Enforcing Semantics-Aware Security in Multimedia Surveillance*

Naren Kodali[2], Csilla Farkas[3,4], and Duminda Wijesekera[1,2]

[1] Center for Secure Information Systems
[2] Dept of Info. Systems and Software Eng.,
George Mason University, Fairfax, VA 22030–4444
nkodali@gmu.edu, dwijesek@gmu.edu
[3] Information Security Laboratory
[4] Dept. of Computer Science and Engineering,
University of South Carolina, Columbia, SC-29208
farkas@cse.sc.edu

Abstract. Continuous audio-visual surveillance is utilized to ensure the physical safety of critical infrastructures such as airports, nuclear power plants and national laboratories. In order to do so, traditional surveillance systems place cameras, microphones and other sensory input devices in appropriate locations [Sch99]. These facilities are arranged in a hierarchy of physical zones reflecting the secrecy of the guarded information. Guards in these facilities carry clearances that permit them only in appropriate zones of the hierarchy, and monitor the facilities by using devices such as hand-held displays that send streaming media of the guarded zones possibly with some instructions. The main security constraint applicable to this model is that any guard can see streams emanating from locations with secrecy levels equal to or lower than theirs, but not higher. We show how to model these surveillance requirements using the synchronized multimedia integration language (SMIL) [Aya01] with appropriate security enhancements. Our solution consists of imposing a multi-level security model on SMIL documents to specify surveillance requirements. Our access control model ensures that a multimedia stream can only be displayed on a device if the security clearance of the display device dominates the security clearance of the monitored zone. Additionally, we pre-process a set of cover stories that can be released during emergency situations that allow using the services of guards with lower clearances without disclosing data with higher sensitive levels. For this, we create a view for each level, and show that these views are semantically coherent and comply with specified security polices.

1 Introduction

Physical structures such as air-ports, nuclear power plants and national laboratories are considered critical, and therefore are guarded continuously. Although the

* This work was partially supported by the National Science Foundation under grants CCS-0113515 and IIS-0237782.

S. Spaccapietra et al. (Eds.): Journal on Data Semantics II, LNCS 3360, pp. 199–221, 2004.

ultimate security providers are human guards, they are aided by physical surveillance instruments consisting of networked audio-visual devices such as cameras and microphones. Additionally, such facilities also have a hierarchical structure reflecting the levels of secrecy of the information contained in them. Accordingly, people accessing a zone carry appropriate clearances. For example, common areas such as ticket counters are accessible to all people, but baggage areas are accessible to an authorized subset only. Furthermore those that are allowed in the control towers are further restricted, reflecting the sensitivity of the control information. Thus audio-visual monitoring of these facilities must respect these sensitivities. For example, the guards that are only allowed in baggage area have no need to see the cameras monitoring the control towers. Consequently, there is a need to restrict the distribution of surveillance streams to only those guards with appropriate levels of clearances. Doing so using the *synchronized multimedia integration language (SMIL)* [Aya01] is the subject matter of this paper. Here we provide a framework to do so by using SMIL to specify the streaming media requirements, and a multi-level security (MLS) model for security aspects. Consequently, we decorate SMIL documents with security requirements so that appropriate MLS model is imposed. We further show how to utilize the services of guards with lower clearances to aid in emergencies that may occur in high security zones by showing appropriately constructed multimedia cover stories.

We use SMIL because of two choices. Firstly, most display devices are now SMIL compatible [Spy, Nok], and secondly, by using W3C standards and recommendations, our framework can be Web-enabled. Toolkit support to integrate XML compliant services across various platforms [PCV02, Nok, Bul98, EUMJ] are available commercially and freely. Therefore, our framework can be implemented with appropriate tools and ported to a wide range of general-purpose mobile multimedia devices such as those available in automobile navigation systems and hand-held devices.

Secondly, although SMIL is an XML-like language for specifying synchronized multimedia, unlike XML formatted textual documents, multimedia constructs have semantics that predates XML. Therefore, it is necessary to specify SMIL documents that capture those semantics while enforcing specified security policies. We address this issue by proposing a *Multi Level Secure Normal Form (mlsNF)* for multimedia documents. Accordingly, we create secure views appropriate at each level of our MLS model.

Thirdly, given the runtime delays of an operational platform, we show how to generate an executable appropriate for a candidate runtime, which we refer to as *a display normal form* of a SMIL document. We then encrypt media streams in display normal form and transmit them to intended recipients under normal and emergency operating conditions.

The rest of the paper is organized as follows. Section 2 introduces multimedia surveillance and a running example for the problem domain. Section 3 provides a summary of related work and Section 4 reviews SMIL, the XML-like language for multimedia. Section 5 describes the algorithm for transforming to the *Multi Level Secure Normal Form (mlsNF)*and Section 6 proves the correctness of the

transformation algorithm. Section 7 addresses compile time issues and runtime activities including encryption and resource management. Section 8 concludes the paper.

2 Multimedia Surveillance

Physical safety of critical infrastructure such as airports, nuclear power plants and national laboratories require that they be continuously monitored for intrusive or suspicious activities. In order to so, traditional surveillance systems place cameras, microphones [Sch99] and other sensory input devices in strategic locations. Appropriate combinations of such continuously flowing information streams provide a clear understanding of the physical safety of the facility under surveillance. Mostly, secured facilities have several degrees of sensitivities, resulting in categorizing intended users according to their accessibility to physical locations. Similarly, guarding personnel are also categorized according to the sensitivity of the information they are authorized to receive under normal operating conditions. However, in response to unusual circumstances (e.g., emergencies) security personnel may be required to perform actions that are outside their normal duties leading to the release of data about the unauthorized areas. For example, in case of a fire in a high security area emergency workers who are unauthorized to access this area may still be required to obtain fire fighting materials. For this, they need to know what is the extent of the fire and what type of fire extinguisher to obtain. However, they should not be able to know the exact type of the material burning or any information about the burning area that is not directly necessary for their emergency duties. We address this problem by providing our multimedia surveillance system with a semantically rich, pre-orchestrated multimedia cover story repository, so that in emergencies cover stories can be released to lower security levels.

The main difference between a traditional MLS system and MLS for live surveillance feeds during day-to-day operations is the need to disseminate classified information continuously to appropriate personnel for the latter. We assume a multilevel security classification of physical areas depending on their geographical location and their corresponding surveillance data is considered to have the same classification. We develop a methodology to express multimedia compositions with their rich runtime semantics, techniques to enforce integrity and access control, and enable exploitation of cover stories to disseminate relevant material to unauthorized users during emergencies. In addition to enforcing MLS, we propose to record all sensory inputs obtained using the input devices, to be used for forensic analysis, as well as to improve the quality of cover stories.

Figure 1 shows a hypothetical research facility with varying levels of sensitivity. Assume that the area enclosed by the innermost rectangle ABCD contains weapons with highest degree of sensitivity and is accessible (and therefore guarded) by personnel with the highest level of clearance, say top secret (TS). The area between the rectangles PQRS and ABCD is classified at medium level of sensitivity and therefore requires personnel with secret (S) security clearances.

