Secure Digital Libraries

Noha Ibrahim Mohamed ElSherbiny

Thesis submitted to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Master of Science

In

Computer Science and Applications

Edward Alan Fox, Chair

Mohamed Kholief

Danfeng (Daphne) Yao

22 June 2011

Alexandria, Arab Republic of Egypt

Keywords: digital libraries, security, 5S, 5SL

Secure Digital Libraries

Noha Ibrahim Mohamed ElSherbiny

ABSTRACT

Digital libraries are an integration of complex computer and information systems that could benefit from a formal approach to design. There are various design aspects to consider in a digital library; a crucial aspect is security. Security often is a requirement in digital libraries that should be considered during the design process and not as an add-on feature.

5S provides a DL modeling framework, to define all the aspects of a digital library. It covers the different formats and types of digital objects stored, how they are grouped and organized, what sequence of operations occur in the digital library, how the objects will be represented, and who is part of the digital library community. However, the 5S descriptive language (5SL) previously did not cover the essential security requirements in a digital library.

The goal of this research is to extend the 5S framework to describe the security requirements in digital library. An XML schema was developed to describe the necessary security requirements in a digital library, and some of the essential features of the digital library design. This work explains the key security requirements needed in a digital library from the 5S perspective and how the framework can be extended to include these requirements. The extended 5SL was applied to a case study on the Egyptian University Libraries Consortium.

Acknowledgements

I would like to express my sincere gratitude to my advisor and masters committee chairman Dr. Edward Fox for his continuous guidance and support during this research work. His patience and motivation have inspired me to persevere and explore this research field further. I am truly grateful to him for providing me with the necessary resources to pursue the research.

I would also like to thank my other committee members, Dr. Mohamed Kholief and Dr. Danfeng Yao, for their valuable comments, suggestions, and review of this research.

I would like to thank Dr. Donald Shoemaker, Dr. Yasser Fouad, Seungwon Yang, and Amine Chigani for their time to help me with the requirements gathering of my research.

I thank the Arab Academy for Science and Technology for funding my education throughout my Masters degree and especially Dr. Yasser Hanafy for his constant guidance and interest.

I thank my friends and colleagues at AAST and Virginia Tech for their support and feedback throughout my Masters Degree.

My greatest thanks go out to my family for their support but I would especially like to thank Yaya and Hami for the constant support, unconditional love, and the unimaginable sacrifice they continue to provide.

Table of Contents

List of Figures

List of Tables

Chapter 1. Introduction

1.1 Motivation

Security is an important issue in digital library design. Security weaknesses in digital libraries, coupled with attacks or other types of failures, can lead to confidential information being inappropriately accessed, or loss of integrity of the data stored. These in turn can have a damaging effect on the trust of publishers or other content providers, can cause embarrassment or even economic loss to digital library owners, and can even lead to pain and suffering or other serious problems if urgently needed information is unavailable [41].

There are many security requirements to consider because of the variety of different actors working with a digital library. Each of these actors has different security needs [20]. Thus, a digital library content provider might be concerned with protecting intellectual property rights and the terms of use of content, while a digital library user might be concerned with reliable access to content stored in the digital library. Requirements based on these needs sometimes are in conflict, which can make the security architecture of a digital library even more complex.

The design of the security architecture of a digital library must go beyond simply adding one or a few modules to a previously designed system. This is because there may be security holes in pre-existing modules, and because difficulties can arise when attempting to integrate the modules. The security architecture of a digital library must be designed so that security concerns are handled holistically. A security system designer must view the whole architecture and consider all of the applicable security factors when designing a secure digital library. The nature of a security attack may differ according to the architecture of the digital library; a distributed digital library has more security weaknesses than a centralized digital library.

The 5S framework [27] uses streams, structures, spaces, scenarios, and societies to describe the functionality of a digital library. A DL designer would analyze the digital library using 5S and then describe the digital library using the 5SL descriptive language, 5SL [26]. The 5SL constructs could be fed to a DL generator, which would generate the specified digital library. 5SL mostly uses XML to define each of the 5Ss; XML is used because it is a platform independent way of defining the constructs.

The 5S language did not previously provide constructs for defining the security aspects of a digital library.

1.2 Research Question

Over the years researchers have proposed different digital library models. Researchers at Cornell University developed a digital library architecture called Dienst [31], which provides a protocol and implementation for internet access to a decentralized, distributed collection. In Europe researchers have developed the DELOS reference model that aims to set a foundation on which to build digital libraries. Another approach is the 5S model, which is used herein to describe design requirements of a digital library. All these models have limited to no consideration of the security requirements that are necessary to consider when developing a digital library. As mentioned before it’s not enough to add security modules to a previously designed system, since many security requirements may necessitate changes to other modules, and call for additional changes related to system integration. Therefore, we should seek formalization for a secure digital library. The following research points are covered in this thesis:

- **What are the possible security vulnerabilities of a digital library?**
- **What are the necessary security requirements of a digital library?**
- **Can these security requirements be integrated into, and described using, the 5S framework?**

1.3 Hypothesis

The main hypothesis of this thesis is that the 5SL can be extended to describe the security requirements of a digital library.

1.4 Research Methodology

First a literature review of the field of security issues in digital libraries was performed; the results of this review highlighted the recent developments in digital library and information system security and the technologies developed for use. Having gained a broad appreciation of the field, a detailed study of 5S and 5SL was performed to understand how to incorporate security into the framework. Requirements gathering and analysis were performed to collect the essential security requirements of a digital library. Experts in the field of software engineering, digital library design, and people working on current digital library projects facing security problems, were consulted. An expert in the fields of sociology was consulted to gain a perspective on the nature of privacy and security related to user information stored in digital libraries. Finally, the requirements were mapped to each of the 5S sub-models and the 5SL was extended to include the security requirements defined.

1.5 Outline of the Thesis

This document is organized as follows. Chapter 1 is an introduction to the problem, motivation, hypothesis, claims, evidence, and the research methodology used. Chapter 2 describes digital libraries, the 5S model, and security issues related to digital libraries. Chapter 3 introduces the security issues of the 5SL and proposes an extension to include security requirements. Chapter 4 is a summary of the work and future work, which is followed by the references and Appendices: Societies Schema, and Structures Schema.

Chapter 2. Survey

In this chapter we present digital libraries and the 5S framework, which is the basis of the work in Chapter 3 that is followed by an extensive discussion of the security issues that affect a digital library; the technologies described will be used in the extension of the 5S framework. This chapter is organized as follows. Section 2.1 is an overview of digital libraries and their applications. Section 2.2 is a description of the DELOS reference model. Section 2.3 describes the 5S framework and Section 2.4 is a discussion of the security issues that affect a digital library.

2.1 Digital Libraries

The developments in the Internet, the World Wide Web, and other information and communication technologies has changed the way in which we generate, distribute, access, and use information. There's a shift from using printed information resources to digital resources. Digital resources are now generated, disseminated, and stored electronically making it easier for them to be accessible to anyone from anywhere. This advancement has led to the development of digital libraries, which allow remote access to resources from anywhere in the world.

A digital library is a system "that carries out interactions among information and knowledge producers, librarians, and information and knowledge seekers" [14]. A DL aims to provide a means to collect, store, and retrieve information electronically.

A DL includes a repository for digital content where this content can be in many different formats. Digital content can be audio, video, images, and various formats of text, as well as metadata about digital and non-digital resources. A DL employs technologies that assist in the management of the DL, information storage and retrieval, interface design, networking, etc. The Digital Library provides services to its community; a community can be viewed as the users and stakeholders of the DL. This community has certain social, legal, cultural, and political issues that affect the digital library.

There are many advantages to the development of a digital library [20],

- **Information is available to the user from anywhere**
  As mentioned before, DLs are available to any user from anywhere in the world, making information more accessible. This is not only beneficial to the user of the DL but also to publishers who wish to market and sell their content.
- **Better information retrieval and manipulation**
  DLs provide sophisticated methods for information retrieval, which enable improved information access.
- **Facilities for information sharing**
  Not all the digital content in DLs are accessible with a fee. Many digital libraries such as NDLTD [8] provide open access to resources.
- **Up-to-date access to information**
  DL users get up-to-date information. The time lag between the physical creation of the resource and its access in a traditional library is significant. However, in a DL this time lag is reduced greatly, making it easier for users to obtain the latest information.

2.1.1 Digital Library Applications

Over the past decade there have been various contributions to digital library research, which resulted in the development of many digital libraries. Some of those digital libraries are mentioned below.

Some digital libraries were developed to support institutional publications such as the ACM [1] digital library, which offers access to the ACM journals, magazines, and conference proceedings.

Other digital libraries were developments of national libraries such as the Library of Congress's THOMAS [12], established in January of 1995 to provide comprehensive information on federal legislation, and the Digital Library of Canada [3] which provides access to information on music, history, and literature.

In the USA, there has been development of different digital libraries at universities such as The Stanford Encyclopedia of Philosophy [11], which offers a DL with references in the field of Philosophy, which is maintained by a group of experts in the field. The Digital Morphology [10] library is a DL that provides high resolution X-Ray CT descriptions of biological specimens; it was developed at the University of Texas.

Some digital libraries have been developed as part of a research project funded by an institution or a foundation. Some examples of these are shown below:

- The National Gallery of Spoken Word [7] is an online searchable DL of spoken word collections of the 20th Century.
- The Networked Digital Library for Theses and Dissertations [8] provides access to student's ETDs (electronic theses and dissertations) from various institutions and universities from all over the world.
- The Crisis, Tragedy, and Recovery Network (CTRnet) [2] is a DL that provides a range of services related to different kinds of tragic events.
- The European Libraries and Electronic Resources in Mathematical Sciences (EULER) [6] provides a digital library of mathematical publications.

2.1.2 Digital Library Design

The varying nature of the information content stored in a digital library, along with the various users and their needs, lead to challenges for DL designers. They must integrate the different resources and the complex services available to the different users of the digital library. Since DLs can sometimes communicate with various systems, issues of interoperability and integration arise.

One of the major challenges of DL design is caused by the differences in the computer systems, formats, information organization, file structures, and retrieval features of the different information systems or collections that are accessible through a digital library.

Therefore, various models are available to assist DL designers in creating a digital library that can deal with the complex issues of integration and make DL design much easier.

According to [15] the following are the eight general design principles of a digital library:

- **The technical framework exists within a legal and social framework.**
  Digital libraries have changed the way people create, disseminate, and use information. In order to ease the design of digital libraries, changes must be made in the legal and social framework of society, to protect the stakeholders of the information.
- **Understanding of the digital library needs standard terminology.**
  Standard terminology must be developed and used among all involved in the design of the digital library. This is important to avoid confusion between any of the actors involved in the design, since most of them would come from different disciplines, facilitating collaboration between them.
- **The underlying architecture should be separate from the content stored in the library.**
  An underlying architecture that is applicable to all the different types of digital material should be present, to help in the identification of the materials. A special effort should be made to handle each content type such as music, video, text, software, and so on, and their different formats.
- **Names and identifiers are the basic building blocks for the digital library.**
  Each object in the digital library should have its own unique identifier or name that should be valid for a long period of time.
- **Digital library objects are more than collections of bits.**
  A digital object may have more than one type of content; e.g., it could be made up of audio and text. Each object also must have metadata that characterizes the object, e.g., describing its creator, date of creation, title, intellectual property rights, and access rights.

- **A digital library object should not be tied to a particular technology.**
  One of the benefits of having a digital library is that the information can be viewed by anyone anywhere. The digital objects should be as hardware and software independent as possible, making it more flexible for users to access them.
- **Repositories must look after the information they hold.**
  Repositories store information on the digital content as well as the metadata of the content that describes all the access rights and access methods. It's necessary that a repository provides essential security on the digital objects and only allows operations that comply with the access rights of the object. While many DLs have a repository as a single entity, in the case of the bucket approach [35], the repository may be distributed, e.g., implemented at the level of buckets.
- **Users want intellectual works not digital objects.**
  DL architectures must store the objects in a way that makes it easy for them to be manipulated and presented to the user in any abstract form. When DLs manage complex objects, for example musical performances where there is a composition, sheet music, a performance by an orchestra, a recording of the performance, different masters from the recording based on different technologies, and various releases on DVDs and CDs, then FRBR [17] can help with related conceptualization and management.

In addition to discussions by Arms of general principles, models with more specific definitions have been proposed [18, 27]. These models provide DL designers a way to define digital libraries. Two of these models are discussed in Sections 2.2 and 2.3.

2.2 DELOS Reference Model

The DELOS reference model was developed by DELOS, a network of excellence on Digital Libraries. The model was designed to “set the ground rules for the field and lead to the development of reference documents that will capture the full spectrum of concepts that play a role in Digital Libraries” [18].

According to the model, the creation of a digital library can be viewed as a process that produces 3 systems that make up a 3-tier framework. Digital Library Management System, Digital Library System, and finally Digital Library are the 3 tiers. The diagram below shows the 3-tier framework.

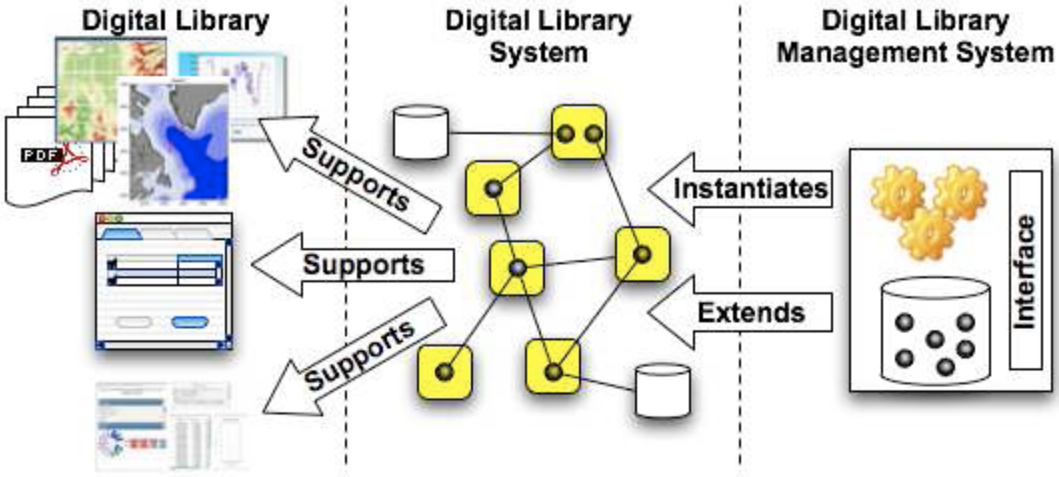


Figure 1. DL, DLS, and DLMS framework [18] **(Used under fair use guidelines, 2011)**

- **Digital Library (DL)**
  Is an organization that collects, manages, and preserves digital content. It offers specialized functionality on that content to its users, according to certain predefined policies.
- **Digital Library System (DLS)**
  Is a software system that provides the functionality of the Digital Library, based on a defined architecture. The users of a DL interact with a Digital Library through the Digital Library System [18].
- **Digital Library Management System (DLMS)**
  Is a software system that provides an interface for user interaction with the DL and an infrastructure to produce and administer fundamental Digital Library System functionalities and to provide a means to integrate additional software to add more functionality to the DL.

Having understood the function of each of the 3-tiers, the model then goes on to explain the digital library universe. There are 6 main concepts in a digital library universe: content, user, functionality, architecture, quality, and policy. Below is a diagram from [18] showing the different concepts in the digital library universe.

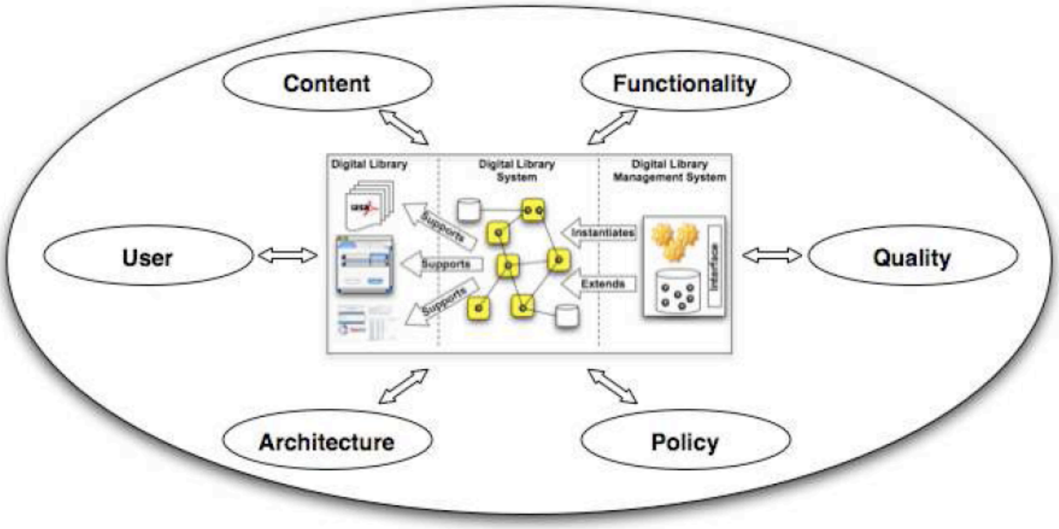


Figure 2. The digital library universe: main concepts [18] **(Used under fair use guidelines, 2011)**

- **Content**
  Content represents the information available for the users of the digital library.
- **User**
  User represents the actors that interact with the system.
- **Functionality**
  Functionality represents the facilities supported by the system and is available to the users.
- **Architecture**
  Architecture represents the hardware and software that make up the system.
- **Quality**
  Quality represents the parameters that can be used to describe and assess the content and actions performed by a Digital Library.

