An Assessment Model for Security-Critical Enterprise Systems

Bandar M. Alshammari

***Abstract*—This paper presents a model for assessing security of enterprise systems. It focuses on the structural properties of enterprise systems' architectures in order to quantify their overall security. The model is built on the well-known three-tier architecture model and aims to identify the ways in which security-critical data values may be transferred between various components of the system's architecture. This paper extends the three-tier architecture model to add a fourth layer which defines a set of low-level security metrics developed based on systems' structural characteristics, such as data accessibility, coupling, cohesion and complexity. These metrics then are linked to relevant components of the three layers in the three-tier architecture model and hence defining a single security metric for each component. By combining security metrics of each layer's components, a single security index is defined that forms the security value of each layer. Finally, the entire system's security is summarised as a single security value. These metrics allow different architecture of the same system, or different systems with similar functionalities, to be compared for their relative security at a number of different abstraction levels at an early stage of development for any enterprise system.**

***Index Terms*—Security models, three-tier architecture, security metrics, enterprise systems.**

I. Introduction

Much existing software is designed with poor consideration of information security which makes it vulnerable to many threats including malicious attacks [1]. Software patches are one of the suggested solutions for many of the security attacks facing software [1] but they are expensive to develop and deploy, and do not solve basic design weaknesses in the program code. Another solution to achieve a secure product is by following a trustworthy security process [2]. Security processes, in general, consider many aspects of system design, coding, testing, and auditing [2] (e.g., international security standards such as the Common Criteria [3] or the Trusted Computer Criteria [4]). Another common approach for achieving a secure computer program is by following certain coding guidelines which focus on the level of individual program statements (e.g., to avoid/detect buffer overflows [5]). However, these solutions do not always work effectively and may, in general, even introduce new vulnerabilities to existing software [1].

The most promising approach is the one which is capable of quantifying security of a given system at an early stage of development (i.e., security metrics). Several types of metrics have been defined in the literature which aim to measure security of programs. These include metrics which assess security at the abstract system architecture level [6], at the design phase [7], [8], and at the low level of program code [9]-[11]. However, none of this work to date is capable of measuring the overall security of a given enterprise system based on its architecture with respect to information flow.

This paper, instead, defines several new security metrics for a complete enterprise system based on its architecture design. The model builds on the three-tier architecture by defining a set of low-level security metrics extracted from the well-defined system's structural design properties. Then, each layer will be linked to a number of these security metrics. An index which forms the overall security of a given enterprise system is computed based on the security metrics of the three layers. These metrics can be used to compare different architectures at different level of abstractions for the same system and identify the most secure one. This allows system's architects and developers to assess the relative security of their systems from an early stage of development with respect to the potential flow of classified data.

II. Related Work

Measuring the security of the system's architecture is an important aspect of identifying the overall security of a given program. One of the studies in this field is by Antonino *et al.* [12] who define an evaluation technique for measuring the security of an existing service-oriented architecture. This evaluation technique is based on two types of metrics: severity and credibility. Severity relates to the value of tagged security artifacts while credibility is the probability of correctly assigning a tag to its relevant system component [12].

Further work in this area was conducted by Liu *et al.* who proposed a model called the "User System Interaction Effect (USIE)" [13]. The USIE model is responsible for providing a systematic approach to identify security defects from the architecture of a service-oriented system [13].

Another approach for measuring security based on an architecture by Manadhata *et al.* [6] measures security with regard to the attack surface size. The system's attack surface measurement is an indicator of the risk of attack [14]. It is based on the set of possible resources which an attacker could use to attack the system [14], including methods, data and channels [14]. A method is described by Manadhata *et al.* as a system entity which could send data (exit point) or receive data (entry point) [6]. Data in their approach is any entity which is visible in the current system such as files, cookies and database records [6]. They also define channels as system

Manuscript received September 14, 2013; revised November 25, 2013.

B. M. Alshammari is with the Information Technology Department, University of Aljouf, Saudi Arabia (e-mail: bmshammeri@ju.edu.sa).