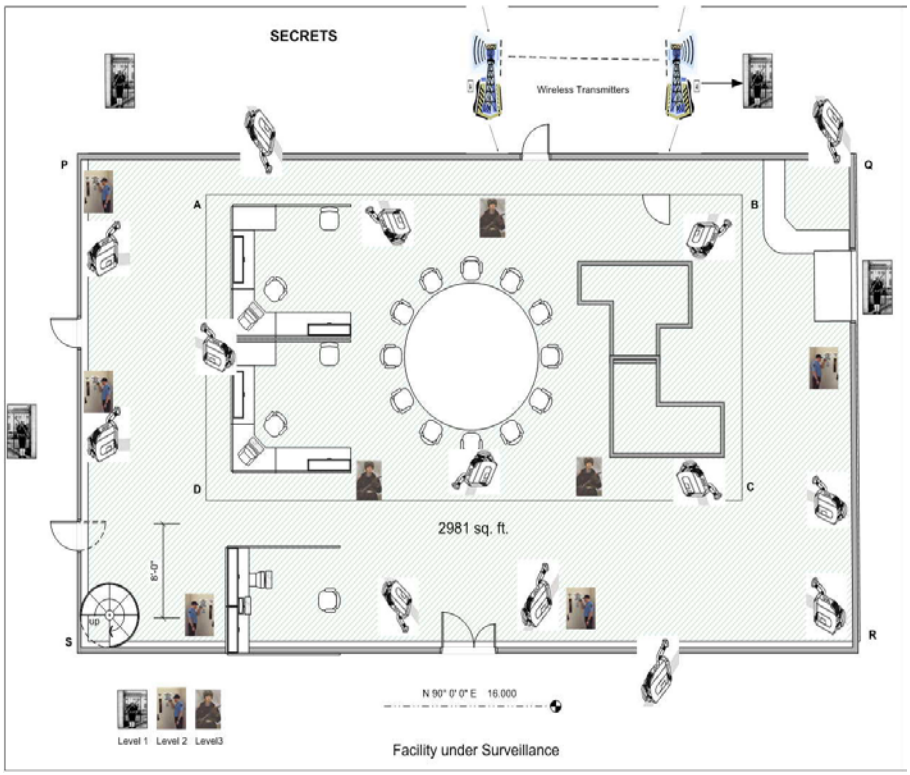

Fig. 1. A hypothetical facility under Surveillance

The area external to PQRS contains least sensitive material, and can be accessed by unclassified personnel, like visitors and reporters. We classify the areas into Top-Secret (TS), Secret (S) and Unclassified (UC) security levels with application domains, e.g., Dom as categories. Security labels form a lattice structure. For simplicity, we omit the application domain and use TS, S, and UC as security labels. The area inside ABCD is TS, the area inside of PQRS, but outside of ABCD is S, and the area outside PQRS is UC. Employees, guards, support services personnel, and general public have TS $>$ S $>$ UC clearances, where $>$ corresponds to the dominance relation defined in MLS systems. As shown in Figure 1, an area with higher level of sensitivity is a sub-part of areas with all lower levels of sensitivities. Therefore, a guard with top-secret clearance may be used in the classified area, but not vice versa. For surveillance purposes, cameras (infrared and normal light) and other devices such as microphones are situated throughout the facility. Multimedia streams emanating from these devices are continuously used to monitor the facility. We propose a design where all multimedia data is transmitted to a centralized control facility and then directed to handheld devices of appropriate security personnel.

3 Related Work

A distributed architecture for multi-participant and interactive multimedia that enables multiple users to share media streams within a networked environment is presented in [Sch99]. In this architecture, multimedia streams originating from multiple sources can be combined to provide media clips that accommodate look-around capabilities. Multilevel security (MLS) has been widely studied to ensure data confidentiality, integrity, and availability [Osb]. MLS systems provide controlled information flow based on the security classification of the protection objects (e.g., data items) and subjects of the MLS system (e.g., applications running in behalf of a user). To provide information confidentiality, data is allowed to flow only from low security levels to higher security levels [Low99]. Information security policies in databases aim to protect the *confidentiality* (secrecy) and *integrity* of data, while ensuring *availability* of data. In Multilevel Secure (MLS) systems direct violations of data confidentiality are prevented by mandatory access control (MAC) mechanisms, such as those based on the Bell-LaPadula (BLP) [San93] model. Mandatory (or lattice-based) policies are expressed via security classification labels that are assigned to *subjects*, i.e., active computer system entities that can initiate requests for information, and to *objects*, i.e., passive computer system entities that are used to store information. Security labels are composed from two components: 1) a hierarchical component, e.g., *public* $<$ *secret* $<$ *top-secret*, and 2) a sub-set lattice compartment, e.g., $\{\} \subset \{navy\} \subset \{navy, military\}$ and $\{\} \subset \{military\} \subset \{navy, military\}$, however, there is no subset relation between $\{military\}$ and $\{navy\}$. Security labels are formed by combining the two components together, i.e., (top-secret, $\{navy\}$), (secret, $\{navy, military\}$), etc. Security labels form a mathematical lattice structure with a dominance relation among the labels. If no dominance relation exists among the labels, then they are called incompatible. MAC policies control read and write operations on the data objects based on the classification labels of the requested data objects and the classification label (also called clearance) of the subject requesting the operation. For simplicity, in this work we only use the hierarchical component of the security labels, i.e., *public* $<$ *secret* $<$ *top-secret*. However, our results hold on full lattice-based access control models.

Regulating access to XML formatted text documents has been actively researched in the past few years offering a multitude of solutions. Bertino et al. [BBC$^+$00] have developed Author-X, a Java based system to secure XML documents that enforces access control policies at various granularities and corresponding user credentials. Author-X encodes security policies for a set of XML documents in an XML file referred to as the policy base containing both permissions and prohibitions. Damiani et al. [DdVPS00, DdVPS02] developed an access control model where the tree structure of XML documents is exploited using XPATH expressions to control access at different levels of granularity. The smallest protection granularity is an XPATH node, and security policies specify permissions or prohibitions to all descendent objects of a node.

Damiani et al. [DdV03] discuss feature protection of XML format images. Its primary focus is controlled dissemination of sensitive data within an image. They propose an access control model with complex filtering conditions. This model uses SVG to render the map of a physical facility. While this model could be used to represent our application, it is limited in flexibility and adaptability to certain issues related to physical security using MLS.

Bertino et al. [BHAE02] provides a security framework to model access control in video databases. They provide security granularity, where objects are sequences of frames or particular objects within frames. The access control model is based on the concepts of security objects, subjects, and permitted access modes, like viewing and editing. The proposed model is provides a general framework for the problem domain, but does not explain how access control objects to be released are formalized and enforced.

Stoica at al. [SF02] present cover stories for XML with the aim of hiding non-permitted data from the naive user. The work is motivated by the need to provide secure release of multilevel XML documents and corresponding DTD files in a semantically correct and inference free manner where security sensitivity is not monotonically increasing along all paths originating from the node. Substantial amounts of contemporary research addresses real-time moving object detection and tracking them from stationary and moving camera platforms [VCM], object pose estimation with respect to a geospatial site model, human gait analysis [VSA], recognizing simple multi-agent activities, real-time data dissemination, data logging and dynamic scene visualization. While they offer valuable directions to our research model, they are not a panacea to physical security.

None of the above approaches are completely satisfactory for multimedia surveillance . They primarily address textual documents and exploit the granular structure of XML documents. Multimedia for various reasons as stated has to be treated differently. Synchronization and integration of diverse events to produce sensible information is non-trivial when compared to textual data. The process of retrieval without losing the sense of continuity and synchronization needs better techniques and algorithms which all of the above models do not completely address. Kodali et al. [KW02, KWJ03, KFW03] propose models for multimedia access control for different security paradigms. A release control for SMIL formatted multimedia objects for pay-per-view movies on the Internet that enforces DAC is described in [KW02]. The cinematic structure consisting of acts, scenes, frames of an actual movies are written as a SMIL document without losing the sense of a story. Here access is restricted to the granularity of an *act* in a movie. A secure and progressively updatable SMIL document [KWJ03] is used to enforce RBAC and respond to traffic emergencies. In an emergency response situation, different roles played by recipients determine the media clips they receive.