- **Policy**
  Policy represents all the rules and conditions that define the usage of the digital library.

2.3 5S Model

The DELOS Reference Model [18] looked at the design of a digital library from 6 main concepts in a digital library universe. However other frameworks view digital library design differently. The 5S framework views the specification of a digital library as the definition of constructs [26] building on each of the 5Ss. Below is a detailed view of each 'S'.

2.3.1 Streams

Streams define the different types of content stored in the digital library and their formats. It can represent static content or dynamic content. Static material can be textual material while dynamic content could be coordinates from a GPS of a moving object. Dynamic streams are useful to represent the communication that occurs in a digital library, for example a number of frames that are sent to a user that make up a virtual reality scenario. Static content, on the other hand, could be a stream of characters that make up a text document.

2.3.2 Structures

Structures cover the organization of a digital object, as a whole, or parts of it. For example a book can be structured into chapters, sections, subsections, and paragraphs. In information retrieval systems indexing helps to support future requests and helps categorize these requests to "generate an organizational structure for the document space" [27]. A lot of the material being added to digital libraries is "semistructured", meaning that there is a structure to the data but not a rigid or complete structure as used by database management systems.

Structures are overlaid on other constructs in the 5S framework, especially on Streams. Thus, documents are structured streams, while protocols involve scenarios applied to structured communication streams.

2.3.3 Spaces

A space can be defined as “a set of objects together with operations on those objects that obey certain constraints” [27]. The operations on the objects along with the constraints on those objects, distinguishes spaces from streams and structures. Spaces cover distributed aspects, as well as representations related to 1D, 2D, 3D, and higher dimensional spaces. These include feature, measure, metric, probability, vector, and topological spaces – used throughout computer and human systems.

2.3.4 Scenarios

Scenarios cover functions, operations, requirements, services, and tasks. An important scenario is one describing all the possible ways to use a system to accomplish a user function. Scenarios describe what happens to the streams, in the spaces, through the structures, and who in the society is responsible for these operations.

2.3.5 Societies

A society can be defined as “a set of entities and the relationships between them”. The entities refer to humans, hardware, and software components. For example, human societies in a digital library could be the publishers, editors, administrators, and staff who maintain or use the digital library. Societies cover software actors, agents, components, modules, etc.; this also encompasses related architectural issues. The 5S framework supports Societies and their needs, and the different roles and relationships that they may have, covering all aspects related to information users, privacy, intellectual property rights, and associated policies.

The security issues relating to the 5S model are described in more detail in Chapter 4.

2.4 5SL

The 5S language (5SL) discussed in [26] is “a declarative language for conceptual modeling and generation of digital library applications”. 5SL provides a precise DL tool for specification, and facilitates prototyping. In 5SL, the specification of a DL covers five dimensions: Stream model, Structure model, Spatial model, Scenario model, and Societal model. The 5SL model primitives are defined as XML elements, because they can enclose other sublanguages that help to define DL concepts [26]. 5SL uses XML, because of the various supporting software tools that can be used to facilitate the construction of DL generators.

Table 1. 5SL Overview (adopted from [26] **(Used under fair use guidelines, 2011)**

Models	Primitives	Formalizations	Objectives
Streams Model	Text; video; audio; picture; software program	Sequences; types	Describes properties of the DL content such as encoding and language for textual material or particular forms of multimedia data
Structures Model	Collection; catalog; hypertext; document; metadata; organization tools	Graphs; nodes; links; labels;	Specifies organizational aspects of the DL content
Spaces Model	User interface; index; retrieval model	Sets; operations; spaces; vector space; measure space; probability space	Defines logical and presentational views of several DL components
Scenarios Model	Service; event; condition; action	Sequence diagrams; collaboration diagrams	Details the behavior of DL services
Societies Model	Community; managers; actors; classes; relationships; attributes; operations	Object-oriented modeling constructs; design patterns	Defines managers, responsible for running DL services; actors, that use those services; and relationships among them

Table 1 summarizes all the 5SL models, their primitives, formalizations, and objectives. Each model is explained below.

2.4.1 Streams model

Stream model defines the different types of multimedia information the DL supports. MIME types form the basis for encoding streams because they offer reusability and standardization.

2.4.2 Structures model

Structural model determines how the information in a digital library is structured and organized. In this model the internal structure of the digital objects, the metadata standards, and the properties of the collections are described. Knowledge organization tools are also defined; they help in the structuring of the digital library. XML Schema and/or RDF Schema are the main tools used for describing structures. XML Schema is a standard promoted by the W3C, and the W3C recommendation for RDF Schema was released in February 2004.

2.4.3 Spaces model

Spatial model describes the operations of DL components and the various logical and presentational properties of the components. This model gives details about the indexing methods for the collections and catalogs and the appearance of the user interface. Properties for the indexing include defining types of stemming and/or set of stopwords. In the spatial model the User Interface Markup Language (UIML) and MathML are used to characterize some of the features of spaces.

2.4.4 Scenarios model

Scenarios model describes the behavior of the DL services. It is a sequence of actions that the different actors and managers perform. UML sequence diagrams can be used to represent these actions. The 5SL can translate the diagrams to generate a serialized set of XML elements defining a particular scenario.

2.4.5 Societies model

The aim of the societies model is to define the different community members such as actors and managers of services that operate together to carry out the DL behavior. This model is based on the object-oriented paradigm and uses attributes, methods, and relationships to describe the community and its members and their behavior. There are mainly 2 types of communities: managers and actors. Managers are responsible for the administration of the DL services while actors use those services to meet their information needs.

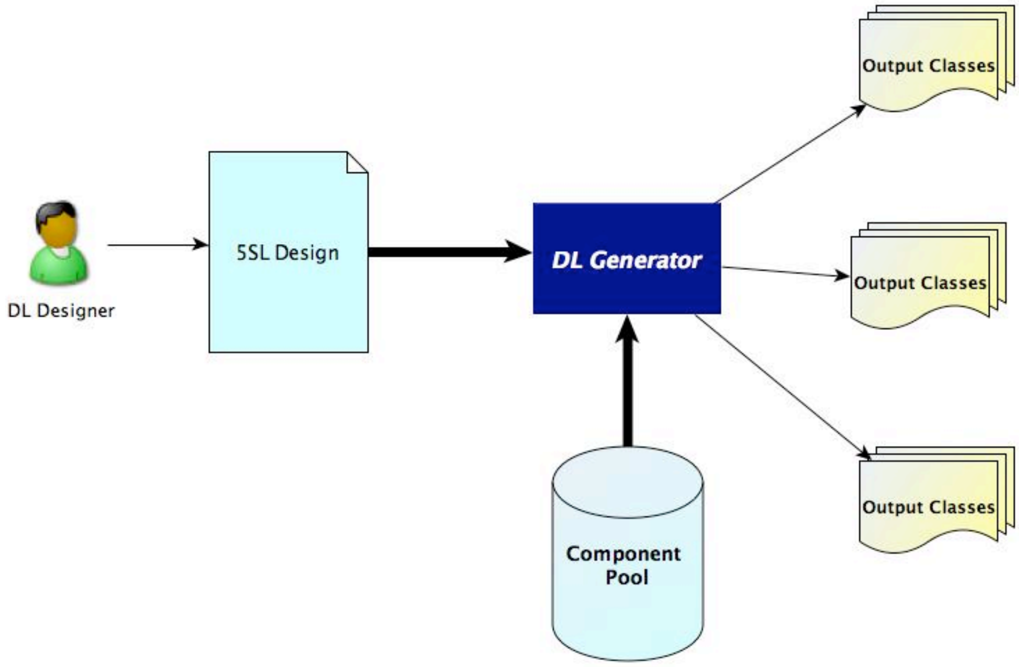


Figure 3. Generation of DL using 5SL

Figure 3 shows how a digital library could be generated using the 5SL. The DL designer would formalize a description of the digital library using the language concepts. The declarative specification in 5SL is provided to a DL generator, and the output produced will provide a customized DL suitable for a certain platform and meeting certain requirements. The output is dependant on the nature of the generator; for example in [26] the MARIAN DL generator was based on a DOM XML parser so it automatically generated 4 kinds of output: a set of class managers, indexing classes, the analyzer, and user interfaces. Another example is dlGen [28], a DL generator that focuses on DSpace implementations. DSpace is used by academic institutions to manage the problem of large amounts of scholarly work created by faculty and staff that was not accessible and suffered from maintenance issues.

2.5 Security issues with digital libraries

Having explained the 2 digital library design models, the DELOS Reference Model and the 5S model, we view security from the perspective of both models, to be as comprehensive and theory neutral as possible. The DELOS Reference Model has been developed top down while the 5S model has been developed bottom up. To provide broad coverage but incomplete formal details, we extend both by formally considering security.

According to the DELOS Reference Model [18] there are 6 main concepts in a digital library universe: content, user, functionality, architecture, quality, and policy. Each of these concepts has security issues that affect it. These issues are explored below.

2.5.1 Content

The content of a digital library includes the information objects that a digital library provides to the users. Some of the security issues involved are integrity and access control. Integrity requires that each object/resource has not been altered or changed by an unauthorized person. Access control encompasses two security requirements. The first is authentication where the user must log into the system, while the second is confidentiality, which means that the content of an object is inaccessible by a person unless they have authorization. Not all digital libraries are free; often content is provided to digital library users for a certain fee, whereupon access control is needed to protect the content. Further, some content is inappropriate for some users, or targeted to particular user groups; there are a whole host of such other reasons for access control.

Logical attacks such as hacking and message tampering can affect the integrity and confidentiality of the content. Improved information access in digital libraries has raised many issues that affect the management of digital libraries. Content Management, or more specifically Digital Rights Management, refers to the protection of content from the different logical security attacks and issues relating to intellectual property rights and authenticity.

2.5.1.1. Digital Rights Management

DRM provides content protection by encrypting the content and associating it with a digital license [41]. The license identifies the user allowed to view the content, lists the content of the product, and states the rights the user has to the resource in a computer readable format using a digital rights expression language (DREL) or extensible Rights Markup Language (XrML) that also describes constraints and conditions.

There are 7 technologies used to provide DRM [23]. Table 2 summarizes the DRM components and supporting technology.

Table 2. DRM Components and Protection Technologies, adapted from [23]

(Used under fair use guidelines, 2011)

Component	Protection Technology
Access and usage control	Encryption (e.g., symmetric, asymmetric), passwords
Protection of authenticity and integrity	Watermarks, digital signatures, digital fingerprints
Identification by metadata	Allows description of an object in suitable categories, covering the digital content, rights owner, and conditions.
Specific hardware and software	Includes all hardware and software used by the end-device through which the digital content is being played, viewed, or printed.
Copy detection systems	Search engines, which search the network for illegal copies and use watermarking.
Payment systems	Can be seen as a certain type of protection technology as it requires user registration, or credit card authentication, which also require a trust relationship between the content provider and the customer.
Integrated e-commerce systems	DRMS must include systems, which support contract negotiation, accounting information, and usage rules.

Each of these components involves mechanisms used to provide DRM:

- **Encryption:** Encryption techniques such as symmetric and asymmetric ciphers can be used to provide access control; public-key encryption is used in payment systems that control how and by whom the content is used.

  Symmetric ciphers using DES, 3DES, AES, and RC4 algorithms require the use of a shared secret key to encrypt data before it is sent. At the receiver's end the cipher text is decrypted using the same secret key. Symmetric ciphers depend on both the sender and receiver knowing the shared key.

  Asymmetric ciphers use a pair of keys, public and private, for each of the sender and the receiver. The public keys of both the sender and the receiver are known but the private key is kept secret. If encryption is performed using the public key then only the private key can be used for decryption and vice versa.
- **Passwords:** Stored strings must be matched by users desiring access.
- **Watermarking:** Characters or images are added to reflect ownership. Steganography is used to conceal data inside audio, video, or images [29]. Different watermarking techniques have different aims; some watermarks might be visible while others invisible. Some watermarks are reversible [32]; it depends on the desired use of the watermark and what is being protected.
- **Digital signature:** Asymmetric encryption can be used. Likewise, hash algorithms such as MD5 and SHA can be used to create a signature [39].
- **Digital fingerprint:** Digital fingerprints are a more powerful technique involving digital signatures and watermarking. The creator of the content creates a unique copy of the content marked for each user; the marks are user-specific hence called fingerprints. Should a user illegally distribute the content, the creator can use search robots to find those copies [38].
- **Copy detection systems:** Search engines also can help locate such copied objects. Copy-detecting browsers can protect digital content too.
- **Payment systems:** Users must divulge personal information to pay for content. Installing payment systems can help protect digital content.

There is no standard mechanism for providing DRM, mainly due to the lack of regulations [20], however there are various systems and protocols introduced to provide content management and support fair usage policies.

2.5.2 Performance

There is a tradeoff between security and performance. Nadeem and Javed used a Pentium-4, 2.4 GHz machine running the Microsoft Windows XP operating system, to encrypt 20527 bytes to 2323398 bytes of data using DES, 3DES, and AES. For 20527 bytes of data it took 2 seconds to encrypt using the DES algorithm and 4 seconds to encrypt using the AES algorithm [33]. It can be seen that the more complex the encryption algorithm the longer it takes to encrypt the data. In another study, encrypting data with the RSA algorithm using a key size of 1024 took 0.08 milliseconds/operation on an Intel Core 2 1.83 GHz processor under Windows Vista in 32-bit mode, while using a key size of 2048 took 0.16 milliseconds/operation [21].

2.5.3 User

The User in a digital library refers to “the various actors (whether human or machine) entitled to interact with digital libraries” [18]. Digital libraries connect the different actors with the information they have and allow the users to consume old or generate new information. Security issues relating to the users of a digital library intersect with content issues discussed above. A main logical security issue relating to users and content is access control. Different access control requirements arise for distributed systems [40] to ensure both confidentiality and authentication:

- **Access control must be applied and enforced at a distributed platform level, so should be scalable and available at various levels of granularity.**
- **Access control models should allow a varied definition of access rights depending on different information and must be dynamic where changes to policies are easily made and easy to manage.**
- **“Access control models must allow high-level specification of access rights.” [40]**

Digital library users may need to be authenticated before they can access content in a digital library. Global/universal identification may not suffice. A service provider that provides content based on a non-identity based criteria like age will not benefit from global identification because there is no way to verify the authenticated user’s personal information. Usernames and passwords are not efficient ways to provide authentication.

One of the most widely used authentication protocols is Kerberos. It [36] is a client–server model, which secures communication with servers on a local network. Developed at MIT in the 1980s to provide security across a large campus network, it is based on the Needham-Schroeder protocol and has now been standardized and included in many operating systems such as UNIX, Linux, Windows 2000, NT and XP.

Kerberos is used as an authentication protocol in cases where attackers monitor network traffic to intercept passwords. It secures communication, provides single sign on and mutual authentication, and does not send a user’s password in the clear on an insecure network.

An alternative solution suitable for digital libraries [42], is to represent information about an individual using credentials. Credentials are “abstract objects which contain statements expressing knowledge or information from a definite context.” Credentials do not specify direct information about a client and their attributes; they describe the local environment and context in which the requests originate [19].

Digital credentials can be used as a means of authentication in providing DL access control [42]. Two agents can be used to assist in the management: a personal security assistant and a server security assistant, to manage digital credentials using a client/server model. The server must notify the client of the credentials required for the current request. The client must have some trust of the server to give its credentials, which raises privacy issues.

The personal security assistant is used to obtain credentials on behalf of the client, store the credentials, parse and interpret the required credentials, and manage the acceptance policies [42]. A server security assistant is available to specify the credential acceptance policies and their usage.

There is a tradeoff between flexibility and security that must be considered when choosing an access control model, as is discussed below.