DOI: 10.7763/IJIET.2014.V4.422

entities which can be used by an attacker to invoke the system's methods such as sockets and pipes [14]. A smaller attack surface indicates a smaller number of potential attacks, and thus a more secure system. They use this metric to compare the attack surface size of different versions of two IMAP servers and two open source FTP demons [14]. However, these approaches do not allow enterprise systems' designers to assess the overall security of a given system based on its architecture.

Recent studies conducted by Alshammari *et al.* [7], [8], [15] had defined several security metrics for UML class designs, and described a tool for automatically evaluating such metrics [16]. The defined metrics assess the potential flow of security-critical data by measuring the accessibility of such data based on the security design principles of "granting least privilege" [17]-[19] and "reducing the size of the attack surface" [20], [14]. Nevertheless, this approach cannot have an indication of the security of enterprise systems especially those developed using a three-tier layer architecture.

Instead, this paper extends the three-tier layer architecture to propose a model that defines a number of security metrics that are capable of measuring overall security of a given enterprise system from an early stage of development based on many of its compositional properties and information flow principles. These metrics will also allow enterprise systems developers to compare the relative security of different versions of the same system.

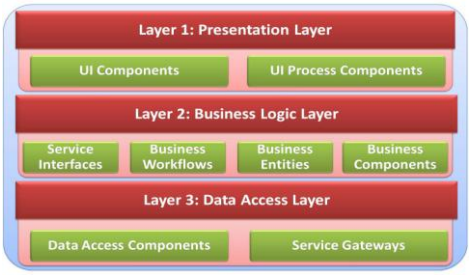

Fig. 1. Three-tier application architecture.

III. Three-Tier Layer Application Architecture

Due to the increasing necessity for developing large-scale enterprise systems (e.g, ERP and DBMS) using three-tier architecture model [21], [22], this section illustrates the various layers of the three-tier layer architecture, their responsibilities and their main components as shown in Fig. 1 the three-tier layer architecture consists of three different layers: Presentation Layer, Business Logic Layer, and Data Access Layer [23]-[25] which are explained as follows.

A. Presentation Layer

This layer is responsible for providing the System's User Interface (UI) (such as windows forms) which allows users to interact with the system [25]. It consists of two main components. One is the UI Components which provide simple forms [25]. The other is the UI Process Components which are used to encapsulate dependencies between the UI form in the case of complex User Interfaces [25].

B. Business Logic Layer

This layer is the core one and is responsible for implementing the application business functionalities [25]. It consists of four components. One is the Business Components which are responsible for encapsulating the business logic [25]. The second is the Business Workflows which can be represented as the low-level activities of a certain system [25]. The third is the Business Entities which are responsible for hiding details of specific data representation formats [25]. The fourth component is the Services Interfaces which provide some of the system's functionalities as a service to the outside world [25].

C. Data Access Layer

The main responsibility of this layer is to provide access to the data [25]. It consists of two components. One is the Data Access Component which exposes the data stored in the databases to the business logic layer [25]. The other component is the Service Gateways which are used by other layers to interact with the data they require [25].

IV. Enterprise Systems Security Assessment Model

This section explains how the proposed security assessment model provides an easy to use approach for assessing the relative security of enterprise systems. The model can provide guidance for the development of secure enterprise systems using a three-tier architecture approach. It ensures that attention is given to lower-level detail in any given system such as how cohesive security-critical components of that system.

As a result, this leads to defining a security measurement of an enterprise system's security in terms of its higher-level components.

The model for assessing the security of a given enterprise system is shown in Fig. 2. It builds on the three-tier architecture development model [23]-[25] commonly followed to develop enterprise systems [21], [22]. Each of its four levels defines a set of metrics for the security of a given enterprise system at a certain abstraction level. The bottom level defines several security metrics from the perspective of the potential flow of classified data in the system based on a number of relevant enterprise system's structural properties. The next level defines security from the perspective of each of the components of the the three layer in the three-tier architecture. Each component in each of the three layers is linked to a number of the security structural properties metrics which have direct influence on them in terms of the potential flow of classified data. The next level up measures security with respect to each of the three layers. This means that security metrics of the components for each layer will form the total security measurement of that layer. Finally, the top level provides a single security measurement which summarises the total security of the entire enterprise system, allowing it to be compared easily with other similar systems. Each of the higher-level sets of metrics is defined based on those metrics at the level beneath it.

