Modeling and Validating the Clinical Information Systems Policy Using Alloy

Ramzi A. Haraty and Mirna Naous

Department of Computer Science and Mathematics
Lebanese American University
Beirut, Lebanon
rharaty@lau.edu.lb

Abstract. Information systems security defines three properties of information: confidentiality, integrity, and availability. These characteristics remain major concerns throughout the commercial and military industry. In this work, we focus on the integrity aspect of commercial security applications by exploring the nature and scope of the famous integrity policy - the Clinical Information Systems Policy. We model it and check its consistency using the Alloy Analyzer.

Keywords: Clinical Information Systems Model, Consistency, and Integrity.

1 Introduction

The goal of information systems is to control or manage the access of subjects (users, processes) to objects (data, programs). This control is governed by a set of rules and objectives called a security policy. Data integrity is defined as "the quality, correctness, authenticity, and accuracy of information stored within an information system" [1]. Systems integrity is the successful and correct operation of information resources. Integrity models are used to describe what needs to be done to enforce the information integrity policies. There are three goals of integrity:

- Prevent unauthorized modifications,
- Maintain internal and external consistency, and
- Prevent authorized but improper modifications.

Before developing a system, one needs to describe formally its components and the relationships between them by building a model. The model needs to be analyzed and checked to figure out possible bugs and problems. Thus, formalizing integrity security models helps designers to build a consistent system that meets its requirements and respects the three goals of integrity. This objective can be achieved through the Alloy language and its analyzer.

Alloy is a structural modeling language for software design. It is based on first order logic that makes use of variables, quantifiers and predicates (Boolean functions) [2]. Alloy, developed at MIT, is mainly used to analyze object models. It translates constraints to Boolean formulas (predicates) and then validates them using the Alloy

G. Huang et al. (Eds.): HIS 2013, LNCS 7798, pp. 1–17, 2013.

Analyzer by checking code for conformance to a specification [3]. Alloy is used in modeling policies, security models and applications, including name servers, network configuration protocols, access control, telephony, scheduling, document structuring, and cryptography. Alloy's approach demonstrates that it is possible to establish a framework for formally representing a program implementation and for formalizing the security rules defined by a security policy, enabling the verification of that program representation for adherence to the security policy.

There are several policies applied by systems for achieving and maintaining information integrity. In this paper, we focus on the Clinical Information Systems Security Policy [4] and to show how it can be checked for consistency or inconsistency using the Alloy language and the Alloy Analyze.

The remainder of this paper is organized as follows: Section 2 provides the literature review. Section 3 discusses the Clinical Information System Security Model, and section 4 concludes the paper.

2 Literature Review

Hassan and Logrippo [5] proposed a method to detect inconsistencies of multiple security policies mixed together in one system and to report the inconsistencies at the time when the secrecy system is designed. The method starts by formalizing the models and their security policies. The mixed model is checked for inconsistencies before real implementation. Inconsistency in a mixed model is due to the fact that the used models are incompatible and cannot be mixed. The authors demonstrated their method by mixing the Bell-LaPadula model [6] with the Role Based Access Control (RBAC) model [7] in addition to the Separation of Concerns model [8]. Two modes are used for combination of models: mixed and hybrid. The system that adopts mixed-mode secrecy implements policies following any parent model. Mixed models combine the parent model's policies and their properties. On the other hand, hybrid models inherit properties from parent models or include other properties not available in the parent. In a mixed secrecy model there is always inconsistency. The authors addressed two types of inconsistencies: model, and system. Model inconsistency is the logical conflict between properties and meta policies. System inconsistency is the conflict between user policies or between user policies and meta policies.

Zao et al. [9] developed the RBAC schema debugger. The debugger uses a constraint analyzer built into the lightweight modeling system to search for inconsistencies between the mappings among users, roles, objects, permissions and the constraints in a RBAC schema. The debugger was demonstrated in specifying roles and permissions according and verifying consistencies between user roles and role permissions and verifying the algebraic properties of the RBAC schema.

Hassan et al. [10] presented a mechanism to validate access control policy. The authors were mainly interested in higher level languages where access control rules can be specified in terms that are directly related to the roles and purposes of users. They discussed a paradigm more general than RBAC in the sense that the RBAC can be expressed in it.

