IOSR Journal of Computer Engineering (IOSRJCE)
ISSN: 2278-0661 Volume 4, Issue 4 (Sep-Oct. 2012), PP 01-05
www.iosrjournals.org

Designing Secure Systems Using AORDD Methodologies in UML System Models

[1]Ushasree R, [2]P. Raja Rajeswari, [3]Dr. D. vasumathi

Abstract: *We propose a AORDD methodology, based on Aspect-Oriented Modeling (AOM), for incorporating security mechanisms in an application. The functionality of the application is described using the primary model and the attacks are specified using aspects. The security mechanism, modeled as security aspect, is composed with the primary model to obtain the security treated model. We illustrate how this can be done and show how the resulting system can be evaluated to give assurance that it is resilient to the given attack. In this paper we describe an aspect-oriented modeling (AOM) approach that eases the task of exploring alternative ways of addressing concerns during software modeling.*

Keywords: *Alloy, Aspect-oriented modeling, secure systems design, Security analysis*

I. Introduction

In the commercial world, designing secure applications is impacted by various parameters, such as time-to-market, cost and effort involved. We propose a risk driven development approach for designing such applications.

For example, a Role Based Access Control (RBAC) model can be used to describe a solution to the banking system's access control concern. A decision to address a concern in a particular manner can give rise to other concerns. For example, the RBAC solution to the access control problem gives rise to new concerns pertaining to the management of roles and permissions. In risk-driven development (RDD) security risks are identified, evaluated, and treated as an integrated part of the development.

II. Background

2.1 ALLOY

we show how to formally verify that a security mechanism incorporated into a system is effective in protecting against a given security breach. we show how a system modeled using UML is converted to a form that can be automatically verified using the Alloy Analyzer.The Alloy analyzer translates a model into Boolean expression and analyses it using SAT-SOLVERS meta model element .

An Alloy model consists of a number of signature and relation declarations. The Alloy Analyzer translates a model into a Boolean expression and analyzes it using embedded SAT-solvers. The user needs to specify a scope to the tool.

2.2 ASPECT ORIENTED MODELING

An aspect oriented modeling approach of the following artifacts

1. A *primary model* that describes the business logic of the application.
2. A set of *generic aspect models*, where each model is a generic description of a crosscutting feature.
3. A set of bindings that determine where in the primary model the aspect models are to be composed.4. A set of composition directives that influence how Aspect models are composed with the primary model.

2.3 SECURE SYSTEM DESIGNS

We also include the project-specific consequence of incorporating a security mechanism to prevent the attack, in the form of variables related to the development effort in terms of cost and time. First, we perform a formal security analysis to give assurance that the system, created by integrating a security mechanism model, is indeed resilient to the targeted attack. We transform a UML misuse model into Alloy and use the Alloy Analyzer to reason about its security properties.

The results of the analysis either give assurance that the security properties exist, Alloy is a fully declarative first-order logic language designed for modeling and analyzing complex systems.

2.4 SECURITY ANALYSIS

In security assessment and management several techniques for identifying and assessing security problems in an information system are combined into a process that ensures that there is continuous review and update of its security controls.

• *Eavesdropping*. The attacker may observe the communications channel.
• *Replay*. The attacker records messages she has observed and re-sends them at a later time.
• *Man-in-the-middle*. The attacker intercepts the messages sent between the parties *C* and *S* and replaces these with her own messages.

III. CASE STUDY

EXAMPLE E-COMMERCE SYSTEM

Our example is an e-commerce platform called ACTIVE. ACTIVE provides services for electronic purchasing of goods over the Internet. The project identified several security risks, including attacks against user authentication in the login service. Here we defines two models are primary model and context-model primary model that describes a user management system in which The*UserMgmt* class defines operations for adding a user to the repository (*addUser*) and for deleting a user from the repository (*deleteUser*).
the diagrams of primary and context model.

