# Support for Multi-Level Security Policies in DRM Architectures

Bogdan C. Popescu
Vrije Universiteit
Amsterdam, The Netherlands
bpopescu@cs.vu.nl

Bruno Crispo
Vrije Universiteit
Amsterdam, The Netherlands
crispo@cs.vu.nl

Andrew S. Tanenbaum
Vrije Universiteit
Amsterdam, The Netherlands
ast@cs.vu.nl

## ABSTRACT

Digital rights management systems allow copyrighted content to be commercialized in digital format without the risk of revenue loss due to piracy. Making such systems secure is no easy task, given that content needs to be protected while accessed through electronic devices in the hands of potentially malicious end-users; in this context, intrusion tolerance becomes a very useful system property. In this paper we point out a limitation shared by all current DRM architectures, namely their weakness in reacting to possible device compromise and confining the damage caused by such a compromise. As a solution, we propose a paradigm shift - moving from the original DRM system model where all devices are equally trustworthy and have discretionary control over all protected content, to a new model where information flow is controlled through a multi-level security policy that differentiates between devices based on their tamper-resistance properties. We show that besides improved intrusion-tolerance, supporting such policies has other advantages, such as the ability to define more flexible business models for supplying content. We also show that for a given DRM architecture, the type authentication protocol used when accepting new devices in the system has a big impact on how well multi-level security policies can be supported, and that a number of protocols currently being considered are not very well suited for this job.

## 1. INTRODUCTION

In the past few years there has been an increasing interest in developing software/hardware architectures facilitating digital rights management (DRM). The main purpose of such architectures is providing digital data content (mostly home entertainment-related) in a way that is convenient for consumers, and also safe and secure from the content providers (CP) point of view.

Building such systems is no easy task, given the strong attack model that has to be taken into account: copyrighted content needs to be protected while being accessed through devices in the hands of potentially malicious end-users. Although making devices tamper-resistant helps, it has been shown [9] that even sophisticated tamper-resistance mechanisms can be circumvented by motivated and technically skilled attackers. In such a context, intrusion tolerance becomes a very useful property; as recent history has demonstrated, protection mechanisms that lack this property (such as the DVD copy protection scheme), have little chance to succeed.

The paradigm currently pondered when designing DRM architectures is that compliant devices can be indiscriminately assumed as part of the trusted computing base (TCB) of the system and given discretionary access to all protected content. In this paper we show that when a DRM system comprises devices with different tamper-resistance properties, following the above paradigm is bad security practice, because it goes against the basic principle of least privilege, making the overall intrusion-resistance of the system to be dictated by the intrusion-resistance of its weakest components.

As a solution, we propose a paradigm shift - moving from the original DRM system model where equally trustworthy devices have discretionary control over all protected content, to a new model where information flow is controlled through a multi-level security policy based on intrinsic device properties (tamper-resistance being one of them). In this paper we point out the advantages of such a shift: better intrusion-tolerance of DRM systems, but also the ability to define more flexible business models for supplying content. We also show that for a given DRM architecture, the type protocol used for authenticating compliant devices has a big impact on how well multi-level security policies can be supported, and that one particular class of such protocols - those based on broadcast encryption [8] - are not very well suited for this job.

The rest of this paper is organized as follows: in Section 2 we discuss about the business models governing the commercial digital content distribution world, and explain how they have shaped the design of DRM architectures. In Section 3 we make the case for the need to enforce multi-level security policies over the data flow in a DRM system, and show how this would improve the overall intrusion tolerance of the system. In Section 4 we discuss device authentication protocols, and show why some of them are not well suited to

support data flow policies. In Section 5 we hint at a possible solution, and in Section 7 we present our conclusions.

## 2. BUSINESS MODELS IN THE DRM WORLD

There are three major players in the commercial digital content distribution world: the content providers, the device manufacturers and the consumers. Each of them has different agendas and priorities; mediating among their often conflicting interests is essential for designing a successful DRM system.

From the CPs point of view, the biggest threat is illegal copy and unlimited distribution of their copyrighted digital content; for this reason, DRM systems must focus on mechanisms allowing providers to control the way digital content is distributed and processed. The key concept for supporting this is the *compliant device* - a device that by its construction is guaranteed to process digital content only in ways sanctioned by the owners of the content.

