A Comparative Study of Software Security Pattern Classifications

Conference Paper · August 2012

DOI: 10.1109/ARES.2012.43

Aleem Khalid Alvi
Queen's University

A Comparative Study of Software Security Pattern Classifications

Aleem Khalid Alvi
School of Computing
Queen's University
Kingston, ON, Canada
aleem@cs.queensu.ca

Mohammad Zulkernine
School of Computing
Queen's University
Kingston, ON, Canada
mzulker@cs.queensu.ca

Abstract— **Software security patterns can be the building blocks of secure software systems. They provide reliable solutions for recurring security problems. The rapid increase in the number of security patterns creates difficulty in the selection of appropriate security patterns for particular security problems. Researchers provide several classification schemes based on unique selection criteria for choosing appropriate security patterns. These schemes are very helpful for software designers to select security patterns for particular security problems. In this paper, we survey various security pattern classification schemes. Further, we compare and contrast these classification schemes using their classification objectives, attributes, dimensions, and quality metrics. The result is helpful for selecting a suitable classification scheme based on the desirable classification attributes and quality metrics. The right selection of classification improves the capability of software designers to select appropriate security patterns for recurring security problems in a specific security context.**

Keywords: software security, security patterns, pattern classification.

I. Introduction

In software engineering, design patterns capture expert knowledge and package it to provide easy and fast implementation, domain independence, and reusability. In the area of software security, the subsets of processes and design patterns that support security requirements are recognized as security patterns. Security patterns are used to achieve the desired security objectives. They provide ease of use to software developers and designers, who may not have a deep understanding of security. Security patterns are defined as a solution of "a particular recurring security problem that arises in a specific security context and presents a well-proven generic scheme [2]."

Security pattern catalogs provide enumeration and are introduced to help software developers using the security patterns in software development [3-7]. A scheme for precise classification of security patterns is vital because security pattern catalogs are not enough, as designers are not always sure when, where, and how to use them [8]. The rapid increase in the number of software security patterns invites a problem in the selection process. The selection of appropriate security patterns for fulfilling a security loophole is complex. The complexity increases with the increasing number of security patterns in an existing collection. The security patterns in the collection help provide a secure environment for large software applications. The security pattern system and language elaborate the security pattern collaboration for achieving desirable security objectives. Therefore, collaboration, communication, and integration of security patterns into a solution of security problem create further complexity. The solution to reduce complexity is a good and efficient classification. A classification helps in the selection process of security patterns and its related patterns that may be used for the collaborative solution of a recurring security problem.

Researchers [6, 9, 10-16, 40] develop the classifications of security patterns based on a number of different perspectives, such as applicability, enterprise architectural spaces, desirable properties, security objectives, logical architecture tiers, threat modeling, software lifecycle phases, text processing approach and learning techniques, attack patterns, and security flaws. A classification provides appropriate security patterns as solutions to recurring security problems caused by software security vulnerabilities.

The rest of the paper is organized as follows. In Section II, we provide an extensive survey on security pattern classifications. In Section III, we compare and contrast the existing classifications. Finally, we conclude and discuss open issues for future research.

II. Security Pattern Classifications

In this section, we explore the landscape of security pattern classifications and study the existing approaches. The security pattern classifications are characterized by the attributes that show the perspective of the classification of security patterns. In the next subsections, we describe the existing classifications of security patterns in the chronological order of their publications. The list of classifications is based on the following attributes.

- product and process
- purpose
- abstraction, lifecycle, and problem domain
- applicability
- organizing table
- undesirable properties
- context of security views
- context of an enterprise framework
- partition schemes
- transactions
- development phases and security objectives
- logical architecture tiers

- core security viewpoints
- formal concept analysis
- software lifecycle
- architectural concerns, layers, and linguistic similarities
- categorization for users
- dimension and pattern graphs
- ontology-based approach
- text processing approach and learning techniques
- levels of abstraction
- attack patterns
- application domain
- security flaws

A. *Classification based on Product and Process*

Kienzle *et al.* [7, 17] categorize a repository with 29 security patterns into two classes: 16 structural and 13 procedural patterns. They organize the pattern repository based on four strategies: subject matter, life cycle stage, pattern type, and audience. They develop a system called "public Web repository system" as an experimental bed for security patterns. The repository is used to store the design and process level security patterns.

B. *Classification based on Purpose*

Cheng *et al.* [18, 19] organize security patterns into the following categories: creational, structural, and behavioral. They adopted the classification criteria from Gang of Four (GoF) classification scheme of design patterns [20]. The security level for each security pattern is determined by using Viega and McGraw's ten principles [21]. They address the abstraction levels of applications, *i.e.*, application-level, host-level, and network-level, where applications are running on client, server, and network, respectively.