2.5.3.1. Access Matrix Model

This conceptual model specifies the rights that each subject possesses for each object [40]. Actions on objects are allowed or denied based on the access rights specified. There are 2 implementations of the AMM:

- **An Access Control List provides a direct mapping of each object the subjects are allowed to access, and their usage rights (owner, read, or write).**
- **A Capability List defines the objects each subject is allowed to access and the usage rights.**

Access control lists and capability lists are not suitable for distributed systems. Their limitations lead to multiple problems [34]. ACL provides limited expressibility of policies. Any change in the policies will propagate in the system/application. Authentication in a system that uses ACL solely is a problem because using username & password in a distributed system is not practical. In a distributed system, administration of the system should be decentralized by delegation to reduce the overhead. The owner of the object specifies a policy in ACL. If an overall policy is specified by an entity higher than the object owner, then conflicts may occur in the access rights. The number of administrative entities in a distributed system can be very large. Not all the administrators may have trust amongst themselves, resulting in incorrectly defined policies. For example, admin A may trust B but not C, however B may trust C. If A were to define a policy for B then it would be implicitly applicable to C, causing problems.

2.5.3.2. Role-based Access Control

Role-based access control involves policies that regulate information access based on the activities the users perform. Such policies require the definition of roles in the system: "a set of actions and responsibilities associated with a particular working activity" [37]. Permissions are assigned to roles instead of individual users. Specifying user authorization involves 2 steps: first assigning the user to a role, second defining the access control that the role has over certain objects.

RBAC is easier to manage and is more extensible than ACL. However RBAC doesn't flexibly handle constraints, where a user with a specific role may need specific permission on an object. An example of an RBAC architecture addressing key limitations is OASIS [16], for use in distributed systems. Role management in OASIS is decentralized and service specific. OASIS is integrated with an event-based middleware that notifies applications of any environmental changes. Roles are parameterized by applications and services to define their client roles, and to enforce policies for role activation and service invocation within each session. Role membership certificates (RMC) are returned to each user on successful login, to be used as a credential to activate other roles [16].

RBAC is suitable for use with digital libraries because it supports decentralized architectures and varying roles, however RBAC doesn't allow for the definition of different roles in a collaborative group.

2.5.3.3. Task Based Access Control

The task based access control model extends subject/object access control by allowing the definition of domains by task-based contextual information [40]. Steps required to perform the task are used to define access control; the steps are associated with a protection state containing a set of permissions for each state, which change according to the task. TBAC uses dynamic management of permissions.

TBAC systems are limited to defining contexts in relation to activities, tasks, or workflow progress. Since it is implemented by recording usage and validity of permissions, therefore, TBAC requires a central access control module to manage permissions activation and deactivation in a just-in-time fashion.

2.5.3.4. Team Based Access Control

RBAC doesn't address cases where group members of different roles want to collaborate in a single group. The TMAC model defines collaboration by user context and object context. "User context provides a way of identifying specific users playing a role on a team at any given moment" [40] while object context defines the objects required.

TMAC offers the advantages of RBAC along with ability to specify fine-grained control on users and on object instances. A scalable access control data structure can be used with large collections, applying concepts of team based access control, focusing mainly on the access control data structure, and employing an access control framework called Document Access Control Method (DACM) with a Document Storage System (DocSS) [25]. DACM allows the decentralized administration of privileges, the definition of different rule sets to control a single collection, and different delegation patterns as models.

Current object access control policies use an array of rules to record the privileges each subject is allowed to each object. This is impractical to manage in the large data collections found in digital libraries. DACM solves this problem by finding symmetries in a permission function to allow a brief expression without losing important distinctions.

2.5.3.5. Content Based access control

Another approach to access control models involves defining models according to content. This approach is applicable in digital libraries and distributed systems [14], where the access rights to the user are dynamic and may change with each login. Content based access control policies are very well suited for digital libraries and distributed systems. Recent research has proposed different models; most use digital credentials for authentication, but vary in the definition/storage of the policy.

An important content based access control model [22], introduces a content based access control system, Digital Library Authorization System, that utilizes the Digital Library Authorization Model (DLAM). Subject, object, and privilege sets can't be used to define policies in digital libraries mainly because DLs are dynamic with large collections of data and subjects. DLAM defines access control policies based on subject qualifications and characteristics. DLAM provides a means to specify the qualifications and characteristics of subjects. It uses content dependant and independent access control and allows the definition of policies with varied granularity.

2.5.4 Functionality

The concept of functionality encompasses the services that a digital library offers to its users. The minimum functions of a digital library include adding new objects to the library or searching and browsing the library and other functions relating to DL management. A security attack that can affect the functionality of the digital library is a Denial of Service attack, which can affect the performance of the system and prevent users from accessing the system.

2.5.5 Architecture

Digital libraries are complex forms of information systems, interoperable across different libraries and so require an architectural framework mapping content and functionality onto software and hardware components [18]. There are various models for architecture, e.g., client-server, peer-to-peer, and distributed. All these require the protection of the communication channels between 2 parties, where sensitive data might be transferred [30]. Securing the connections involves different layers - Internet, transport, or application layer - depending on the architecture of the system.

The distributed model is scalable and flexible. It is useful when building a digital library with changing content from different sources and offers potential for increased reliability. The security requirements for a distributed digital library are challenging, since the content and operations are decentralized. Fault tolerance and error recovery are issues that affect a distributed system. Replication is used to increase the availability of the system. While this approach solves problems with denial of service attacks, it complicates the protection of the content because one or more replicas of the content exist.

The client-server model doesn’t have the same security problems as a general distributed model; however, it presents a major security weakness, the server being a single point of failure. Attacks can be concentrated on one server rather than on the multiple replicas of a distributed model.

In Section 2.2 the 3-tier architecture of the DELOS reference model was explained. There are security issues relating to each of the 3 tiers; on the Digital Library (DL) tier issues of intellectual property rights, digital rights management, data confidentiality, and data integrity affect the digital content. On the Digital Library Systems (DLS) tier availability and access control are security issues because the DLS is the interface between the DL and the users. The user interface of the digital library, the DLMS, is the third tier; I suggest security issues such as access control and privacy of the data should be considered.

2.5.6 Quality

The content and behavior of a digital library is characterized and evaluated by quality parameters. Quality is a concept not only used to classify functionality and content, but also used with objects and services. Some of the parameters are automatically measured and are objective while others are considered subjective; some are measured through user evaluations.

2.5.7 Policy

Policy is the concept that represents the different regulations and conditions that govern the interaction between the digital library and users. Policy supports both extrinsic and intrinsic situations [18], and their definition and modification.

Examples of security issues relating to policies include providing digital rights management, privacy, and confidentiality of the content and users, defining user behavior, and collection delivery.

Digital libraries should be secure. This is an important quality that affects all aspects, as has been shown above using the DL characterization of the DELOS Reference Model [18]. Many of the security issues discussed affect more than 1 concept, suggesting the overlapping of issues between digital library concepts.

Having discussed the 5S framework, 5SL, and the basic security features required in a digital library, we further explore the security features in each of the 5S constructs.

Chapter 3. 5SL extension

To extend the 5SL to include security we must first view the different security requirements needed for a digital library from the perspective of the 5Ss.

3.1 Security issues concerning the 5S

Having viewed the security requirements we now look at how each of them maps to the 5Ss.

3.1.1 Streams

The streams model describes the different formats of the digital content that's to be stored in the digital library. As discussed in Section 2.1.2, publishers or suppliers of the digital content must have certain access rights defined for each digital object. A major security threat is the disclosure of content, meaning that a person has violated an access right defined for the digital object. To prevent this attack we need to provide data confidentiality where data is protected from disclosure [39]. Encryption of the data is a mechanism used to provide data confidentiality. As discussed in Section 2.4.1.1 there are various encryption methods available for use.

Another security vulnerability in the streams model is the modification of data; again digital content access rights might not allow a certain user to modify the contents of a digital object. Any change in the content would be a violation of the access rights defined. Therefore to detect modification made to the digital object we must have data integrity which is the guarantee that the data received is exactly as sent by an authenticated entity (i.e., contains no modification, insertion, deletion, or replay) [39]. There are various methods to provide data integrity, some of which are hash functions such as SHA-1 and MD5, or message authentication codes (MAC) such as DAA and keyed Hash MAC (HMAC). These mechanisms can be used with encryption methods to provide data confidentiality.

3.1.2 Structures

The structures model covers the organization of the digital content and its metadata in the digital library. Again disclosure is another security threat where the data objects may be revealed, violating a predefined access right; to combat this attack we can use each of these services on its own, or a group of them together. Authorization is a desired service of the digital library. Authorization is giving permission to use a resource by defining its access rights. Access control is another required security service; it is the prevention of unauthorized use of a resource [39] (i.e., this service controls who has access to a resource, under what conditions access can occur, and what those accessing the resource are allowed to do). In Section 2.5.3 various access control models were described; any of these models can be used to provide access control.

Another security vulnerability in the structures model is the modification of catalog data. The catalog data has all the information about the metadata; any change in the content would cause a variety of problems. An attacker might change the access rights of certain documents, causing a violation of the access rights defined by the owner of the resource. Therefore to detect modification made to the catalog data we must have data integrity. Again, this can be provided by using hash functions, encryption techniques, or MAC. Another important feature that is required is data consistency, which is the integrity, validity, and accuracy of data between applications. In the structures model, organization tools are used to describe the structure in the digital library. An attacker may change the relationships between certain entities, which would make the model inconsistent. By applying the same mechanisms described for data integrity we can provide data consistency for the structures model.

Illegal use of data is another security attack that can occur in the structures model. An attacker may have gained access to a resource by legal means, but then abuses his privileges by illegally using the data, such as making a copy of the document. Here an attacker would be violating his access rights; this can be prevented by providing access control as well as protecting the intellectual property rights of the data. A term is used to describe the protection of content from the different logical security attacks and issues relating to intellectual property rights and authenticity, *digital rights management (DRM)* [24]. In order to have digital rights management, authorization and privacy of the data are implied. To provide DRM we can use a group of technologies such as encryption, passwords, digital signatures, and watermarking.

3.1.3 Spaces

The spaces model describes the user interface of the digital library and the index retrieval model. The index retrieval model is not being used in the generation of the DL, it is only described for purposes of documentation [26]. Disclosure is a major security concern in the spaces model; only authorized personnel should be able to view the user interface details. Modification of the data is also another concern; any changes in the model could cause major changes in the interface. Therefore data integrity, data confidentiality, and authorization are each a requirement. Mechanisms such as hash functions and message authentication codes could be used to provide data integrity, allowing us to detect any changes made to the data. Encryption can be used to provide data confidentiality where all the data can be encrypted and stored along with authorization of the user before he/she is allowed access to the data; this can prevent inappropriate disclosure of the data.

3.1.4 Scenarios

Scenarios describe how the digital library actors behave and how the services of the digital library are carried out. Another security attack is denial of service, which would prevent the digital library from providing any services to its users. Availability is a security requirement that describes a system property that allows authorized users to access and use the system upon demand [39]. There are various types of DoS attacks; pingflood attack involves sending large amounts of ICMP echo command (ping) data packets to the server to attempt to overload it. A counter mechanism requires a network administrator to obtain the IP address of the attacker and block access from accessing the network.

TCP smurf is another DoS attack. It involves the attacker communicating with the victim using the victim's IP address. This causes confusion on the victim's network resulting in a flood of traffic sent to the victim's network device. Firewalls can be used to prevent TCP smurf. A similar DoS attack is UDP fraggle, which is also used to confuse the victim's network, but using UDP. Again UDP fraggle can be prevented by having a firewall or simply by blocking any ports that could be used for fraggle, such as port 7, echo port, or port 17.

Distributed denial of service attack is more complex than the other attacks discussed; it involves ping flood but from various computers. The computers attacking might not be aware they're being used; a Trojan or a virus might have given a hacker control over the devices. There is no simple solution to overcome DDoS attack, but buying an intrusion detection system would help prevent the attack.

3.1.5 Societies

As discussed before, the societies model describes the entities that make up the community of the digital library. Masquerade is another security attack where an attacker falsifies his identity, pretending to be someone else. This is a major problem because certain users may be allowed access to certain content while others are not; if an attacker masquerades as an entity that's allowed to access certain content then a violation of the access rights of the content occurs. Authentication is the guarantee that the entity communicating is who it claims to be [39], thus preventing masquerade. Another form of masquerade that occurs on a lower level (Network Layer of TCP/IP model) is IP address spoofing. IP spoofing is the creation of Internet protocol (IP) packets with a fictitious source IP address to hide the identity of the sender or impersonate another computer system, thus possibly gaining access to confidential content. Providing authentication in the system by using packet filtering can prevent IP spoofing.

A different security attack is misuse of privileges where a user violates the access rights of a digital object. This can be prevented by providing access control, using any of the methods discussed previously.

Another security attack could be session hijacking, which is the use of a session to gain unauthorized access to information or services. Mainly, it refers to the theft of an HTTP magic cookie, which a user uses to authenticate to a remote server. Here authentication is an important feature but it's not enough to authenticate the user at the start of the session; authentication should continue throughout the session. A mechanism to prevent session hijacking is encryption of the traffic between the communicating entities such as SSL.

Authentication bypass is a security threat where an attacker could perform some action that is restricted to authenticated users without providing authentication. This could lead to various other vulnerabilities such as disclosure of protected data or modification of data. Authentication bypass is easy to avoid by providing authentication of the user. We need to ensure the credentials submitted are valid before performing any action. Mechanisms such as Kerberos and X.509 Authentication service verify the user's identity once before the data exchange starts.

Source/Sender non-repudiation is preventing sender or receiver repudiation. Repudiation is the "denial of by one of the entities involved in a communication of having participated in all of or part of the communication" [39]. This is a problem when a user denies communicating in an e-commerce environment. Some digital libraries may require payment to access certain information; the payment process must be secure and must prevent any repudiation by the user. Non-repudiation is a desired requirement that can be satisfied by having users digitally sign any transaction. In the case of payment, an authorized third party may be used to handle the communications, such as PayPal [9].

The table below shows the different security attacks that can occur at each of the 5Ss, what service is required to prevent or detect the attack, and what corresponding mechanism is required to provide that service.

Table 3. The possible security attacks that can occur at each of the 5Ss; the 'S's are color-coded: Red for Scenarios, Blue for Streams, Green for Spaces, Orange for Structures, and Purple for Societies

5S	Security Attack	Security Service	Security Mechanism
Streams	**Disclosure**	Data Confidentiality	Encryption
	Modification of Data	Data Integrity	Hash functions Message Authentication Codes
Structures	**Disclosure**	Access Control + Data Confidentiality + Authorization	Encryption Access Matrix Model Access Control List Role Based AC Task BAC Team BAC Content BAC
	Modification of catalog data	Data Integrity + Data Consistency	Hash functions Encryption Message Authentication Codes
	Illegal use of data	Digital Rights Management + Privacy + Authorization	Encryption Passwords Digital Signatures Watermarking
Spaces	**Disclosure**	Data confidentiality + Authorization	Encryption
	Modification of Data	Data Integrity	Hash functions Message Authentication Codes
Scenarios	**Denial of Service Attacks: Ping flood/ TCP smurf/ UDP fraggle/ DDoS**	Availability	Block IP addresses of attacker Firewalls Block Echo port 17 Intrusion detection systems
	Disclosure	Access Control + Data confidentiality + Authorization	Encryption
Societies	**Masquerade**	Authentication	Encryption Digital signatures

			Passwords
	IP Spoofing	Authentication	Packet filtering
	Session hijacking	Access Control + Authentication	Encryption of traffic between communicating parties
	Authentication bypass	Authentication + Access Control	Kerberos X.509 Authentication Service
	Source/Sender repudiation	Non-repudiation	Digital signatures Kerberos
	Misuse of privileges	Access Control	Access Matrix Model Access Control List Role Based AC Task BAC Team BAC Content BAC

The security services mentioned above are related to one another. For example, by definition, in order to have access control, one must have authentication of the user and confidentiality of the data. Therefore access control involves authentication and confidentiality. Other broad terms such as intellectual property rights include having copyrights on content or more broadly having digital rights management. Figure 4 shows a concept map of the different security issues of a digital library and how they all relate to each other.

Secure digital libraries require data consistency of the content and the catalog of the digital library. Data consistency as defined before is the integrity, validity, and accuracy of data between applications. Here, data integrity is of the different digital library components and is a security requirement that can prevent replay attacks, data insertion, data modification, and data deletion.

Another security issue in digital libraries is availability, which is a security requirement used to prevent denial of service attacks. The possible types of DoS attacks that can occur on a digital library are distributed denial of service attack, ping flood, TCP smurf, and UDP fraggle.