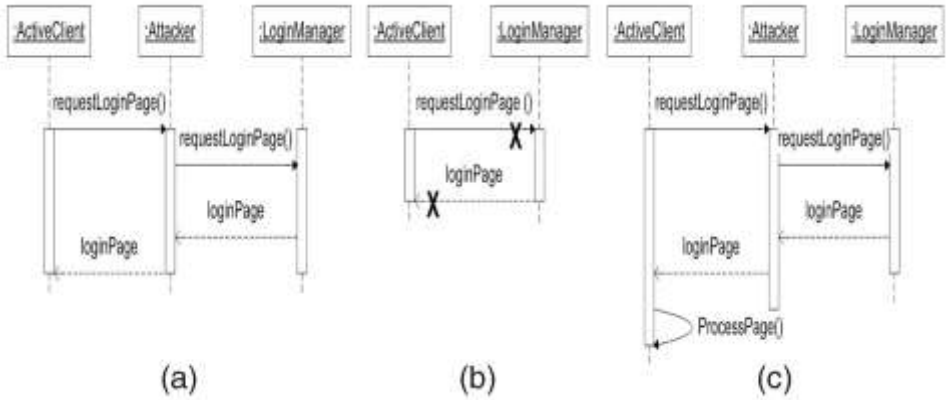

Figure 1: Man-In-Middle-Attack

misuse model of original
(a) ACTIVE login sequence and MIM attack, created by primary login sequence model with context-specific MIM passive attacks models. The communication between ACTIVE CLIENT and
(b) LOGIN MANAGER through ATTACKER.The attack is successful is ©ATTACKER obtains home page, or username,password.

3.1. THE MAN-IN-THE-MIDDLE ATTACK

In this section, we show how to represent the man-in-the-middle attack as a *generic aspect*. Messages between a requestor and authenticator are intercepted by an attacker. authenticator. The risk assessments performed as part of the CORAS project identified the login process as being vulnerable to man-in-the-middle attacks. During this kind of attack, user information can be obtained directly, or an attacker can intercept user names and passwords, to be used at later times to impersonate a valid user. The sequence diagram shows all messages between the |*Requestor* and *Authenticator* passing through the |*Attacker*. Secret information can be changed by the|*Attacker* as shown by the |*checkSecretInfo* message from the |*Requestor* to the |*Attacker*, and the |*checkSecretInfoAt* message passed on to the *Authenticator.* This generic aspect must be instantiated to create a *context-specific* aspect that can then be composed with the primary model to create a misuse model.

3.2. SECURITY MECHANISMS To COUNTER MAN-IN-THE-MIDDLE ATTACKS

System designers must identify security properties relevant to mitigating a risk to system assets. We identify properties according to the ISO/IEC TR 13335:2001 Information Technology—Guidelines for Management of IT Security [9]. The UML2Alloy tool to transform a UML model into Alloy. Its input consists of a UML class diagram in XML Metadata Interchange (XML) format , and an accompanying OCL specification of behavior. We therefore begin with the Abstract & Transform activity as the first activity in AORDD analysis. The next activity, Create Alloy Model using UML2Alloy, applies UML2Alloy to the XMI representation. UML2Alloy implements transformation rules to create an Alloy model. This model is input to the next activity Analyze with Alloy Analyzer.There are a number of OCL constraints that cannot be directly expressed in Alloy and are thus not supported by UML2Alloy (for example, the OCL "iterate" construct). Another issue is that OCL lacks inherent support to capture temporal properties. As a result, different methods have been proposed to extend OCL with the ability to express temporal constraints . It is, however, possible to

depict simple but crucial constraints related to time if a designer models time explicitly and uses conventional OCL to express constraints. Details on exactly which OCL statements are supported by UML2Alloy.

3.3. MISUSE MODEL OF SECURITY-TREATED PRIMARY MODEL

The SRP security-treated misuse model . However, the active attack differs in three ways:

1) Attacker substitutes its own expression and name in the startComm message (aExpr and a name)
2) Attacker generates its own key and token (key and aTok)
3) Attacker substitutes its token for the ActiveClient in the verify message (aTok).