C. *Classification using Abstraction, Lifecycle, and Domain*

Schumacher and Roedig [2] propose the context of a security pattern, *i.e.*, classify security patterns based on software lifecycle phases or based on a layered approach. Using both approaches, they consider the enterprise, system and application as levels of abstraction, and the architecture, design and operation as the most important software lifecycle phases. This 2-dimensional classification approach can easily be extended in both directions to increase the navigability property. In addition, the problem domain of a security pattern may be considered as the third dimension. Threats and attacks also can be used to classify security patterns.

D. *Classification based on Applicability*

Blakley *et al.* [12] provide classification for structural design patterns and develop a technical guide for developers with a consistent set of security design patterns. Security patterns are categorized into two classes based on applicability called the available system patterns and protected system patterns. The technical guide provides a system of software security patterns and a design methodology for constructing a secure software system. However, this catalog does not support two or more overlapping instances of "protected system patterns" for protecting resources in common. It is recommended to avoid overlapping situations for protected system patterns.

E. *Classification based on Organizing Table*

Trowbridge *et al.* [14] introduce a tabular classification scheme for patterns based on the Enterprise Architectural Space Organizing Table (EASOT). The table is used for organizing patterns built on four pillars: the Zachman framework for enterprise architecture [22], the architectural standards description from IEEE 1471-2000 [23], the enterprise architecture framework [24], and the principles of test driven development [25]. The interrogatives from the Zachman framework and test driven development are illustrated by the seven columns: purpose (why), data (what), function (how), timing (when), network (where), people (who), and scorecard (test). The organizing table divides enterprise architecture into five broad enterprise viewpoints, *i.e.*, business architecture, integration architecture, application architecture, operational architecture, and development architecture. The authors use the EASOT to understand the context of patterns throughout the enterprise; for example, security as a context.

F. *Classification based on Undesirable Properties*

Laverdiere *et al.* [10] classify security patterns using undesirable properties. They consider 12 security patterns that use in the architecture and design phases of software lifecycle. They consider the following undesirable properties: over-specified, under-specified, lacking generality, lacking consensus, and misrepresented. Laverdiere *et al.* [10] use the result of undesirable properties and Six-Sigma approach to find out the desirable properties of security patterns. These desirable properties prevent flaws in new security patterns and their templates.

G. *Classification based on the Context of Security Views*

Schumacher *et al.* [1] develop a classification based on context of security views using security taxonomy. Security taxonomy has three fundamental architectural layers: security strategy and policy, services, and mechanisms and implementations. The authors present 46 security patterns from the following eight areas: enterprise security and risk management, identification and authentication, access control model, system access control architecture, operating system access control, accounting, firewall architecture, and secure internet applications. Schumacher *et al.* [1] map the security patterns based on the areas mentioned above to the layers of the security taxonomy. For example, eight patterns from the "enterprise security and risk management" area are mapped to security property, risk management, and security approaches, which are part of the first layer of the security taxonomy (*i.e.*, "security strategy and policy" layer). They classify all patterns based on the context of security views using the security taxonomy.

H. *Classification based on the Context of Enterprise Framework*

Schumacher *et al.* [1] propose a classification based on the Zachman framework using systems viewpoints and interrogatives. The Zachman framework represents six architectural views as columns: data, function, network, people, time and motivation, and the levels of information models as five rows: scope, business model, system model, technology model, and detailed representation. Schumacher *et al.* [1] add the security view as a seventh column. The security view addresses all levels of information from scope to detailed representation. At this stage, the Zachman framework's rows are connected to the security taxonomy that includes the mapping with security patterns (as mentioned in Section G). In this way, security patterns integrate with the enterprise architecture framework and provide partitions for the security pattern landscape based on enterprise framework.

I. *Classification based on the Partitioning Schemes*

Hafiz and Ralph [11] extend the Microsoft's tabular pattern classification scheme [14] and highlight three disadvantages: lack of use of pattern's characteristic, presence of skewness, and navigation problem. They propose the solution to these problems by selecting and integrating the classification schemes of security patterns with the tabular classification. The extended classification proposals are described as follows.

1) *Classification based on the CIA model*

Confidentiality, integrity, and availability (CIA) are the three key security objectives of a secure software system and considered as classification attributes. However, security patterns also support one or more security objectives at the same time. One of the disadvantages is the overlapping nature of classification.