A. Enterprise Systems Structural Quality Properties Security Metrics

This section illustrates the security metrics for enterprise systems architecture designs which are defined based on the analysis of the structural quality properties for software

systems [25]-[27]. These properties include: accessibility, composition, coupling, cohesion, extensibility, inheritance, design size, complexity, Abstraction, and modularity. To define the proposed security metrics for enterprise systems, each property and its relevance to designing secure enterprise systems has to be studied. The relevant quality structural properties are defined below.

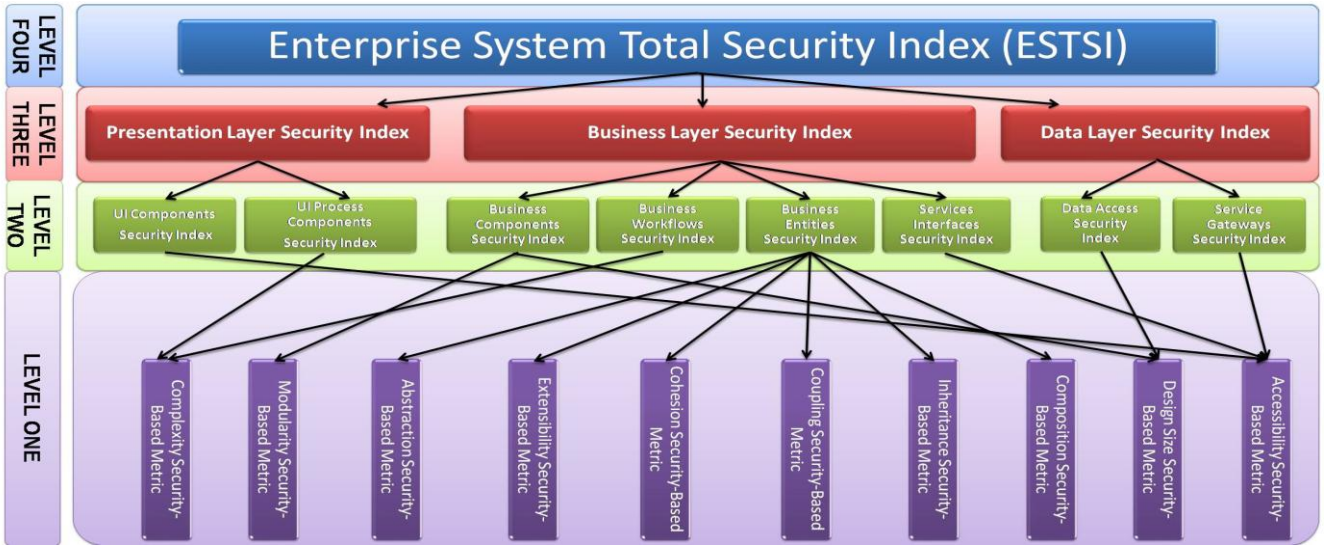

Fig. 2. Enterprise system security assessment model.

1) *Security Accessibility Metrics:* Metrics under this category aim to statically measure the potential flow of information from an accessibility perspective of security-critical objects such as attributes, methods, classes, and components which may contain classified data.
2) *Security Composition Metrics*: Composition (Aggregation) is defined as an association between two or more system components/objects [28], [29], hence metrics under this section will be defined to measure the potential flow of classified information through composed objects.
3) *Security Coupling Metrics*: Coupling-based security metrics will be defined to statically measure the potential flow of information from an inter-object interaction perspective of security-critical objects such as attributes, methods, classes, and components.
4) *Security Cohesion Metrics*: Cohesion-based security metrics will be defined to statically measure the potential flow of information from an intra-object interaction perspective of security-critical objects such as attributes, methods, classes, and components.
5) *Security Extensibility Metrics*: Extensibility is the property which allows a certain object (e.g, class, component, etc) to be extended by other objects [28], [29]. Therefore, metrics under this category aim to measure the potential flow of classified data as a result of extending objects which might contain security-critical values.