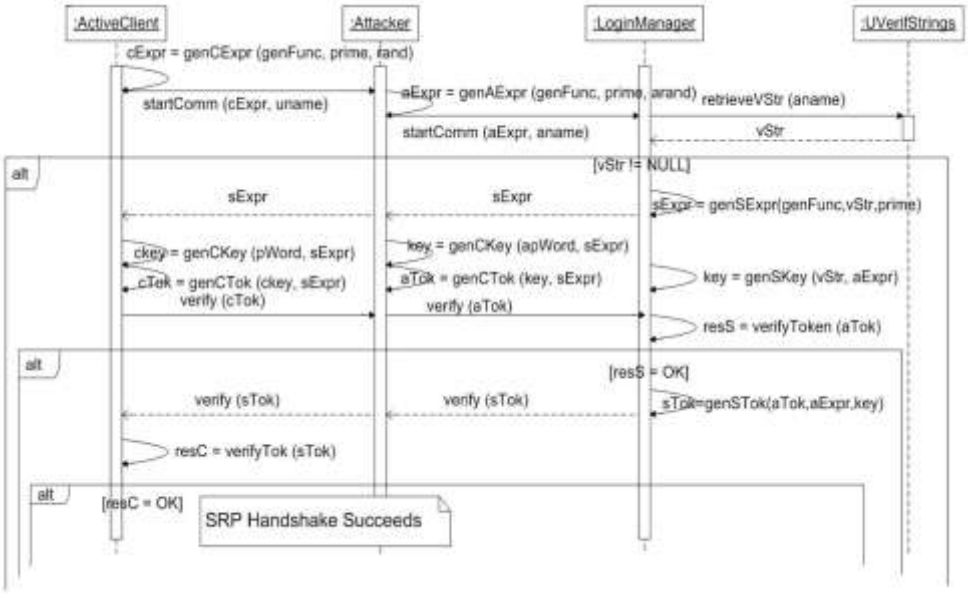

Figure-2: Portion of SRP security-treated misuse model including active MiM attack.

Misuse model of original ACTIVE login sequence and MiM attack, created by composing primary login sequence model with context specific MiM passive attack models. All communication between ActiveClient and LoginManager goes through Attacker. The attack is successful if Attacker obtains homePage, or uname and pword.If the profile does not exist or the user cannot be authenticated, a visitorPage is returned that does not contain any user-specific information.

IV. Implementation Environment

A constraint is a requirement which leaves no design option. e.g the developers could use any language they like then say so. Otherwise describe just the constraint.When referring to system interfaces, legacy systems anddatabases refer to the design documentation for these. Add important diagrams to Appendix A and refer to them in the text. If there is insufficient information about these external systems then mention that this information will need to be completed for the purposes of the development of this system.

```
Context ActiveClient                          /*attacker protocol successful*/
 ActiveClient.allInstance ( )->
 forAll(ac:ActiveClient |
 ac.loginAborted=ResultType::r_false implies
 (ac.at.key < > keyType : : Symmkey and
   ac.at.im.key=KeyType : : symmkey and
    ac.key =keyType ::Symmkey))
```

In the above the OCL assertion that Attacker has not generated the same key a Activeclient and LoginManager if the SRP protocol is successful.

```
Assert ok{all ac:ActiveClient |     /*same key could not be generated by attacker*/
          Ac.loginAborted = r_false =>
{ac.at.Key != symmKey && ac. at.Im.Key=symmKey
           && ac.key = symmkey}}
```

Alloy translation of OCL assertion as shown in the above algorithm .If the protocol does not abort, the Attacker has not generated the same key as that generated by Active client and login manager.

Scope	**Time required**
10	2 seconds

14	5seconds
20	27seconds

UML Alloy tool is used to create an Alloy model from the class diagram and associated OCL specification. The OCL specification defines system behavior, and users must create an XMI format of the class diagram and OCL specification, using a UML design tool.