Shaffer [11] described a security Domain Model (DM) for conducting static analysis of programs to identify illicit information flows, such as control dependency flaws and covert channel vulnerabilities. The model includes a formal definition for trusted subjects, which are granted privileges to perform system operations that require mandatory access control policy mechanisms imposed on normal subjects, but are trusted not to degrade system security. The DM defines the concepts of program state, information flow and security policy rules, and specifies the behavior of a target program.

Misic and Misic [12] addressed the networking and security architecture of healthcare information system. This system includes patient sensor networks, wireless local area networks belonging to organizational units at different levels of hierarchy, and the central medical database that holds the results of patient examinations and other relevant medical records. In order to protect the integrity and privacy of medical data, they proposed feasible enforcement mechanisms over the wireless hop.

Haraty and Boss [13] showed how secrecy policies can be checked for consistency and inconsistency by modeling the Chinese Wall Model [14], Biba Integrity Model [15], Lipner Model [16] and the Class Security Model [17-18]. The authors used the Alloy formal language to define these models and the Alloy Analyzer to validate their consistency. In their work, they listed the ordered security classes (Top Secret, Secret, Confidential, and Unclassified) and their compartments (Nuclear, Technical, and Biological) and defined those using signatures. Then, the possible combinations and the relationships between classes and compartments were specified. Facts were used to set the model constraints and to prove that the model is consistent. In the Biba Integrity model, the authors listed the subject security clearance and the object security classes and then modeled the constraints of how subjects can read/write objects based on “NoReadDown” and “NoWriteUp” properties.

3 Clinical Information Systems Security Model

Security of medical records is a very important issue in clinical information systems. Security policies have to be carefully designed in order to limit the number of users that can access patient records and to control the operations that may be applied to the records themselves. Thus, it is very critical to protect confidentiality of records and their data integrity. Anderson [4] developed a policy for clinical information systems that combine confidentiality and integrity to assure patient privacy and record validity.

The policy assumes that personal health information concerns one individual at a time and is contained in a medical record. As stated in [4], the policy “principles are derived from the medical ethics of several medical societies, and from the experience and advice of practicing clinicians”. It is expressed based on two sets of principles: The Access Principles set deals with confidentiality and the creation, deletion, confinement, aggregation, and Enforcements Principles set handles the integrity:

- Confidentiality principles:
  - Access Principle 1: Each medical record has an access control list containing the individuals and groups who are able to read and append information to the record. The system must restrict access to record to those identified on the list and deny access to anyone else.
  - Access Principle 2: One of the clinicians (responsible clinician) on the access control list must have the right to add other clinicians to the access control list.
  - Access Principle 3: The responsible clinician must notify the patient of the names on the access control list whenever the patient's medical record is opened.
  - Access Principle 4: The name of clinician, the date and the time of the access of medical record must be recorded. Similar information must be kept for deletions.

    Note that auditors cannot access original medical records; instead copies are provided for this purpose to prevent auditors from changing the original records. Medical records can be read and altered by the clinicians by whom the patients have consented to be treated.

- Integrity principles:
  - Creation Principle: A clinician may open a record, with the clinician and the patient on the access control list. If the record is opened as a result of referral, the referring clinician must also be on the access control list.
  - Deletion Principle: Clinical information cannot be deleted from a medical record until the appropriate time has passed.
  - Confinement Principle: Information from one medical record may be appended to a different medical record if and only if the access control list of the second record is a subset of the access control list of the first.
  - Aggregation Principle: Measures for preventing the aggregation of patient data must be effective. In particular, a patient must be notified if anyone is to be added to the access control list for the patients' record and if that person has access to a large number of medical records.
  - Enforcement Principle: Any computer system that handles medical records must have a subsystem that enforces the preceding principles. The effectiveness of this enforcement must be subject to evaluation by independent auditors.

Therefore, based on Clinical Information Systems Security Policy highlighted rules, and assuming the existence/respect of the non-highlighted rules, the system to be implemented must ensure confidentiality by maintaining an access control lists containing users able to read and append original or copied records and grant access to other users. Moreover, the system must maintain data integrity by allowing the appending of information from one record to another if and only if the access control list of the second record is a subset of the access control list of the first. Responsible clinicians must notify patients regarding the names of clinicians on the access control list.