TABLE I: SECURITY METRICS FOR THE ARCHITECTURE LAYERS

Name	Metric	Definition
PLSM	Presentation Layer Security Metric	UIC + UIPC
BSLSM	Business Logic Layer Security Metric	SI + BW + BE + BC
DSLSM	Data Access Layer Security Metric	DAC + SG

6) *Security Inheritance Metrics*: Inheritance is a mechanism which allows programmers to provide objects (i.e., classes, packages, methods, and attributes) with generalization and specialization relationships [28], [29]. Inheritance could allow sub-components to acquire privileges over classified data in super-components, therefore, metrics under this category are defined to measure the chance of potential classified information flow as a result of inheritance.
7) *Security Design Size Metrics*: A metric for measuring design size in object-oriented designs defined by Bansiya and Davis [27] is called Design Size in Classes (DSC). With respect to security of enterprise systems, design size-based security metrics measure the proportion of security-critical components such as classes and packages to the overall components in an enterprise system.
8) *Security Complexity Metrics*: The complexity design property aims to identify how difficult it is to understand the internal and external structure of a given system [27] and hence the complexity metric measures this. In this paper, the security complexity-based metric assesses how complex structure of a certain enterprise system affect the security level of that system from the perspective of potential flow of classified data.
9) *Security Abstraction Metrics*: Abstraction is a design property that aims to hide internal details of a given system's component (e.g., a method's implementation) by providing an interface layer which provides abstract details of that component [25]. With regard to the effect of this property on security of a given enterprise system, metrics under this category will assess the potential flow of classified data as a result of using this property to design that system.
10) *Security Modularity Metrics*: In designing object-oriented systems, modularity is the process of dividing a system into a set of functional units which have a collection of related components [30]. Metrics under this section will be able to identify the impact of

using this property to design enterprise systems on the overall security of that system. This means that those metrics will show whether modularity will improve or worsen security of a certain enterprise system.

TABLE II: THE ENTERPRISE SYSTEM TOTAL SECURITY INDEX

Name	Metric	Definition
ESTSI	Enterprise System Total Security Index	PLSM + BSLSM + DSLSM

B. Security Metrics for the Architecture Components

This section illustrates the security metrics for each of the eight components defined in the three layer architecture. Each component will be associated with a number of the structural properties security metrics which are mostly concerned with. Hence, this section defines eight security enterprise components metrics as shown below.

1) *Security Metric for the UI Components (UIC):* This security metric consists of the security metrics of enterprise systems' structural properties related to the UI components. Hence UIC can be represented by the Accessibility security-based metric.
2) *Security Metrics for the UI Process Components (UIPC):* This metric can be represented by the complexity security-based metric.
3) *Security Metrics for the Business Components (BC):* This metric is the sum of the Modularity security-based metric and the Accessibility security-based metric.
4) *Security Metrics for the Business Workflows (BW):* This metric can be represented by the complexity security-based metric.
5) *Security Metrics for the Business Entities (BE*): This metric consists of the sum of the Abstraction security-based metric, Extensibility security-based metric, Coupling security-based metric, Cohesion security-based metric, the Inheritance security-based metric, the Composition security-based metric and the Design Size security-based metric.
6) *Security Metrics for the Service Interfaces (SI):* The SI metric is represented by the Accessibility security-based metric.
7) *Security Metrics for the Data Access Components (DAC):* The DAC metric is the value of the Design Size security-based metric.
8) *Security Metrics for the Service Gateways (SG):* The SG metric can b represented be the Accessibility security-based metric.

C. Security Metrics for the Architecture Layers

This section defines three security metrics which each relates to one of the three layers. These are smaller, more easily understood, set of security metrics which still contains sufficient information to give an accurate measure of a program's security. Each of those three is the sum of the security components metrics which are associated with that layer. The layer security metrics are shown in Table I

D. The Enterprise System Total Security Index (ESTSI)

The top layer of the proposed model defines a single value which represents the total security index of the entire enterprise system. This value gives a simple approach for assessing the overall security of a certain enterprise system based on information obtained from lower-level security metrics. The ESTSI is simply the sum of the security metrics of the three layers as shown in Table II.

