The fact that the execution of actions is supported by only a few mechanisms today restricts the set of policies that can be enforced, but supporting it technically isn't a problem.

Implementing mechanisms that employ modification or delays shouldn't be very difficult either. An increased use of modifying or delaying mechanisms could significantly enhance the user friendliness of control mechanisms. Imagine a policy that prohibits playing a movie in full quality for those who haven't paid for it, but lets everyone play it with reduced quality. Pure enforcement by inhibition would mean that if the user tries to play the movie in full quality without prior payment, nothing happens. Enforcement by modification means that the movie is automatically played in reduced quality. The use of delays could, for example, mean that a mechanism delays playing a movie until it has triggered a necessary payment. In other words, modification and delays can take some work off the user and ensure that as much functionality as possible is available to the user. This enhances a mechanism's usability and is likely to increase user acceptance for such technologies.

Observation mechanisms

We haven't found any readily available mechanisms that implement the idea of observation mechanisms. This is likely a consequence of the difficulties of obtaining trustworthy signals. As mentioned in the introduction, however, this rather lightweight approach to usage control is particularly appealing in situations in which consumers are unlikely to allow the provider more control over their IT infrastructure. We could argue that if a mobile phone or handheld device, for instance, implements trusted observation mechanisms, then it will probably be equipped with suffi-

Observation mechanisms are particularly appealing when consumers want to restrict the providers' influence.

ciently powerful technology to allow for full control. Because usage control isn't only a matter of technology but also of expediency, we see the applicability of observation mechanisms in the context of business information systems rather than mobile phones or handheld devices.

Some of the gaps we've identified in this article are at least partly addressed by current developments. For example, the layer of embedding of control mechanisms seems to shift more and more toward the hardware layer. At least some parts of the control mechanisms, such as keys or random number generators, are likely embedded in hardware in many of the upcoming mechanisms or are based on the security that trusted platform modules provide.

Another trend is that DRM technologies are increasingly used to support business processes in the form of document protection, such as in Adobe LiveCycle Rights Management or Microsoft's RMS (www.microsoft.com/windowsserver2003/technologies/rightsmgmt/default.mspx). Eventually, this development might lead to a wider range of requirements and application domains that such mechanisms support. Document protection, for example, isn't only important in connection with intellectual property but also with privacy. We're convinced, however, that only an increased use of observation mechanisms will lead to enforcement that's suited for a wide range of business applications. Implementing such mechanisms requires implementing trustworthy signaling components on the consumer side. In terms of enforcement by control mechanisms, we see potential in approaches that differ from inhibition. The execution of actions can increase the range of supported requirements, whereas modification and delay increase user convenience.

There is a further gap—namely, the one between expressions in usage control policies ("delete" and "distribute," to name two) and the mapping to commands or sequences of commands on the consumer's devices. Today, rights expression and policy languages don't provide a precise semantics of such terms. We see such a mapping as one of the fundamental problems in the area of usage control that aren't yet solved.

Although we've only examined single mechanisms in this article, the interplay of different mechanisms should also be examined (such as the Coral Consortium [www.coral-interop.org/main/news/Coral.whitepaper.pdf] and others[17,18]). Currently, many mechanisms aren't yet interoperable with each other, meaning protected content can only be played on particular devices. This seriously affects the control mechanisms' usability because many of today's devices support shared media. A movie might be played on a PC, a TV that receives the content from a media server, and even on a mobile device, but DRM often prohibits that movie from being played on all these platforms. The lack of interoperability increases users' reluctance to accept DRM technologies.

Usage control is both a very important and a challenging problem. Some solutions for the enforcement of usage control already exist, but to cover a wide range of application areas, these solutions need to mature and expand. This leaves many interesting areas of current and future research that are of great practical relevance. □