Intellectual property rights is a major security concern in digital libraries since the content stored on the digital library might not be for free; the creator of the content might wish to enforce certain access rights on the content. The content might have copyrights that determine how the content is to be used. Digital Rights Management is a sub-category of intellectual property rights, which refers to the protection of content from the different logical security attacks and issues relating to intellectual property rights and authenticity. In order to enforce DRM certain sub-requirements are needed, for example, digital signatures and access control. These can be used to preserve the authenticity of the object.

Access Control as stated above is basically having data confidentiality and authentication of the user along with a series of access usage definitions for each of the objects. These definitions describe who can access a resource, under what conditions, and what can be done with the resource.

As seen in the DELOS reference model an important aspect of digital library design involves the different policies, including the security policies. Security policies are the different regulations and conditions that govern how a system stores, manages, protects, and distributes sensitive information. There are various security policies in digital libraries that define the terms and conditions for use of the digital library; some policies may define the privacy issues relating to a digital library. This includes the privacy from the user's perspective as well as the collection and distribution of the data. There are various legal issues concerning privacy, especially the personal information being stored, for example, in certain digital libraries that store sensitive personal information about people, their salaries, and their income. DL designers must make sure that the personal information is confidential and not available to anyone or may risk facing legal charges.

During the requirements gathering phase, various digital libraries were studied to understand the nature of the security requirements. In the CTRnet project [2], Professor Donald Shoemaker, from the Sociology Department at Virginia Tech, described the sensitive nature of information stored on CTRnet. Information such as survey results, which may include sensitive information such as the income of studied subjects, must not be revealed to just anyone.

Certainly some members of the study group may view this information, however, the information must be stored in a way that would give an abstract view of the study findings without revealing sensitive personal information to someone who does not have the right to view the information. An example of privacy violation could result when personal information could be deduced after multiple queries are conducted, using the digital library search agent. All these issues are privacy issues that need to be addressed. An important aspect of privacy is authorization, which in turn requires specifying access control.

Another security issue that affects digital libraries is SPAM. During the discussion with the CTRnet project members, it was stated that SPAM was a serious security issue. The digital library offers forums for victims of crises or tragedies to exchange ways to deal with grief and recovery. The DL is free to any user; it’s only required that they create an account the first time they use the DL. Spammers have been using the forum to publish adverts and other spam messages.

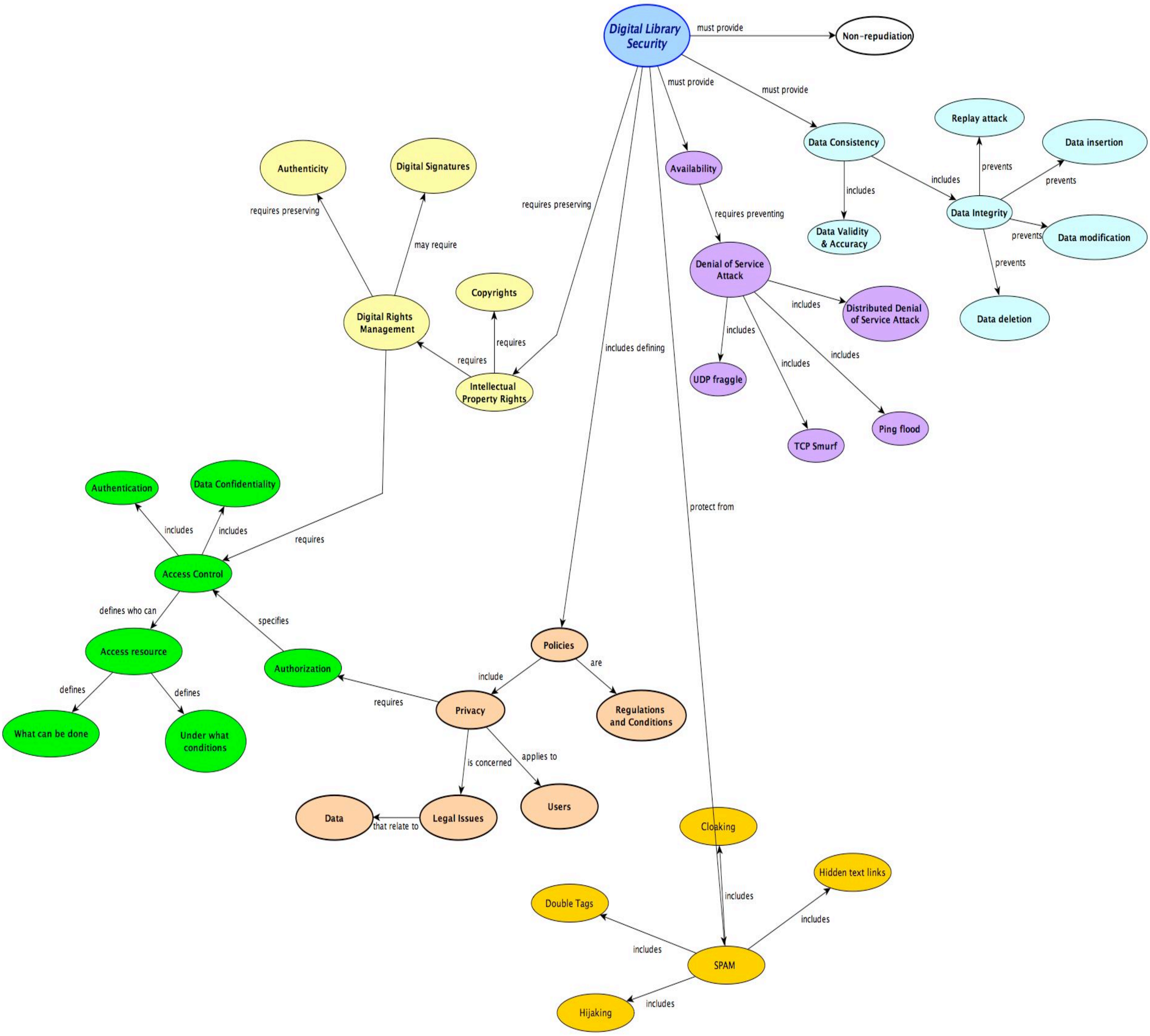


Figure 4. Concept map of the issues related to digital library security

3.2 Secure 5SL XML Schema

The hypothesis of this thesis is that the 5SL can be extended to describe the security requirements of digital libraries, e.g., those discussed in the previous section. We show that, but cannot prove completeness. What is discussed below is a set of additions to what has been specified earlier with regard to 5SL [25], hence we do not tie closely to previously defined 5SL constructs. Further, even with regard to just security concerns, the additions are incomplete; see more about future work that can be done in Section 5.2. Also, not all scenarios related to security, and not all interconnections that run across the Ss, are given below; these issues could lead to future work too.

3.2.1 General Schema

The XML schema is based on the 5SL; each of the 5S model's security requirements are encapsulated in an element. There are 5 elements, secure streams, secure structures, secure societies, secure scenarios, and secure spaces. Each of these elements defines the necessary security requirements discussed above along with the mechanisms to provide these requirements, giving details of configuration. EditiX [4] was used to validate the schema.

Figure 5 shows the outline of the schema.

```
<xs:schema>
    <xs:element name= “secure_streams”>
      ..
    </xs:element>
    <xs:element name= “secure_structures”>
      ..
    </xs:element>
    <xs:element name= “secure_spaces”>
      ..
    </xs:element>
    <xs:element name= “secure_scenarios”>
      ..
    </xs:element>
    <xs:element name= “secure_societies”>
      ..
    </xs:element>
</xs:schema>
```

Figure 5. Outline of the extended 5SL schema

3.2.1 Streams Schema

There are 2 main security requirements in the streams model, data confidentiality and data integrity. Below is an outline of the basic structure of the secure_streams model.

```
<xs:element  name= “secure_streams” >
  <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" name=”DataType”>
          <xs:complexType>
          <xs:sequence>
              <xs:element minOccurs="1" maxOccurs="1"  name= “DataConfidentiality”>
                  …
                  …
              </xs:element>
              <xs:element minOccurs="1" maxOccurs="1"  name= “DataIntegrity”>
                  …
                  …
              <xs:element>
          </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
  </xs:complexType>
</xs:element>
```

The 5SL schema of the streams model was based on MIME types [26] which consist of a type, subtype, language, and an encoding. Performing encryption using symmetric encryption models or asymmetric encryption models is the same for any “type” of data stored, i.e., all text files that are PDFs could be encrypted using the RSA algorithm with a public key. Figure 6 shows a graphical outline of the schema.

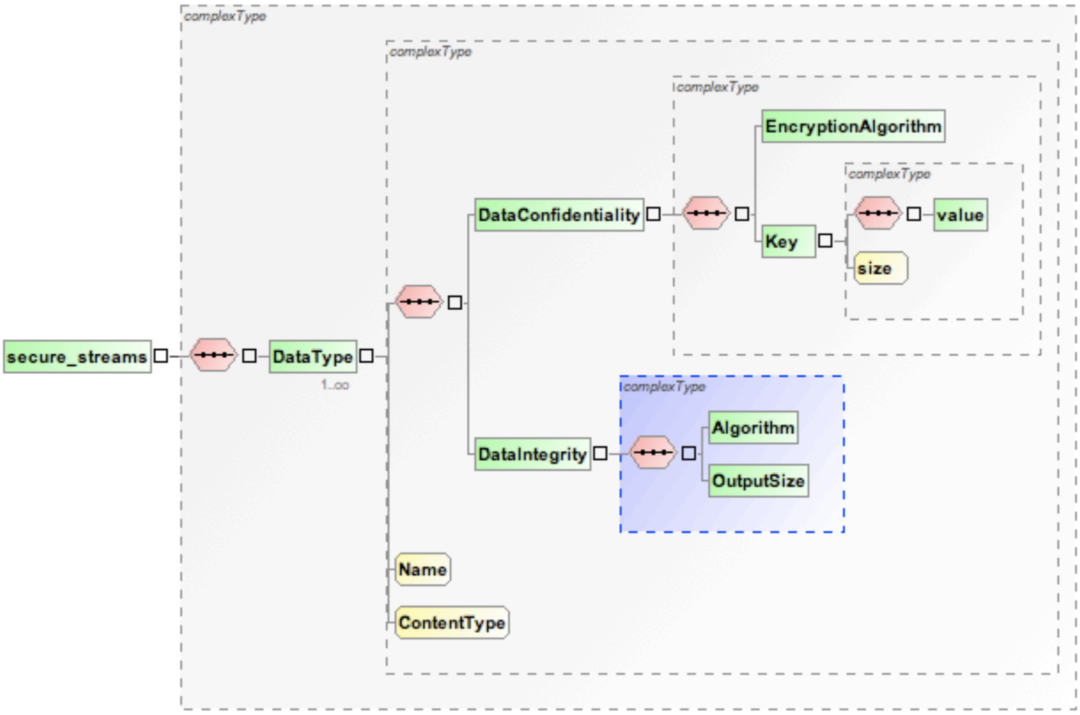


Figure 6. Graphical View of XML Schema for Secure Streams

From Figure 6 we can see that the secure_streams element is made up of a sequence of elements and subelements. DataType is a child element of the secure_streams element; there should be at least 1 DataType element. Each DataType has 2 children elements; DataConfidentiality and DataIntegrity, only 1 occurrence of both elements is required. DataConfidentiality describes how each datatype will be encrypted, using which algorithm (DES, AES, RSA, or Blowfish), the key, and its size. DataIntegrity is the second child, which defines what integrity mechanism each DataType will use. This is defined by having an Algorithm element (SHA1, MD5, DAA) and the OutputSize (160,128,64). The OutputSize is restricted to a minimum of 16 bits because the minimum size of a hashed value or a MAC should not be less than 16 bits.

```
<xs:element maxOccurs="unbounded" name="DataConfidentiality">
    <xs:complexType>
    <xs:sequence>
      <xs:element maxOccurs="unbounded""  name="EncryptionAlgorithm" type="xs:string”/>
      <xs:element maxOccurs="unbounded" name="Key" type="xs:string">
      <xs:complexType>
            <xs:sequence>
            <xs:element name="value" type="xs:string"/>
            </xs:sequence>
            <xs:attribute name="size" type="xs:integer" use="required"/>
      </xs:complexType>
      </xs:element>
    </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element maxOccurs="unbounded" name="DataIntegrity">
      <xs:complexType>
      <xs:sequence>
       <xs:element maxOccurs="unbounded" name="Algorithm" type="xs:string"/>
       <xs:element maxOccurs="unbounded" name="OutputSize">
         <xs:simpleType>
         <xs:restriction base="xs:integer">
         <xs:minInclusive value="16"/>
         </xs:restriction>
       </xs:simpleType>
       </xs:element>
      </xs:sequence>
      </xs:complexType>
</xs:element>
```

Figure 7. Data Confidentiality and Data Integrity in Streams model

3.2.2 Structures Schema

The structures model defines the internal structure of every digital object, the metadata about that object, and properties of collections and catalogues. Data confidentiality, data integrity, digital rights management, authorization, and access control are all issues affecting the structures model. Below is an outline of the schema.

```
<xs:element  name= “secure_structures” >
  <xs:complexType>
  <xs:sequence>

  <xs:sequence>
  <xs:complexType>
</xs:element >
```

Figure 8 shows a graphical outline of the secure structures model shown below. Data confidentiality and data integrity are requirements that overlap with both the streams and structures models. Since the data confidentiality and data integrity of each type of digital content was defined in the streams model, it does not need to be defined in the structures model, especially since in the 5SL framework the data types defined in streams are enumerated in the structures model. The metadata of each document needs to be encrypted and a method of integrity must be used to detect any modification in the metadata.

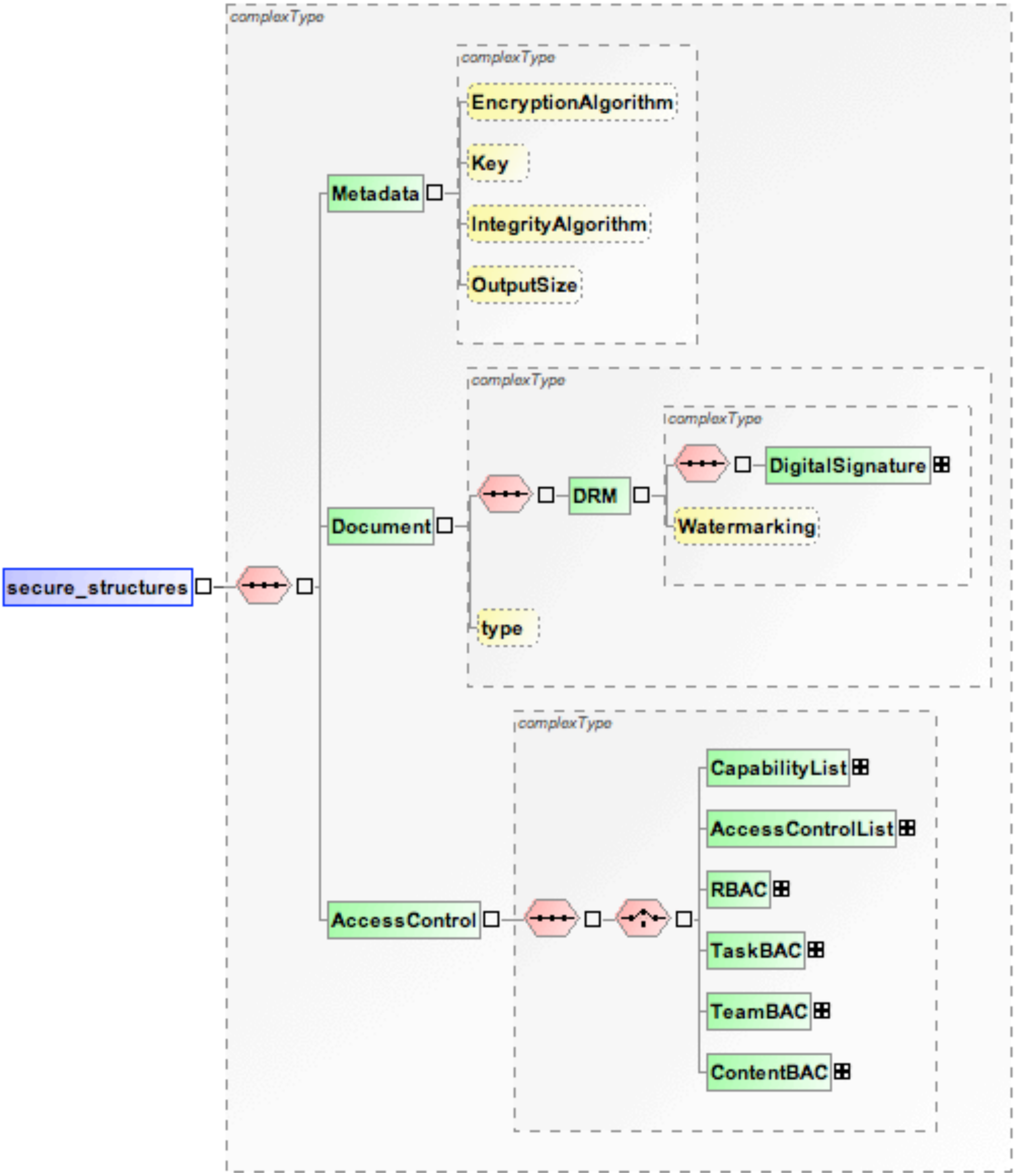


Figure 8. Graphical Display of the XML Schema for secure_structures

From the figure above the secure_structures is made up of a sequence of 3 elements: metadata, document, and AccessControl. As discussed before, the metadata could be modified by an attacker, therefore the DL designer may wish to encrypt the metadata using any of the encryption techniques discussed. Therefore, he'd define the attribute Encryption and the attribute Key, which refers to the size of the key. Or if he just wants to have a checksum or hashed value to detect any changes in the metadata, the attributes IntegrityAlgorithm and OutputSize are defined.