3.1 Clinical Information Systems Security Policy Implementation

In order to implement the model, a clinical information system is used as an example to demonstrate consistency with respect to Clinical Information Systems Security Policy. The system users can access medical records if their names are contained in access control lists attached to the medical records - one list per record. Medical records cover the personal health information of individuals. There are two versions of medical records: the original version and the copy version. Access to original copies is restricted only to clinicians and patients defined in the access control lists. Auditors are allowed to alter only the copy version of medical records. Regarding the framework used for implementation, the Alloy language and the Alloy Analyzer (based on its available features and its ability to check system consistency and to generate instances) were used for implementation.

Table 1 lists system users. There are three types of users: patients, clinicians and auditors. However, and according to Access Principle 2, the system has an additional user named the responsible clinician who has the right to add other clinicians to an

Table 1. Clinical Information System Users

Users	Description
Ptns	Patients
Clns	Clinicians
ClnsR	Responsible clinicians
Adts	Adults

Table 2. Clinical Information System Patients

Patients	Description
P1	Patient 1
P2	Patient 2

Table 3. Clinical Information System Clinicians

Clinicians	Description
C1	Clinician 1
C2	Clinician 2
C3	Clinician 3
C4	Clinician 4
C5	Clinician 5

Table 4. Clinical Information System Responsible Clinicians

Responsible clinicians	Description
CR1	Responsible Clinician 1
CR2	Responsible Clinician 2

Table 5. Clinical Information System Auditors

Auditors	Description
A1	Auditor 1
A2	Auditor 2
A3	Auditor 3

access control list. Tables 2, 3, 4 and 5 list available patients, clinicians, responsible clinicians, and auditors, respectively. Accordingly, the system has two patients (P1 and P2), five clinicians (C1, C2, C3, C4, and C5), two responsible clinicians (CR1 and CR2), and three auditors (A1, A2, and A3).

Table 6 shows two versions of medical records: the original version MRO and the copy version MRC. The system covers three medical records (mc1, mc2 and mc3) defined in table 7 and their respective copies (mo1, mo2 and mo3) shown in table 8. Each medical record contains personal health information that concerns one individual at a time.

Table 6. Clinical Information System Medical Records Versions

Medical records	Description
MRC	Medical Records Copies
MRO	Medical Records Original

Table 7. Clinical Information System Medical Records (Original)

Medical records	Description
mc1	Medical record 1
mc2	Medical record 2
mc3	Medical record 3

Table 8. Clinical Information System Medical Records (Copy)

Medical records	Description
mo1	Medical record 1
mo2	Medical record 2
mo3	Medical record 3

Table 9 determines the different access control lists used by the clinical information system. There are two types of lists: lists accessible by patients and clinicians by whom the patients have consented to be treated and they are labeled lst1, lst2 and lst3, and lists accessible by the auditors who are given access to copies of patients medical records and they are labeled lstA1, lstA2 and lstA3.

Table 9. Clinical Information System Access Control Lists

Access control lists	Description
lst1	Access control list 1 covering one patient and clinicians.
lst2	Access control list 2 covering one patient and clinicians.
lst3	Access control list 3 covering one patient and clinicians.
lstA1	Auditor Access control list covering the auditors.

Table 10 shows the domain of system access control lists. For instance, lst1 is the access control list of the medical record mo1 that concerns patient P1. In addition to P1, the clinicians C1, C4, C5 and CR1 are in lst1 and are allowed to access mo1, whereas CR1 is the responsible clinician who can add other clinicians to lst1. Similarly, lst2 is the access control list of mo2. It contains P2, C2, C3, C4 and CR2. However, list lst3 concerns mo3 of the patient P2; it contains C2 and CR2. Note that mo2 and mo3 are two different medical records for the same patient P2 assuming that the system can open a new record for a patient regardless if he/she has a previous medical record in the system, with the possibility to append the information from an existing record to the new record as per the Confinement Principle, and then discard the old record information after a certain period of time as per the Deletion Principle.