2) *Classification based on the Application Context*

This classification considers the structure of the secure software system that is partitioned into three parts: core, perimeter, and exterior. The core security considers internal security mechanism related issues, perimeter security considers the authentication, authorization and security filtering related issues, and exterior security considers communication security related issues. This classification uses application context to partition the landscape of security patterns.

3) *Classification using the Security Wheel*

The security wheel consists of spokes in which every spoke represents a security service, and if any service fails then the overall system faces potential risks and vulnerabilities. There are 12 spokes in a security wheel representing the security objectives of the system: confidentiality, integrity, policy, auditing, management, availability, compliance, logging, public key infrastructure, labeling, authentication, and authorization [15]. The classification of security patterns using the security wheel has fine level of partitions for security pattern landscape.

4) *Classification based on the McCumber Cube*

The McCumber cube is a model framework used for information security and it has three dimensions: security views (confidentiality, integrity, and availability), information states (storage, transmission, and processing), and safeguards (policy and practices, human factors, and technology) [26]. Security patterns categorize based on security views, information states, and safeguards.

5) *Classification based on the Threat Modeling*

Hafiz and Ralph [11] use the STRIDE model, a threat modeling framework, to classify security patterns [13]. The abbreviation STRIDE comes from the first letters of six known threat names, *i.e.*, spoofing identity, tampering with data, repudiation, information disclosure, denial of service, and elevation of privilege. They use 12 security patterns to classify based on the STRIDE model. Few security patterns could not be classified because they use all the security concepts of the STRIDE model, *e.g.*, minefield and defense in depth patterns.

6) *Classification based on Hierarchy*

The hierarchy-based classification scheme is the combination of the hierarchical classification mechanism and the partitioning scheme or classical table. Therefore, a new security pattern is easily accommodated to the hierarchical classification by changing some of its parameters. For example, it considers the application context in the first level and the STRIDE model in the second level classification. This classification considers structural patterns only. It has flat and hierarchical views that support both directions. However, this scheme inherits disadvantages from the classifications used in different level of hierarchy.

Security patterns that are located at the internal nodes of the hierarchical tree support more security concepts and objectives than those which are at the leaves. For example, "defense in depth" pattern is at the root node in case of a hierarchical classification based on the application context and the STRIDE model.

J. *Classification based on Transactions*

Arteaga *et al.* [27] propose a classification of security patterns for secure transactions with the web service. They consider nine specific security patterns for providing secure transactions. These security patterns are responsible to provide a secure communication between senders and receivers. This classification helps in the selection of security patterns to design the security model for avoiding security risks related to transactions between a requester and a web service. The classification provides the identification of suitable security patterns. These patterns provide high efficiency and performance for maintaining security objectives confidentiality, integrity, availability, accountability, and authentication at the different stages of a transaction. The classification supplies a reliable security model to software designers for transaction-level security.

K. *Classification based on Development Phases and Security Objectives*

Yskout and Heyman *et al.* [6] provide a classification based on the development phases that use the perspective of applications and systems. They classify security pattern landscape based on the application architecture, design, and system. They further classify security patterns based on the security objectives: confidentiality, integrity, availability, accountability, non-repudiation, anonymity, and privacy. Moreover, the authors include security sub-objectives, *i.e.*, secure data transmission, controlled access, and identification. The primary security objectives CIAA (confidentiality, integrity, availability, and accountability) are extended to complete a set of security objectives used for the classification. However, Yskout and Heyman *et al.* [6] could not find any security pattern as an example for fulfilling the security objectives such as non-repudiation, anonymity, and privacy.

L. *Classification based on Logical Architecture Tiers*

Steel *et al.* [15] describe the classification of security patterns based on a tiered approach. They classify patterns according to logical architectural tiers. Security patterns are categorized based on the support they provide to the system's layers. They classify landscape of security patterns based on four tiers: web, business, web service, and identity.

M. *Classification based on Core Security Viewpoints*

Rosado *et al.* [28] provide a classification of architectural patterns with a security perspective using Avgeriou and Zdun [30] concepts for design patterns. They provide a set of security viewpoints to depict architectural security patterns. Their catalog contains seven core security views, *i.e.*, logical, process, development, physical, deployment, operational, and misuse case. Therefore, they categorize the architectural security pattern catalog into seven viewpoints. The idea of viewpoints is further divided into sub-viewpoints using a security pattern template.