Each document has an attribute type, which refers to one of the DataTypes defined in the streams model. These documents must have their intellectual property rights preserved; an element DRM is defined which has an attribute watermarking, where the DL designer would define the watermarking technique to be used on this document. The DL designer may wish to preserve the property rights of the document by defining a digital signature to be attached to the document. There is a choice of 3 techniques to use, HashFunction, MAC, or PrivateKey; each of these XML elements is used to define how the document is to be signed.

Finally, access control of the documents is defined; it is not a child element of Document because access control should be an overall system property and not just related to the document; it spans all the structural elements of the model. There is a choice of elements to use: CapabilityList, AccessControlList, RoleBAC, TaskBAC, TeamBAC, or ContentBAC; each one of these elements refers to the access control models described in Section 2.5.3. The full XML schema of secure_structures is in Appendix B.

3.2.3 Spaces Schema

The spaces model in the 5SL is based on the User Interface Markup Language (UIML) [13] which defines user interfaces using XML. The security requirement for spaces is confidentiality. The outline of the secure spaces model is shown below. Figure 9 shows a graphical outline of the schema.

```
<xs:element  name= “secure_spaces” >
  <xs:complexType>
  <xs:sequence>
  <xs:element name= “App” type= “xs:string”>
  …
  …
  </xs:element>
  </xs:sequence>
  </xs:complexType>
</xs:element >
```

In a UIML description of the user interface the tag app is made up of multiple groups; each group is made up of multiple elements, each describing the name of the element and its class, for example, if it's a menu item or toolbar. Data confidentiality is concerned with the whole interface definitions; therefore any encryption mechanism should be applied to the whole application (app) and not per group. The same applies to data integrity. We want to be able to detect any change made to the schema, on a per app basis, as is shown in Figure 10.

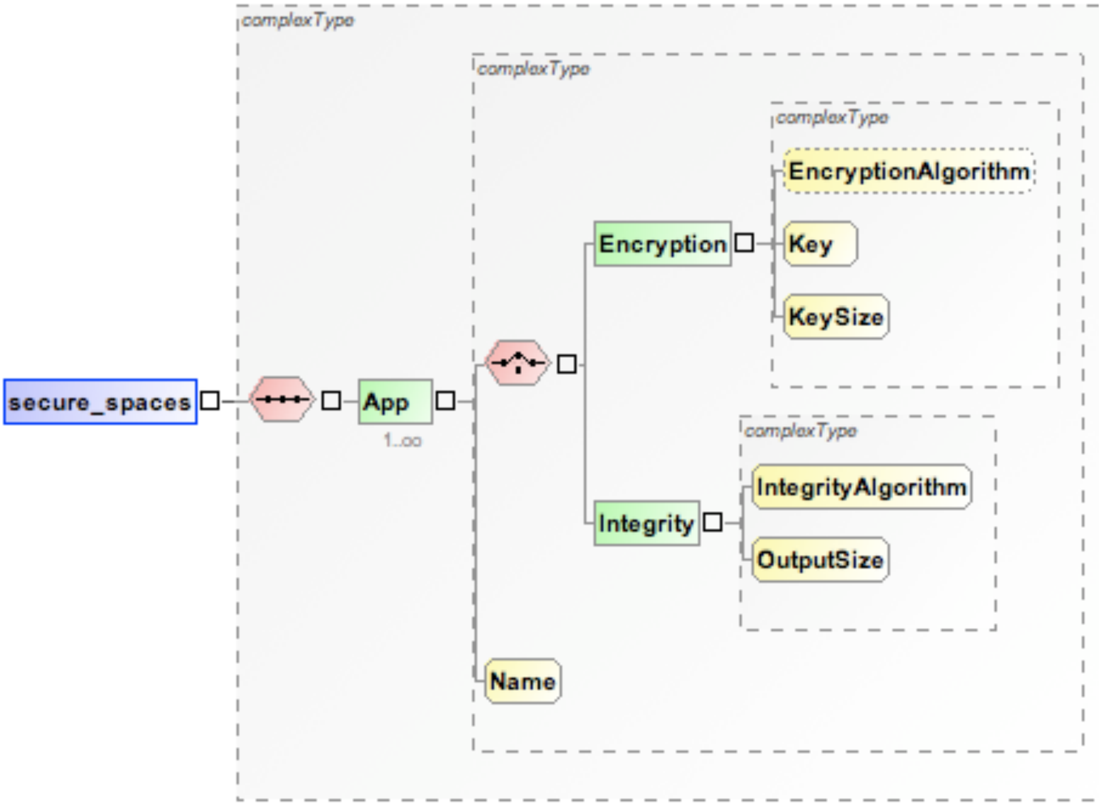


Figure 9. Graphical View of XML Schema of Secure Spaces

In the figure above, the secure_spaces element is made of the App element, which is made up of 2 elements, Encryption and Integrity, along with the attribute Name of the App. The App element is an UIML tag, which specifies an interface application, e.g., an App could be a toolbar that’s available in the application. There is a choice of either using encryption methods on the App or Integrity methods on the App.

```
<xs:element maxOccurs="unbounded" minOccurs="1" name="App">
    <xs:complexType>
    <xs:choice>
        <xs:element name="Encryption">
        <xs:complexType>
        <xs:attribute name="EncryptionAlgorithm"/>
        <xs:attribute name="Key" use="required"/>
        <xs:attribute name="KeySize" use="required"/>
        </xs:complexType>
        </xs:element>
        <xs:element name="Integrity">
        <xs:complexType>
        <xs:attribute name="IntegrityAlgorithm" use="required"/>
        <xs:attribute name="OutputSize" use="required">
        <xs:simpleType>
            <xs:restriction base="xs:integer">
            <xs:minInclusive value="16"/>
            </xs:restriction>
        </xs:simpleType>
        </xs:attribute>
        </xs:complexType>
        </xs:element>
    </xs:choice>
    <xs:attribute name="Name" use="required"/>
    </xs:complexType>
</xs:element>
```

Figure 10. XML Schema of Secure Spaces

3.2.4 Scenarios Schema

There are 2 main security requirements in the scenarios model, data confidentiality and availability. Below is an outline of the basic structure of secure_scenarios model.

Figure 11 shows a graphical outline of the structure of the schema.

```
<xs:element  name= “secure_scenarios” >
  <xs:complexType>
      <xs:element name= “DataConfidentiality” >
      <xs:complexType>
      ..
      ..
      </xs:complexType>
      </xs:element>
      <xs:element name= “Availability” >
      <xs:complexType>
      ..
      ..
      </xs:complexType>
      </xs:element>
  </xs:complexType>
</xs:element >
```

In the 5SL model [26] the scenarios were modeled using UML sequence diagrams and then the diagrams were serialized into XML elements. Where each scenario had an event, each event had a sender, receiver, action, and multiple parameters for the action. This model is used in the secure_scenarios; our main concern however is the actions performed and the parameters of these actions. Any one of these parameters may be confidential; therefore as in Figure 12, encryption techniques are applied to any parameter.

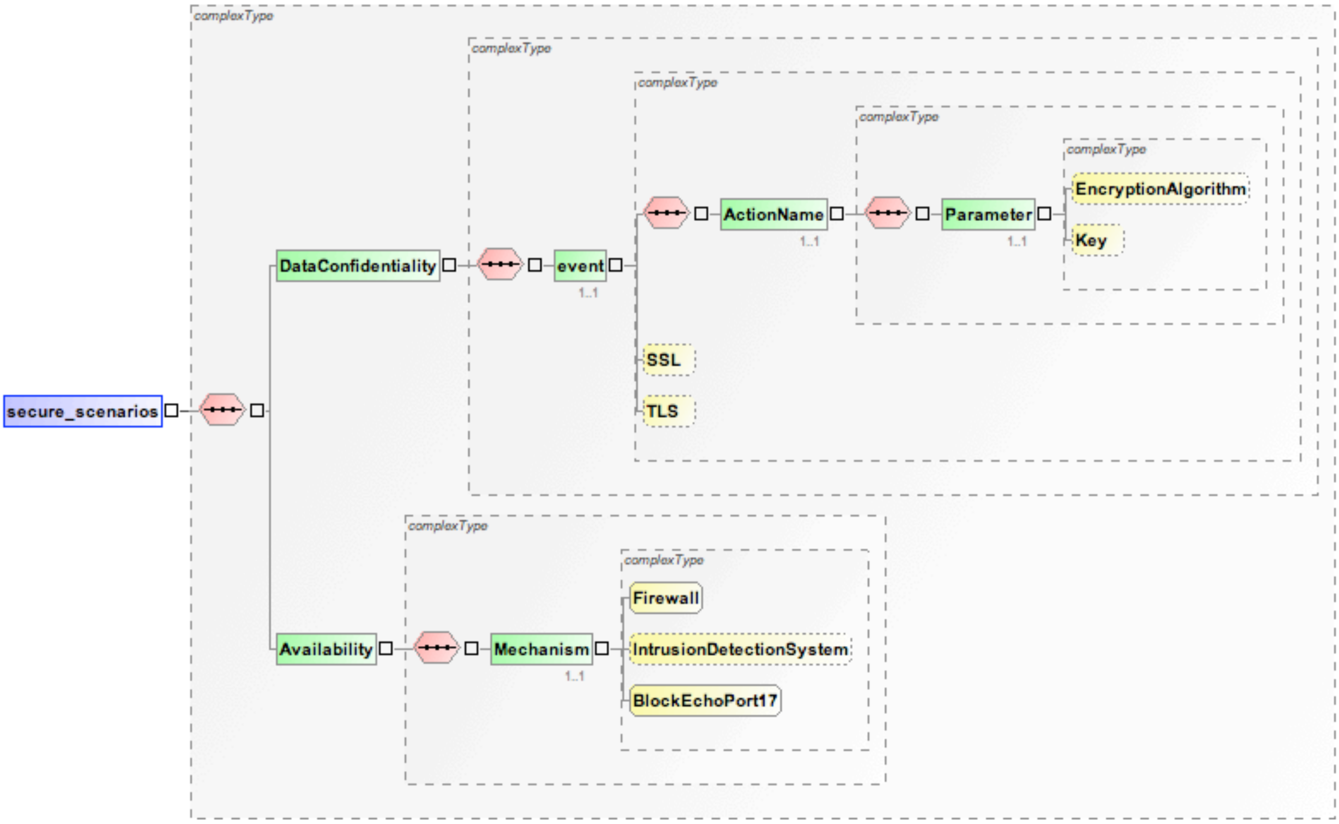


Figure 11. Graphical view of XML Schema of Secure Scenarios

Other than providing confidentiality to each parameter sent to/from sender and receiver of an action, the session itself can be secured. Secure Socket Layer, Transport Layer Security, and Secure HTTP (HTTPs) are methods for protecting the communications channel, providing the necessary data confidentiality.

```
<xs:element name="DataConfidentiality">
        <xs:complexType>
        <xs:sequence>
        <xs:element minOccurs="1" name="event">
                <xs:complexType>
                <xs:sequence>
                <xs:element minOccurs="1" name="ActionName">
                  <xs:complexType>
                  <xs:sequence>
                    <xs:element minOccurs="1" name="Parameter">
                    <xs:complexType>
                    <xs:attribute name="EncryptionAlgorithm" type="xs:string"/>
                    <xs:attribute name="Key" type="xs:string"/>
                    </xs:complexType>
                    </xs:element>
                  </xs:sequence>
                  </xs:complexType>
                </xs:element>
                </xs:sequence>
                </xs:complexType>
        </xs:element>
        </xs:sequence>
        </xs:complexType>
</xs:element>
```

Figure 12. Data Confidentiality element of the Secure Scenarios model

Availability is a quality service that’s applicable on the whole system; therefore it is not defined for each event but is used on the whole system by defining 1 to 3 of the following attributes; firewalls which is a required attribute, intrusion detection system, and block echo port 17, which also is required. The availability schema is shown in Figure 13.

```
<xs:element name="Availability">
     <xs:complexType>
     <xs:sequence>
          <xs:element minOccurs="1" name="Mechanism">
          <xs:complexType>
             <xs:attribute name="Firewall" type="xs:string" use="required"/>
             <xs:attribute name="IntrusionDetectionSystem" type="xs:string"/>
             <xs:attribute name="BlockEchoPort17" type="xs:boolean" use="required"/>
          </xs:complexType>
          </xs:element>
          </xs:sequence>
          </xs:complexType>
          </xs:element>
     </xs:sequence>
     </xs:complexType>
</xs:element>
```

Figure 13. Availability element of the Secure Scenarios model

3.2.5 Societies Schema

The societies model in the 5SL classifies the community of the digital library as managers or actors. Managers maintain the digital library and its services while the actors use these services. The XML schema of the 5SL societies model describes each actor and manager and the different operations that he/she may perform and the relationships between them. The outline of the secure societies model is shown below; Figure 14 shows a graphical outline of the schema. The model defines methods for user authentication. To provide authentication one can use usernames and passwords, digital credentials, or digital signatures. Access control overlaps with both the societal model and the structural model. Although access control is an issue in the societies model, it will not be defined here since it was defined in the structures model.

```
<xs:element  name= “secure_societies” >
  <xs:complexType>
  <xs:sequence>
      <xs:element name = “Authentication”>
```

```
          <xs:complexType>
          …
          …
          </xs:complexType>
            </xs:element>
      </xs:sequence>
      </xs:complexType>
    </xs:element >
```

Figure 14 shows the secure_societies element being made up of Authentication element; authentication is a security requirement that is to be applied on both Actors and Managers. An element Type is defined; the attribute value refers to whether the Type is an actor or a manager. Each Type has a Community, and, as in the 5SL framework, each community member must have a name, ID, and service defined in the attributes name and ID (and the latter in the child element service). As discussed in Section 2.5.3 a user needs an identifier, which could be a username and password or a digital credential. Username_Password element defines the username and password properties of a user by allowing the DL designer to specify certain restrictions, as shown in Figure 15 (see Appendix A for detailed XML schema). The Credentials element defines a user credential by allowing the DL designer to specify how long the user name is and how it should be entered, the ID, the department, and the title of the user credential (see Appendix A).

