Nuovo DRM Paradiso: Towards a verified fair DRM protocol

H.L. Jonker, S. Krishnan Nair, and M. Torabi Dashti

[1] Technische Universiteit Eindhoven, h.l.jonker@tue.nl
[2] Vrije Universiteit Amsterdam, srijith@few.vu.nl
[3] CWI, Amsterdam, dashti@cwi.nl

Abstract. The NPGCT DRM scheme, that proposes a unique concept of DRM-preserving content redistribution, has some security issues. These issues are addressed in this paper by an extension of NPGCT. A security mechanism that provides fairness in unsupervised exchanges is introduced, and the mechanism of detecting and revoking circumvented devices is reexamined devices. The resulting DRM scheme, Nuovo DRM, and its requirements are formally specified. A finite model of the scheme is subsequently model-checked and shown to satisfy its design requirements.

Keywords: DRM, formal verification, optimistic fair exchange protocols.

1 Introduction

In recent years, there has been a rapid increase in the popularity of personal devices capable of rendering digital contents, some of them also having peer-to-peer communication facilities. With the advent of these personal digital devices, content providers are looking at various secure business models to tap into this growing market for selling their copyrighted materials, necessitating the development of mechanisms to protect digital contents from illegal access and unauthorized distribution. Technologies used to enforce predefined policies controlling access to digital contents are referred to as Digital Rights Management (DRM) techniques. The main goal of DRM systems is to enforce DRM policies after contents has been distributed to consumers. The current approach to solve this problem is to limit the distribution of protected contents to only the so-called *compliant* devices, which by construction are guaranteed to always observe the DRM policies associated with the digital contents they render.

A unique concept of DRM-preserving content *redistribution* was proposed in [21] (hereafter called NPGCT scheme), where users act also as content distributors. This potentially allows a business model where a consumer can not only buy the rights to use a content, but also to redistribute or resell the content in a controlled manner. From security point of view, this is technically challenging, as the resulting system forms a network of independent peer-to-peer devices, each of them a consumer, authorized distributor, and also a potential attacker. Moreover, sobering experience [15] has shown that DRM techniques are inherently complicated and when carelessly enforced can infringe on customers' rights. These concerns are addressed by applying formal methods for verifying DRM protocols to provide both content vendors and customers a certain degree of confidence in the security and fairness of the system. This paper presents the formal specification and verification of an extended protocol based on the concepts presented in [21].

Contributions Our contributions in this paper are threefold. First, we identify some security issues with the NPGCT scheme. These issues are addressed in our extended scheme, dubbed *Nuovo*

DRM. In particular, the exchange protocol for peer customers is augmented with a light-weight recovery subprotocol, which proves essential in preventing unfair exchange scenarios. The resulting scheme can be seen as an *optimistic fair exchange* DRM scheme.

Second, we formally specify and verify the proposed DRM scheme. The protocol participants and the intruder model are specified in the μCRL process algebraic language [13]. The requirements for effectiveness, content secrecy, fairness in exchange and resistance to content masquerading are formalized in regular alternation-free μ-calculus [18]. These requirements are then verified on a finite model of the system using a model-checker from the CADP toolset [11]. The proposed protocols are shown to indeed achieve their design goals under the perfect cryptography assumption [9].

Third, the mechanism for detecting and revoking compromised devices in NPGCT is revised, allowing Nuovo DRM to detect and revoke circumvented devices at a lower communication cost.

Related work The probabilistic synchronous fair exchange protocol of [3] is similar to our work in that it relies on trusted computing devices in exchange. In contrast, we present a deterministic asynchronous protocol that achieves strong as opposed to probabilistic fairness. In this paper we do not explicitly model semantics and derivations of rights associated with DRM-protected contents. We refer the interested reader to the related literature on this, such as [23, 14, 20].

Structure of the paper We start by explaining the notations and cryptographic assumptions used in the paper in Section 2. Section 3 summarizes the NPGCT scheme, which provides the basis for our refined scheme. Section 4 presents Nuovo DRM, its assumptions and its goals. Nuovo DRM is formally analyzed in Section 5 and shown to achieve its goals. Section 6 presents the Nuovo's mechanism for detecting and revoking compromised devices. Finally, Section 7 concludes the paper.

2 Notations and assumptions

In this section we introduce the notations and assumptions for the rest of the paper.

Notations Through the paper we use C and D to denote compliant customer devices. These are respectively owned by $owner(C)$ and $owner(D)$. P denotes a trusted legitimate content provider. M denotes DRM-protected content. The finite set of all protected contents in the protocol is denoted $Cont$. It is assumed that unique descriptions (such as hash values) of all $M \in Cont$ are publicly known. The (finite) set of all possible rights in the protocol is denoted $Rgts$. The term $R_X(M)$ represents the rights of device X for content M.