Device manufacturers, on the other hand, are particularly keen on cost savings: since DRM functionality needs to be incorporated into mass-produced consumer electronic (CE) devices, even marginal cost reductions can lead to significant competitive advantages. In this context, security mechanisms that rely on public key cryptographic algorithms are seen as a disadvantage, since they require cryptographic accelerator hardware (which increases the overall cost) in order to operate efficiently. For this reason, lot of interest is given to DRM architectures solely relying on symmetric key cryptographic algorithms; one class of such algorithms known as *broadcast encryption* [8, 18, 16] appears to be particularly well suited for this application scenario, and has lately received considerable attention [15].

Finally, consumers have certain interoperability expectations: when buying digital content, they want to be able to access this content from any of the devices they own, just as with traditional media (CDs and DVDs). Previous experience has shown strong negative consumer reaction when copyright protection mechanisms have disrupted interoperability expectations, such as audio CDs not playing in PC CDROM drives [13].

### 2.1 Compliant devices

The typical business model in the DRM world is to have CPs and device manufacturers set up a consortium (licensing organization) which develops standards for compliant devices [3], and delegates the right to produce these devices to the participating manufacturers. The idea is then to distribute digital content in such a way that it can only be accessed through licensed compliant devices, which are the ones to enforce the CPs copyrights. It is therefore essential to be able to distinguish between compliant and non-compliant devices.

To achieve this, compliant devices normally incorporate cryptographic keys that facilitate *compliance checking* (proving to CPs and to other devices they are indeed compliant), and are manufactured as *tamper-resistant* to prevent malicious users from extracting these keys and build DRM circumvention devices. The licensing organization has ultimate control over the key material incorporated into devices, but typically delegates the right to issue device keys to the participating manufacturers.

By construction, compliant devices will only process digital content in ways sanctioned by the CPs. Because of this property, once a device is authenticated as compliant, it is implicitly added to the DRM system's *trusted computing base* (TCB), and given *discretionary* access to the data content. This is considered safe, exactly because of this self-enforcing property of compliant devices: before processing any piece of content the device will *always* check the usage rules associated with the content and only proceed if allowed. An example of this is giving a compliant digital video recorder full access to piece of video marked "no copy": because the recorder is compliant, the content owner can be assured it will never make a copy, even though it has the ability to do so.

### 2.2 DRM systems as personal private networks of compliant devices

A DRM system, no matter how secure, is next to useless, if it is not accepted by consumers. As mentioned, key to gaining this acceptance is the ability to match consumer's inter-operability expectations.

One way to deal with the interoperability requirement is to design DRM systems as a *personal private networks* (PPNs) of compliant devices. The idea is to have one such PPN for every household, and allow legally acquired content to seamlessly flow from device to device; this gives the consumer the same "content anytime, anywhere" experience as more traditional content distribution models (CDs, DVDs). A number of DRM architectures based on the PPN abstraction have been proposed [7, 17, 2, 5]. Although they differ in the technical mechanisms employed, they are more less based on the same system model, which assumes compliant devices incorporating three types of functionality:

- Management functionality: this involves accepting new devices in the PPN. Only compliant device are allowed in the network, so an essential part of the management functionality is compliance checking. Besides this, there may be other requirements that need to be enforced on the structure of a PPN, the most obvious being size restrictions (in order to avoid creating a PPN comprising all devices in the world!).

- Access functionality: this involves bringing new content to the PPN. The means by which this new content is retrieved may vary and include broadcast, Internet, pre-recorded media, and other proprietary copy protection systems. It is important to understand that copyrighted content is always transferred together with its usage rules. These rules can be expressed in various rights management languages [6, 4] and describe what actions are allowed on the content (e.g. play once, play many, copy once, copy many, etc).

- Presentation functionality: this involves rendering digital content. Before performing any operation, a compliant device will first ensure that operation is allowed according to the usage rules associated with the content (the *self-enforcing property* of compliant devices).

One functional requirement that needs to be taken into account in every DRM design concerns network connectivity: given that these DRM systems are centered around the home environment, continuous connectivity among all devices in the PPN, as well as return data channels between devices and the licensing organization cannot be assumed. An additional reason to stick with this restriction is that in this way more traditional content distribution mechanisms - such as CDs and DVDs - can be modeled as special cases of a one-way broadcast channel from CPs to consumers. As we will see in the next section, this restriction has major implications on the choice of cryptographic primitives used in the compliance checking protocols.