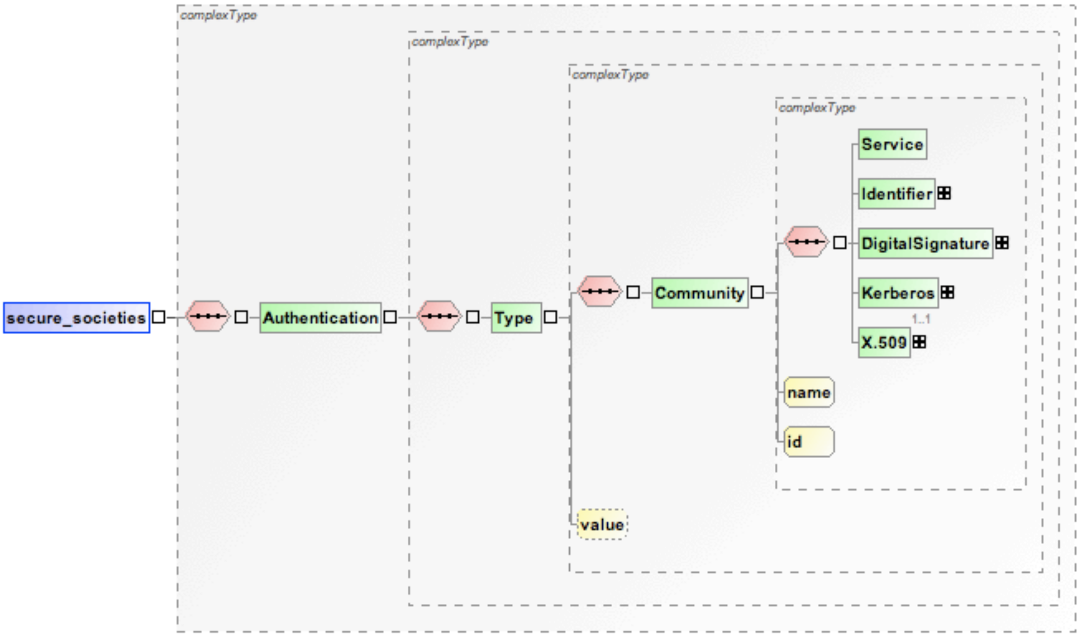


Figure 14. General XML Schema of Authentication in the Societies Model

A user may be authenticated using his digital signature; the DigitalSignature element specifies the type of signature used. Other authentication techniques are Kerberos and X.509 authentication service, as shown in Figure 16; each of these techniques has a corresponding element that is used to define the configuration of the services (see Appendix A). In Kerberos the realm, hostname, and Kerberos password are defined. In X.509 the certificate authority (CA) is defined, whether it is a privately defined or a trusted third party.

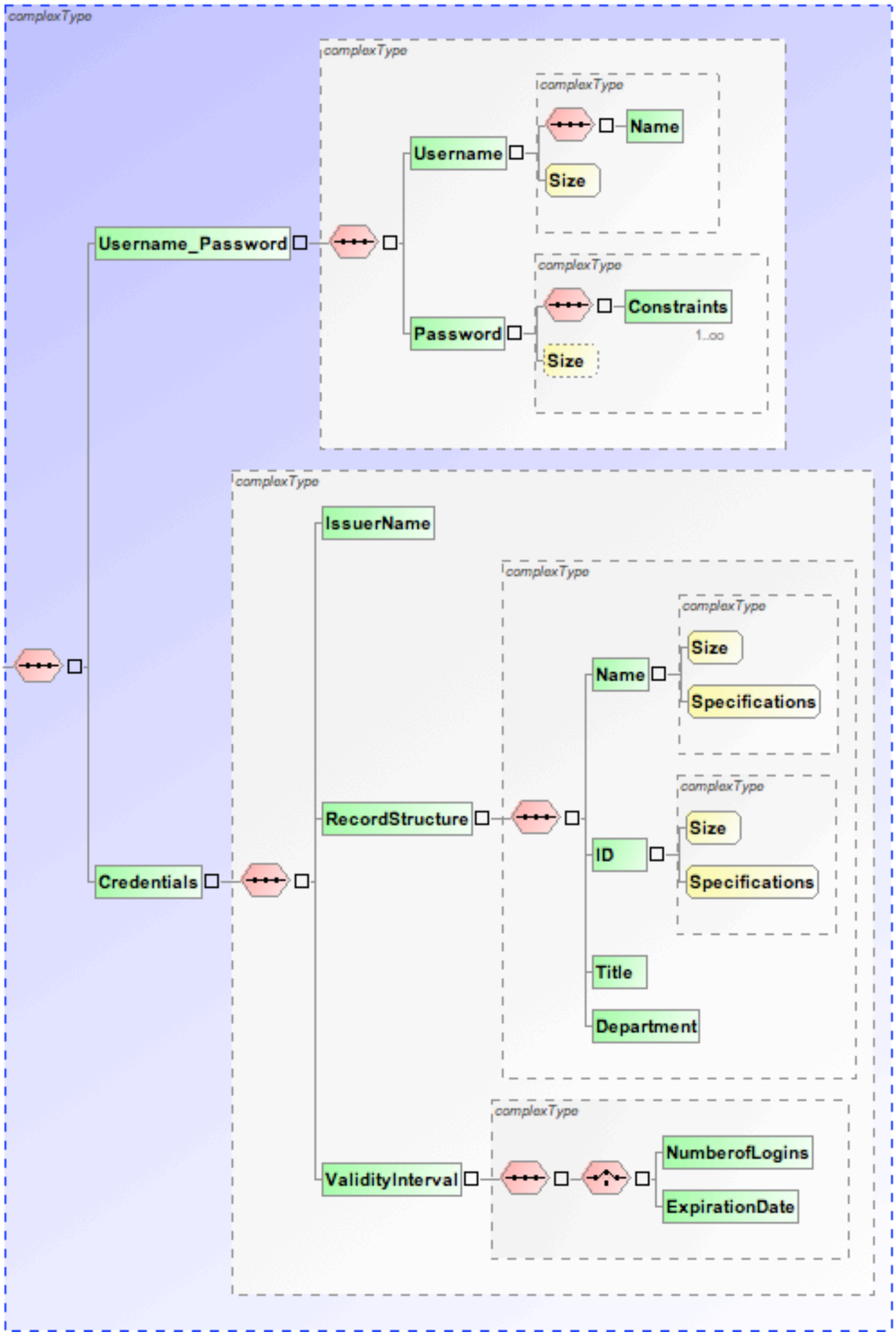


Figure 15. General XML Schema of Identifier element in the Societies Model

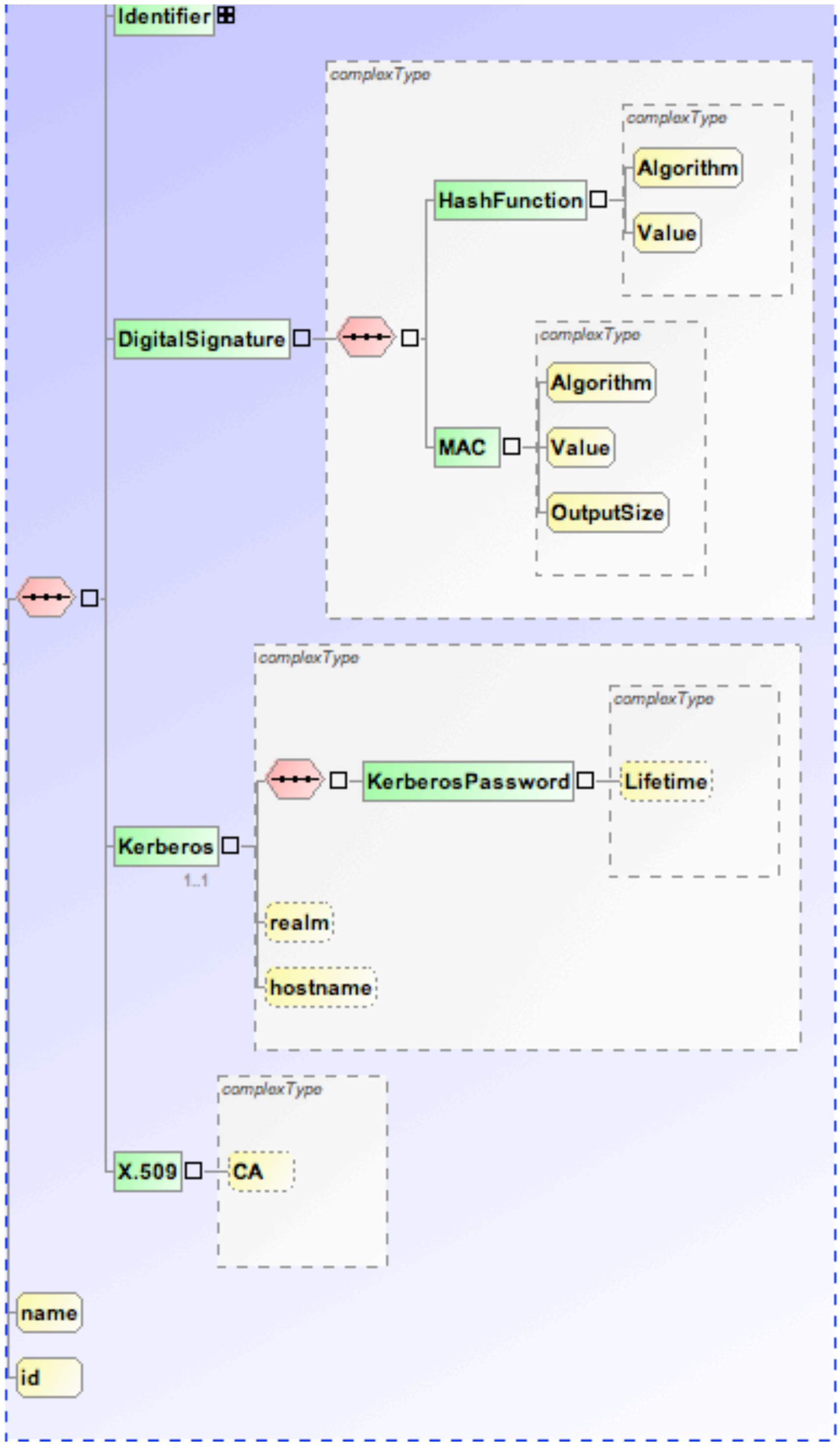


Figure 16. XML Schema of Kerberos, X.509, and Digital Signature elements

It is important to note that the extended 5SL defines the security features to be added to the 5SL. It is supposed to be used with the 5SL to define security requirements when needed. In certain digital libraries, some parts of the library might need security considerations while other parts might provide open access to users; those parts that require security could use the extended 5SL along with the basic 5SL.

Chapter 4. DL Case Study with secure 5SL

In the previous section we introduced the secure 5SL; in this section we will apply the secure 5SL to a case study, discussed in a moderate amount of detail.

Egyptian Universities Library Consortium

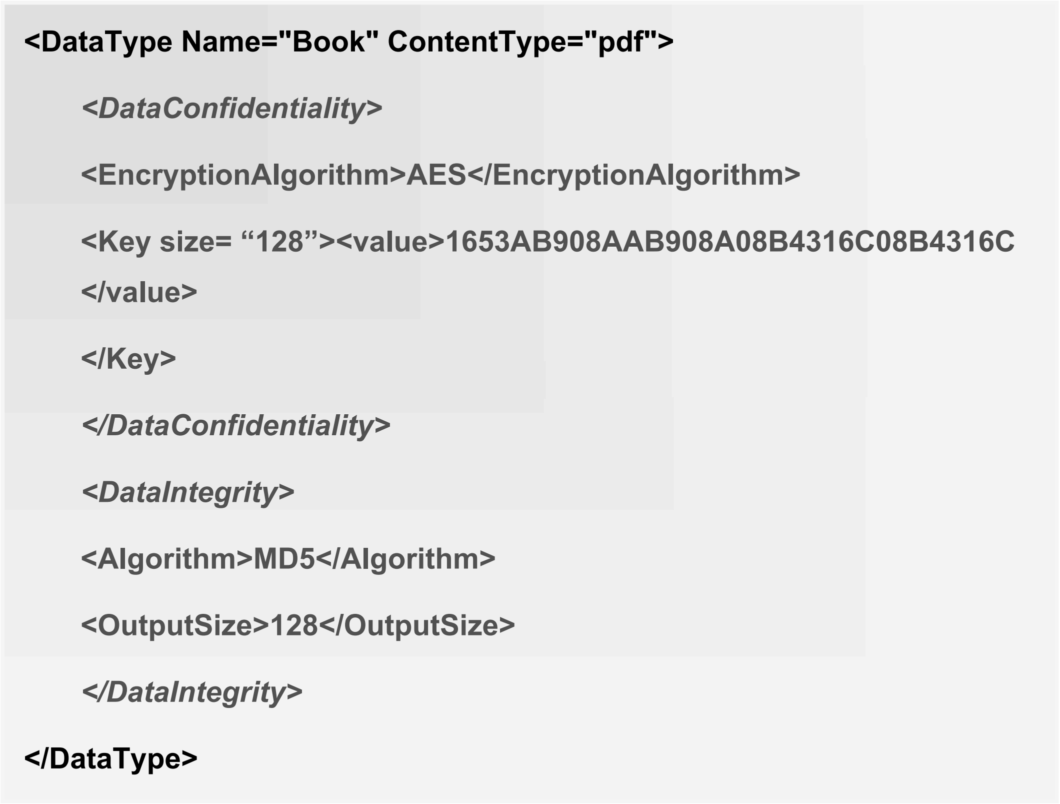
The Egyptian universities library consortium (EULC) [5] is a digital library that holds theses, dissertations, and research works from universities from all over Egypt. It has information about any publication from an Egyptian university or researcher and about the local periodicals. It offers to its members a range of online electronics resources to view either from the EULC database, or information about the books available to borrow from one of the Egyptian University Libraries as well as information about resources from other libraries around the world.

Each of the universities in Egypt has its own site that is supervised by a group of site managers. There are 25 different sites for the different universities and research institutions participating in EULC.

4.1.1 Streams

The EULC deals with many types of streams: text, images, audio, and video. The EULC also allows users to search for objects that may be found in their local university library. Content such as projector slides, cartographic material, notated music, and even computer files are among the different types of objects stored.

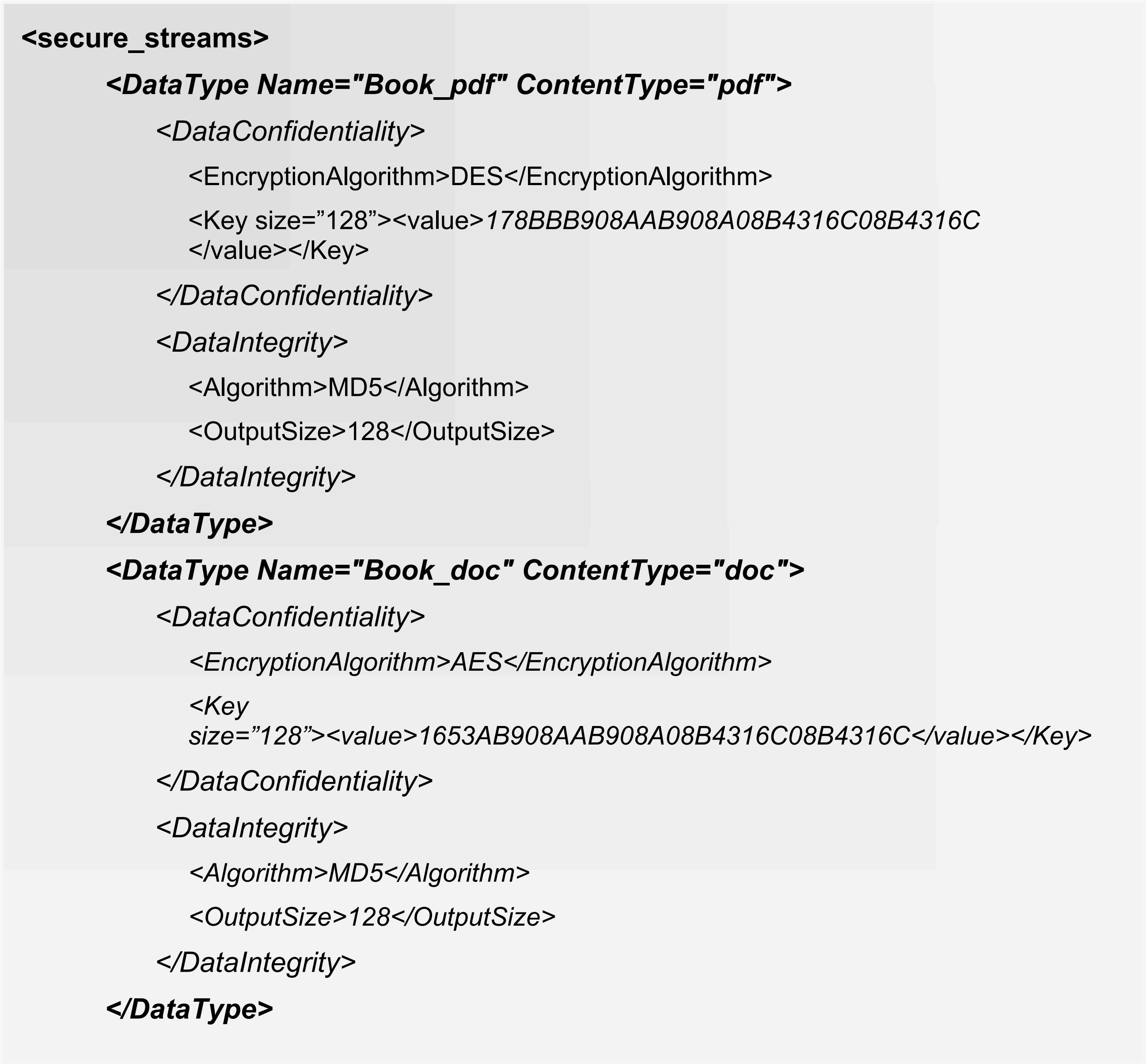
There are books, dissertations, theses, research papers, and periodicals stored in the EULC digital library; we aim to define the security requirements for them using secure_streams; a sample is shown below.



```
<DataType Name="Book" ContentType="pdf">
    <DataConfidentiality>
    <EncryptionAlgorithm>AES</EncryptionAlgorithm>
    <Key size= “128”><value>1653AB908AAB908A08B4316C08B4316C
    </value>
    </Key>
    </DataConfidentiality>
    <DataIntegrity>
    <Algorithm>MD5</Algorithm>
    <OutputSize>128</OutputSize>
    </DataIntegrity>
</DataType>
```

Figure 17. Sample of DataType definition using secure_streams

For each type of book a similar XML segment will be defined, but changing the ContentType accordingly. The same applies to the research papers, periodicals, dissertations, and theses. The EncryptionAlgorithm and key are to change accordingly; the Algorithm and Outputsize are to change accordingly too. A broader example is shown in Figure 18.