In [KFW03] an MLS application for secure surveillance of physical facilities is described, where guards with different security classification in charge of the physical security of the building are provided live feeds matching their level in

the security hierarchy. This paper is an extended version of [KFW03], in which multimedia surveillance is described with limited operational semantics.

4 SMIL

SMIL [Aya01, RHO99] is an extension to XML developed by W3C to allow multimedia components such as audio, video, text and images to be integrated and synchronized to form presentations [RvOHB99]. The distinguishing features of SMIL over XML are the syntactic constructs for timing and synchronization of streams with qualitative requirements commonly known as QoS. In addition, SMIL provides a syntax for spatial layout including constructs for non-textual and non-image media and hyperlink support. SMIL constructs for synchronizing media are ⟨seq⟩, ⟨excl⟩ and ⟨par⟩. They are used to hierarchically specify synchronized multimedia compositions. The ⟨seq⟩ element plays the child elements one after another in the specified sequential order. The ⟨excl⟩ construct specifies that its children are played one child at a time, but does not impose any order. The ⟨par⟩ element plays all children elements as a group, allowing parallel play out. For example, the SMIL specification ⟨par⟩⟨video src=camera1⟩⟨audio src = microphone1⟩⟨/par⟩ specify that media sources camera1 and microphone1 are played in parallel. In SMIL, the time period that a media clip is played out is referred to as its active duration. For parallel play to be meaningful, both sources must have equal active durations. When clips do not have same active durations, SMIL provides many constructs to make them equal. Some examples are begin (allows to begin components after a given amount of time), dur (controls the duration), end (specifies the ending time of the component with respect to the whole construct), *repeatCount* (allows a media clip to be repeated a maximum number of times). In addition, attributes such as *syncTolerance* and *syncMaster* controls runtime synchronization, where the former specifies the tolerable mis-synchronization (such as tolerable lip-synchronization delays) and the latter specifies a master-slave relationship between synchronized streams. In this paper, we consider only the basic forms of synchronization construct which means, we do not specify *syncMaster* and *syncTolerance*. Thus we assume that components of ⟨par⟩ have equal play out times and they begin and end at the same time.

An important construct that we use is ⟨switch⟩ allowing one to switch among many alternative compositions listed among its components. These alternatives are chosen based on the values taken by some specified attributes. For example, ⟨switch⟩ ⟨audio src="stereo.wav" systemBitrate⟩25⟩⟨audio src="mono.wav" systemBitrate ⟨ 25⟩⟨/switch⟩ plays stereo.wav when the SMIL defined attribute *systemBitrate* is at least 25 and mono.wav otherwise. We use this construct to specify our surveillance application. In order to do so, we define two custom attributes *customTestMode* that can take values "normal" and "emergency" and *customTestSecurity* that take any value from ("TS","S","UC"). The first attribute is used to indicate the operating mode that can be either normal or emergency and the second attribute indicates the security level of streams that can

be top secret, secret or unclassified. SMIL also requires that every application-defined attribute (custom attribute in SMIL terminology) have a title and a default value. It further has a special flag *override* that makes the value *hidden* or *visible*. When override takes the value hidden, the player is not allowed to change the value of the custom attributes. That feature is useful in specifying security attributes that are not to be altered by SMIL players.

Surveillance requirements, such as those in the example given in the SMIL fragment below specifies which multimedia sources have to be displayed under the two operating conditions. We assume that the source document specifies the security label of each source and that MLS policies are used to ensure that guards are permitted to view only those multimedia sources that are dominated by their security clearances. For this, we preprocess a given MLS multimedia document and produce views that are permitted to be seen by guards for each security classification. Then, we separately encrypt and broadcast multimedia documents for each category, to the appropriate locations by efficient use of bandwidth. In order to achieve this objective, we first transform every SMIL document with proposed security and mode attributes to three SMIL documents, where all security labels in each document consists of solely one *customTestSecurity* attribute, namely the one that is appropriate to be seen by guards with the label value. We now formally state and prove that this can be done for an arbitrary SMIL document with our security labels.

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language">
<customAttributesMODE>
      <customTestMode="Normal" title="Normal Mode"
        defaultState="true" override="hidden"
       <customTestMode id="Emergency" title="Emergency Mode"
        defaultState="true" override="hidden"
<customAttributesMODE> <customAttributesSecurity>
      <customTestSecurity id="TS" title="Top-Secret"
        defaultState="true" override="hidden"/>
      <customTestSecurity id="S" title="Secret"
        defaultState="true" override="hidden"/>
      <customTestSecurity id="UC"  title="Unclassified"
        defaultState="true" override="hidden"/>
</customAttributesSecurity>
<body>
<switch>
//Classification is TS(Top-Secret)
<par customTestMODE= "Normal">
<video src="CameraTS1.rm" channel="video1" customTestSecurity="TS"/>
<audio src="CameraTS1.wav" customTestSecurity="TS" />
//Classification is S(Secret)
<video src="CameraS1.rm" channel="video1" customTestSecurity="S"/>
<audio src="CameraS2.wav" customTestSecurity="S"/>
//Classification is U(Unclassified)
<video src="CameraU1.rm" channel="video2" customTestSecurity="S"/>
<audio src="CameraU1.wav" customTestSecurity="S" /> </par>
```