N. *Classification based on Formal Concept Analysis*

Sarmah *et al.* [30] use linguistic metaphors and formal concept analysis (FCA) for the classification of security patterns. They adopt this approach from Hasso and Carlson's [31] work on design patterns. The FCA is a formal model to organize security patterns. They use a trust-based security model, trust elements, and security patterns. These elements are used with their formal concepts leading to a security pattern lattice. The result is a categorization of security patterns using the FCA technique of scaling. The authors use this approach to reduce redundancy among security patterns.

O. *Classification based on Software Lifecycle*

Yoshioka *et al.* [9] describe a security pattern classification based on software lifecycle phases. The classification provides the required corresponding security patterns based on software lifecycle phases (*i.e.*, requirement, design, and implementation) to solve recurring security problems. The authors discuss the security patterns on phase bases; however, they do not mention the hierarchy of security patterns within phases. Therefore, every phase needs to be classified in more detail in order to solve the selection problem.

P. *Classification based on Architectural Concerns, Layers, and Linguistic Similarities*

Fernandez *et al.* [8] use the software architecture classification approach from Avgeriou and Zdun [29] to classify architectural patterns addressing the types of concerns and propose three viewpoints for the classification of security patterns: architectural concern, architectural layer, and linguistic similarity (relationship between patterns). A classification dimension according to architectural concerns means patterns for access control, cryptography, file control, identity. From architectural point of view, the whole computer system is the hierarchy of layers. A classification dimension according to architecture layers means the services of the database, operating systems, and hardware layers. Security patterns in all layers ensure security in the software system. Security patterns and their interrelation are based on the system layer connections or interrelations. Furthermore, the authors apply an automated relationship analysis technique for patterns. This technique uses the text processing techniques called the "term frequency–inverse document frequency" and "vector space model." The automated relationship analysis technique extracts patterns from documents and finds the pattern relationships based on document similarities. The relationship extraction is helpful in the selection process of security patterns.

Fernandez *et al.* [8] combine and express architectural concern, architectural layer, and linguistic similarity viewpoints using pattern diagrams. Pattern diagrams summarize the whole picture of relevant patterns and their interrelationships. They provide easy navigation to designers for pattern selection.

Q. *Classification based on Categorization for Users*

VanHilst *et al.* [32] propose a classification of security patterns based on various security concerns that developers may face in many phases of software lifecycle. They describe six primary dimensions to classify security patterns independently. Security patterns are classified in the following dimensions: lifecycle stages, component source types, security response types, architecture layers, constraint levels, and domain types. The six dimensions in the classification are further divided into the regions of concern.

These regions are further classified as hierarchical, disjoint, or overlapping.

R. *Classification based on Dimension and Pattern Graphs*

Washizaki *et al.* [33] improves the classification based on "categorization for users" (as mentioned in Section Q) by introducing two new models: pattern and dimension graphs. They develop a meta-model using security patterns and classification dimensions. The meta-model is used to describe the relationships of pattern-to-dimension and pattern-to-pattern uniformly. They use a dimension graph to describe the security pattern relationships to several classification dimensions. Further, they use a pattern graph to describe the security pattern relationships to each other. For the dimension graph, they select the following six dimensions: lifecycle stage, concern, type of pattern, architectural level, constraint, and domain.

S. *Classification using Ontology-based Approach*

Khoury *et al.* [34] introduce a classification based on ontological interfaces. The classification is based on the profiles of software developers. Software developer profiles are differentiated based on designing and coding skills. A "design-oriented developer" profile shows designing skills for designing several programming tasks. An "implementation-oriented developer" profile shows coding skills for software development tasks. The selection of a security pattern depends on ontological mapping between requirements (using the ontological interface for the design-oriented developer profile) and threat models, security bugs and errors (using the ontological interface for implementation-oriented developer profile).

T. *Classification based on Text Processing Approach and Learning Techniques*

Hasheminejad *et al.* [35] propose text processing and learning techniques for the classification of security patterns. They automate the process for identifying the security pattern related to the security requirements. They have found that Naive Bayes is the best learning technique for the selection method of security patterns. This research automates the paradigm of a classification or selection process and gets the appropriate security pattern for the requirement.

U. *Classification based on Levels of Abstraction*

Dougherty *et al.* [3] describe the categorization of secure design patterns using the three classes based on the levels of abstraction, *i.e.*, the architecture, design, and implementation. They classify 15 security design patterns: three secure design patterns for architectural-level, six secure design patterns for design level, and six secure design patterns for implementation level.

V. *Classification based on Attack Patterns*

Wiesauer *et al.* [13] define a security pattern classification based on attack patterns employing the STRIDE model. They use the common attack pattern enumeration and classification (CAPEC) catalog [36]. They map security patterns to attack patterns of the CAPEC catalog. The important implication of this taxonomy is finding security loopholes through the attack patterns and blocking them by using the corresponding security patterns. Therefore, the classification is given as a rational approach for the selection of security patterns to counter the security loopholes.