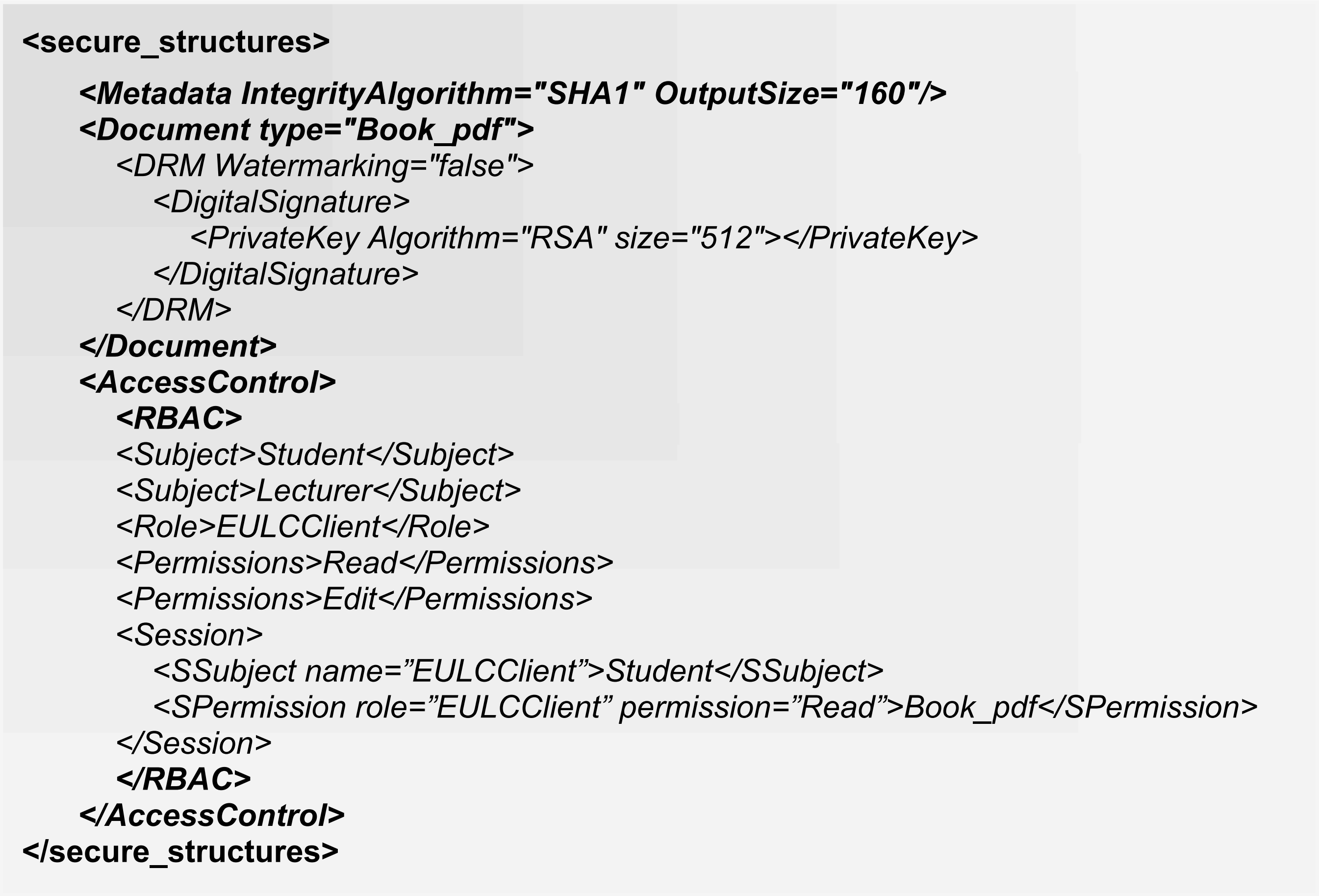
```
<secure_streams>
        <DataType Name="Book_pdf" ContentType="pdf">
            <DataConfidentiality>
                <EncryptionAlgorithm>DES</EncryptionAlgorithm>
                <Key size=”128”><value>178BBB908AAB908A08B4316C08B4316C
                </value></Key>
            </DataConfidentiality>
            <DataIntegrity>
                <Algorithm>MD5</Algorithm>
                <OutputSize>128</OutputSize>
            </DataIntegrity>
        </DataType>
        <DataType Name="Book_doc" ContentType="doc">
            <DataConfidentiality>
                <EncryptionAlgorithm>AES</EncryptionAlgorithm>
                <Key
                size=”128”><value>1653AB908AAB908A08B4316C08B4316C</value></Key>
            </DataConfidentiality>
            <DataIntegrity>
                <Algorithm>MD5</Algorithm>
                <OutputSize>128</OutputSize>
            </DataIntegrity>
        </DataType>
```

Figure 18. Example of secure_streams with multiple datatypes

4.1.2 Structures

EULC DL stores some digital objects discussed in Section 3.1.1 such as books, periodicals, and so on, and has information about other physical objects available in the University Library. The metadata of the EULC digital objects is described using the Dublin Core standard.

To use the secure_structures to describe the security of EULC DL can be done as shown in the example in Figure 19. The DL designer would first identify the Integrity mechanism to be used on the Metadata. Then, for each Document with one of the DataTypes defined in streams, watermarking is selected along with a choice of techniques for Digital Rights Management. Here for Document of type Book in PDF format there will be no watermarking used, but the book will be digitally signed using a PrivateKey of size 512 generated by the RSA algorithm.



```
<secure_structures>
    <Metadata IntegrityAlgorithm="SHA1" OutputSize="160"/>
    <Document type="Book_pdf">
      <DRM Watermarking="false">
        <DigitalSignature>
          <PrivateKey Algorithm="RSA" size="512"></PrivateKey>
        </DigitalSignature>
      </DRM>
    </Document>
    <AccessControl>
      <RBAC>
      <Subject>Student</Subject>
      <Subject>Lecturer</Subject>
      <Role>EULCClient</Role>
      <Permissions>Read</Permissions>
      <Permissions>Edit</Permissions>
      <Session>
        <SSubject name=”EULCClient”>Student</SSubject>
        <SPermission role=”EULCClient” permission=”Read”>Book_pdf</SPermission>
      </Session>
      </RBAC>
    </AccessControl>
</secure_structures>
```

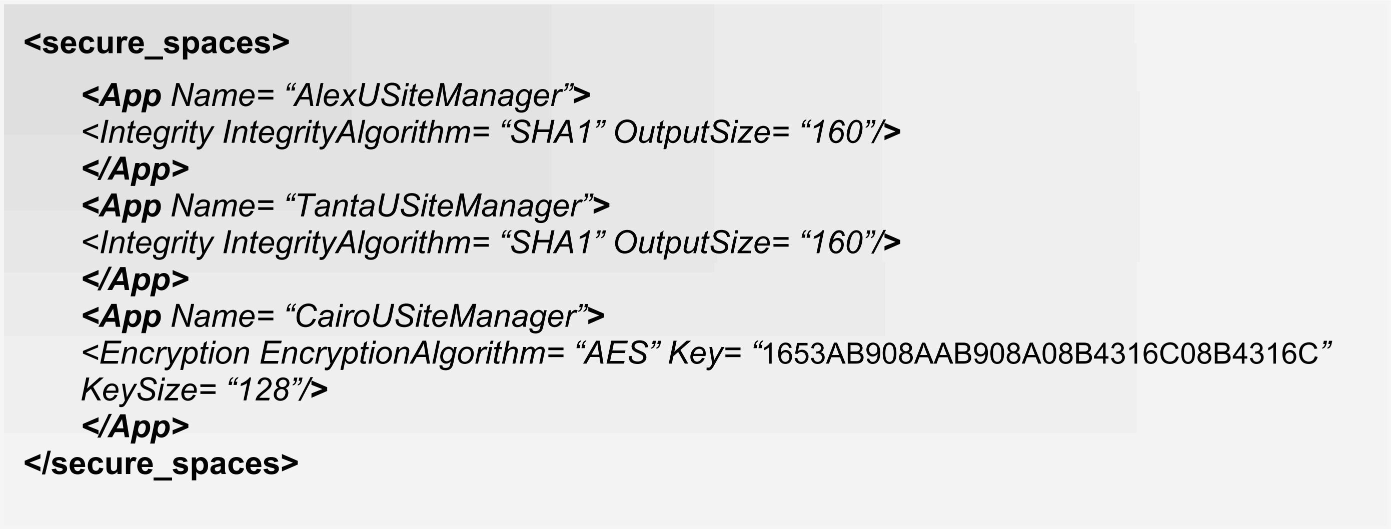
Figure 19. Example of secure_structures

The example in Figure 19 also shows how Access Control could be defined using the RBAC model. The model requires a series of different Subjects. Again, these are from the definition in the secure_socieities along with the roles available; for example EULCClient is a role describing the users such as Students and Lecturers who use the DL for access to content and are active members of a EULC university or institution.

Permissions describes the different authorization that the different roles can have, such as read or edit a document. Finally Session is a mapping of roles to permissions and subjects to roles; for example here a student has the role EULCCLient; the EULCClient role has read permissions.

4.1.3 Spaces

A spatial aspect of the EULC digital library is that the DL is an integration of various sub-libraries at different physical locations. Each Egyptian university and some institutions have their own local website, therefore each of these websites have their own abstraction of the data customized for the university or institution. Figure 20 shows how spaces could be used to define the integrity and confidentiality of the different interfaces.



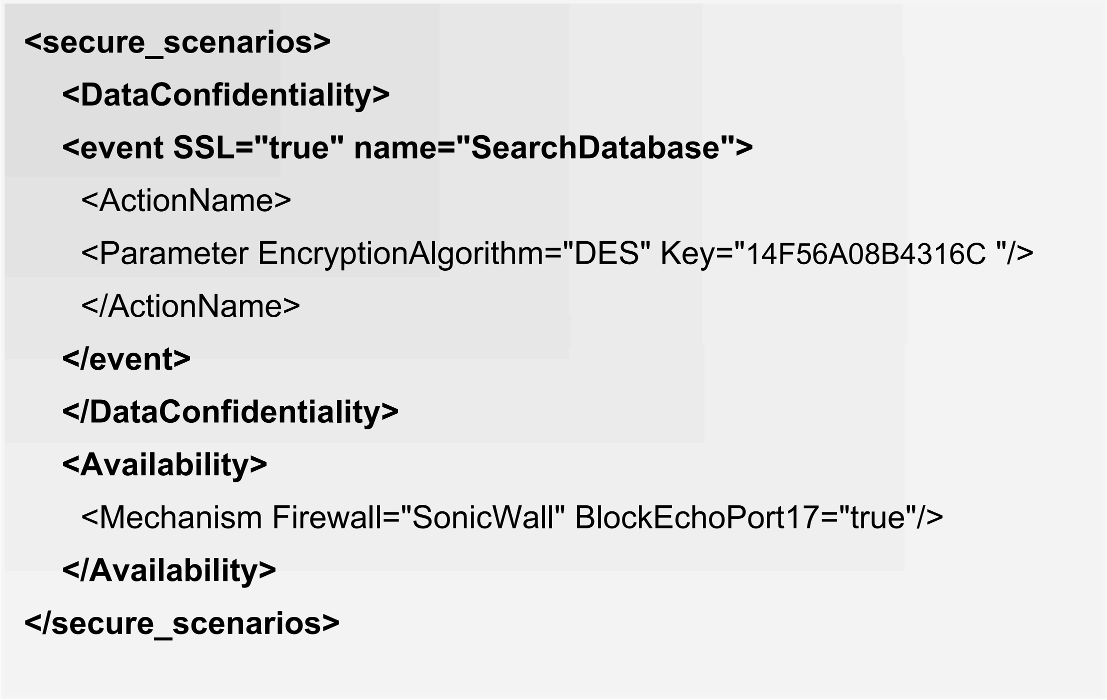
```
<secure_spaces>
    <App Name= “AlexUSiteManager”>
    <Integrity IntegrityAlgorithm= “SHA1” OutputSize= “160”/>
    </App>
    <App Name= “TantaUSiteManager”>
    <Integrity IntegrityAlgorithm= “SHA1” OutputSize= “160”/>
    </App>
    <App Name= “CairoUSiteManager”>
    <Encryption EncryptionAlgorithm= “AES” Key= “1653AB908AAB908A08B4316C08B4316C”
    KeySize= “128”/>
    </App>
</secure_spaces>
```

Figure 20.. Example of EULC spaces defined by Secure_Spaces

4.1.4 Scenarios

Each community member of EULC performs some actions or procedures in the digital library. There are various processes that go on. There is the process of submission, where the employees of the DL enter and edit metadata about the different resources. The managers of the sites make sure that the system is accessible all the time.

The students and faculty members may search the collection for a desired book, thesis, or dissertation. This could be written using the secure_scenarios model; this is shown in Figure 21.



```
<secure_scenarios>
    <DataConfidentiality>
    <event SSL="true" name="SearchDatabase">
      <ActionName>
      <Parameter EncryptionAlgorithm="DES" Key="14F56A08B4316C "/>
      </ActionName>
    </event>
    </DataConfidentiality>
    <Availability>
      <Mechanism Firewall="SonicWall" BlockEchoPort17="true"/>
    </Availability>
</secure_scenarios>
```

Figure 21. An example of using secure_scenarios for the event SearchDatabase

4.1.1 Societies

The primary community addressed through the EULC society is university students, either graduate or undergraduate. The digital library aims to provide students with publications to assist them in their education, and to help students who are preparing either a thesis or dissertation.

Another community involved in EULC is that of faculty. Faculty members might also search the DL for publications or references that might be useful to their work.

The overall site of the EULC is managed by employees of Mansoura and Zagzig University; however, each independent site has its own site manager who supervises the functionality of the DL and employees who control and update the data on the site. These administrators provide a third community in EULC.

The EULC societies actors would be faculty and students; these can be called EULCClients, while the employees and site managers are of type Managers.

Figure 22 shows how Student could be defined using the secure_societies model. The student is identified using credentials; the issuer name is the EULC; the specifications of the name and ID are defined. A validity interval of 10 logins is set for this credential type.

```
<Identifier>
<Credentials>
        <IssuerName>EULC</IssuerName>
        <RecordStructure>
                <Name Size="15" Specifications="(a-zA-Z)+|(a-zA-Z0-9)+"></Name>
                <ID Size="10" Specifications="(0-9)+"></ID>
                <Title>Student</Title>
                <Department>Alexandria University </Department>
        </RecordStructure>
        <ValidityInterval>
        <NumberofLogins>10</NumberofLogins>
        </ValidityInterval>
</Credentials>
</Identifier>
```

Figure 22. Shows how an Identifier for a member of a community could be defined

```
<Community name= “”>
<Kerberos realm= ‘LOCALNET” hostname= “eulc”>
        <KerberosPassword lifetime= “60h”>EULC</IssuerName>
</Kerberos>
<X.509 CA= “privateCA”>
</X.509>
</Community>
```

Figure 23. Using Kerberos and X.509 to define Authentications

Figure 23 shows how X.509 could be used to provide authentication by specifying a private Certificate Authority and Kerberos.

Chapter 5. Summary and Conclusions

5.1 Conclusions

The security requirements of a digital library can be defined using 5S. For each of the streams, structures, spaces, scenarios, and societies models the detailed security requirements are described. It is possible to describe these security requirements and their configuration by using the 5SL. The extended 5SL gives DL designers an XML schema to conform to when describing the DL security requirements. The XML schema was validated by using an XML editior (EditiX). The extended 5SL is a superset of the basic 5SL and is not intended to replace the original 5SL model.