```
<par customTestMODE= "Emergency">
//All 3 above together (Total of 6 feeds)
//Here are the secret cover stories
<par>
<video src="CoverstoryTS-to-S1.rm" channel="video1"
id="TS-to-Secret" customTestSecurity="S"/>
<audio src="CoverstoryTS-toS1.wav" customTestSecurity="S"/>
</par>
//Here are the unclassified cover stories
<par>
<video src="CoverstoryTS-to-U1.rm" channel="video1"
id="TS-toUC1" customTestSecurity="U"/>
<audio src="CoverstoryTS-to-U1.wav" customTestSecurity="U"/>
<video src="CoverstoryS-to-U1.rm" channel="video1" id="Secret-toUC1"
customTestSecurity="U"/>
<audio src="CoverstoryS-to-U1.wav" customTestSecurity="U"/>
</par>
//Followed by normal the TWO UC camera feeds.
</switch>
    </body>
</smil>
```

As the fragment shows, the document consists of two sections, where the first section defines the custom attribute *customTestMode* with values "Normal" and "Emergency". Because the second and the fourth lines of fragment specify that *customTestMode* is hidden, the value of this attribute corresponding to each stream cannot be reset later. The second part of the file consists of a switch statement consisting of collection of media streams connected by ⟨par⟩ constructs. Notice that there are two section inside the ⟨switch⟩ statement, where the first one begins with the line ⟨par customTestMODE= "Normal"⟩ and the second one begins with the line ⟨par customTestMODE= "Emergency"⟩. That specifies that the streams inside be shown under normal and emergency operating conditions. In this example, each area has a camera and a microphone to record audio and video streams to be transmitted to appropriate guards. They are named CameraTS1.rm, CamerU1.wav etc. The security classification of each source is identified by the application defined SMIL attribute customTestSecurity. For example, ⟨video src="CameraTS1.rm" channel="video1" customTestSecurity="TS"/⟩ specifies that the video source named CameraTS1.rm has the Top Secret security level. The intent being that this source is to be shown only to top-secret guards. As the second half of the document shows, there are three audio-visual cover stories named CoverstoryTS-to-S1.rm to CoverstoryS-to-UC1.wav are shown with the appropriate security level specified with the attribute customTestSecurity. The main composition is encoded using a ⟨switch⟩ statement that is to be switched based on the operating mode (normal or emergency).

5 MLS Normalform and the Translation Algorithm

In this section we define the Multi Level Secure Normal Form (mlsNF) and also provide the algorithm for transforming an arbitrary SMIL specification into its MLS normal form.

Definition 1 (MLS Normal Form) *We say that a SMIL specification S is in Multi Level Secure Normal Form (mlsNF) if it is of one of the following forms:*

1. *It is of the form ⟨ par ⟩ Cts(S) Cs(S) Cu (S) Cud(S) Cod ⟨ /par ⟩ where all attributeTestSecurity attributes in Cts(S), Cs(S), Cu(S) are respectively TS, S and U. In addition, Cud(S) has no attributeTestSecurity and Cod(S) has two different value set for attributeTestSecurity.*

2. *It is of the form ⟨ par ⟩ Cts(S) Cs(S) Cu (S) Cud(S) Cod(S) ⟨ /par ⟩ with one or two components of ⟨ par ⟩ may be missing. Here Cts(S), Cs(S) and Cu(S), Cud(S) Cod(S) satisfy requirements stated above.*

3. *It is of the form Cts(S), Cs(S), Cu(S), Cud(S), Cod(S) where Cts(S), Cs(S), Cu(S), Cud(S) and Cod(S) satisfy requirements stated above. We say that Cts(S), and Cs(S) and Cu(S) are respectively the top secret, secret and unclassified views of the specification S. Cud(S) is the view with missing security classifications and Cod(S) is the view with contradictory security classifications.*

As stated in Definition 1, a SMIL specification in mlsNF is one that is parallel composition of at most three specifications, where each specification belongs to one security class, that are said to be the views corresponding to the respective security classes. Notice that in Definition 1, the latter two cases are degenerate cases of case 1 where one or more views of the specification become null. In attempting to create views from an arbitrary SMIL document, one encounters two undesirable situations. The first is the missing security classifications resulting in a non-null Cud(S). The other is the situation with contradictory security classification due to over specification. An example under specified SMIL specification is ⟨ audio src= "myAudio.wav" ⟩ , and an example contradictory specification is ⟨ video src= "myMovie.rm" attributeTestSecurity=TS attributeTestSecurity=S ⟩. Thus, it is tempting to avoid such situations by applying completeness and conflict resolution policies [JSSS01] designed to be used in XML formatted and databases. Note, that completeness and conflict resolution polices were intended to be used for inheritance hierarchies. Because SMIL hierarchies are not due to inheritances and instead they are syntactic constructs for media synchronization, blindly applying such policies to resolve under and over specification of SMIL documents destroys the synchronized play out semantics of media streams. In this paper, we use the neutral policy of discarding under and over specified fragments Cud(S) and Cod(S) of a SMIL specification S.

The Algorithm 1 details the mechanics of conversion from an arbitrary SMIL specification into mlsNF. It describes how the rewrite should be done when we encounter different time containers, some of which are nested. The generated output would have atmost three parallel compositions each corresponding to a unique security level. The MLS paradigm has an unique property which allows subjects with a higher classification access to the view of the lower classified subjects. This algorithm takes this property into consideration when generating smilNF.

Algorithm 1 TOmlsNF (Conversion to MLS Normal form)

INPUT : Arbitrary SMIL fragment. Possible classifications Top-Secret, Secret, Unclassified.
OUTPUT : mlsNF
(s) is an arbitrary SMIL specification (as described in 4 with a possible Security classification.
if (s) is ⟨ seq ⟩ $s_1 s_2$ ⟨ /seq ⟩ **then**
C_{ts} (s) = ⟨ seq ⟩ ⟨ par ⟩ $C_{ts}(s_1)$ ⟨ /par ⟩ ⟨ par ⟩ $C_{ts}(s_2)$ ⟨ /par ⟩ ⟨ /seq ⟩
C_s (s) = ⟨ seq ⟩ ⟨ par ⟩ $C_s(s_1)$ ⟨ /par ⟩ ⟨ par ⟩ $C_s(s_2)$ ⟨ /par ⟩ ⟨ /seq ⟩
C_u (s) = ⟨ seq ⟩ ⟨ par ⟩ $C_u(s_1)$ ⟨ /par ⟩ ⟨ par ⟩ $C_u(s_2)$ ⟨ /par ⟩ ⟨ /seq ⟩
else if (s) is ⟨ par ⟩ s_1 s_2 ⟨ /par ⟩ **then**
C_{ts} (s) = ⟨ par ⟩ $C_t s(s_1)$ ⟨ /par ⟩ ⟨ par ⟩ $C_t s(s_2)$ ⟨ /par ⟩
C_s (s) = ⟨ par ⟩ $C_s(s_1)$ ⟨ /par ⟩ ⟨ par ⟩ $C_s(s_2)$ ⟨ /par ⟩
C_u (s) = ⟨ par ⟩ $C_u(s_1)$ ⟨ /par ⟩ ⟨ par ⟩ $C_u(s_2)$ ⟨ /par ⟩
end if
if either of $C_x(s_i)$ are empty for some x ∈ {TS,S,U} and i ∈{1,2} **then**
$C_x(s_i)$ in the right hand sides above must be substituted by ϕ (S_i) where ϕ (s_i) is defined as ⟨ audio or video src = empty ⟩
end if
If Security classification =Top-Secret, then C_{ts} (s) = (s)
If Security classification =Secret, then C_{ts}(s) = ϕ ,C_s (s) = (s)
If Security classification=Unclassified, then C_{ts} (s) = ϕ , C_s(s) = ϕ , and C_u (s)= (s).
Then let mlsNF (s) = ⟨ seq ⟩ ⟨ par ⟩ C_{ts} ⟨ /par ⟩ ⟨ par ⟩ (s) C_s ⟨ /par ⟩ ⟨ par ⟩ (s) C_u (s) ⟨ /par ⟩ ⟨ /seq ⟩ .

We now have to ensure that Algorithm 1 preserves semantics. That is, top secret, secret and unclassified viewers of a specification S will view Cts(S), Cs(S) and Cu(S) respectively. This proof is easy, provided that we have a formal operational semantics for SMIL. While providing such semantics is not difficult, it does not exist yet. Therefore, while we are working on it, we provide a rudimentary operational semantics for the purposes of showing that our algorithms work as expected.

5.1 Operational Semantics for SMIL

In this section, we provide a simple operational semantics for media streams and SMIL documents constructed using ⟨par⟩, ⟨seq⟩ and ⟨switch⟩ commands. The sole objective of this exercise is to show that Algorithm 1 transforms a SMIL document to a collection of other SMIL documents to respect this semantics. The latter is referred to as semantic equivalence [Mul87]. Following customary practices in programming language semantics, our operational semantics and the proof of semantic equivalence will be inductive in nature. It is worth noting that our semantics is only applicable to our application scenario and syntactic constructs, and its extension to other purposes and constructs form our ongoing work.

Definition 2 (Timed Display Instance) *We say that a quadruple (S, T-begin, T-end, Security Set) is a timed display instance provided that:*

1. *S is a basic media element with a finite active duration $\delta \geq 0$ and T-begin leq T-end are arithmetic expressions of a single real variable* `t` *satisfying T-end = T-begin + δ.*
2. *Security set a subset of TS, S, U consisting of attributeTestSecurity attribute values of S.*
3. *We say that a set of timed display instances is a timed display set provided that there is at least one timed display element with t as its T-begin value.*
4. *Taken as expressions containing the variable t, the smallest T-begin value of a timed display set is said to be the origin of the timed display set. We use the notation O(TDI) for the origin of the timed display set TDI.*
5. *Taken as expressions containing the variable t, the largest T-begin value of a timed display set is said to be the end of the timed display set. We use the notation E(TDI) for the end of the timed display set TDI.*

The following two elements tdi_1 and tdi_2 are examples of timed display instances.

1. tdi_1 = (⟨video, src= "myVideo.rm", dur=5, attributeTestSecurity=TS ⟩, t, t+7, TS)
2. tdi_2 = (⟨audio, src= "myAudio.rm", dur=10, attributeTestSecurity=U⟩, t+7, t+17, U)

Therefore, {tdi_1, tdi_2} is timed display set with its origin t and end t+17. The intent here is to consider TDI= {tdi_1,tdi_2} as a possible playout of the SMIL specification ⟨seq⟩ ⟨video, src= "myVideo.rm", dur=5, attributeTestSecurity=TS⟩, ⟨audio, src= "myAudio.rm", dur=10, attributeTestSecurity=U ⟩ ⟨/seq⟩ that begin at an arbitrary but thereafter fixed time t and ends at t+17. Now we describe some algebraic operations on timed display sets that are necessary to complete the definition of our operational semantics of SMIL. The first is that of origin substitution defined as follows.

Definition 3 (Algebra of Timed Display Sets: Substitution) *Suppose TDS is a timed display set with the formal time variable* `t` *and* `s` *is any*

arithmetic expression possibly containing other real valued variables. Then TDS(s/t) is the notation for the timed display set obtained by syntactically substituting all timing values (that is T-begin and T-end values) `t` *by* `s` *in all expressions of TDS.*

For the example TDI given prior to Definition 3, TDI(2t+7/t) consists of tdi_1(2t+7/t),tdi_2(2t+7/t) where tdi_1(2t+7/t) and tdi_2(2t+7/t) are defined as:

1. tdi_1(2t+7/t) = (⟨video, src= "myVideo.rm", dur=5, attributeTestSecurity= TS⟩, 2t+7, 2t+21, {TS})
2. tdi_2(2t+7/t) = (⟨audio, src= "myAudio.rm", dur=10, attributeTestSecurity= U⟩, 2t+21, 2t+31, {U})

The reason for having Definition 3 is that in order to provide formal semantics for the ⟨seq⟩ operator, it is necessary to shift the second child of the ⟨seq⟩ by the time duration of its first child and recursively repeat this procedure for all of ⟨seq⟩'s children. To exemplify the point, the first example the TDI= {tdi_1,tdi_2} is infact {tdi_1} ∪ TDI′(t+7/t)} where TDI′ is given by tdi′ = (⟨audio, src = "myAudio.rm", dur=10, attributeTestSecurity = U⟩, t, t+10, {U}). We are now ready to obtain operational semantics for SMIL specifications, provide the following assumptions are valid.

Definition 4 (Basis Mapping) *Suppose M is the set of basic media elements of S. Then any mapping [[ ]] from M to a set of Timed Display Instances TDI is said to be a basis mapping for a denotation iff all T-begin elements of M have the same value t, where t is a real variable. Then we say that [[ ]] is a basis mapping parameterized by t.*

Lemma 1 (Existence of basis mappings). *Suppose M is a set of basic media streams with time durations. Then M has a basis mapping.*

Proof: For each media stream m= ⟨type, src= "...", dur=value, attributeTestSecurity= "..." type⟩, in M, let [[M]] map to (m, t, t+value, Att Values). Then [[ ]] is a basis mapping.
We now use a basis mapping to define operational semantics of any SMIL specification S as follows.

Definition 5 (Operational Semantics for SMIL) *Suppose S is a SMIL specification and [[ ]] is a basis mapping for the basic media elements B of S with the formal parameter t. Then we inductively extend [[ ]] to S as follows.*

1) [[Null]] = Φ

2) [[⟨seq⟩ S1 S2⟨/seq⟩]] = [[S1]] U [[S2]](end([[S1]])/t)

3) [[⟨par⟩ S1 S2⟨/par⟩]] = [[S1]] U [[S2].

4) [[⟨switch⟩ S1 S2 ⟨/switch⟩]] = [[S1]] if S1 satisfies the attribute of the switch. = [[S2]] otherwise if S2 satisfies the attribute of the switch. = Φ otherwise.

We now say that the extended mapping [[ ]] is a semantic mapping parameterized by `t`. It is our position that the informal definition given by the SMIL specification is captured by our operational semantics, provided we are able to evaluate the attribute of the switch. This can be easily formalized using customary practices of program language semantics, and is therefore omitted here for brevity. We now formally state and prove the semantic equivalence of Algorithm 1. That shows that rewritten specification has the same operational semantics as the original that we offer as the correctness argument for the rewrite.

6 Correctness of the Translation Algorithm

Theorem 1 (Correctness of Algorithm 1) *Suppose that S is a SMIL specification and [[ ]] is a semantic mapping parameterized by t. Then [[S]] = [[mlsNF(S)]].*

Proof: As stated earlier, this proof also proceeds by induction on the structure of S. Thus, for the sake of brevity, we show one base case and one inductive case.
Example Base Case:
Suppose S is ⟨ type src=" ", dur=n, attributeTestSecurity="S" type ⟩. Then, by Algorithm 1,