W. *Classification based on Application Domain*

Bunke *et al.* [37] surveyed the whole bandwidth of security patterns from 1997 to 2010, extensively. They identify 360 security patterns and propose a classification of security patterns based on the application domain concept. They consider the classification natural, if it is categorized according to a discipline or application domain. They first completely scan the security pattern landscape and collect the keywords: user, password, operating system, enterprise, or process. They extract the keywords from patterns to unify common groups of patterns. In this way, they found five pattern groups: enterprise, software, cryptographic, user, and network.

Bunke *et al.* [37] propose the expansion of the classification by developing a classification using view points at the second level where the first level classification uses the application-domain concept.

X. *Classification based on Security Flaws*

Alvi and Zulkernine [40] provide a natural classification scheme for software security patterns. They propose a classification of software security patterns based on security flaws. The classification scheme is associated with software lifecycle phases. Security flaws are incorporated in the classification of software security patterns with security objectives in the requirement phase, security properties in the design phase, and attack patterns in the implementation phase. The classification scheme uses the software weakness taxonomy (called common weakness enumeration [42]), attack pattern taxonomy (called common attack pattern enumeration and classification [36]), and sources of software flaw taxonomy [41]. These taxonomies provide security flaws and attack patterns based on the software lifecycle phases.

III. Comparision of Secureity Pattern Classifications

Security pattern classifications are compared with respect to their objectives, attribute descriptions, and dimensions as shown in Table I. The goal of these efforts is to provide ease in the selection of a security pattern classification for software designers or pattern miners from the list of classifications explained from Sub-section A to Sub-section X.

TABLE I. COMPARISON BASED ON OBJECTIVES, ATTRIBUTES, DIMENSIONS AND QUALITY METRICS

Paper References	***Sub-sections***	***Classification Types/Objectives***	***Classification Attribute Descriptions***	***Classification Dimensions***	***Desirable Classification Quality Metrics* [39]**								
					***Navigability* [38]**	***Completeness***	***Acceptability***	***Comprehensibility***	***Determinism***	***Mutual Exclusivity***	***Repeatability***	***Unambiguity***	***Usefulness***
[7]	*A*	Structural and procedural-oriented.	Classification based on product and process.	M		X	X	X	X	X	X	X	
[18]	*B*	Creational, structural, and behavioral.	Classification based on purpose.	M			X	X	X	X	X	X	
[2]	*C*	General.	Classification using abstraction, software lifecycle, and problem domain.	M	X	X	X	X	X		X	X	
[12]	*D*	System-oriented.	Classification based on applicability.	M			X	X	X	X	X	X	X
[14]	*E*	Enterprise architectural-oriented.	Classification based on organizing table (tabular classification).	M		X	X	X	X	X	X	X	X
[10]	*F*	General.	Classification based on undesirable properties.	S					X		X	X	
[1]	*G*	Security views-oriented.	Classification based on the context of security views using security taxonomy.	M		X	X		X	X	X	X	X
	H	Enterprise framework-oriented.	Classification based on the context of an enterprise framework	M	X	X	X		X	X	X	X	X
[11]	*I*	Partitioning schemes-oriented.	Classification based on the CIA model.	M		X	X	X	X		X	X	X
			Classification based on the application context.	M			X	X	X		X	X	X
			Classification based on the security wheel.	M		X	X		X			X	X
			Classification based on the McCumber cube.	M		X	X		X	X	X	X	X
			Classification based on the threat modeling framework.	M			X		X		X	X	X
			Classification based on the hierarchy.	M	X	X	X	X	X		X	X	X
[27]	*J*	Transaction-oriented.	Classification based on transactions.	M	X		X		X	X	X	X	X
[6]	*K*	General.	Classification based on development phases and security objectives.	M	X	X	X	X	X	X	X	X	X
[15]	*L*	Architectural-oriented.	Classification based on logical architecture tiers.	M			X	X	X	X	X	X	X
[28]	*M*	Architectural-oriented.	Classification based on core security viewpoints.	M	X		X			X	X	X	X
[30]	*N*	General.	Classification based on formal concept analysis.	S	X	X	X			X	X	X	
[9]	*O*	General.	Classification based on software lifecycle.	M		X	X	X	X	X	X	X	X
[8]	*P*	Architectural-oriented.	Classification based on architectural concerns, layers, and linguistic similarities.	M	X	X	X		X	X	X	X	X
[32]	*Q*	General.	Classification based on categorization for users.	M	X	X	X		X		X	X	
[33]	*R*	General.	Classification using dimension and pattern graphs.	M	X	X	X		X		X	X	
[34]	*S*	Ontology-oriented.	Classification using ontology-based approach.	S		X	X		X		X		X
[35]	*T*	Automation-oriented.	Classification based on text processing approach and learning techniques.	S		X			X		X		X
[3]	*U*	Architectural design-oriented.	Classification based on three classes on the levels of abstraction.	M			X	X		X	X	X	X
[13]	*V*	General.	Classification based on attack patterns.	S		X	X	X	X		X		X
[37]	*W*	General.	Classification based on application domain concept.	S		X	X	X	X	X	X	X	
[40]	*X*	Software lifecycle-oriented.	Classification based on security flaws using software lifecycle phases.	M	X	X	X	X	X		X	X	X