References

1. J. Park and R. Sandhu, "The UCON ABC Usage Control Model," *ACM Trans. Information and Systems Security*, vol. 7, no. 1, 2004, pp. 128–174.
2. A. Pretschner, M. Hilty, and D. Basin, "Distributed Usage Control," *Comm. ACM*, vol. 49, no. 9, 2006, pp. 39–44.
3. M. Hilty et al., *A System Model and an Obligation Language for Distributed Usage Control*, tech. report I-ST-20, DoCoMo Euro-Labs, Dec. 2006; www.docomoeurolabs.de/pdf/A_System_Model_and_an_Obligation_Language_for_Distributed_Usage_Control.pdf.
4. C. Bettini et al., "Provisions and Obligations in Policy Rule Management," *J. Network and System Management*, vol. 11, no. 3, 2003, pp. 351–372.
5. D. Povey, "Optimistic Security: A New Access Control Paradigm," *Proc. Workshop on New Security Paradigms*, ACM, 1999, pp. 40–45.
6. M. Hilty et al., "Usage Control Requirements in Mobile and Ubiquitous Computing Applications," *Proc. Int'l. Conf. Systems and Networks Communications*, IEEE CS Press, 2006.
7. S. Guth and R. Iannella, *Open Digital Rights Language (ODRL) Version 2 Requirements*, Feb. 2005; odrl.net/2.0/v2req.html.
8. D. Parrott, *Requirements for a Rights Data Dictionary and Rights Expression Language*, tech. report, Reuters, 2001; xml.coverpages.org/Reuters-mpeg-response-v10-public.pdf.
9. European Union, "Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 On The Protection of Individuals with Regard to the Processing of Personal Data and on the Free Movement of Such Data," *Official Journal L 281*, Nov. 1995, pp. 31–50; http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31995L0046:EN:NOT.
10. Privacy and Identity Management for Europe (Prime) Project, "Requirements v. 1," June 2005; www.prime-project.eu/prime_products/reports/reqs/.
11. M. Hilty et al., "A Policy Language for Distributed Usage Control," *Proc. European Symp. Research in Computer Security*, Springer-Verlag, 2007, pp. 531–546.
12. X. Wang et al., "XrML—eXtensible Rights Markup

Language," *Proc. ACM Workshop on XML Security* (XMLSEC 02), ACM Press, 2002, pp. 71–79.
13. M. Hilty et al., *Enforcement for Usage Control—An Overview of Control Mechanisms*, tech. report I-ST-18, DoCoMo Euro-Labs, July 2006; www.docomoeurolabs.de/pdf/Enforcement_for_Usage_Control_an_Overview_of_Control_Mechanisms.pdf.
14. Digital Content Protection, *High-bandwidth Digital Content Protection System – v. 1.1*, June 2003; www.digital-cp.com/files/static_page_files/8006F925-129D-4C12-C87899B5A76EF5C3/HDCP_Specification%20Rev1_3.pdf.
15. R. Sandhu et al., "Client-Side Access Control Enforcement Using Trusted Computing and PEI Models," *J. High Speed Networks*, vol. 15, no. 3, 2006, pp. 229–245.
16. P. Sevinç, M. Strasser, and D. Basin, "Securing the Distribution and Storage of Secrets with Trusted Platform Modules," *Proc. IEEE Workshop in Information Security Theory and Practices*, LNCS 4462, Springer-Verlag, 2007, pp. 53–66.
17. Marlin Developer Community, "Marlin Architecture Overview," 2006; www.marlin-community.com/images/wp/MarlinArchitectureOverview.pdf.
18. Marlin Developer Community, "The Role of Octopus in Marlin," 2006; www.marlin-community.com/images/wp/RoleofOctopusinMarlin.pdf.

Alexander Pretschner *is a senior researcher at the Swiss Institute of Technology (ETH Zurich). His research interests include distributed usage control, model-based testing, automotive software engineering, and information retrieval. Pretschner has MScs in computer science from Aachen University of Technology and the University of Kansas and a PhD in computer science from Munich University of Technology. He is a member of the ACM and the German Gesellschaft für Informatik. Contact him at pretscha@inf.ethz.ch.*

Manuel Hilty *is a security architect for AdNovum Informatik AG in Switzerland. His research interests are usage control, identity management, and information security in general. Hilty has an MSc in electrical engineering and a PhD in computer science from the Swiss Federal Institute of Technology (ETH Zurich). Contact him at hilty@adnovum.ch.*

Florian Schütz *is a security architect and business consultant for RUAG Electronics, Switzerland. His research interests include usage control, security engineering, and information security in general. Schütz has an MSc in computer science from the Swiss Federal Institute of Technology (ETH Zurich). He is a student member of the IEEE. Contact him at fschuetz@ieee.org.*

Christian Schaefer *is a researcher for DoCoMo Euro-Labs in Munich, Germany. His research interests are the enforcement of security policies in distributed systems with a focus on usage control and security of mobile handsets. Schaefer has an MSc in computer science from the University of Karlsruhe, Germany. He is a member of the IEEE. Contact him at schaefer@docomolab-euro.com.*

Thomas Walter *is a senior manager in the Smart and Secure Services Group of DoCoMo Euro-Labs. His research interests include security of software and services for mobile devices, security policies, and access and usage control in distributed environments. Walter has an MSc in computer science from the University of Hamburg and a PhD in computer science from the Swiss Federal Institute of Technology (ETH Zurich). He is a member of Gesellschaft für Informatik (German computer society) and the IEEE. Contact him at walter@docomolab-euro.com.*