```
Cts(S) = Null, Cs(S) = S, Cu(S) = Null, Cud(S) = Null and Cod(S) =
Null. Therefore, [[<par>Cts(S) Cs(S) Cu(S) Cud(S) Cod(S) </par>]]=
[[<par>Null S Null Null Null</par>]] = [[Null]] U [[S]] U [[Null]]
U [[Null]] U [[Null]] = [[S]]. Hence [[mlsNF(S)]] = [[S]].
```

Example Inductive Case:
Suppose S is ⟨seq⟩ S1 S2⟨/seq⟩. Then, from Algorithm 1,

```
[[mlsNF(S)]] =[[<par>Cts(S) Cs(S) Cu(S) Cud(S) Cod(S) </par>]] =
[[Cts(S)]] U [[Cs(S)]] U [[Cu(S)]] U [[Cud(S)]] U [[Cod(S)]] =
[[<seq>Cts(S1) Cts(S2)</seq>]] U [[<seq>Cs(S1) Cs(S2)</seq>]] U
[[<seq>Cu(S1) Cu(S2)</seq>]] U [[<seq>Cud(S1) Cud(S2)</seq>]] U
[[<seq>Cod(S1) Cod(S2)</seq>]]
    =
    [[Cts(S1)]] U [[Cts(S2)]](end([[Cts(S1)]])/t) U[[Cs(S1)]]
    U [[Cs(S2)]] (end([[Cs(S1)]])/t)
    U [[Cu(S1)]] U [[Cu(S2)]](end([[Cu(S1)]])/t) U[[Cud(S1)]]
    U [[Cud(S2)]] (end([[Cud(S1)]])/t)
    U [[Cod(S1)]] U [[Cod(S2)]](end([[Cod(S1)]])/t)
```

Conversely,