Legends:
The symbol "X" shows that the quality metric is significant in a classification.
'S' represents single and 'M' represents multiple attributes.

In Table I, the first column provides the list of security pattern classification paper references. The second column lists the corresponding work discussed in Section II. In the third, fourth, and fifth columns security pattern classifications are compared with respect to their types/objectives, attribute descriptions, and dimensions, respectively. The last column is used for desirable classification quality metrics. It is divided into nine sub-columns to represent nine desirable quality metrics for the security pattern classifications.

The classification that has the relevant classification attributes and satisfies the appropriate number of desirable classification quality metrics may provide reasonable benefit for software designers to use it in the selection process of security patterns. We consider the following nine desirable classification quality metrics: navigability [38], completeness, acceptability, comprehensibility, determinism, mutual exclusivity, repeatability, unambiguity, and usefulness [39]. These classification quality metrics are taken from [38, 39] and defined as follows.

- Navigability provides an ability to direct a software designer among collaborative and related patterns.
- Completeness of a classification schema accounts for covering all security patterns and provides categories accordingly.
- An acceptable classification schema should be structured in a way that it provides help in partitioning the security pattern landscape and becomes generally approved.
- A classification is comprehensive only, if it is easily understood by both a novice and expert developer.
- The clear definition of the process of a classification of security patterns makes a classification deterministic.
- The ability to categorize each security pattern into at most one category makes a classification mutually exclusive.
- A classification should be repeatable and reusable over time.
- Each category of a classification must be clearly defined so that there is no ambiguity left in the classification of security patterns.
- The usefulness of a classification is defined by its ability to be used in a secure software development industry, especially by a development team.

Table I lists the comparison among classifications using quality metrics. The selection (shown by symbol “X” in Table I) of the quality metrics is used to evaluate the classifications subjectively. This is useful for the selection of an appropriate security pattern. For example, using Table I, we can instantly get the knowledge about the classification proposed by Rosado et al. 2007 [28], *i.e.*, it is architecture-oriented and based on core security viewpoints with multiple dimensions and possesses navigability, acceptability, mutual exclusivity, repeatability, unambiguity, and usefulness qualities. A classification based on a single dimension represents a direction of classification using one classification attribute, on the other hand, a classification based on multiple dimensions utilize two or more classification attributes.

CONCLUSION

We have presented a survey of the existing classifications of security patterns. At the end of the survey, the comparison of the classifications has been provided using classification objectives, attribute descriptions, dimensions, and quality metrics. The survey provides a bird’s-eye view to the current status of the existing classification schemes. The analysis of the classification schemes provides the selection choices for a scheme to software designers based on desirable security requirements. The classification scheme that matches with software design goals provides appropriate security patterns for solving the concerned security problems in the development of the secure software system.

Security patterns are spread as a heterogeneous landscape in many directions. For classifying the comprehensive list of security patterns, it may be fruitful to use multiple dimensions with disjoint or non-redundant partitions of the security pattern collection. However, the classification quality metrics should be maintained as much as possible. The appropriate selection of a classification scheme will enhance the possibility of fast and accurate selection of security patterns. In addition, combining a number of security patterns to achieve desired complex security properties is an open problem.

ACKNOWLEDGMENT

This work is partially supported by Bell Canada and the Ontario Research Fund (ORF).

REFERENCES

[1] M. Schumacher, E. Fernandez-Buglioni, D. Hybertson, F. Buschmann, and P. Sommerlad, *Security Patterns: Integrating Security and Systems Engineering.* Chichester, West Sussex, England: John Wiley & Sons Ltd., 2006.