Table 11 displays a matrix showing the system relationships among objects in the clinical information system. The objects and relations are used to model and validate the clinical information system security policy. In this table and as per the Confinement Principle, the information of mo2 can be appended to mo3 since lst3 is a subset of lst2. The auditors in the system are used to enforce the system principles (Enforcement Principle). They are given access to the copies of the system medical records for checking purposes. Auditors can update these copies without any constraints because their alteration will not affect system integrity. Additionally,

patients are allowed to read their medical records only, whereas clinicians are given read and append access to the medical records according to the access control lists. Moreover, responsible clinicians can add new clinicians to access control lists. They must notify patients about the names of clinicians eligible to read and append their records (i.e., clinicians available in the access control list) and also whenever a clinician is to be added to the medical record access control list (Aggregation Principle).

Table 10. Clinical Information System Access Control Lists Domain

Access control list	Medical record	Patient	Clinicians
lst1	mo1	P1	C1, C4, C5, CR1
lst2	mo2	P2	C2, C3, C4, CR2
lst3	mo3	P2	C2, CR2
lstA1	mc1, mc2, mc3	P1, P2	A1, A2, A3

Table 11. Clinical Information System Clinical System Relations

Object/ Relation	accessed by	read	append	notify	infoappendedto	addnew Clnsto
mo1	lst1				-	
mo2	lst2				mo3	
mo3	lst3				-	
mc1	lstA1					
mc2	lstA1					
mc3	lstA1					
P1		mo1	-	-		
P2		mo2, mo3	-	-		
C1		mo1	mo1	-		-
C2		mo2	mo2	-		-
C3		mo2	mo2	-		-
C4		mo1, mo2	mo1, mo2	-		-
C5		mo1	mo1	-		-
CR1		mo1	mo1	P1		lst1
CR2		mo2, mo3	mo2, mo3	P2		lst2, lst3

The clinical information system can now be represented using the Alloy language. First, the system main entities represented in the above tables are declared as shown in the following sections of code:

- Section 1 declares the main entities of the clinical information system:
  - Users: users set covering all system users.
  - HcUsrs: health care users set as part of users set.
  - Ptns: set of patients as part of users set.
  - Clns: set of clinicians as part of HcUsrs set.
  - ClnsR: set of responsible clinicians as part of Clns set.
  - Adts: set of auditors as part of users set.
  - MRs: set of all medical records defined in the system.
  - MRO: set of original medical records as part of MRs set.
  - MRC: set of medical records copies as part of MRs set.
  - ACL: set of access control lists attached to the original medical records.
  - AACL: set of auditor access control lists attached to the copy version of medical records.

```
//declaration section
abstract sig Users{}//all users
sig HcUsrs extends Users{}//health care users

sig Ptns extends Users{read:one MRO}//Patients
sig Clns extends HcUsrs{read:some MRO,append:some MRO}//clinicians
sig ClnsR extends Clns{notify:some Ptns, addnewClnsto:some ACL}//responsible clinicians
sig Adts extends Users{read:some MRC,append:some MRC}//auditors

abstract sig MRs{} //all medical records
sig MRO extends MRs{accessedby:one ACL}//Medical Records original
sig MRC extends MRs{accessedby:one AACL}//Medical Records Copy

abstract sig ACL{contains: some Ptns, Contains: some Clns} //Access control lists
abstract sig AACL{contains: some Adts}//auditors access control list
```

Section 1. Clinical Information System Entities

Section 1 also defines the relations among entities. The relation "read" between Ptns and MRO means that patients can read information from their original medical records. Similarly, "read" and "append" relations between Clns and HcUsrs means that clinicians can read and append patients medical records MRO. The responsible clinicians ClnsR using the "addnewClnsto" relation are responsible to add new clinicians to access control lists and "notify" patients accordingly. Auditors are part of system users. They are allowed to alter the copy version of system medical records. Medical records set – MRO - is "accessedby" access control lists set while MRC set is accessed by auditors access control list AACL. The access control list ACL "contains" patients and clinicians; however, AACL contains only auditors.