V. Results And Discussions

Solutions to design concerns (e.g., security and fault tolerance concerns) may crosscut many modules of a design model. The cross-cutting nature of these solutions can make understanding, analyzing and changing the solutions difficult. This complexity can be addressed through the use of aspect-oriented modeling (AOM) techniques,where the design of a cross-cutting solution is undertaken in an independent fashion,and the resulting *aspect* models are composed with *primary* models of core functionality to create a complete system design. Composition is necessary to identify conflicts across aspect and primary models, and to identify undesirable emergent properties in composed models.

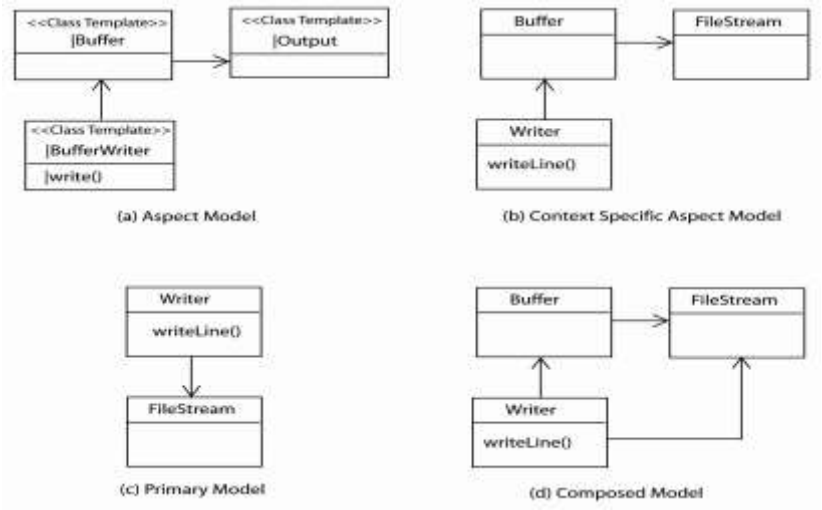

Figure 3: MODELS

Example 1. Consider the example in Figure . In the context specific aspect model, the *UserMgmt* class contains a operation called *getRepositorySize()* that retrieves the size of *SystemMgmtAuthRepository*. a different operation. To resolve this conflict, the rename directive can rename one or both operations, and the replace References directive can update any references to the old Name. The following composition directives are applied:

(1) rename aspect::UserMgmt::getRepositorySize() **to** aspect::UserMgmt::getAuthRepositorySize()
(2)replaceReferencesaspect::UserMgmt::getRepositorySize()**with** aspect::UserMgmt::getAuthRepositorySize()
(3)renameprimary::UserMgmt::getRepositorySize() primary::UserMgmt::getUserRepositorySize()
(**4) replaceReferences**primary::UserMgmt::getRepositorySize
After Application and note the changes

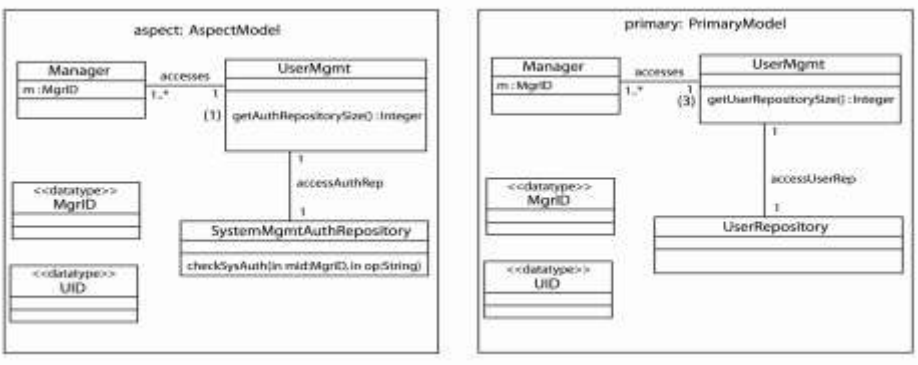

Figure 4: Node results

The result of applying the directives is shown in Figure 3. Where applicable, the effects of the composition directives are denoted in the composed model using the corresponding numbers. The names of *getRepositorySize()* in aspect and primary are changed to *getAuthRepositorySize()* and *getUserRepositorySize()*, respectively. The references to the operation names are changed throughout each model to reflect the name change, and to avoid reference conflicts.