## 3. THE NEED FOR MULTI-LEVEL SECURITY POLICIES

Adding all compliant devices to the TCB of the DRM system seems a good idea, since it greatly simplifies the inter-device interaction: once two devices have authenticated each other as compliant, they are free to exchange any content over a secure channel. Furthermore, DRM management operations are greatly simplified, since actual device identity is not important anymore (as [15] notes - "a device cares only about connecting to another compliant device, not exactly which device it connects to").

The problem with the above solution is that it treats all compliant devices as equally trustworthy. This works fine when devices are tamper-proof, or at least they have the same degree of tamper-resistance. In practice however, tamper-proofness is extremely hard to achieve (if not even impossible), and tamper-resistance always comes in different degrees [9]. Furthermore, since it is very expensive to achieve a high tamper-resistance degree, it can be expected that inexpensive electronic devices will only come with relatively low tamper-resistance.

Since a DRM system will most likely contain compliant devices of different tamper-resistance degrees, it is easy to see that adding them all to the system's TCB with equal privileges is a bad idea from a security point of view: an attacker can compromise the security of the entire system by compromising the device with the lowest tamper-resistance degree (thus, the overall system security is dictated by its weakest component).

The solution we advocate for the above problem is a paradigm shift: move from a flat security policy model where all compliant devices are given discretionary access to all content to a *multi-level security policy model*, where the system enforces *mandatory control* over the information flow between devices. An example of this is the Bell-LaPadula [12] security model, which specifies how a set of subjects are allowed to access a set of protected objects. The model, derived from security policies used in the military, associates with each subject and object two attributes - the *clearance* level and *need-to-know* requirements for subjects, and *class* (how sensitive the information is) and *category* (for which tasks is the information necessary) for objects. In the Bell-LaPadula model, subjects can only access objects whose class is lower or equal to their clearance, and whose category is included in their need-to-know requirements.

As an example of how this may work in our application scenario, consider a simple, Bell-LaPadula-like security policy adapted for a DRM system:

- The compliant devices in the DRM system are treated as the subjects in the original Bell-LaPadula model:
    - The *clearance* subject attribute is set to the tamper-resistance degree of the given device.
    - The *Need To Know* attribute is set to the media types supported by the given device.
- The protected media items in the DRM system are treated as the objects in the original Bell-LaPadula model:
    - The *class* attribute is set to the digital quality of a given piece of media.
    - The *category* attribute is set to the media type of a given piece of media.

With this model in place, a compliant device with a high tamper-resistance degree will never send high quality content to a low tamper-resistant device. Furthermore, devices will only get the data they need (and audio device will never get video content).

A DRM architecture supporting such a multi-level security policy would be much more robust to security breaches due to device compromise: first of all, a device compromise would only cause *localized* damage (compromising an audio player would only give an attacker access to audio content), and second, the attacker's gain will be proportional to its efforts (compromising a low-tamper-resistant device will only give access to low quality content). Most importantly, the restrictions brought in place by such a multi-level security model will have little negative impact on the normal functioning of the correct devices in the domain: there is no reason an audio player would ever need video data, and it is quite likely that cheap devices with low tamper-resistance degree will also provide only low quality analog output, so there is no reason to give them access to high quality digital data.

Additionally, the ability to support such data flow policies would facilitate defining novel business models for supplying digital content. For example, it would make possible alliances between content providers and device manufacturers, allowing providers to target their content only to certain classes of devices. This would be particularly interesting for companies that are *both* content providers and device manufacturers (Sony Music and Sony Electronics for example), which would be able to make their device offering more attractive by providing "premium" content that could only be accessed through these devices. We want to stress that in this paper we do not advocate for any particular business model; our belief is that a greater variety of *possible* business models is always good for the consumer, and view this as an additional argument for the paradigm shift we propose.

# 4. COMPLIANCE CHECKING MECHANISMS

In the previous section, we have shown how multi-level security policies controlling the data flow can improve the overall intrusion-tolerance of DRM systems. In this section we show that the types of data flow policies a given DRM architecture can support depend on the compliance checking protocol used.