Cryptographic assumptions In our analysis the cryptographic operations are assumed to be ideal, as in Dolev and Yao [9]: We assume a secure collision-resistant one way hash function h; hence $h(x)$ uniquely describes x. When a message M is encrypted with key K, denoted $\{M\}_K$, we assume that M can only be extracted using K. Notations $pk(X)$ and $sk(X)$ denote the public and private keys of entity X, respectively. In symmetric encryption we have $\{\{M\}_K\}_K = M$ and in asymmetric, $\{\{M\}_{sk(X)}\}_{pk(X)} = \{\{M\}_{pk(X)}\}_{sk(X)} = M$. Encrypting with a private key denotes signing and we assume, for convenience, that M is retrievable from $\{M\}_{sk(X)}$.

Trusted devices assumptions We assume that compliant devices are able to locally perform atomic actions. Multiple actions may be logically linked in these devices, such that either all or none of them are executed[4]. They also contain a limited amount of non-volatile secure memory. A legitimate trusted third party (TTP) to ensure fair exchange between customers (in our scheme this is the content provider), is assumed impartial in its behavior and infinitely often available to compliant devices.

[4] Note that distributed atomicity, as opposed to this local atomicity, corresponds to fairness in exchange [24].

3 The NPGCT DRM scheme

The NPGCT scheme [21] was proposed as a DRM-preserving digital content redistribution system where a consumer can also act as a content reseller. The scheme consists of two main protocols: the first (provider-to-client, P2C) distributes contents from provider P to client C, the second (client-to-client, C2C) allows C to resell contents to another client D.
The P2C protocol is initiated by the owner of C who wants to buy item M with rights R from provider P. Adapted from [21]:

1. $C \rightarrow P$: Request content
2. $C \leftrightarrow P$: Mutual authentication, [payment]
3. $P \rightarrow C$: $\{M\}_K, \{K\}_{pk(C)}, R, metadata(M), \Lambda$
   $\Lambda = \{h(P, C, M, metadata(M), R)\}_{sk(P)}$

Λ acts as a certification that C has been granted rights R and thus acts as proof of C's right to redistribute M to other clients. It also binds $metadata(M)$ to M, which prevents spoofing attacks.
The C2C protocol is initiated by the owner of D who wants to buy M with rights R' from C. Adapted from [21]:

1. $D \rightarrow C$: Request content
2. $C \leftrightarrow D$: Mutual authentication
3. $C \rightarrow D$: $\{M\}_{K'}, \{K'\}_{pk(D)}, R_C(M), R', metadata(M), \Lambda, \Lambda'$
   $\Lambda' = \{h(C, D, M, metadata(M), R')\}_{sk(C)}$
4. D : Verifies $metadata(M)$, Λ' and $R_C(M)$ using Λ
5. $D \rightarrow C$: $\psi, [payment]$
   $\psi = \{h(C, P, \{M\}_{K'}, metadata(M), R')\}_{sk(D)}$

ψ acts as a receipt from D that it has received M with the rights R', while Λ and Λ' form a chain of rights that proves that D has been granted R'.

3.1 Security issues of NPGCT

The NPGCT scheme suffers from two security flaws that undermine its goals. First, in both protocols, a malicious client can feed rights from a previous session to her trusted device, as freshness of the authentication phase is not extended to guarantee freshness of the delivered content-rights bundle. In this case P will not be paid for the rights that C can accumulate. A common remediation is to use nonces to ensure freshness of the entire exchange. See Section 4 for such an adaption.

Second, in the C2C protocol, payment is not bound to the request/receive messages. Thus, D can abort the protocol after receiving M and avoid paying C. In Section 4 of this paper, a recovery phase is added to the protocol to prevent such unfair protocol runs. Furthermore, a fallback mechanism is included in our proposal to ensure fairness.

4 The Nuovo DRM scheme

This section describes an extended version of NPGCT, dubbed Nuovo DRM, which in particular addresses the security concerns noted in Section 3.1. As stated there, freshness of the exchange is ensured and a recovery subprotocol is added to the C2C exchange.

Second, the payment scheme was left open in the NPGCT scheme, so various online (requiring contact with the bank, e.g. payment using checks) and off-line (not requiring contact with the bank, e.g. payment by cash) payment systems could be used. In contrast, Nuovo DRM is limited

to online payment systems, where payment orders exchanged for contents can be encashed only at a central banking entity. The process of chaining rights, used in NPGCT to detect non-compliant devices, is not needed in Nuovo DRM as detection is achieved by inspecting payment orders (see Section 6). The following assumptions are made for Nuovo DRM:

A1. Consumer compliant devices are assumed tamper-proof (relaxed later in Section 6). Owners of compliant devices are untrusted. They may collude to subvert the protocol and can, in particular, halt their own devices.
A2. We assume an asynchronous resilient communication model with no global clock, i.e. the communication media delivers each transmitted message intact in a finite but unknown amount of time. Under certain reasonable assumptions, this is attainable [4].
A3. There is a hierarchy of public keys, with the public key of the root embedded in each compliant device and available to content providers. Using this hierarchy, a device can prove its identity or verify identities without contacting the root.
A4. Protocol participants negotiate the price of content in advance. In Nuovo DRM, the price of the content being traded is bundled with the requested rights.