```
[[S]]   = [[<seq> S1 S2 </seq>]] = [[S1]] U
[[S2]](end(S1)/t)
= [[Cts(S1)]] U [[Cs(S1)]] U[[Cu(S1)]] [[Cud(S1)]] U [[Cod(S1)]]
U( [[Cts(S2)]] U [[Cs(S2)]] U[[Cu(S2)]] [[Cud(S2)]] U [[Cod(S2)]])
(end(S1)/t)
```

by the inductive assumption.

But notice that

```
([[Cts(S2)]] U [[Cs(S2)]] U[[Cu(S2)]] [[Cud(S2)]]U
[[Cod(S2)]]) (end(S1)/t)
    =
[[Cts(S2)]](end([[Cts(S1)]])/t) U
[[Cs(S2)]](end([[Cs(S1)]])/t) U [[Cu(S2)]](end([[Cu(S1)]])/t) U
[[Cud(S2)]](end([[Cud(S1)]])/t) U [[Cod(S1)]] U
[[Cod(S2)]](end([[Cod(S1)]])/t)
```

Therefore [[mlsNF(S)]] = [[S]], thereby justifying the inductive case.

6.1 Representing Secure Views in SMIL

On rewriting the SMIL fragment in Section 4 into the MLS Normal form we create different views for each of the following cases represented as a separate SMIL document. In the SMIL fragment represented below , we have the format of such a specification denoting the entire structure of a "Top-Secret" view in the normal mode and a "Secret" view in the emergency mode.

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language">
<customAttributesMODE>
      ---
<customAttributesSecurity>

<seq>
<switch>
<par customTestMode="Normal" customTestSecurity = "TS">
 <par> <video src="Tsamera1.rm" channel="video1" dur="45s"/>
      <audio src="TSCamera1.wav"  />
 </par>
 <par> <video src="TSCamera2.rm" channel="video1" />
       <audio src="TSCamera2.wav"/> </par> </par>
<par customTestMode ="Normal" customTestSecurity = "S">
XXXXXXXXXX //Normal Form View for Normal Mode "S" Class
</par>
<par customTestMode ="Normal" customTestSecurity = "UC">
XXXXXXXXXX //Normal Form View Normal Mode "UC" Class
</par>
<par customTestMode ="Emergency" customTestSecurity = "TS">
XXXXXXXXXX //Normal Form View for Normal Mode "TS" Class
</par>
<par customTestMode ="Emergency" customTestSecurity = "S">
 <video src="SCamera1.rm" channel="video2" dur="25s"/>
 <audio src="SCamera1.wav"  /> </par>
<par> <video src="Scamera2.rm" channel="video2"/>
      <audio src="Scamera2.wav"  /> </par>
<par>
<video src="CoverstoryTS1.rm" channel="video1" id="TSCoverstory1"/>
```

```
<audio src="CoverstoryTS1.wav" />
</par>
<par>
<video src="CoverstoryTS1.rm" channel="video1" id="TSCoverstory1"/>
<audio src="CoverstoryTS1.wav"  />
</par>
<par customTestMode ="Emergency" customTestSecurity = "UC">
 XXXXXXXX//Normal Form View for Emergency Mode "UC" Class
</par> </switch> </seq>
```

Each view will be made into a SMIL document and named as follows ModeNClassTS.smil, ModeEClassTS.smil, ModeNClassS.smil, ModeEClassS.smil, ModeNClassUC.smil, ModeEClassUC.smil depending on it mode and classification attributes.

7 Runtime Operations

In the most general case, a SMIL specification in mlsNF is of the form ⟨par⟩ Cts Cs Cu Cod Cud ⟨/par⟩ where Cts Cs Cu Cod and Cud respectively have top secret, secret, unclassified, over specified and under specified security levels. How one resolves under specification and over specification is a matter of policy, and is not addressed in this paper. Independently, Cts, Cs, Cu are to be shown to guards with top secret, secret, and unclassified clearances. In addition, in order to respond to emergencies, these specifications have a mode switch encode using a custom attribute *attributeTestMode*. As observed in Figure 2, this attribute is to be evaluated at the beginning of a ⟨switch⟩ statement. That is unsatisfactory for intended purposes, after this switch statement is executed, the operating mode could vary many times. Because the ⟨switch⟩ is evaluated only once, the SMIL specification is now oblivious to such changes in application situations. In this section, we show how to rewrite a SMIL document with one ⟨switch⟩ statement for changing a mode to that one that makes the *attributeTestMode* be evaluated at regular intervals. Although in theory any system could switch its operating mode in an arbitrarily small time intervals, practical considerations limits this interval to a minimum. This minimum switching granularity may depend upon many parameters such as hardware, software and the inherent delays in of switching on firefighting and other emergency related equipment. Therefore, given a switching delay D, we rewrite the given SMIL document so that the mode attribute *attributeTestMode* re-evaluated every D time units. How that is done is discussed in the next section.

7.1 Informal Display Normal Form

The following SMIL specification given below, has the same structure as the fragment considered in Section 4. If we want to break up this specification so that the *attributeTestMode* is tested each D units of time and the switch reevaluated, then the fragment S1 can be translated as shown in S2.

```
S1 =<switch> <par  attributeTestMode= "normal"> XX </par> <par
attributeTestMode= "emergency"></par> </switch>

S2 = <par dur=D, repeatCount="indefinite"><switch> <par
attributeTestMode="normal"> XX</par> <par
attributeTestMode="emergency">YY </par> </switch> </par>
```

Notice that the outer ⟨par⟩ construct specifies that enclosing specification be executed for duration of D time units and repeated indefinitely. However, the outer ⟨par⟩ construct has only one element, namely the switch. Therefore, the ⟨switch⟩ construct is executed for infinitely many times, and each time the *attributeTestMode* is tested. Given a SMIL specification with the *attributeTestMode* specified in the form where the switch is reevaluated every D time units is said to be in display normal form for the attribute *attributeTestMode* and time duration D. We can now informally say that every SMIL document where the *attributeTestMode* is used in the stated form can be translated into its display normal form. We stress the informal nature of our argument because of our commitment to limited operational semantics. However these semantics can be enhanced so that this construction will preserve semantic equivalence.

7.2 Operational Semantics for Making Display Normal Form Semantically Equivalent

In this section, we briefly show how our operational semantics of SMIL can be enhanced so that any SMIL construction with a specified structure and its display normal form are semantically equivalent. First, we close timed display sets under finite concatenations and re-interpret SMIL semantics with respect to them.