- Section 2 defines the instances of the sets declared in section 1. Accordingly, there are three original medical records instances mo1, mo2 and mo3. Following the Confinement Principle of the Clinical Information Systems Policy, original medical records information can be appended to other medical records. Thus, the relation "infoappendedto" is defined to serve this purpose. In addition to the original medical records, the system defines three instances of copy version medical records: mc1, mc2 and mc3. Moreover, lst1, lst2 and lst3 are three instances belonging to ACL and lstA1 is one instance belonging to the AACL set. Also the system declares five clinicians: C1, C12, C3, C4 and C5, two patients: P1 and P2, three auditors: A1, A2 and A3, and two responsible clinicians: CR1 and CR2.

```
//assume a medical record concerns one individual at a time
one sig mo1,mo2,mo3 extends MRO{Infoappendedto: one MRO} //medical orginal record instances
one sig mc1,mc2,mc3 extends MRC{ } //3 different copies of medical records

one sig lst1,lst2,lst3 extends ACL {} //access control list instances
one sig lstA1 extends AACL{} //auditors access control list instances

one sig C1,C2,C3,C4,C5 extends Clns{}//clinicians instances
one sig CR1,CR2 extends ClnsR{}//responsible clinicians instances

one sig P1,P2 extends Ptns{} //2 patients
one sig A1,A2,A3 extends Adts{}// 3 auditors
```

Section 2. Clinical Information System Instances

```
fact
{

//define content of lst1: C1, C4,C5 and CR1
lst1.contains=P1
lst1.Contains=C1+C4+C5+CR1

//define content of lst2: P2 and C2, C3, C4 and CR2
lst2.contains=P2
lst2.Contains=C2+C3+C4+CR2

//define content of lst3: P2, C2 and CR2 -> lst3 is subset of lst2
lst3.contains=P2
lst3.Contains=C2+CR2

//copy of medical record is accessed by ONE AND ONLY ONE auditors access control list only
//lstA1 contains A1, A2 and A3
lstA1.contains=A1 +A2+A3
```

Section 3. Clinical Information System Constraints (part 1)

- Section 3 highlights the beginning of the fact{} procedure where the system constraints are set. The first two lines specify explicitly the contents of the access control list lst1. List lst1 is accessed by patient P1 and clinicians C1, C4, C5 and CR1 where CR1 is the responsible clinician for lst1. List lst2 contains P2, C2, C3, C4 and CR2, and lst3 contains P2, C2 and CR2. lst3 is a subset of lst2 since P2, C2 and CR2 are also part of lst2. The last line of this section states that the auditor access control list lstA1 contains A1, A2 and A3. The symbol "+" is used for union.
- Section 4 determines that the medical records mo1, mo2 and mo3 are accessed by lst1, lst2 and lst3, respectively.

```
//orginal medical record is accessed by ONE AND ONLY ONE access control list only
mo1.accessedby=lst1
mo2.accessedby=lst2
mo3.accessedby=lst3
```

Section 4. Clinical System Constraint (part 2)

- Section 5 shows the constraints related to the read and append relations. It determines records that can be read or appended by clinicians. This section makes use of negation when there are multiple options and explicit declaration when there is only one option. For instance, patient P1 can read mo1 only. However, P2 has two records in the system mo2 and mo3. In this case, the second line states that P2 is not able to read mo1; thus, he/she can read either mo2 or mo3 at a time. The same applies to the subsequent lines of code: clinician C1 is a member of lst1 only. For this reason C1 can read/append mo1 only. However, C2 is part of lst2 and lst3. C2 cannot read/append mo1 but he/she has the option to access mo2 or mo3. Regarding the responsible clinicians CR1 and CR2, they have additional roles in the system other than reading/appending medical records. CR1 is able to add new clinician to the access control list lst1 and notify C1, accordingly. The last 4 lines of code in the section determine that CR1 notifies P1 when he/she adds new clinician to lst1 and CR2 notifies patients other than P1 (i.e., P2) upon adding new clinician to a list different than lst1 (i.e., lst2 or lst3).