[2] M. Schumacher and U. Roedig, “Security Engineering with Patterns,” In *Proc. of the 8th Conference on Pattern Languages of Programs*, Illinois, Sept. 2001.

[3] C. Dougherty, K. Sayre, R. C. Seacord, D. Svoboda, and K. Togashi, “*Secure Design Patterns*,” Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Tech. Rep. CMU/SEI-2009-TR-010, 2009.

[4] J. Yoder and J. Barcalow, “Architectural Patterns for Enabling Application Security,” In *Proc. of Pattern Languages of Programs*, Illinois, 1997.

[5] S. Romanosky, “*Security Design Patterns: Part 1*,” Online at CGISecurity, version 1.4, Nov. 2001. Available: http://www.cgisecurity.com/lib/securityDesignPatterns.html.

[6] K. Yskout, T. Heyman, R. Scandariato, and W. Joosen, “*A System of Security Patterns*,” Department of Computer Science, Catholic University of Leuven, Celestijnenlaan, Heverlee, Belgium, Tech. Rep. CW-469, 2006.

[7] D. M. Kienzle, M. C. Elder, D. Tyree, and J. Edwards-Hewitt, *Security Patterns Repository, Version 1.0*, 2002. Available: http://www.scrypt.net/~celer/securitypatterns /repository.pdf

[8] E. B. Fernandez, H. Washizaki, N. Yoshioka, A. Kubo, and Y. Fukazawa, “Classifying Security Patterns,” In *Proc. of the 10th Asia-*

Pacific Web Conference, Shenyang, China. *Springer LNCS 4976*, pp.342-347, April 2008.

[9] N. Yoshioka, H. Washizaki, and K. Maruyama, "A Survey on Security Patterns," *Progress in Informatics*, vol. 5, pp. 35-47, 2008.

[10] M-A. Laverdiere, A. Mourad, A. Hanna, and M. Debbabi, "Security Design Patterns: Survey and Evaluation," In *Canadian Conference on Electrical and Computer Engineering*, Ottawa, ON, pp.1605-1608, May 2006.

[11] M. Hafiz and E. J. Ralph, "Security Patterns and their Classification Schemes," University of Illinois at Urbana Champaign, Dept. of Computer Science, Tech. Rep., August 2006. Available: https://netfiles.uiuc.edu/mhafiz/www/research/patterns/secpat classify.pdf.

[12] B. Blakley, C. Heath, and members of The Open Group Security Forum, "*Security Design Patterns*," The Open Group, Tech. Rep. G031, April 2004.

[13] A. Wiesauer and J. Sametinger, "A Security Design Pattern Taxonomy based on Attack Patterns," In *International Joint Conference on e-Business and Telecommunications*, INSTICC Press, Milan, Italy, pp. 387-394, July 2009.

[14] D. Trowbridge, W. Cunningham, M. Evans, L. Brader, and P. Slater, "Describing the Enterprise Architectural Space," *MSDN*, June 2004.

[15] C. Steel, R. Nagappan, and R. Lai, Core Security Pattern: Best Practices and Strategies for J2EE, Web Services, and Identity Management. Upper Saddle River, NJ: Prentice Hall PTR / Sun Micros, 2006.

[16] A. Sarmah, S. M. Hazarika, and S. K. Sinha, "Security Pattern Lattice: A Formal Model to Organize Security Patterns," In *International Workshop on Database and Expert Systems Application*, Turin, Italy, pp. 292-296, Sept. 2008.

[17] D. M. Kienzle and M. C. Elder, "*Final Technical Report: Security Patterns for Web Application Development*," Defense Advanced Research Projects Agency, DARPA Contract. F30602-01-C-0164, 2002.

[18] B. H. C. Cheng, S. Konrad, L. A. Campbell, and R. Wassermann, "Using Security Patterns to Model and Analyze Security Requirements," In *IEEE Workshop on Requirements for High Assurance Systems*, Monterey, California, pp. 13-22, Sept. 2003.

[19] R. Wassermann and B. H. C. Cheng, "*Security Patterns*," Software Engineering and Network Systems Laboratory, Department of Computer Science and Engineering Michigan State University, Tech. Rep. MSU-CSE-03-23, August 2003.

[20] E. Gamma, R. Helm, R. Johnson, and J. M. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented So,mftware*. Upper Saddle River, New Jersey: Addison-Wesley Professional, 1995.

[21] J. Viega and G. McGraw, *Building Secure Software: How to Avoid Security Problems the Right Way*. 1st ed., Upper Saddle River, New Jersey: Addison-Wesley Professional, 2001.