The goals of Nuovo DRM are to provide *secrecy* and resist *content masquerading* (as NPGCT aims to[5]). Additionally, Nuovo aims to provide *fairness* by inclusion of subprotocols for providing fairness in exchange. The goals of Nuovo DRM are thus:

G1. Effectiveness If honest participants run the protocol, the protocol terminates successfully, i.e. the correct content-right bundle is exchanged for the corresponding payment order.
G2. Secrecy No outsider may learn *secret* contents. NPGCT and Nuovo DRM achieve this by encrypting protected for intended compliant devices. Hence, DRM-protected contents should never appear unencrypted.
G3. Resist content masquerading Content M should not be able to pass for content M'.
G4. Strong fairness Informally, strong fairness states that if Alice and Bob run a protocol to exchange their items, finally either both or none of them should have received the other party's item.

Note that Nuovo aims at strong fairness because the contents which are exchanged in the system can be retrieved from the trusted third party (the content provider). Strong fairness also guarantees timeliness (see [2]), informally: honest protocol participants can safely terminate their role in the protocol after a finite amount of time, without altering the degree of achieved fairness. As fairness is a liveness property (see [1]), a resilience assumption on the communication channels is required for fairness to hold in general (see [2]) – in Nuovo DRM, this is assumption A2.

4.1 Nuovo DRM protocols

Nuovo DRM consists of two main protocols: a P2C protocol and a C2C protocol. In the P2C protocol, the owner of C wants to buy item M with rights R from content provider P. C and P, but not $owner(C)$, are assumed trusted.

[5] The other requirements of NPGCT mainly concern attacks on storage backup systems, which are out of the scope of our analysis.

1. $owner(C) \rightarrow C : P,\ h(M),\ R$
2. $C \rightarrow P : C,\ n_C$
3. $P \rightarrow C : \{n_P,\ n_C,\ C\}_{sk(P)}$
4. $C \rightarrow P : \{n_C,\ n_P,\ h(M),\ R,\ P\}_{sk(C)}$
5. $P \rightarrow C : \{M\}_K,\ \{K\}_{pk(C)},\ \{R,\ n_C\}_{SK(P)}$

In the first step, the owner specifies what content he wants, from whom, and with which rights. (Note that due to assumption A4, $owner(C)$ and P have agreed upon the price.) P and C can both check the other's legitimacy (assumption A3). In steps 2 and 3, C and P mutually authenticate eachother. The fourth step constitutes a *payment order* from C to P. After receiving this message, P checks if R is same as previously agreed upon (assumption A4) and if so, stores the payment order (for future/immediate encashing). P then generates a random fresh key K and concludes the transaction by sending the content to C in step 5. C checks whether the received M matches the requested $h(M)$, and whether the received nonce is n_C. If so, C updates $R_C(M)$ with R. Note that C may already own rights to M (denoted by $R_C(M)$), hence $R_C(M)$ is augmented with R. As we abstract away from rights semantics, the update phase is left unspecified in this paper.

In the C2C protocol, the owner of D wants to buy item M with rights R' from compliant device C. To ensure fairness in exchange (i.e. both parties finish, or neither does), a TTP is required (see [10]). If a TTP need only act in case of a conflict, the burden on the TTP is reasonable (assuming most participants act honestly). Nuovo DRM uses such an *optimistic* fair exchange protocol. Hence there are two protocols: the optimistic exchange protocol and the conflict resolution. The optimistic exchange runs similar to the P2C protocol, and goes as follows:

1. $owner(D) \rightarrow D : C,\ h(M),\ R'$
2. $D \rightarrow C : D,\ n_D$
3. $C \rightarrow D : \{n_C,\ n_D,\ D\}_{sk(C)}$
4. $D \rightarrow C : \{n_D,\ n_C,\ h(M),\ R',\ C\}_{sk(D)}$
5. $C \rightarrow D : \{M\}_K,\ \{K\}_{pk(D)},\ \{R',\ n_D\}_{sk(C)}$

At step 5, C issues the rights to D and, more importantly, atomicly updates the rights associated with M (reflecting that rights have been used for reselling M), and stores the payment order signed by D. The atomicity of these actions is required as it guarantees that C does not store the payment order without simultaneously updating the right $R_C(M)$. Checking for compliance is not as easy as in the P2C protocol, mainly because of compromised devices. This issue is examined in Section 6 in more detail. We refer to [21] for a thorough discussion of revocation of circumvented devices. In the C2C protocol, a malicious $owner(C)$ can abort before delivering message 5 to D or this message can simply get lost due to a hardware failure. To prevent such unfair situations for D, we provide a recovery mechanism to obtain the lost content.