V. CONCLUSION AND FUTURE WORK

This paper has presented a new security assessment model which quantifies the security of enterprise systems. It provides a simple and transparent approach for assessing security of any enterprise system at various levels of abstraction. The model builds on the well-known three layer architecture which is commonly used to develop enterprise systems. It considers low-level structural properties of an enterprise system architecture in order to quantify the security of higher-level components and layers of the three-tier layer application architecture. Security metrics of the three layers are then summed to a single value that defines the overall security measurement of a certain enterprise system. This assessment can provide systems' architects and developers with a simple approach for quantifying any enterprise system's security. It also allows them to compare different versions of a system from the perspectives of classified data confidentiality and integrity at an early stage of development.

Future work will include defining the structural properties metrics in more detail. It will also demonstrate the validity of the proposed security assessment model empirically using existing enterprise systems. To do this, a tool needs to be developed in order to automatically calculate these metrics from an architectural perspective.

REFERENCES

[1] B. Fernandez, "A methodology for secure software design," in *Proc. the International Conference on Software Engineering Research and Practice, SERP '04*, vol. 1, Las Vegas, Nevada, USA, 2004, pp. 130–136.

[2] M. Howard and D. LeBlanc, *Writing Secure Code*, Redmond, Wash: Microsoft Press, 2002.

[3] CC Recognition Arrangement. The Common Criteria for Information Technology Security Evaluation (CC). [Online]. Available: http://www.commoncriteriaportal.org/.

[4] The National Computer Security Center (NCSC). (1985). The Trusted Computer System Evaluation Criteria (TCSEC). [Online]. Available: http://csrc.nist.gov/publications/history/dod85.pdf.

[5] R. C. Seacord, *Secure Coding in C and C++*, Upper Saddle River, NJ: Addison-Wesley, 2006.

[6] P. K. Manadhata, K. M. C. Tan, R. A. Maxion, and J. M. Wing, "An approach to measuring a system's attack surface," *Tech. Rep. CMU-CS-07-146*, Carnegie Mellon University, Pittsburgh, PA, August 2007.

[7] B. Alshammari, C. J. Fidge, and D. Corney, "Security metrics for object-oriented class designs," in *Proc. the Ninth International Conference on Quality Software (QSIC 2009)*, 2009, IEEE, Jeju, Korea, pp. 11–20.

[8] B. Alshammari, C. J. Fidge, and D. Corney, "Security metrics for object-oriented designs," in *Proc. the Twenty-First Australian Software Engineering Conference (ASWEC 2010)*, Auckland, 6–9 April, Eds. J. Noble and C. J. Fidge, California, USA, IEEE Computer Society, 2010, pp. 55–64.

[9] I. Chowdhury, B. Chan, and M. Zulkernine, "Security metrics for source code structures," in *Proc. the Fourth International Workshop on Software Engineering for Secure Systems*, 2008, ACM, Leipzig, Germany, pp. 57–64.

[10] I. Chowdhury and M. Zulkernine, "Can complexity, coupling, and cohesion metrics be used as early indicators of vulnerabilities?" in *Proc. the 2010 ACM Symposium on Applied Computing, SAC '10*, 2010, ACM, New York, NY, USA, pp. 1963–1969.

[11] B. Alshammari, C. J. Fidge, and D. Corney, "Security metrics for java bytecode programs," in *Proc. the Twenty-Fifth International*