Note that the section 5 does not cover any constraints regarding the copy version of medical records accessed by the auditors since their alteration will not affect system integrity.

```
//each patient can read his/her record only.
P1.read=mo1
P2.read!=mo1

C1.read=mo1
C1.append=mo1

C2.read!=mo1
C2.append!=mo1

C3.read=mo2
C3.append=mo2

C4.read=mo2
C4.append=mo2

C5.read=mo1
C5.append=mo1

CR1.read=mo1
CR1.append=mo1

CR2.read!=mo1
CR2.append!=mo1

CR1.notify=P1
CR2.notify!=P1

CR1.addnewClnsto=lst1
CR2.addnewClnsto!=lst1
```

Section 5. Clinical Information System Constraints (part 3)

3.2 Clinical Information System Security Policy and Alloy Analysis

Figure 1 displays the clinical information system meta model generated by the Alloy system. It shows multiple users of the system: auditors - Adts, health care users - HcUsrs, patients – Ptns, and clinicians – Clns, as part of health care users. Additionally, the figure displays the system access control lists – ACL, and auditors' access control lists – AACL, and their instances. Moreover, the model shows two types of medical records: original version of medical records (MRO) and the copy version (MRC) and their instances. However, the meta model does not show any constraints. Executing the system using the Alloy Analyzer will generate instances based on defined constraints.

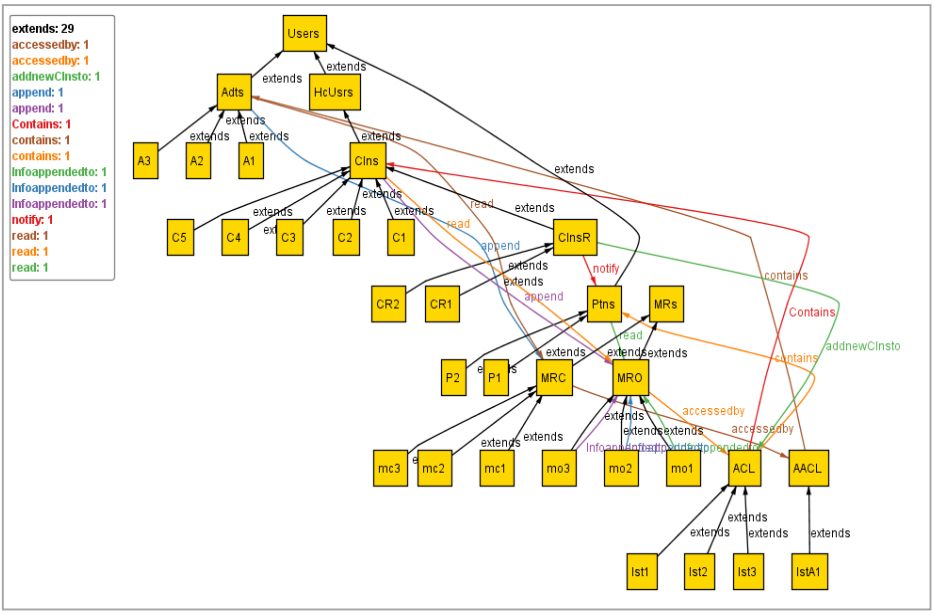

Fig. 1. The Clinical Information System Meta Model

Testing system consistency is done by running the system predicates and generating possible instances then validating them. In order to test the constraints specified in the fact procedure, we need to write a predicate and run it.

- Section 6 declares an empty predicate used to test the system consistency based on the defined facts. Executing the example() will produce the output specified in figure 2. The figure shows that an "instance found" and "Predicate is consistent". It takes around 78ms to determine consistency and find an instance.

```
// run example to show consistencies/ inconsistencies
pred example(){
}
run example
```

Section 6. Clinical Information System Predicate

```
Executing "Run example"
  Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
  292 vars. 124 primary vars. 291 clauses. 78ms.
  Instance found. Predicate is consistent. 78ms.
```

Fig. 2. Clinical Information System Consistent Alloy Analyzer Output

Clicking the link "**Instance**" will show figure 3. More instances can be generated by pressing the "Next" button located at the top of the screen of figure 3.