The goal of the recovery protocol is to ensure a fair state for D in case of a failure in communicating message 5 in the C2C protocol. D can start a recovery session with the content provider P at any time after sending message 4 in the C2C protocol. Once the recovery protocol has been initiated, D ignores messages from the optimistic run of C2C. The recovery runs as follows:

5^r. $D : resolves(D)$
6^r. $D \rightarrow P : D,\ n'_D$
7^r. $P \rightarrow D : \{n_P,\ n'_D,\ D\}_{sk(P)}$
8^r. $D \rightarrow P : \{n'_D,\ n_P,\ \langle n_D,\ n_C,\ h(M),\ R',\ C\rangle,\ P\}_{sk(D)}$
9^r. $P \rightarrow D : \{M\}_K,\ \{K\}_{pk(D)},\ \{R',\ n'_D\}_{SK(P)}$

This protocol is similar to the P2C protocol and its detailed description is skipped here. The way P resolves failed exchanges deserves more attention and is explained below.

Assume that D tries to resolve an unsuccessful exchange with C. As a result of the atomicity of C's actions in the optimistic subprotocol, only the following situations are possible: either C has updated $R_C(M)$ and has the payment order of message 4 (which it is thus entitled to have), or C does not have the payment order and has not updated $R_C(M)$. In the latter case, resolving the exchange boils down to a P2C exchange. However, when C owns the payment order from D, two different cases can happen: (1) D resolves, then C tries to encash the payment order. P reimburses C, as D has already paid P. Note that P is not paid for providing D with M – it has already been paid by C when C bought the right to resell M. (2) C encashes the payment order, then D resolves. P will not charge D, because D has already paid the price and C has updated its right $R_C(M)$, for which P has already (directly or indirectly) been paid.

One can argue that the recovery subprotocol may also fail due to lossy communication channels. As a way to mitigate this, persistent communication channels for content providers can be built, e.g. by using an FTP server as an intermediary. The provider would upload the content, and the device would download it from the server. Such resilient communication channels (see A2) are generally necessary to guarantee fairness [2].

As a final note, we emphasize that we only consider compliant devices here (assumption A1) and this protocol can trivially be attacked if the devices are tampered. Section 6 discusses methods for revoking tampered devices and resisting systematic content pirating.

5 Formal analysis of Nuovo DRM

Cryptographic protocols are notorious for being error prone. Formal verification techniques can bring to light issues that have been overlooked by protocol designers and provide a certain level of confidence in the correctness and security of protocols.

In this section we briefly describe the necessary steps to formally verify whether Nuovo DRM achieves its design goals. We stick to assumption A1 here and detection of circumvented devices is out of the scope of our current analysis. Our approach is based on finite-state model-checking [7], which is (usually) fully automatic and thus requires negligible human intervention and, moreover, always produces concrete counterexamples, i.e. attack scenarios, if the design fails to satisfy a desired property. It can therefore be efficiently integrated in the design phase. However, a prefect security proof of the system cannot, in general, be established by model-checking. For an overview on formal methods for verifying security protocols see [19].

Our formal verification can be seen as a sequence of steps: specify the protocol and the intruder model, state the desired properties and verify the protocol. Below we describe these steps in detail.

5.1 Formal specification of Nuovo DRM

The complex structure of Nuovo DRM, for instance in having stateful participants who need to make internal decisions, calls for an expressive specification language. We have formalized the Nuovo DRM scheme in μCRL, a language for specifying and verifying distributed systems and protocols in an algebraic style [13]. A μCRL specification describes a labeled transition system, in which states represent process terms and edges are labeled with actions. The μCRL toolset, together with CADP [11] which acts as a back-end for the μCRL toolset, features visualization, simulation, state space generation, symbolic reduction, model checking and theorem proving capabilities. It has already been successfully used in some industrial-scale case studies [22, 12].

We model a security protocol as an asynchronous composition of a finite number of acyclic non-deterministic named processes. These processes model roles of honest participants of the protocol. Processes communicate by sending and receiving messages. A message is a pair $m = (p, c)$, where p is the identity of the intended receiver process and c is the content of the message. To send or receive a message m, a participant p performs the actions $\mathbf{send}(p, m)$ or $\mathbf{recv}(p, m)$, respectively. Apart from **send** and **recv**, all other actions of processes are assumed internal, i.e. not communicating with other participants. These are symbolic actions that typically denote security claims of protocol participants (see Section 4.1 for some examples). Due to space constraints we do not present the formal specification of the participants of the Nuovo DRM scheme in this paper. For a complete specification we refer to [17].

Communication models We consider two different communication models. The first is a synchronous communication model and is used to verify the effectiveness property (goal G1). In this model there is no intruder and all participants are honest. A process p can send a message m to q only if q at the same time is in a position to receives it from p. The synchronization between these is denoted **com**, which formalizes the "$p \rightarrow q : m$" notation of Sections 3 and 4. In order to verify the remaining properties (G2–G4), an asynchronous communication model is used, where the intruder has complete control over the communication media. When a process p sends a message m with the intention that it should be received by q, it is in fact the intruder that receives it, and it is only from the intruder that q may receive m. The communications between participants of a protocol, via the intruder, is thus asynchronous and, moreover, a participant has no guarantees about the origins of the messages it receives.