Conference on Software Engineering and Knowledge Engineering (SEKE 2013), 2013, Boston, 27–29 June, Knowledge Systems Institute.
[12] P. Antonino, S. Duszynski, C. Jung, and M. Rudolph, "Indicator-based architecture-level security evaluation in a service-oriented environment," in *Proc. the Fourth European Conference on Software Architecture: Companion Volume, ECSA '10*, 2010, ACM, New York, NY, USA, pp. 221–228.
[13] Y. Liu, I. Traore, and A. Hoole, "A service-oriented framework for quantitative security analysis of software architectures," in *Proc. IEEE Asia-Pacific Services Computing Conference*, 2008, pp. 1231–1238.
[14] P. Manadhata and J. Wing, "An attack surface metric," *IEEE Transactions on Software Engineering*, vol. PP, no. 99, pp. 1, 2010.
[15] B. Alshammari, C. Fidge, and D. Corney, "Security assessment of code refactoring rules," in *Proc. the National Workshop on Information Assurance Research, Riyadh*, Saudi Arabia, 2012, April 18, VDE, pp. 1–10.
[16] B. Alshammari, C. Fidge, and D. Corney, "An automated tool for assessing security-critical designs and programs," in *Proc. the National Workshop on Information Assurance Research*, Riyadh, Saudi Arabia, 2012, April 18, VDE, pp. 1–10.
[17] J. H. Saltzer and M. D. Schroeder, "The protection of information in operating systems," in *Proc. the IEEE*, vol. 63, 1975, pp. 1278–1308.
[18] M. Bishop, *Computer Security: Art and Science*, Boston: Addison-Wesley, 2003.
[19] G. McGraw, *Software Security: Building Security In*, Upper Saddle River, NJ: Addison-Wesley, 2006.
[20] M. Howard, "Attack surface: Mitigate security risks by minimizing the code you expose to untrusted users," *Microsoft MSDN Magazine*, vol. 11, 2004.
[21] D. T. Liu and X. W. Xu, "A review of web-based product data management systems," *Computers in Industry*, vol. 44, no. 3, pp. 251–262, 2001.
[22] S. Dumbrava, D. Panescu, and M. Costin, "A Three-tier Software Architecture for Manufacturing Activity Control in ERP Concept," in *Proc.f the 2005 International Conference on Computer Systems and Technologies, CompSysTech' 2005*, 2005, Technical University, Varna, Bulgaria.
[23] W. W. Eckerson, "Three tier client/server architecture: Achieving scalability, performance, and efficiency in client server applications," *Open Information Systems*, vol. 10, no. 1, 1995.
[24] M. Fowler, *Patterns of Enterprise Application Architecture*, Boston: Addison-Wesley, 2003.
[25] Microsoft. (October 2009). Microsoft application architecture guide. [Online]. Available: http://msdn.microsoft.com/en-us/library/ff650706.aspx.
[26] Open Group. Architecture principles. [Online]. Available: http://pubs.opengroup.org/architecture/togaf9-doc/arch/chap23.html.
[27] J. Bansiya and C. G. Davis, "A hierarchical model for object-oriented design quality assessment," *IEEE Transactions on Software Engineering*, vol. 28, pp. 4–17, 2002.
[28] S. Bennett, S. McRobb, and R. Farmer, *Object-Oriented Systems Analysis and Design Using UML*, 3rd ed., 2006, Maidenhead: McGraw-Hill Higher Education.
[29] M. Fowler, *UML Distilled: A Brief Guide to the Standard Object Modeling Language*. 2003, 3 ed., Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc..
[30] MSDN Library. (June 2010). Patterns and Paractices. [Online]. Available: http://msdn.microsoft.com/en-us/library/ff921069(v=pandp.20).aspx.

Bandar M. Alshammari is an assistant professor of software engineering at the Department of Information Technology, University of Aljouf, Saudi Arabia. He earned his BS from the School of Information Technology, University of Canberra, Australia. He holds a Master degree in Computer and Communications Engineering from the Queensland University of Technology, Australia. He earned his PhD in 2012 in Software Security Engineering from the Queensland University of Technology and his PhD title was "Quality Metrics for Assessing Security-Critical Computer Programs". He is currently the Vice Dean of Admission and Registration at the University of Aljouf. He obtained a number of prizes including Prince Abdulrahman Alsudairy scholarship to study undergraduate and graduate studies in Australia in 2000 and the Faculty of Built Environment and Engineering Award for Excellent Academic Performance at the Queensland University of Technolog in 2007. He has published several papers in the area of software security mainly focusing on software security metrics.

.