Definition 6 (Algebra of TDS: Downward Closure and Concatenation) *Suppose* tdi_1*=(<type src="xx", ..dur=d1, attributeTestSecurity="y", T-begin1,T-end1>, {y}) and* tdi_2*=(<type src="xx", ..dur=d2, attributeTestSecurity="y", T-begin2, T-end2>, {y}) are two timed display units with the same source, attributeTestSecurity values, security components satisfying T-end1=T-begin2.*

1. *Then we say that* tdi_3*=(⟨type src="xx", ..dur=d1, attributeTestSecurity ="y", T-begin1,T-end2⟩, {y}) is the concatenation of* tdi_1 *and* tdi_2*. We denote the concatenation of* tdi_1 *and* tdi_2 *by* tdi_1*;*tdi_2*.*
2. *We say that a timed display set TDS is concatenation closed if* tdi_1*,*tdi_2 $\in$ *TDS* $\Rightarrow$ tdi_1*;*tdi_2 $\in$ *TDS.*
3. *We say that a timed display set TDS is downward closed if .=(⟨type src="xx", ..dur=d1, attributeTestSecurity="y", T-begin1,T-end1⟩, {y}) ∈ TDS, then =(⟨type src="xx", ..dur=d1, attributeTestSecurity="y",T-begin1′,T-end1′ ⟩,{y})∈ TDS for any T-begin′ > T-begin and T-end′ < t-end.*

According to Definition 6, downward closure allows any timed display set to include all segments of already included media streams. Concatenation closure allows piecing together successive segments of the same stream to obtain longer streams.

Lemma 2 (Minimal Concatenation Downward Closure of TDS: CD Closure).

Given a timed display set TDS, the concatenation closure of TDS, TDS is defined as follows:*

1. $TDS^0 = \{(< type, src"x", attTestValue = Y >, t, t, \{Y\}) \mid (¡type, src"x", attTestValue = Y >, t_1, t_2, \{Y\}) \in TDS \text{ and } t_1 \leq t \leq t_2\}$
2. $TDS^1 = TDS$
3. $TDS^{n+1} = TDS^n ; TDS$
4. $TDS^* = \cup \{TDS^n \mid 0 \leq n\}$
5. $TDS^\wedge = \{(¡type, src"x", attTestValue = Y ¿, t_1, t_2, \{Y\}) \mid (¡type, src"x", attTestValue = Y >, t_3, t_4, \{Y\}) \in TDS \text{ and } t_1 \geq t_3 \text{ and } t_4 \leq t_2\}$

Then, $(TDS^*)^\wedge$ is the minimal timed display set containing TDS that is both concatenation and downward closed.

Proof: Omitted

We now enhance the semantics of SMIL by using CD closure sets of base sets. Hence, we strengthen definition 5 as follows.

Definition 7 (Enhanced Semantics for SMIL) *Suppose S is a SMIL specification and [[ ]] is a basis mapping for the basic media elements B of S with the formal parameter t. Then we inductively extend [[ ]]+ to S as follows.*

1) [[Null]]+ = Φ.

2) [[S']]+ = ([[S']])^ for all basic media streams S' of S.*

3) [[< seq > S1 S2 < /seq >]]+ = ([[S1]]+ U [[S2]]+(end([[S1]]+)/t))^*

4) [[<par> S1 S2</par>]]+ = [[S1]]+ U [[S2]]+.

5) [[<switch> S1 S2 </switch>]]+ = [[S1]]+ if S1 satisfies the attribute of the switch. = [[S2]]+ otherwise if S2 satisfies the attribute of the switch. = Φ otherwise.

We now say that the enhanced mapping [[ ]]+ is a semantic mapping parameterized by `t`. Now we show how this semantics preserves the display normal form. Notice that the difficulty of the semantics given in definition 5 was with respect to piecing together successive segments of the same stream. By taking concatenations, this problem was solved in definition 5. Downward closures were taken to permit taking all subintervals of permitted streams.

Lemma 3 (Equivalence of Display Normal Form).

The two specifications S1 and S2 have the same semantics.

Informal Proof First observe that if S1 is the specification given on the left and S2 is the specification given on the right, then tdi $\in$ [[S1]]+ iff tdi^n

$\in$ [[S2]]+. The reason being that S2 executes S1 arbitrarily many times. But, [[S2]]+ is concatenation and downward closed. Therefore, $\text{tdi}^n \in$ [[S2]]+ iff tdi $\in$ [[S2]]+. The reader will now see that downward closure was required in order to obtain tdi $\in$ [[S2]]+ from $\text{tdi}^n \in$ [[S2]]+.

7.3 Dynamic Runtime Activity

As explained, any given SMIL specification S for surveillance is statically translated into its MLS normal form mlsNF(S). Then, when the runtime provides D, mlsNF(S) is translated into its display normal form, say DNF(mlsNF(S),D). Then the runtime takes each the set of streams within the switch that has duration of D, evaluates the switch, and depending upon the mode encrypts and transmits either the streams corresponding to normal operating mode or those that correspond to the emergency operating mode. The SMIL fragment below shows the display normal form for the *Secret View*

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"> <head>
<customAttributesMODE>
      <customTestMode="Normal" title="Normal Mode"
        defaultState="true" override="hidden"
        uid="ControllerChoice" />
       <customTestMode id="Emergency" title="Emergency Mode"
        defaultState="false" override="hidden"
        uid="ControllerChoice" />
</customAttributesMODE>
<customAttributesSecClass>
      <customTestsecClass id="TS" title="Top-Secret"
        defaultState="true" override="hidden"/>
<customTestsecClass id="S" title="Secret"
        defaultState="true" override="hidden"/>
<customTestsecClass id="UC"  title="Unlassfied"
        defaultState="trye" override="hidden"/>
</customAttributesSecClass> <body>
    <switch>
<ref src="ModeNClassS.smil" customTestMode ="Normal"
customTestsecClass ="S" /> <ref src="ModeEClassS.smil"
customTestMode ="Emergency" customTestsecClass ="S" />

<ref src="ModeEClassUC.smil" customTestMode ="Emergency"
customTestsecClass ="UC" />
    </switch>
  </body>
</smil>
```

Similarly views for all classification in both the Normal and the Emergency modes can be created. The mode evaluation procedures for setting of the *mode value* associated with a `customTestMODE` is as follows:

1. The initial setting is taken from the value of the `defaultState` attribute, if present. If no default state is explicitly defined, a value of false is used.
2. The URI (Controller Choice) defined by the uid attribute is checked to see if a persistent value has been defined for the custom test attribute with the associated id (Normal, Emergency). If such a value is present, it is used instead of the default state defined in the document (if any). Otherwise, the existing initial state is maintained.
3. As with predefined system test attributes, this evaluation will occur in an implementation-defined manner. The value will be (re) evaluated dynamically.

7.4 Quality of Service(QoS) and Encryption Issues

The Service Level Agreement(SLA) determines the specifications and restrictions that have to be communicated between the client and the server in order to maintain good quality [WS96]. The requirements of the processors and memory (primary and secondary), and other technicalities such as tolerable delay, loss, pixels have to be negotiated prior or sometimes during the transfer process. HQML [GNY+01] proposes an XML based language for the exchange of processor characteristics. The most important characteristic is the amount of buffer, in terms of memory that the recipient device should have in order to maintain continuity. These specifications would be represented within the SMIL document, so that the recipient device will first prepare or disqualify itself for a reception. In the proposed model, the QoS parameters are generally negotiated prior to the display. They could be set as custom defined attributes that have to resolve to true for the display to happen. We can use some of the standard attributes of the switch statement `systemRequired`, `systemScreenDepth`, and `systemScreenSize` to enforce regulation. The SMIL fragment depicted above shows the QoS Negotiation TAGS in accordance with HQML [GNY+01] and [WS96]and the Encryption tags applied to the display normal form of the secret view to achieve fidelity and confidentiality

```
<smil>
 <App name = "Surveillance Facility#3">
     <Configuration id = "Level1Guard">
             <UserLevelQoS> high </UserLevelQoS>
              <UserFocus> memory </UserFocus>
      </Configuration>
      <Configuration id = "Level2Guard">
                  <MemUnit mem = "Mbytes"> 5MB </mem>
                  <UserLevelQoS> Average </UserLevelQoS>
```