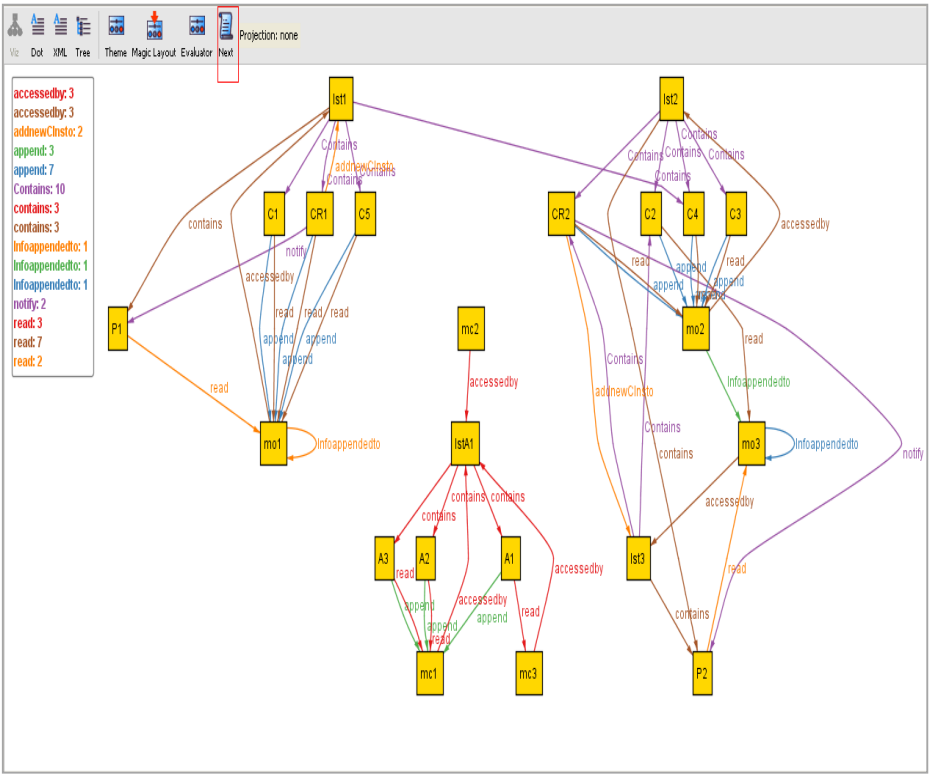

Fig. 3. Clinical Information System Instance

The generated instances demonstrate the consistency of the system as shown in tables 12- 16.

Table 12. Clinical Information System Consistency Checking (part 1)

Object/Relation	contains	consistent
lst1	P1, C1, CR1, C5, C4	Yes
lst2	CR2, C2, C4, C3, P2	Yes
lst3	P2, C2, CR2	Yes

Table 13. Clinical Information System Consistency Checking (part 2)

Object/Relation	accessedby	consistent
mo1	lst1	Yes
mo2	lst2	Yes
mo3	lst3	Yes
mc1	lstA1	Yes
mc2	lstA1	Yes
mc3	lstA1	Yes

Table 14. Clinical Information System Consistency Checking (part 3)

Object/Relation	read	append	consistent
P1	mo1	-	Yes
P2	mo3	-	Yes
C1	mo1	mo1	Yes
C2	mo3	mo2	Yes
C3	mo2	mo2	Yes
C4	mo2	mo2	Yes
C5	mo1	mo1	Yes
CR1	mo1	mo1	Yes
CR2	mo2	mo2	Yes
A1	mc3	mc1	Yes
A2	mc1	mc1	Yes
A3	mc1	mc1	Yes

Table 15. Clinical Information System Consistency Checking (part 4)

Object/Relation	infoappendedto	consistent
mo1	mo1	Yes
mo2	mo3	Yes
mo3	mo3	Yes

Table 16. Clinical Information System Consistency Checking (part 5)

Object/Relation	addnewClnsto	notify	consistent
CR2	lst3	P2	Yes
CR1	lst1	P1	Yes

However, specifying a wrong predicate such as stating that lst1 contains the patient P2 (section 7, line 2) will cause inconsistency and the result of running the example() is displayed in figure 4.

```
// run example to show inconsistencies
pred example(){

lst1.contains=P2 //inconsistent bcz lst1 contains only one patient
//P1.append=mo2 // inconsistent bcz patient cannot append medical record

}
run example
```

Section 7. Clinical Information System Inconsistent Predicate

```
Executing "Run example"
  Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
  293 vars. 124 primary vars. 294 clauses. 47ms.
  No instance found. Predicate may be inconsistent. 0ms.
```

Fig. 4. Clinical Information System Alloy Analyzer Inconsistent Output

Thus, the clinical information system adopting the Clinical Information Security Policy is a consistent system. Any misbehavior will results in an inconsistency where no instances are found.