Intruder model We follow Dolev and Yao's approach to model the intruder [9]. The Dolev-Yao intruder has complete control over the network: it intercepts and remembers all transmitted messages, it can encrypt, decrypt and sign messages if it knows the corresponding keys, it can compose and send new messages using its knowledge and it can remove or delay messages in favor of others being communicated. As it has complete control over communication media, we assume it plays the role of the communication media. All messages are thus channeled through the intruder. Under the perfect cryptography assumption [9], this intruder has been shown to be the most powerful attacker model [6]. In our formalization, this intruder can be seen as a non-deterministic process that blindly exhausts all possible sequences of actions, resulting in a labeled transition system which can subsequently be formally checked. The intruder process may also have a legitimate role in the protocol.

The intruder model used here is however different from the Dolev-Yao intruder in certain aspects (for a formal specification of our intruder model see [17]). These differences stem from the characteristics of the DRM protocol and its requirements, as stated below.

I1. Trusted devices, that play a crucial role in this protocol, limit the power of the intruder significantly[6]. However, the intruder has the ability to deliberately turn its (trusted) devices off. This is a threat to fairness in exchange, because the devices will deviate from the protocol description. This ability has been implemented in our model by letting the intruder's device(s) non-deterministically select between continuing the protocol and aborting communications at each step, except when performing atomic actions.

I2. Resilient channels are generally required to guarantee liveness of protocols. No liveness property can be proved in the Dolev-Yao model, since the intruder can simply block all communications. To verify fair exchange properties, resilient communication channels (abbreviated as RCC) are often assumed. They guarantee that all transmitted messages will *eventually* reach their destination, provided there is a recipient for them [2]. The behavior of our intruder model is limited by RCC,

[6] In our formalization we dismiss the possibility of tampering a trusted devices.

i.e. it cannot indefinitely block the network[7]. Since the intruder is a non-deterministic process in our model, the part of its behavior that violates *RCC* has to be purged afterward. The action $\mathbf{com}^{\dagger}$, used in Section 5.2, represents communication actions not required by RCC. Therefore, a protocol has to achieve its goals even when executions containing $\mathbf{com}^{\dagger}$ actions are avoided. A formalization of an intruder model that respects *RCC* is given in [5].

I3. To indicate violation of the secrecy requirement, the intruder process performs the abstract action *revealed* when it gets access to a non-encrypted version of any DRM-protected content. This action does not include the case where the intruder can merely render an item using its trusted device, which is a normal behavior in the system.

5.2 Analysis results

In this section we describe the results obtained from the formal analysis of the Nuovo DRM scheme. Our analysis has the following properties: The intruder is allowed to have access to unbounded resources of data (like fresh nonces), should it need them to exploit the protocol. Here we consider only a finite number of concurrent sessions of the protocol, i.e. each participant is provided a finite number of fresh nonces to start new exchange sessions. Although this does not constitute a proof of correctness or security for a protocol in general, in many practical situations it suffices. The fact that the problem of the security of cryptographic protocols is not decidable (e.g. see [8]) implies that a trade-off has to be made between completeness of the proofs and their automation. Our analysis method is fully automatic and the verification algorithms do not need human interventions. Similar to [9], we assume perfect cryptography and do not consider attacks resulting from weaknesses of the cryptographic functions used in protocols. Type flaw attacks[8] are also omitted from our analysis. They can, in any case, be easily prevented [16].

The formal analysis considers two scenarios. The first verifies the effectiveness while using the synchronous communication model of Section 5.1. The second scenario uses the asynchronous communication model of Section 5.1 to verify the remaining properties. Both scenarios consist of two compliant devices C and D that are controlled (not tampered) by the intruder of Section 5.1. Below, P, as always, represents the trusted content provider.

The following results have originally been encoded in the regular alternation μ-calculus and model-checked using CADP toolset [11]. The regular alternation-free μ-calculus is a fragment of μ-calculus that covers the Nuovo DRM's design goals in its entirety, both safety and liveness parts, naturally incorporates data parameters that are exchanged in the protocols and, moreover, can be efficiently model-checked [18]. Below, only an intuitive description of these properties is presented. The verified properties are formally specified in [17], where a short account of the used logic is also given. The properties use abstract actions to improve the readability of the proved theorems. These actions are explained in Section 4.1. A precise specification of these actions can be found in the complete formalization of the Nuovo DRM scheme presented in [17].

Honest scenario S_0: The communication network is assumed operational and no malicious agent is present. C is ordered to buy an item from P. Subsequently C resells the purchased item to D. This scenario was model-checked using the CADP toolset, confirming that it is deadlock-free, and effective as specified below.

Result 1 *Nuovo DRM is effective for scenario* S_0*, meaning that it satisfies the following properties:*

[7] E.g. a wireless channel provides *RCC* for mobile devices, assuming that jamming can only be locally sustained.

[8] A type-flaw attack happens when a field in a message that was originally intended to have one type is interpreted as having another type.