VI. Conclusion And Future Work

In this paper, we propose a methodology for developing secure systems that are resilient to given attacks. We first perform risk assessments to identify the types of attacks that are typical for such applications. We show how to evaluate the application against such attacks. If the results of this evaluation indicate that the assets may be compromised, then some security mechanism must be incorporated into the application. The resulting system is then formally analyzed to ensure that it is indeed resilient to the given attack. We validated our approach on a real-world e-commerce application. Our approach does not detect new vulnerabilities but it can be used for assessing whether a given vulnerability poses sufficient risk that necessitates its mitigation. The main benefit of our approach is that it simplifies the design of complex systems. The primary models and the aspects can be analyzed in isolation to ensure that individually they satisfy the functional and security properties respectively. The models can be composed and the analysis of the composed model will give assurance that the resulting system also satisfies the properties. Another benefit of our approach is that it allows one to experiment with various security mechanisms to see which one is most suitable for preventing a given attack on the application. When a system is required to enforce different security properties, multiple aspects must be integrated with the application. This will allow one to study and formalize the interaction between aspects.

Our on-going and future work concentrates efforts in three areas. We are in the process of developing detailed algorithms to support the abstraction of complex UML diagrams and their conversion to OCL specifications, so that the approach can be automated. This ability will aid developers using the approach by reducing the chances that simplifying abstractions made by the developer leave out crucial items for the analysis. We are also investigating the broader applicability of the approach to other security mechanisms that are more appropriately specified by UML diagrams other than sequence diagrams. Finally, we are also investigating application of the approach to other stages in the development lifecycle of complex software systems, especially to the requirements phase.

References

[1] ISO 14508, Common Criteria for Information Technology Security Evaluation, in Version 3.1, Revision 2, 2007.

[2] S.H. Houmb, "Decision Support for Choice of Security Solution:The Aspect-Oriented Risk Driven Development (AORDD) Framework,"Dept. of Math. Sciences, Norwegian Univ. of Science and Technology, 2007.

[3] S.H. Houmb et al., "Cost-Benefit Trade-Off Analysis Using BBN for Aspect-Oriented Risk-Driven Development," Proc. IEEE Int'lConf. Eng. Complex Computer Systems, pp. 195-204, 2005.

[4] S.H. Houmb et al., "An Integrated Security Verification and Security Solution Design Trade-Off Analysis Approach," Integrating Security and Software Eng.: Advances and Future Vision,H. Mouratidis and P. Giorgini, eds., IGI Global, 2007

[5] R. France et al., "A UML-Based Pattern Specification Technique,"IEEE Trans.Software Eng., vol. 30, no. 3, pp. 193-206, Mar. 2004.

[6] R. France et al., "Aspect-Oriented Approach to Design Modeling," IEE Proc. Software, vol. 151, no. 4, pp. 173-186, 2004.

[7] G. Georg, J. Bieman, and R. France, "Using Alloy and UML/OCL to Specify Run-Time Configuration Management: A Case Study," Proc. Workshop pUML-Group Held Together with the UML, A. Evanset al., eds., pp. 128-141, 2001.

[8] G. Georg et al., "An Aspect-Oriented Methodology for Designing Secure Applications," Information and Software Technology, vol. 51, no. 5, pp. 846-864, 2009.

[9] Straw et al., "Model Composition Directives," The Unified Modelling Language: Modelling Languages and Applications (UML), T. Baar et al., eds., pp. 84-97, Springer, 2004.

[10] P. Ziemann and M. Gogolla, "An Extension of OCL with Temporal Logic," Proc. Workshop Critical Systems Development with UML, J. Ju¨ rjens, ed., pp. 53-62, 2002.

[11] T. Wu, "The Secure Remote Password Protocol," Proc Internet Soc.Network and Distributed System Security Symp., pp. 97-111, 1998.GEORG ET AL.: verification and trade-off analysis of security properties in uml system models 355

Ushasree R.