```
                <UserFocus> Delay </UserFocus>
                <Delayunit del = "Minutes"> 7 </del>
                <SLAModel> Conform SLA </SLAModel>
     </Configuration>
     <Configuration id = "Level3Guard">
            <UserLevelQoS> high </UserLevelQoS>
            <UserFocus> clarity </UserFocus>
            <Clarityunit clar= "pixels/inch"> 200 </clar>
     </Configuration>
</App> <customAttributes>
 //Mode and Security defined here
<customAttributes>
   </head>

<body> <seq> <switch>
 <par>
 <media src=" ModeNClassTS.smil " customTest3 = "Normal"/>
 <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'>
 <CipherData>
    <CipherValue>123BAVA6</CipherValue>
 </CipherData>
 </EncryptedData>
 </par>
 <par>
 <media src=" ModeNClassS.smil " customTest3="Emergency"/>
 <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'>
 <CipherData>
   <CipherValue>65APR1</CipherValue>
 </CipherData>
 </EncryptedData>
   </par>
//Other SMIL views.
      </switch> </seq> </body>
</smil>
```

Mobile handheld viewing devices [EUMJ] that have embedded SMIL players are the recipients in our architecture. A smartcard, which enforces access control, is embedded into the display device [KW02, KFW03]. Each display device has a unique smartcard depending on the classification of the guard that utilizes it and his/her classification and any other rules set by the controller. A decryption key associated with the privileges of the guard is also embedded in the smartcard. When a display device receives an encrypted SMIL document, the smartcard decrypts the appropriate segment depending on the available key. We encrypt each view in the document as shown the SMIL fragment with a unique Symmetric Key using the standard XML encryption specification. An inbuilt Cryptix

Parser [KW02] that is programmed in firmware (or in software) to handle the decryption process would enable selective decryption of the appropriate view based on the access privileges as defined in the smartcard. With encryption, we guarantee that nobody tampers the stream in transit even if there is mediate stream acquisition.

8 Conclusions

We provided a framework for audio-video surveillance of multi-level secured facilities during normal and pre-envisioned emergencies. We did so by enhancing SMIL specifications with security decorations that satisfy MLS security constraints during normal operations and provide controlled declassification during emergencies while maintaining the integrity and confidentiality. Then we showed how to transform such a SMIL composition to its MLS normal form that preserve runtime semantics intended by SMIL constructs, and how to create SMIL views compliant with MLS requirements. Given the delay characteristics of a runtime, we showed how to transform a SMIL document in MLS normal form so that the operating mode can be switched with a minimal delay while respecting runtime semantics. Our ongoing work extends this basic framework to incorporate richer multimedia semantics and diverse security requirements such as non-alterable media evidence and two way multimedia channels.

References

[Aya01] Jeff Ayars. *Synchronized Multimedia Integration Language*. W3C Recommendation, 2001. http://www.w3.org/TR/2001/REC-smil20-20010807.

[BBC+00] Elisa Bertino, M. Braun, Silvana Castano, Elena Ferrari, and Marco Mesiti. Author-x: A java-based system for XML data protection. In *IFIP Workshop on Database Security*, pages 15–26, 2000.

[BHAE02] Elisa Bertino, Moustafa Hammad, Walid Aref, and Ahmed Elmagarmid. An access control model for video database systems. In *Conferece on Information and Knowledge Management*, 2002.

[Bul98] David Bulterman. Grins: A graphical interface for creating and playing smil documents. In *Proc. of Seventh Int'l World Wide Web Conf. (WWW7)*. Elsevier Science, New York, April 1998.

[DdV03] Ernesto Damiani and Sabrina De Capitani di Vimercati. Securing xml based multimedia content. In *18th IFIP International Information Security Conference*, 2003.

[DdVPS00] Ernesto Damiani, Sabrina De Capitani di Vimercati, Stefano Paraboschi, and Pierangela Samarati. Securing XML documents. *Lecture Notes in Computer Science*, 1777:121–122, 2000.

[DdVPS02] Ernesto Damiani, Sabrina De Capitani di Vimercati, Stefano Paraboschi, and Pierangela Samarati. A fine grained access control system for xml documents. *ACM Transactions on Information and System Security*, 5, 2002.

[EUMJ] E. Ekudden, U.Horn, M.Melander, and J.Olin. On-demand mobile media- a rich service experience for mobile users.

[GNY+01] Xiaohui Gu, Klara Nahrstedt, Wanghong Yuan, Duangdao Wichadakul, and Dongyan Xu. An XML-based quality of service enabling language for the web, 2001.

[JSSS01] Sushil Jajodia, Pierangela Samarati, Maria Luisa Sapino, and V. S. Subrahmanian. Flexible support for multiple access control policies. *ACM Trans. Database Syst.*, 26(2):214–260, 2001.

[KFW03] Naren Kodali, Csilla Farkas, and Duminda Wijesekera. Enforcing integrity in multimedia surveillance. In *IFIP 11.5 Working Conference on Integrity and Internal Control in Information Systems*, 2003.

[KW02] Naren Kodali and Duminda Wijesekera. Regulating access to SMIL formatted pay-per-view movies. In *2002 ACM Workshop on XML Security*, 2002.

[KWJ03] Naren Kodali, Duminda Wijesekera, and J.B.Michael. SPUTERS: a secure traffic surveillance and emergency response architecture. In *submission to the Journal of Intelligent Transportaion Systems*, 2003.

[Low99] Gavin Lowe. Defining information flow, 1999.

[Mul87] Ketan Mulmuley. *Full abstraction and semantic equivalence*. MIT Press, 1987.

[Nok] Mobile Internet Toolkit: Nokia. www.nokia.com.

[Osb] Sylvia Osborn. Mandatory access control and role-based access control revisited. pages 31–40.

[PCV02] Kari Pihkala, Pablo Cesar, and Petri Vuorimaa. Cross platform smil player. In *International Conference on Communications, Internet and Information Technology*, 2002.

[RHO99] L. Rutledge, L. Hardman, and J. Ossenbruggen. The use of smil: Multimedia research currently applied on a global scale, 1999.

[RvOHB99] Lloyd Rutledge, Jacco van Ossenbruggen, Lynda Hardman, and Dick C. A. Bulterman. Anticipating SMIL 2.0: the developing cooperative infrastructure for multimedia on the Web. *Computer Networks (Amsterdam, Netherlands: 1999)*, 31(11–16):1421–1430, 1999.

[San93] Ravi S. Sandhu. Lattice-based access control models. *IEEE Computer*, 26(11):9–19, 1993.

[Sch99] B. K. Schmidt. An architecture for distributed, interactive, multi-stream, multi-participant audio and video. In *Technical Report No CSL-TR-99-781, Stanford Computer Science Department*, 1999.

[SF02] Andrei Stoica and Csilla Farkas. Secure XML views. In *Proc IFIP 11.3 Working Conference on Database Security*, 2002.

[Spy] Spymake. Integrated surveillance tools http://www.spymakeronline.com/.

[VCM] Mobile VCMS. Field data collection system http://www.acrcorp.com.

[VSA] VSAM. Video surveilance and monitoring webpage at http://www-2.cs.cmu.edu/ vsam/.

[WS96] Duminda Wijesekera and Jaideep Srivastava. Quality of service QoS metrics for continuous media. *Multimedia Tools and Applications*, 3(2):127–166, 1996.