Compliant devices incorporate cryptographic keys that facilitate compliance checking. The licensing organization has ultimate control over this key material, but normally delegates the right to issue "compliant" keys to approved manufacturers. Both symmetric key and public key cryptographic algorithms can be used to do compliance checking.

## 4.1 Mechanisms based on symmetric key algorithms

Traditional symmetric key authentication protocols [14] rely on a trusted, on-line *key distribution center* (KDC). However, continuous on-line access to an external entity cannot be assumed in a DRM system, so an alternative design needs to be considered. The solution is to have symmetric key *pre-distribution* schemes, essentially give every compliant device a shared key with every other compliant device in the world. One class of symmetric key pre-distribution schemes, collectively known as *broadcast encryption*, are particularly well suited in this context, since they give the licensing organization full control over the key material (compromised keys can be easily revoked), without requiring a two-way communication channel to devices.

### 4.1.1 Broadcast encryption

There is not one, but a number of broadcast encryption schemes, with rather different properties [8, 18, 16]. What all these schemes have in common is that they allow a dynamic group of devices to establish a common secret, by receiving messages broadcast by the licensing organization (no back communication channel is necessary). Once a common secret key has been established, it can be used to protect the digital content exchanged by the compliant devices part of the group. We will now briefly describe the functioning of one of these algorithms ([16]), which has been specifically designed for DRM applications.

In [16], key material is organized in a logical binary tree, where each node in the tree corresponds to a symmetric key. The number of leaves in the tree is equal to the maximum number of compliant devices in the world; this may be in the order of hundreds of million of even more in the case of very successful products. Each device is assigned a leaf and contains all the (secret) keys that are on the path between its assigned leaf and the root of the tree (thus, the root key is known to all devices).

The licensing organization decides the group membership; when it wants to create a new group (when compromised devices need to be excluded from the group, or due to membership changes for subscription services) it generates a new random group key, and encrypts it with keys in the tree that cover only the new group members; the encrypted group key can then be safely broadcast to the entire world, since only the group members will have the keys to decrypt it. This scheme works quite well when devices selected to be the broadcast group cluster together in one of the key tree's sub-trees (since the group key only needs to be encrypted under the key at the root of the sub-tree); however, in the general case, when there are few sub-trees covering only group members, the group key may need to be encrypted mostly with the leaf keys corresponding to individual group members, which leads to a broadcast message size linear to the group size. As we will describe in the following section, this is the very reason why broadcast encryption is not well suited for expressing complex security policies governing the data flow between compliant devices.

### 4.1.2 Limitations of broadcast encryption

When broadcast encryption is used for device compliance checking, specifying multi-level security policies becomes more difficult. In essence, each distinct attribute value in a multi-level security policy requires creating an additional secure subgroup of compliant devices (this subgroup comprises all devices that are assigned that particular attribute value). Although this is not impossible (for example the CPRM [2] architecture specifies separate authentication groups for audio and video media), it is clear that complex multi-level security policies ultimately lead to an un-manageably large number of such subgroups. Furthermore, given the discussion in the previous section, it is clear that fine-grained multi-level security policies will lead to subgroups of devices that are unlikely to cluster into sub-trees in the logical key tree, and because of this, they will require a large broadcast message size.

What makes things even worse is that defining a subgroup requires knowledge of the subgroup members' keys in the logical tree (in order to encrypt the common subgroup secret such that only subgroup members can decrypt it). This leaves the licensing organization with two un-appealing options: first it can share the key material with all content providers in order to allow them to directly create the subgroups they need. However, sharing cryptographic material is very bad security practice. The second option is to have the licensing organization actively participating in creating every device subgroup any content provider may need. Considering there can be a very large number of content providers, each of them who may want to frequently create new subgroups (in order to have per-content item data flow policies for example), this is second option is clearly not scalable.

## 4.2 Mechanisms based on public key algorithms

A number of DRM architectures [7, 17] rely on public key cryptographic algorithms for device compliance checking. The idea is to give each device a unique public/private key pair, with the public key certified through a chain of digital certificates issued by the licensing organization and manufacturers (with the licensing organization acting as the certification authority). Devices can then prove their compliance by simply exchanging certificates and then engaging in a public key authentication protocol.