[22] J. A. Zachman, "A Framework for Information Systems Architecture," In *IBM Systems Journal*, vol.38, no.2-3, pp. 454-470, 1999.

[23] IEEE Recommended Practice for Architectural Description of Software-Intensive Systems, IEEE Standard 1471-2000, 2000.

[24] M. Goodyear, *Enterprise System Architectures: Building Client Server and Web Based Systems*. 1st ed., Boca Raton, Florida: CRC Press, 1999.

[25] K. Beck and C. Andres, *Extreme Programming Explained: Embrace Change*. 2nd ed., Upper Saddle River, New Jersey: Addison-Wesley Professional, 2004.

[26] J. McCumber, "Information Systems Security: A Comprehensive Model" In *Proc. of 14th National Computer Security Conference*, National Institute of Standards and Technology (NIST), Washington, D.C., 1991.

[27] J. M. Arteaga, R. M. Gonzalez, F. J. Alvarez, and M. V. Martin, "A Classification of Security Patterns for the Transactions between a Requester, an Intermediary, and a Web-service," In *Proc. of the Communication, Network, and Information Security*, Cambridge, Massachusetts pp. 132-137, Oct. 2006.

[28] D. Rosado, C. Gutierrez, E. Fernández-Medina, and M. Piattini, "Defining Security Architectural Patterns Based on Viewpoints," In *Computational Science and Its Applications*, vol. 4707, Springer Berlin / Heidelberg, pp. 262-272, 2007.

[29] P. Avgeriou and U. Zdun, "Architectural Patterns Revisited–A Pattern Language," In *Proc. of the 10th European Conference on Pattern Languages of Programs*, Irsee, Germany, July 2005.

[30] A. Sarmah, S. M. Hazarika, and S. K. Sinha, "Security Pattern Lattice: A Formal Model to Organize Security Patterns," In *International Workshop on Database and Expert Systems Application*, Turin, Italy, pp. 292-296, Sept. 2008.

[31] S. Hasso and C. R. Carlson, "A Theoretically-based Process for Organizing Design Patterns," In *Proc. of the 12th Pattern Languages of Programs*, Monticello, Illinois, Sep. 2005.

[32] M. VanHilst, E. B. Fernandez, and F. Braz, "A Multi-dimensional Classification for Users of Security Patterns," *Journal of Research and Practice in Information Technology*, vol. 41, pp. 87-98, 2009.

[33] H. Washizaki, E. B. Fernandez, K. Maruyama, A. Kubo, and N. Yoshioka, "Improving the Classification of Security Patterns," In *20th International Workshop on Database and Expert Systems Application*, pp. 165-170, Liz, Austria, Sep. 2009.

[34] P. E. Khoury, A. Mokhtari, E. Coquery, and M.-S. Hacid, "An Ontological Interface for Software Developers to Select Security Patterns," In *Proc. of 19th International Workshop on Database and Expert Systems Application*, pp. 297-301, Turin, Italy, 2008.

[35] S. M. H. Hasheminejad and S. Jalili, "Selecting Proper Security Patterns Using Text Classification," In *International Conference on Computational Intelligence and Software Engineering*, Wuhan, China, pp. 1-5, Dec. 2009.

[36] Common Attack Pattern Enumeration and Classification (CAPEC), "CAPEC List Release 1.6," Available: http://capec.mitre.org/.

[37] M. Bunke, R. Koschke, and K. Sohr, "Application-Domain Classification for Security Patterns," In *Proc. of the 3rd International Conferences on Pervasive Patterns and Applications*, pp. 138-143, Rome, Italy, Sep., 2011

[38] M. Hafiz, P. Adamczyk, and R. E. Johnson, "Organizing Security Patterns," In *IEEE Software*, vol.24, no.4, pp.52-60, 2007.

[39] S. Hansman and R. Hunt, "A Taxonomy of Network and Computer Attacks," Computers Security, vol. 24, issue 1, pp. 31-43, Elsevier, 2005.

[40] A. K. Alvi and M. Zulkernine, "A Natural Classification Scheme for Software Security Patterns," In *IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing, IEEE*, Computer Society, pp. 113-120, Sydney, Dec. 2011.

[41] F. Dörenberg, "Dependability Impairments: Faults, Errors and Failures," in Analysis and Synthesis of Dependable Electronic Systems, unpublished, 2003. http://www.nonstopsystems.com/ textbook_sample_1.pdf

[42] Common Weakness Enumeration (CWE), "CWE-2000: CWE Dictionary (2.1) ," Available: http://cwe.mitre.org/.