1. *Each purchase request is inevitably responded.*

$$\forall m \in \mathit{Cont},\ r \in \mathit{Rgts}.\ \mathit{update}(C, m, r, P) \text{ inevitably happens after } \mathit{request}(C, m, r, P)$$
$$\forall m \in \mathit{Cont},\ r \in \mathit{Rgts}.\ \mathit{update}(D, m, r, C) \text{ inevitably happens after } \mathit{request}(D, m, r, C)$$

2. *Each received item is preceded by its payment.*

$$\forall m \in \mathit{Cont},\ r \in \mathit{Rgts}.\ \mathit{update}(C, m, r, P) \text{ is always preceded by } \mathit{issue}(P, m, r, C)$$
$$\forall m \in \mathit{Cont},\ r \in \mathit{Rgts}.\ \mathit{update}(D, m, r, C) \text{ is always preceded by } \mathit{issue}(C, m, r, D)$$

Dishonest scenario S_1: The intruder controls the communication network and is the owner of the compliant devices C and D. The intruder can instruct the compliant devices to purchase items from the provider P, exchange items between themselves and resolve a pending transaction. Moreover, the compliant device C can non-deterministically choose between following or aborting the protocol at each step, which models the ability of the intruder to turn the device off (see I1 in Section 5.1). We model three concurrent runs of the content provider P, and three sequential runs of each of C and D. The resulting model was checked with the CADP toolset and the following results were proven.

Result 2 *Nuovo DRM provides secrecy in scenario S_1, i.e. no protected content is revealed to any non-compliant device, in this case, the intruder (see I3 in Section 5.1).*

$$\forall m : \mathit{Cont}.\ \mathit{revealed}(m) \text{ never happens}$$

Result 3 *Nuovo DRM resists content masquerading attacks in S_1, ensuring that a compliant device only receives the content which it has requested.*

$$\forall a \in \{C, D\},\ m \in \mathit{Cont},\ r \in \mathit{Rgts}.\ \mathit{update}(a, m, r, P) \text{ is always preceded by } \mathit{request}(a, m, r, P)$$
$$\forall m \in \mathit{Cont},\ r \in \mathit{Rgts}.\ \mathit{update}(C, m, r, D) \text{ is always preceded by } \mathit{request}(C, m, r, D)$$
$$\forall m \in \mathit{Cont},\ r \in \mathit{Rgts}.\ \mathit{update}(D, m, r, C) \text{ is always preceded by } \mathit{request}(D, m, r, C)$$

Besides, the intruder cannot feed the self-fabricated content m_0 to compliant devices:

$$\forall a \in \{C, D\},\ r \in \mathit{Rgts}.\ \mathit{update}(a, m_0, r, P) \text{ never happens}$$
$$\forall r \in \mathit{Rgts}.\ \mathit{update}(C, m_0, r, D) \text{ never happens}$$
$$\forall r \in \mathit{Rgts}.\ \mathit{update}(D, m_0, r, C) \text{ never happens}$$

Result 4 *Nuovo DRM provides strong fairness in S_1 for P, i.e. no compliant device receives a protected content, unless the corresponding payment has been made to P.*

$$\forall a \in \{C, D\},\ m \in \mathit{Cont},\ r \in \mathit{Rgts}.\ \mathit{update}(a, m, r, P) \textit{ is always preceded by a corresponding } \mathit{issue}(P, m, r, a)$$

Result 5 *Nuovo DRM provides strong fairness in S_1 for D, as formalized below*[9]*:*

1. *As a customer: If a compliant device pays (a provider or reseller device) for a content, it will eventually receive it.*
   Note that there are only finitely many TTPs available in the model, so the intruder, in principle, can keep all of them busy, preventing other participants from resolving their pending

[9] Fairness for C is not guaranteed here, because it may quit the protocol prematurely. A protocol, in principle, guarantees security only for the participants that follow the protocol.

transactions. It corresponds to a denial of service attack in practice, which can be mitigated by putting time limits on transactions with TTPs. As we abstract away from timing aspects here, instead, the action $last_{ttp}$ is used to indicate that all TTPs in the model are exhausted by the intruder. In other words, if this action does not happen, there is at least one TTP available for honest participants.

$\forall m \in Cont,\ r \in Rgts.$ after a $request(D, m, r, C)$ either $resolves(D)$ or $update(D, m, r, C)$ is (fairly) inevitable, with no extra help from the intruder, i.e. with no $\mathbf{com}^\dagger$ *in between.*
$\forall m \in Cont,\ r \in Rgts.$ As long as $last_{ttp}$ does not happen, after each $resolves(D)$ an $update(D, m, r, P)$ is reached, with no extra help from the intruder, i.e. with no $\mathbf{com}^\dagger$ *in between.*
$\forall m \in Cont,\ r \in Rgts.$ after a $request(D, m, r, P)$, $update(D, m, r, P)$ is (fairly) inevitable, with no extra help from the intruder, i.e. with no $\mathbf{com}^\dagger$ *in between.*

2. *As a reseller: no compliant device receives a protected content from a reseller device, unless the corresponding payment has already been made to the reseller.*

$$\forall m \in Cont,\ r \in Rgts.\ update(C, m, r, D) \text{ is always preceded by } issue(D, m, r, C)$$

Note that the strong fairness notion that is formalized and checked here subsumes the timeliness property of goal G4, simply because when D starts the resolve protocol, which it can autonomously do, it always recovers to a fair state with no help from the intruder.