Public key-based compliance checking mechanisms are much better suited for supporting multi-level security policies. Because devices are identified by means of public data (the public key in the device certificate), there is no need for the

licensing organization and the content providers to share any of the secret key material, so they can work independently from each other (in fact, for the licensing organization is irrelevant whether there is one or one million content providers, it only needs to worry about certifying public keys as being assigned to "compliant" devices). The only problem that needs to be solved is a flexible mechanism for efficiently describing device subgroups; as we will show in Section 5, this can be quite easily accomplished by modifying the format of the device certificate.

## 4.3 Discussion

As already mentioned, DRM solutions based on public key cryptographic algorithms are less attractive from an economic point of view: in order to perform public key operations reasonably efficiently, CE devices need to be equipped with cryptographic hardware accelerators, which increase the overall device price. For this reason, the research community expects that solutions based on broadcast encryption will be favored by CE manufacturers.

In the light of the paradigm shift we advocate, the balance may be tipped back in favor of public-key based DRM solutions, since they are clearly better suited to support multi-level security policies. Although they are more expensive due to the additional hardware required, we believe the improved intrusion tolerance and more flexible business models possible under the new paradigm are worth the costs (rigid focusing on cost-cutting is most of the times even more expensive, as proven by the DVD protection scheme blunder). Furthermore, the additional costs may not even be that significant, given that the price of cryptographic accelerators (which would become commodity hardware should they have to be incorporated in all CE devices) is likely to drop significantly as a result of mass production.

## 5. A POSSIBLE SOLUTION

So far we have advocated for the need to enforce multi-level security policies over the data flow in DRM systems. We have shown that taking into account differences between individual CE devices in a DRM system can provide better intrusion tolerance; we have also argued that it should be content providers that set such policies for the content items they distribute, so they can strike the optimal balance (from their point of view) between ensuring portability of their content, and protecting the same content from being "highjacked" from devices with low tamper-resistance properties. We have also shown that by providing the same content at different quality levels it is possible to further discourage malicious users from attempting to pirate this content by tampering low-end compliant devices.

It is important to understand that data flow policies are different from usage rules. Usage rules, which can be specified in specialized policy languages such as [6, 4], describe **how** the content should be used (for example - "play once", "play one month", "play with commercials", etc.). Data flow policies specify **where** the content can be used and at which quality level (for example "send HDTV [11] content only to high-end Sony TV sets"). At the moment, there are no DRM architectures that can enforce data flow policies, and there are no policy languages that can used to express them.

In the case of DRM architectures where compliance checking is done by means of public key authentication [7, 17], support for enforcing data flow policies is relatively easy to add. The idea is to use the device certificate as means to incorporate information allowing flexible and efficient subgrouping of compliant devices for the purpose of defining multi-level security policies. This information would consist of a number of attribute-value pairs, describing intrinsic properties of the device. For example, such attributes may include the name of the manufacturer, the device model, some manufacturer-specific device identifier, the supported media types, possibly the manufacture date. All electronic devices have such properties, so it would be simple for the licensing organization to standardize the attribute names (for example "Manufacturer", "Device Model", "Manufacture Date", etc.).

Content providers can use these attribute-value pairs to define device subgroups; such subgroups can then be used in expressing content flow policies, which are passed to devices with access functionality together with the actual content. With each piece of content it distributes, a content provider associates the following items:

- the **usage policy** - this describes how the content can be used; the XrML standard [6] can be used to express such policies.

- the **data flow policy** - this describes where (on which devices) the content can be used. As we will describe next, this policy is based on the attribute-value pairs included in the device certificate.

- the **quality label** - this describes the quality of the item according to the value levels specified in the data flow policy.

A data flow policy consists of the following items:

- a number of totally ordered **quality levels**; the description of each quality level depends on the media type of the data for which the policy is written (for example, in the case of audio media, the quality level can be expressed as the bit rate, in the case of images this can be the compression rate, and so on).