4 Conclusion

In this paper, we presented the Clinical Information System Security Policy. We used system examples based on the defined security model. We formalized the system according to the model then checked its consistency. Since Alloy allows expressing systems as set of logical constraints in a logical language based on standard first-order logic, we used it to define the system and policy. When creating the model, we specified the system users and subjects then Alloy compiled a Boolean matrix for the constraints, and we asked it to check if a model is valid, or if there are counter examples.

Acknowledgements. This work was funded by the Lebanese American University.

References

[1] Summers, C.: Computer Security: Threats and Safeguards. McGraw Hill, New York (1997) ISBN-13: 978-0070694194

[2] Jackson, D.: Alloy 3.0 Reference Manual (2004), `http://alloy.mit.edu/reference-manual.pdf` (retrieved on September 24, 2012)

[3] Seater, R., Dennis, G.: Tutorial for Alloy Analyzer 4.0 (2011), `http://alloy.mit.edu/tutorial4` (retrieved on September 24, 2012)

[4] Anderson, R.: A Security Policy Model for Clinical Information Systems. In: Proceedings of the 1996 IEEE Symposium and Security and Privacy, pp. 30–43. IEEE Press, Oakland (1996)

[5] Hassan, W., Logrippo, L.: Detecting Inconsistencies of Mixed Secrecy Models and Business Policies. University of Ottawa, Canada, Technical Report (2009)

[6] Bell, L., LaPadula, E.: Secure Computer Systems: Mathematical Foundations. Technical Report 2547, Volume I. The MITRE Corporation (1976)

[7] Ferraiolo, D.F., Kuhn, D.R.: Role-Based Access Control. In: Proceedings of the 15th National Computer Security Conference, Baltimore, MD, USA, pp. 554–563 (1992)

[8] Viega, J., Evans, D.: Separation of Concerns for Security. In: Proceedings of the Workshop on Multi-Dimensional Separation of Concerns in Software Engineering, Limerick, Ireland, pp. 126–129 (2000)

[9] Zao, J., Hoetech, W., Chu, J., Jackson, D.: RBAC Schema Verification using Lightweight Formal Model and Constraint Analysis. In: Proceedings of 8th ACM Symposium on Access Control Models and Technologies, Boston, MA, USA (2003)

[10] Hassan, W., Logrippo, L., Mankai, M.: Validating Access Control Policies with Alloy. In: Proceedings of the Workshop on Practice and Theory of Access Control Technologies, Quebec, Canada (2005)

[11] Shaffer, A., Auguston, M., Irvine, C., Levin, T.: A Security Domain Model to Assess Software for Exploitable Covert Channels. In: Proceedings of the ACM SIGPLAN Third Workshop on Programming Languages and Analysis for Security, Tucson, Arizona, USA, pp. 45–56 (2008)

[12] Misic, J., Misic, V.: Implementation of Security Policy for Clinical Information Systems over Wireless Sensor Networks. Ad Hoc Networks Journal 5, 134–144 (2006)

[13] Haraty, R.A., Boss, N.: Modeling and Validating Confidentiality, Integrity, and Object Oriented Policies using Alloy. In: Security & Privacy Preserving in Social Networks. Springer (2013) ISBN 978-3-7091-0893-2

[14] Brewer, D., Nash, M.: The Chinese Wall Security Policy. In: Proceedings of the IEEE Symposium on Research in Security and Privacy, Oakland, CA, USA, pp. 206–214 (1989)

[15] Biba, K.J.: Integrity Considerations for Secure Computer Systems. Technical Report MTR-3153. The MITRE Corporation (1977)

[16] Lipner, S.B.: Non-discretionary Controls for Commercial Applications. In: Proceedings of the IEEE Symposium on Security and Privacy, Oakland, CA, USA, pp. 2–10 (1982)

[17] Haraty, R.A.: A Security Policy Manager for Multilevel Secure Object Oriented Database Management Systems. In: Proceedings of the International Conference on Applied Modeling and Simulation, Cairns - Queensland, Australia (1999)

[18] Haraty, R.A.: C2 Secure Database Management Systems – A Comparative Study. In: Proceedings of the Symposium on Applied Computing. San Antonio, TX, USA, pp. 216–220 (1999)