Theorem 1. *Nuovo DRM achieves its design goals in scenarios S_0 and S_1.*

Proof. G1 is achieved based on result 1. Result 2 implies G2. Result 3 guarantees achieving G3. Results 4 and 5 guarantee G4.

6 Detection and revocation of compromised devices

The security of Nuovo DRM relies on the compliance of the customer devices. However over time, some of these devices will be compromised. In this section, we examine how to detect compromised devices and propose a method to limit interactions with these devices (i.e. relaxing A1). As in NPGCT, the proposed mechanism aims at detecting systematic content pirating, rather than occasionally misbehaving users. To this end we introduce the following additional assumptions:

A5. The bank (responsible for encashing payment orders) cooperates with content providers[10] to detect malevolent users. Here, for the sake of simplicity, the content provider and the bank are assumed to be the same entity.
A6. When a compliant device signs a payment order, the payment order is encashable. (This enables an attack where signing occurs without sufficient funds – which can be ameliorated, but that is beyond the scope of this paper.)

A compromised device can perform two obvious attacks[11]. First, it can overuse its reselling rights. To detect large scale overselling, the provider reconstructs the the chain of sold rights. This is possible because of assumption A5, i.e. devices need to contact the provider (or the bank) to acquire payment for sold rights.

In detail, the provider maintains a directed weighted graph $G = (V, E)$ for each sold content-rights combination. Each $v \in V$ represents a compliant device and $E \colon V \times V \to \mathbb{N}$ represents right

[10] We believe that the required degree of collaboration makes the assumption practically tenable.
[11] We do not discuss content extraction attacks. For related discussions and countermeasures see [21].

transfers between pairs of compliant devices. For each $v \in V$, the *weight difference* is defined as $\Delta(v) = \sum_{v' \in V} E(v, v') - \sum_{v' \in V} E(v', v)$ (outgoing weight minus incoming weight). Note that for compliant devices v, $\Delta(v) \leq 0$. We define $U \subseteq V$, the set of nodes which has sold a bundle, but has not yet encashed it. If C is a compromised device which engages in large scale overselling, after a reasonable amount of time, the provider will detect C's behavior by noting that $\Delta(v_c)$ plus the number of yet-to-cash rights is positive, i.e. $\Delta(v_c) + \sum_{u \in U} \Delta(u) > 0$. By putting time limits on encashing payment orders (strengthening A2), the time bound on detecting compromised devices can be controlled.

Secondly, a compromised device can attack by not paying for content it receives. This violation of assumption A6 can be detected by the bank. Since compliant devices do not violate the assumptions, any device violating A6 must be compromised. This requires that compliant devices cannot buy more than they can pay for – i.e. they must have some awareness of available credit.

Naturally, in the preferred case, detected compromised devices are confiscated. However, in practice there will be detected, compromised devices interacting with compliant devices. As a practical measure to limit interaction between compliant devices and known compromised devices, a Device Revocation List (DRL), containing public keys of detected compromised devices, can be used. The provider keeps a complete, up to date version of the DRL. There are various ways to distribute the DRL to compliant devices, providing a trade-off between storage required and security provided. We examined various update mechanisms between two extremes: providing each device with the entire DRL upon each contact (as in [21]), and storing only public keys of compromised devices that have had contact with the device. In the latter case, a detected attacker can still contact each compliant device at least once, which is not desirable.

As the DRL will rarely decrease (only when a compromised device is confiscated), we propose an update scheme which operates as follows: Provider P has the complete DRL L. Each device keeps a list of contacted devices; called f_c (friends of C) for device C. Each device partitions the DRL stored onto it in two partitions: $L_c(s)$ (self-contacts) and $L_c(o)$ (contacted by others). On contact with provider P, $L_c(s)$ is updated to $f_c \cap L$. (To avoid an overly large f_c, it can be set to empty after this operation). On contact with a device D, $L_c(o)$ is updated to $L_c(o) \cup L_d(s)$.

This means that C checks all devices it has contacted, each time it contacts P, and when it contacts another device D it extends its own DRL with the compromised devices that have had contact with D. In this manner, a device will only store a small, but highly relevant portion of the DRL – namely that part with which it has had contact, or a direct contact of it has had contact. Due to the inclusion of contacts of direct contacts, the scheme provides a balance between storage requirements and speed of spreading updates.

7 Conclusions

This paper presents an extended version of the NPGCT DRM scheme, which addresses security issues of the original protocols and can detect and revoke circumvented devices at a lower communication cost. We also report on the formal specification and model checking of a non-trivial practical DRM scheme. DRM systems are inherently complicated and, thus, error prone. This calls for expressive and powerful formal verification tools to provide both content vendors and customers a certain degree of confidence in the security and fairness of the system. We have analyzed and validated our design goals on a finite model of the Nuovo DRM scheme.