- a number of **subgroups descriptors** equal to the number of quality levels. Each subgroup corresponds to one quality level, and describes the set of devices that are allowed to receive the data content (at least) at that quality level. Such a descriptor is an expression consisting of ($attribute = range$) pairs connected using the $AND$, $OR$ and $NOT$ logical operators. The attributes in the expression correspond to the attribute names that can appear in the device certificates. The $range$ term can either be a continuous range of values, for attributes that are defined over a scalar domain ("1998-2000" for example, in the case of "Manufacture Time"), or it can be just a list of values separated by commas ("Sony, Toshiba, JVC" for example, in the case of "Manufacturer").

Within a PPN, content migrates from device to device in order to achieve the "content anytime/anywhere" experience for the end user. However, data flow policies are also enforced. More specifically, when content goes form a device $S$ to a device $D$, the following happens:

- $S$ and $D$ authenticate each other as compliant devices (using their device certificates). At the end of this process, each device has a copy of the other device's certificate, and there is a secure channel between the two.

- $D$ requests a piece of data content stored by $S$.

- $S$ uses the attribute values in $D's$ certificate to evaluate the expressions for each of the subgroup descriptors in the content flow policy associated with the data content requested by $D$, starting with the one corresponding to the highest quality level. The first match corresponds to the maximum quality level $D$ can receive the content. If there is no match, it means $D$ is not allowed to receive the content item, under the policy rules set by the CP for that item.

- if $S$ has the content item at a quality level equal or lower to the maximum $D$ is allowed to handle, it can transfer it to $D$ directly over the secure channel.

- if the quality level for the content item is higher than the maximum $D$ can handle, $S$ needs to *downgrade* the content before transferring it to $D$. The mechanism for quality downgrading is media type specific (for example, extra compression for images, coarser sampling for audio, etc.). Once $S$ has done the downgrading, it can transfer the (modified) content item, together with its new quality label and the original data flow policy to $D$. If $S$ does not have the capabilities for doing the downgrading, it should refuse to transfer the content item to $D$.

In this way, content providers can define arbitrary device subgroups *without* the assistance of either the licensing organization or device manufacturers, which are only responsible with assigning (once) the attribute-value pairs for each device they certify. Because CPs define their own (device subgroup - quality level) mappings, the same device can be assigned different quality clearances when processing different pieces of content, which makes possible to define novel business models for supplying content. For example, Sony, as a content provider, can designate all audio devices produced by Sony - as an electronics manufacturer - with the highest quality clearance, so all these devices would be allowed to get digital audio at the highest quality. On the other hand, BMG, may take a more prudent approach for its own content, and restrict its high-quality audio only to the high-end Sony audio players. Thus, a DRM system that supports data flow policies not only has better intrusion tolerance properties, but also facilitates defining new business models for supplying content.

## 6. DISCUSSION

The new paradigm discussed in this paper can be placed the a broader context of what constitutes a secure computing system. This issue has been around for almost forty years by now, and it is interesting to notice that attempts to align DRM systems along more traditional models of secure computation, such as the one presented in [10] raise a number of problems. One problem is scale: a DRM system architecture deals with millions, potentially billions of devices spread all over the world; this overshadows even the most ambitious designs in the area of distributed systems. Furthermore, the notion of "secure based on tamper-resistance assumptions", which is key to all DRM architectures is quite new, so it is not covered by more traditional secure systems designs.

For this reason, we believe a lot of work needs to be done in this area (DRM) in order to identify which of the more traditional security techniques can be re-used, and which aspects require a new design. For example, even the Bell-LaPadula policy model which we use as an example throughout the paper, does not fit perfectly: while in the original model objects without any label need to be placed at the top secrecy level in order to prevent disclosure of legacy, non-labeled information, with DRM this would cause major problems, since it would prevent distribution of copyright-free content, which should be accessible by all devices.

Another example of imperfect alignment between what the new paradigm would require and what traditional security methodology can provide is the classification of tamper resistance properties for compliant devices. At the first glance, it may look tempting to apply the guidelines in the Common Criteria Project [1] for assurance levels. However, the Common Criteria mostly targets software products which can be modified over time to fix security flaws and adapt to new threats. On the other hand, compliant devices once manufactured and sold are extremely difficult to alter. Because of this, a newly detected security flaw in a class of devices will most likely degrade the entire class to a lower assurance level.