5.2 Future work

The 5S framework as extended above yields a series of XML serializations that describe the security requirement definitions. These XML serializations can be fed into a DL generator along with a component pool to generate a tailored digital library. Though this work has not covered the details of DL generation, this is an additional useful extension to consider.

Another extension is the design of a graphical tool that would allow DL designers to easily generate the XML document that conforms to the extended 5SL schema.

An important extension is to define the attack and trust models using 5SL not only for attackers outside the DL, but to also consider attackers from within the digital library, e.g., a rogue librarian who may attempt to learn user's transaction records.

The security issues discussed are about digital libraries that are not distributed. An extension of the work is to explore the security issues that affect distributed digital libraries and the changes that would apply to the extended 5SL if the attack/trust model were viewed in cloud computing settings. Considering more security issues relating to distributed digital libraries, it would be important to consider the policies that can be used to specify restrictions in the information sharing chain, e.g., library A shares copies of digital resources with library B, but wants to apply some restrictions, which may be due to DRM regulations. In this scenario, the policies and enforcement may need to be modified. The definition of policies in the extended 5SL would need to be modified so that policies and derivative policies could be described perhaps by having an inheritance mechanism.

References

1. *ACM*. [cited: 6/3/2011]; Available from: http://portal.acm.org/.
2. *Crisis, Tragedy and Recovery Network (CTRnet)*. [cited: 6/3/2011]; Available from: www.ctrnet.net.
3. *Digital Library of Canada*. [cited: 6/3/2011]; Available from: http://www.nlc-bnc.ca/index-e.html.
4. *EditiX*. [cited: 6/3/2011]; Available from: http://free.editix.com/index.html.
5. *Egyptian Univeristies Library Consortium*. [cited: 6/3/2011]; Available from: http://www.eulc.edu.eg/eulc/Libraries/start.aspx?fn=ChangeLang&DefaultLang=En&Applang=E&ScopeID=1.
6. *EULER*. [cited: 6/3/2011]; Available from: http://www.emis.de/projects/EULER/.
7. *National Gallery of Spoken Word*. [cited: 6/3/2011]; Available from: http://www.ngsw.org/.
8. *NDLTD*. [cited: 6/3/2011]; Available from: www.ndltd.org.
9. *PayPal*. [cited: 6/3/2011]; Available from: https://www.paypal.com/cy.
10. *The Digital Morphology*. [cited: 6/3/2011]; Available from: http://www.digimorph.org/.
11. *The Stanford Encyclopedia of Philosophy*. [cited: 6/3/2011]; Available from: http://plato.stanford.edu/about.html.
12. *THOMAS*. [cited: 6/3/2011]; Available from: http://thomas.loc.gov/home/thomas.php.
13. *UIML*. [cited: 6/3/2011]; Available from: http://www.uiml.org/index.php.
14. Adam, N.R., et al., *A Content-Based Authorization Model for Digital Libraries.* IEEE Transactions on Knowledge and Data Engineering, 2002. **14**(2): p. 296 - 315.
15. *Key concepts in the Architecture of the Digital Library*. [cited; Available from: http://www.dlib.org/dlib/July95/07arms.html.
16. Bacon, J., K. Moody, and W. Yao, *Access control and trust in the use of widely distributed services.* Software- Practice & Experience (Middleware), 2003. **33**(4): p. 375 - 394.
17. Buchanan, G. *FRBR: enriching and integrating digital libraries*. in 6th ACM/IEEE-CS joint conference on Digital libraries (JCDL '06). 2006 of Conference. Chapel Hill, NC, USA: ACM.
18. Candela, L., et al. *The DELOS Digital Library Reference Model*. 2007 http://www.delos.info/index.php?option=com_content&task=view&id=345
19. Ching, N., V. Jones, and M. Winslett. *Authorization in the Digital Library: Secure Access to Services across Enterprise Boundaries*. in Third International Forum on Research and Technology Advances in Digital Libraries. 1996 of Conference. Washington, DC: IEEE.
20. Chowdhury, G. and S. Chowdhury, *Introduction to Digital Libraries*. 2003: Facet Publishing.
21. *Speed Comparison of Popular Crypto Algorithms*. [cited; Available from: http://www.cryptopp.com/benchmarks.html.

22. Ferrari, E., et al., *An Authorization System for Digital Libraries.* The VLDB Journal, 2002. **11**(1): p. 58 - 67.
23. Fetscherin, M. and M. Schmid. *Comparing the Usage of Digital Rights Management Systems in the music, film, and print industry*. in Proceedings of the 5th International Conference on Electronic Commerce. 2003 of Conference. Pittsburgh, Pennsylvania: ACM.
24. Fetscherin, M. and M. Schmid. *Comparing the Usage of DRM systems in the Music, Film and Print Industry*. in ICEC. 2003 of Conference. Pittsbugh, USA: ACM.
25. Gladney, H.M., *Access Control for Large Collections.* ACM Transactions on Information Systems (TOIS), 1997. **15**(2): p. 154 - 194.
26. Gonçalves, M.A. and E.A. Fox. *5SL – A Language for Declarative Specification and Generation of Digital Libraries*. in *JCDL*. 2002. Portland, Oregon, USA: ACM.
27. Gonçalves, M.A., et al., *Streams, Structures, Spaces, Scenarios, Societies (5S): A Formal Model for Digital Libraries.* ACM Transactions on Information Systems (TOIS), 2004. **22**(2).
28. Gorton, D., *Practical Digital Library Generation into DSpace with the 5S Framework*, Masters Thesis in *Computer Science and Applications*. 2007, Virginia Tech:Blacksburg. p.109 http://scholar.lib.vt.edu/theses/available/etd-04252007-161736/unrestricted/dgorton_thesis_final.pdf
29. Johnson, N.F. and S. Jajodia, *Steganography: Seeing the Unseen.* IEEE Computer, 1998: p. 26-34.
30. Kohl, U., J. Lotspiech, and S. Nusser. *Security for the Digital Library - Protecting Documents Rather than Channels*. in Ninth Workshop on Database and Expert Systems. 1998 of Conference. Vienna, Austria.
31. Lagoze, C. and J.R. Davis, *Dienst: An Architecture for Distributed Document Libraries.* Communications of the ACM, 1995. **38**(4): p. 1.
32. Mintzer, F., J. Lotspiech, and N. Morimoto (1997) *Safeguarding Digital Library Contents and Users*. D-Lib Magazine **3** (7/8). http://www.dlib.org/dlib/december97/ibm/12lotspiech.html
33. Nadeem, A. and M.Y. Javed. *A Performance Comparison of Data Encryption Algorithms*. in 1st International Conference on Information and Communication Technologies. 2005 of Conference. Karachi, Pakistan: IEEE.
34. Nagaraj, S.V. *Access Control in Distributed Object Systems: Problems with Access Control Lists*. in Tenth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises. 2001 of Conference. Cambridge, MA: IEEE.
35. Nelson, M.L. and K. Maly, *Buckets: Smart Objects for Digital Libraries*, in Communications of the ACM, 2001. **44**(Issue).
36. Neuman, C. and T. Ts'o, *Kerberos: An Authentication Service for Computer Networks*, in IEEE Communications Magazine, 1994. **32**(Issue): p. 33 - 38.
37. Sandhu, R.S. and P. Samarati, *Access Control: Principle and Practice*, in IEEE Communications Magazine, 1994. **32**(Issue): p. 40 - 48.

38. Schonberg, D. and D. Kirovski. *Fingerprinting and Forensic Analysis of Mutimedia*. in Media Management. 2004 of Conference. New York, USA: ACM.

39. Stallings, W., *Cryptography and Network Security*. 4 ed. 2006: Pearson Prentice Hall.

40. Tolone, W., et al., *Access Control in Collaborative Systems.* ACM Computing Surveys, 2005. **37**(1): p. 29 - 41.

41. Tyrväinen, P. (2005) *Concepts and a Design for Fair Use and Privacy in DRM*. D-Lib Magazine **11** (2). http://www.dlib.org/dlib/february05/tyrvainen/02tyrvainen.html

42. Winslett, M., et al. *Assuring Security and Privacy for Digital Library Transactions on the Web: Client and Server Security Policies*. in IEEE International Forum on Research and Technology Advances in Digital Libraries (ADL). 1997 of Conference. Washington, DC: IEEE.

Appendices

Appendix A: XML Schema for Societies Model

```
<?xml version="1.0" encoding="UTF-8"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="secure_societies">

<xs:complexType>

<xs:sequence>

<xs:element name="Authentication">

<xs:complexType>

<xs:sequence>

<xs:element name="Type">

<xs:complexType>

<xs:sequence>

<xs:element name="Community">

<xs:complexType>

<xs:sequence>

<xs:element name="Service" type="xs:string"/>

<xs:element name="Identifier">

<xs:complexType>

<xs:sequence>

<xs:element name="Username_Password">

<xs:complexType>

<xs:sequence>

<xs:element name="Username">
```

```
<xs:complexType>
<xs:sequence>
<xs:element name="Name" type="xs:string"/>
</xs:sequence>
<xs:attribute name="Size" type="xs:integer" use="required"/>
</xs:complexType>
</xs:element>
<xs:element name="Password">
<xs:complexType>
<xs:sequence>
<xs:element maxOccurs="unbounded" name="Constraints" type="xs:string"/>
</xs:sequence>
<xs:attribute name="Size" type="xs:integer"/>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="Credentials">
<xs:complexType>
<xs:sequence>
<xs:element name="IssuerName" type="xs:string"/>
<xs:element name="RecordStructure">
<xs:complexType>
<xs:sequence>
```

```
<xs:element name="Name">
<xs:complexType>
<xs:attribute name="Size" type="xs:integer" use="required"/>
<xs:attribute name="Specifications" type="xs:string" use="required"/>
</xs:complexType>
</xs:element>
<xs:element name="ID">
<xs:complexType>
<xs:attribute name="Size" type="xs:integer" use="required"/>
<xs:attribute name="Specifications" type="xs:string" use="required"/>
</xs:complexType>
</xs:element>
<xs:element name="Title" type="xs:string"/>
<xs:element name="Department" type="xs:string"/>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="ValidityInterval">
<xs:complexType>
<xs:sequence>
<xs:choice>
<xs:element name="NumberofLogins">
<xs:simpleType>
<xs:restriction base="xs:integer">
<xs:minInclusive value="0"/>
</xs:restriction>
```

```
</xs:simpleType>
</xs:element>
<xs:element name="ExpirationDate" type="xs:date"/>
</xs:choice>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="DigitalSignature">
<xs:complexType>
<xs:sequence>
<xs:element name="HashFunction">
<xs:complexType>
<xs:attribute name="Algorithm" use="required"/>
<xs:attribute name="Value" use="required"/>
</xs:complexType>
</xs:element>
<xs:element name="MAC">
<xs:complexType>
<xs:attribute name="Algorithm" use="required"/>
<xs:attribute name="Value" use="required"/>
```

```
<xs:attribute name="OutputSize" use="required"/>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element maxOccurs="1" name="Kerberos">
<xs:complexType>
<xs:sequence>
<xs:element name="KerberosPassword">
<xs:complexType>
<xs:attribute name="Lifetime">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:pattern value="(0-9)*s"/>
<xs:pattern value="(0-9)*m"/>
<xs:pattern value="(0-9)*h"/>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
</xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute default="LOCALNET" name="realm">
<xs:simpleType>
<xs:restriction base="xs:string">
```

```
<xs:pattern value="[A-Z]+"/>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
<xs:attribute name="hostname">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:pattern value="[a-z]+"/>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
</xs:complexType>
</xs:element>
<xs:element name="X.509">
<xs:complexType>
<xs:attribute default="private" name="CA">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:enumeration value="private"/>
<xs:enumeration value="trustedCA"/>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
</xs:complexType>
</xs:element>
</xs:sequence>
```

```
<xs:attribute name="name" type="xs:string" use="required"/>

<xs:attribute name="id" type="xs:integer" use="required"/>

</xs:complexType>

</xs:element>

</xs:sequence>

<xs:attribute name="value">

<xs:simpleType>

<xs:restriction base="xs:string">

<xs:enumeration value="Actor"/>

<xs:enumeration value="Manager"/>

</xs:restriction>

</xs:simpleType>

</xs:attribute>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:schema>
```

Appendix B: Secure Structures XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="secure_structures">
<xs:complexType>
<xs:sequence>
<xs:element name="Metadata">
<xs:complexType>
<xs:attribute name="EncryptionAlgorithm" type="xs:string">
</xs:attribute>
<xs:attribute name="Key" type="xs:string">
</xs:attribute>
<xs:attribute name="IntegrityAlgorithm" type="xs:string">
</xs:attribute>
<xs:attribute name="OutputSize">
<xs:simpleType>
<xs:restriction base="xs:integer">
<xs:minInclusive value="16">
</xs:minInclusive>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
</xs:complexType>
</xs:element>
```

```
<xs:element name="Document">
<xs:complexType>
<xs:sequence>
<xs:element name="DRM">
<xs:complexType>
<xs:sequence>
<xs:element name="DigitalSignature">
<xs:complexType>
<xs:sequence>
<xs:choice>
<xs:element name="HashFunction">
<xs:complexType>
<xs:attribute name="Algorithm" use="required">
</xs:attribute>
<xs:attribute name="Value" use="required">
</xs:attribute>
</xs:complexType>
</xs:element>
<xs:element name="MAC">
<xs:complexType>
<xs:attribute name="Algorithm" use="required">
</xs:attribute>
<xs:attribute name="Value" use="required">
</xs:attribute>
<xs:attribute name="OutputSize" use="required">
</xs:attribute>
```

```xml
</xs:complexType>
</xs:element>
<xs:element name="PrivateKey" type="xs:integer">
</xs:element>
</xs:choice>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="Watermarking" type="xs:string" use="optional">
</xs:attribute>
</xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="type">
</xs:attribute>
</xs:complexType>
</xs:element>
<xs:element name="AccessControl">
<xs:complexType>
<xs:sequence>
<xs:choice>
<xs:element name="CapabilityList">
<xs:complexType>
<xs:sequence>
<xs:element minOccurs="1" name="Identifier">
```

```
<xs:complexType>
<xs:sequence>
<xs:element maxOccurs="1" minOccurs="1" name="Document">
<xs:complexType>
<xs:attribute name="rights" use="required">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:enumeration value="owner">
</xs:enumeration>
<xs:enumeration value="read">
</xs:enumeration>
<xs:enumeration value="write">
</xs:enumeration>
<xs:enumeration value="read_write">
</xs:enumeration>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
```

```xml
<xs:element name="AccessControlList">
<xs:complexType>
<xs:sequence>
<xs:element minOccurs="1" name="Document">
<xs:complexType>
<xs:sequence>
<xs:element maxOccurs="1" minOccurs="1" name="Identifier">
<xs:complexType>
<xs:attribute name="rights" use="required">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:enumeration value="owner">
</xs:enumeration>
<xs:enumeration value="read">
</xs:enumeration>
<xs:enumeration value="write">
</xs:enumeration>
<xs:enumeration value="read_write">
</xs:enumeration>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
```

```
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="RBAC">
<xs:complexType>
<xs:sequence>
<xs:element minOccurs="1" name="Subject">
</xs:element>
<xs:element minOccurs="1" name="Role">
</xs:element>
<xs:element minOccurs="1" name="Permissions">
</xs:element>
<xs:element minOccurs="1" name="Session">
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="TaskBAC">
<xs:complexType>
<xs:sequence>
<xs:element minOccurs="1" name="Subject">
</xs:element>
<xs:element minOccurs="1" name="Task">
</xs:element>
<xs:element minOccurs="1" name="Permissions">
```

```
</xs:element>
<xs:element minOccurs="1" name="Session">
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="TeamBAC">
<xs:complexType>
<xs:sequence>
<xs:element minOccurs="1" name="Team">
<xs:complexType>
<xs:sequence>
<xs:element minOccurs="1" name="Subject">
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element minOccurs="1" name="Task">
</xs:element>
<xs:element minOccurs="1" name="Permissions">
</xs:element>
<xs:element minOccurs="1" name="Session">
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
```

```
<xs:element name="ContentBAC">
<xs:complexType>
<xs:sequence>
<xs:element name="GLINModel">
<xs:complexType>
<xs:sequence>
<xs:element name="Credentials">
</xs:element>
<xs:element name="Concept">
</xs:element>
<xs:element name="Privilege">
</xs:element>
<xs:element name="Sign">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:enumeration value="+">
</xs:enumeration>
<xs:enumeration value="-">
</xs:enumeration>
</xs:restriction>
</xs:simpleType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
```

```
</xs:complexType>

</xs:element>

</xs:choice>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:schema>
```