As future work we are considering analyzing the accountability of the provider, which is absolutely trustable in this study, addressing possible anonymity concerns of customers and incor-

porating the payment phase into the formal model. We are also currently working on a practical implementation of Nuovo DRM using existing technologies.

References

1. B. Alpern and F. B. Schneider. Defining liveness. Technical Report TR 85-650, Dept. of Computer Science, Cornell University, Ithaca, NY, October 1984.
2. N. Asokan. *Fairness in electronic commerce*. PhD thesis, University of Waterloo, 1998.
3. G. Avoine, F. Gärtner, R. Guerraoui, and M. Vukolic. Gracefully degrading fair exchange with security modules. In *EDCC '05*, volume 3463 of *LNCS*, pages 55–71. Springer, 2005.
4. A. Basu, B. Charron-Bost, and S. Toueg. Simulating reliable links with unreliable links in the presence of process crashes. In *ACM WDAG '96*, volume 1151 of *LNCS*, pages 105–122. Springer, 1996.
5. J. Cederquist, R. Corin, and M. Torabi Dashti. On the quest for impartiality: Design and analysis of a fair non-repudiation protocol. In *ICICS'05*, volume 3783 of *LNCS*, pages 27 – 39. Springer, 2005.
6. I. Cervesato. The Dolev-Yao intruder is the most powerful attacker. In *LICS'01*. IEEE Computer Society Press, 16–19 June 2001.
7. E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking*. The MIT Press, 2000.
8. H. Comon and V. Shmatikov. Is it possible to decide whether a cryptographic protocol is secure or not? *J. of Telecommunications and Information Technology*, 4:3–13, 2002.
9. D. Dolev and A. C. Yao. On the security of public key protocols. *IEEE Trans. on Information Theory*, IT-29(2):198–208, 1983.
10. S. Even and Y. Yacobi. Relations amoung public key signature systems. Technical Report 175, Computer Science Department, Technicon, Haifa, Israel, 1980.
11. J.-C. Fernandez, H. Garavel, A. Kerbrat, R. Mateescu, L. Mounier, and M. Sighireanu. CADP: A protocol validation and verification toolbox. In *CAV '98*, volume 1102 of *LNCS*, pages 437–440. Springer-Verlag, 1996.
12. W. Fokkink, N. Ioustinova, E. Kesseler, J. van de Pol, Y. S. Usenko, and Y. A. Yushtein. Refinement and verification applied to an in-flight data acquisition unit. In *Proc. CONCUR '02*, volume 2421 of *LNCS*, pages 1–23. Springer, 2002.
13. J. F. Groote and A. Ponse. The syntax and semantics of μCRL. In *Algebra of Communicating Processes '94*, Workshops in Computing Series, pages 26–62. Springer-Verlag, 1995.
14. S. Guth. Rights expression languages. In *ACM DRM'03*, pages 101–112, 2003.
15. J. Alex Halderman and Edward Felten. Lessons from the Sony CD DRM episode. In *Proc. the 15th USENIX Security Symposium*, (to appear) 2006.
16. J. Heather, G. Lowe, and S. Schneider. How to prevent type flaw attacks on security protocols. In *CSFW '00*, page 255, Washington, DC, USA, 2000. IEEE Computer Society.
17. H. Jonker, S. Krishnan Nair, and M. Torabi Dashti. Nuovo DRM paradiso: formal specification and verification of a DRM protocol. Technical Report SEN-R0602, CWI, Amsterdam, 2006.
18. R. Mateescu and M. Sighireanu. Efficient on-the-fly model-checking for regular alternation-free μ-calculus. *Sci. Comput. Program.*, 46(3):255–281, 2003.
19. C. Meadows. Formal methods for cryptographic protocol analysis: Emerging issues and trends. *IEEE Journal on Selected Areas in Communication*, 21(1):44–54, January 2003.
20. D. Mulligan and A. Burstein. Implementing copyright limitations in rights expression languages. In *ACM DRM'02*, pages 137–154, 2002.
21. S. Nair, B. Popescu, C. Gamage, B. Crispo, and A. Tanenbaum. Enabling DRM-preserving digital content redistribution. In *7th International IEEE Conference on E-Commerce Technology*, pages 151–158, München, Germany, 19-22, July 2005. IEEE Computer Society.
22. J. Pang, W. Fokkink, R. F. H. Hofman, and R. Veldema. Model checking a cache coherence protocol for a Java DSM implementation. In *Proc. IPDPS 03*, page 238. IEEE Computer Society, 2003.
23. R. Pucella and V. Weissman. A logic for reasoning about digital rights. In *CSFW '02*, page 282, Washington, DC, USA, 2002. IEEE Computer Society.
24. J. Tygar. Atomicity in electronic commerce. In *PODC '96*, pages 8–26. ACM press, 1996.