Finally, we want to stress that the business model we are considering for this paper assumes an open content delivery architecture: basically this means that content providers are interesting in making their content offering available to a class of devices as broad as possible; no device manufacturers would be excluded from this distribution, as long as their devices could guarantee copyright protection. It is important to understand that this business model, although quite popular (there are a number of standards and industry wide-consortiums that push for such an open-architecture framework for enforcing DRM [3]), is not unique. Especially in the past two years new models have emerged, some of which focus on closed systems - the best example of this is the IPod music player produced by Apple which only plays Apple licensed content. However, protection architectures for such closed systems are outside the scope of this paper.

## 7. CONCLUSION

In this paper, we point out a limitation shared by all current DRM architectures, namely their inability to enforce *multi-level security policies* controlling the information flow between devices in the system. We show that supporting such policies has a number of advantages, most importantly improved intrusion tolerance, but also the ability to define more flexible business models for supplying content. We also

show that for a given DRM architecture, the type protocol used for authenticating compliant devices has a big impact on how well multi-level security policies can be supported, and that one particular class of such protocols - those based on broadcast encryption [8] - are not very well suited for this job.

As for future work, we are currently investigating whether other types of multi-level security policies (besides Bell-LaPadula) could be relevant for the DRM application scenario. We are also working on the formalization of a policy language for expressing these data flow policies. Finally, we plan to investigate whether it is feasible to have a public-key infrastructure tailored for supporting DRM systems, in order to minimize the overhead typical with a general purpose trust infrastructure.

# 8. REFERENCES

[1] Common Criteria for IT Security Evaluation. http://csrc.nist.gov/cc/.

[2] Content Protection for Recordable Media. http://www.4centity.com/tech/cprm/.

[3] The Digital Video Broadcasting Consortium. http://www.dvb.org/.

[4] The Open Digital Rights Language Initiative. http://odrl.net/.

[5] xCP Cluster Protocol. http://www.almaden.ibm.com/ software/ds/ContentAssurance/papers/xCP_DVB.pdf.

[6] XrML 2.0 Technical Overview. http://www.xrml.org/reference/ XrMLTechnicalOverviewV1.pdf, May 2002.

[7] Smartright technical white paper. http://www.smartright.org/images/SMR/ content/SmartRight_tech_whitepaper_jan28.pdf, Jan. 2003.

[8] Amos Fiat and Moni Naor. Broadcast Encryption. In *Advances in Cryptology - CRYPTO '93*, pages 480–491, Aug. 1993.

[9] R. Anderson and M. Kuhn. Tamper Resistance - a Cautionary Note. In *Proc. 2nd Usenix Workshop on Electronic Commerce*, pages 1–11, Nov. 1996.

[10] D.E. Bell and L.J. LaPadula. Secure Computer System: Unified Exposition and MULTICS Interpretation. Technical Report MTR-2997, MITRE Corporation, 1976.

[11] Advanced Television Systems Comitee. Digital Television Standard, Revision B, with Amendments 1 and 2. ATSC Standard A/53B, May 2003.

[12] Dieter Gollmann. *Computer Security*. Wiley, 1999.

[13] John A. Halderman. Evaluating New Copy-Prevention Techniques for Audio CDs. In *Proc. 2002 ACM Workshop on Digital Rights Management*, 2002.

[14] J.T. Kohl and B.C. Neuman. The Kerberos Network Authentication Service (Version 5) . Technical report, IETF Network Working Group, 1993. Internet Request for Comments RFC-1510.

[15] Jeffrey B. Lotspiech, Stefan Nusser, and Florian Pestoni. Broadcast encryption's bright future. *IEEE Computer*, 35(1), 2002.

[16] Dalit Naor, Moni Naor, and Jeff Lotspiech. Revocation and Tracing Schemes for Stateless Receivers. In *Advances in Cryptology - CRYPTO '01*, pages 41–62, 2001.

[17] S.A.F.A. van den Heuvel, W. Jonker, F.L.A.J. Kamperman, and P.J. Lenoir. Secure Content Management in Authorized Domains. In *Proc. IBC 2002*, pages 467–474, Sept. 2002.

[18] Chung Kei Wong, Mohamed G. Gouda, and Simon S. Lam. Secure Group Communications Using Key Graphs. In *Proc. of the ACM SIGCOMM*, pages 68–79, 1998.