Application-Domain Classification for Security Patterns

Michaela Bunke, Rainer Koschke, and Karsten Sohr
Center for Computing Technologies (TZI),
Universität Bremen, 28359 Bremen, Germany
{mbunke|koschke|sohr}@tzi.de

***Abstract*—Security patterns are best practices to handle recurring security problems. Existing classifications for security patterns consider only a small number of patterns, and their purpose is often focused on implementations issues. Therefore we identify missing aspects in existing classifications and introduce a new classification scheme based on application domains. This scheme is based on a literature survey on security patterns published in the period of 1997 to 2010 to cover the whole bandwidth of exiting security pattern.**

***Index Terms*—Security Patterns.**

I. Introduction

Software security is an emerging area in software development. More and more vulnerabilities are published and compromise systems and their users [1]. Software designers and programmers are therefore faced with applying security solutions to software systems. In the domain of software development, design patterns have been proposed as specific solutions for recurring problems in software design [2].

Yoder and Barcalow summarized some existing patterns targeting security and introduced the term *security pattern* [3], only three years after Gamma et al. [2] proposed their design patterns. Security patterns are best practices aiming at ensuring security [4], [5].

Existing security pattern classifications are often based on a few security patterns. Their scope is often limited to special areas such as implementation patterns. For instance, Hafiz et al. formed their classification with only 14 security patterns [6], but there exist many more security patterns.

Therefore, we conducted a systematic literature review and collected the published security patterns in the period of 1997 to 2010. We propose a new classification scheme that summarizes 409 security patterns, a much longer list than the one by Yoder and Barcalow [3]. Moreover, our classification scheme supports the selection by application domain, which is relevant for researchers and practitioners who are interested in security patterns for domain-driven tasks.

Our motivation conducting this literature research and shaping a new classification was to get an overview on existing patterns and organize them in application domains. The research focus of our future research is to detect and validate software security patterns implemented in code. To get started, we need a clear picture of what kinds of security patterns exist.

The remainder of this paper is structured as follows. An overview of classifications in general is given in Section II. Existing classification approaches for security patterns will be described in Section III. In Section IV, we describe our literature survey and will introduce our new classification. Afterwards, we will conclude in Section V.

II. Requirements for Classifications

The increasing number of patterns makes it necessary to develop classifications. This sections describes general requirements for classifications in general and on security patterns in particular.

A classification should be based on systematic methodologies and techniques to organize a mass of patterns. A classification organizes patterns into groups of patterns that share one or many properties such as the application domain or a particular purpose. The kind of properties that should be used is not fixed. A pattern can have more than one specific property. Therefore, it may be included in more than one classification category.

According to Buschmann et al., a pattern classification scheme should meet some basic properties [7]. It must too be simple and easy to learn. This should be supported by using only few classification criteria to reduce the complexity and ambiguity for users. In addition, a classification should reflect the main properties of a pattern to classify it. Last but not least, a classification scheme should provide the possibility to classify new patterns.

Fernandez et al. pointed out that a classification should make the application of patterns much easier along the software lifecycle [8]. Due to the fact that it is impractical to look at details of a pattern to pick the right pattern for the problem at hand, a classification should help to understand the essential nature and value of patterns.

A simple and intuitive classification is shown in Figure 1. A natural way to classify pattern is to categorize them according to the discipline where they are applied. Patterns can also be distinguished by their application domain like network, embedded systems or distributed systems. Another way to differentiate patterns is to determine their programming concepts paradigms such as object-oriented or imperative programming language concepts. Moreover, it is possible to categorize them depending on the level of abstraction they address, e.g., design or coding patterns. In addition, the criterion *purpose* represents the kind of problem a pattern solves and when it may be applied.

 ISBN: 978-1-61208-158-8

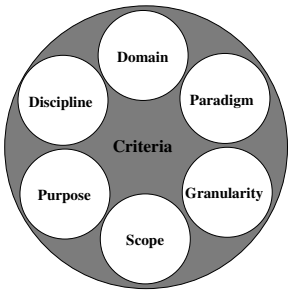

Fig. 1. Intuitive classification

III. Existing Classifications

The presented classification criteria in Section II are simple, but do not always fit for selecting the right pattern for a special purpose because of their generality. Therefore more specific classification schemata based on one or more criteria have been developed to meet special purposes like the *structural* and *behavioural* distinction on object creation and usage by Gamma et al. [2].

One of the simplest classifications for security patterns was used by Kienzle et al. [9]. They presented the *structural* and *procedural* criteria for the differentiation of the patterns described in their final report. If a security pattern is concerned about compositions or structures that are implemented in a final software product, it is *structural*. Otherwise a security pattern is *procedural*.

Konrad et al. [10] proposed a classification method for security patterns re-using the classification for design patterns such as *creational*, *structural* and *behavioural* from Gamma et al. [2]. They enhanced their classification by adding further categories such as network, host, and application. In their work, they considered only the security patterns introduced by Yoder and Barcalow [3].

Schumacher's security patterns book offers a new classification system [11]. The classification is based on Zachman's framework [12] for enterprise architecture. It is presented along two dimensions. One dimension represents different views on the interrogatives "what", "how", "where", "who", "when", and "why". The second dimension shows different information model views such as *business model* or *technology model*. Schumacher et al. enhanced this framework by adding the column *security* to emphasize the security view and to be able to address all model levels. They organized only the patterns contained in the book into their classification.

According to the Java Platform, Enterprise Edition (JEE) pattern classification by Alur et al. [13], Steel et al. [14] classify their JEE security patterns in a similar way. They separate their patterns in layers that are typical for the development in the JEE domain such as *Web*, *Business*, *Web Service* and added a fourth tier that represents the special issue of *identity management*. This classification is only designed for the special purpose of JEE patterns and does not consider other types of patterns.

Rosado et al. related security requirements to security patterns and classified security patterns into two categories: architectural and design patterns [15].

Hafiz et al. proposed a classification based on a tree structure combined with the STRIDE-Model [16] to join the software and security view in terms of security patterns [6]. The STRIDE-Model is normally used for threat modelling including identifying and prioritization of security vulnerabilities. They tested their classification with 14 different security patterns, and compared it against other available security classifications.

VanHilst et al. introduced a multi-dimensional matrix of concerns to classify security patterns [17]. It addresses the problem coverage and pattern classification. Their idea was that each matrix dimension represents a well-defined list of concerns, which is presented along one single axis. To classify security patterns, the primary dimension contains concerns of life-cycle activities like *domain analysis* or *requirements*. The second dimension differentiates security patterns by their component source type such as *new code*, *legacy*, or *wizard-code*. Other dimensions may hold types of security responses like prevention or mitigation, but they can also be further customized to a user's need. Their classification was tested by different members of their team, who added six different security patterns to the classification.

Fernandez et al. state that security patterns are architectural patterns [18]. Therefore, their approach deals with two classifications that differ in different viewpoints of security patterns. On the one hand, they introduced a classification by a hierarchy of layers and on the other hand, they proposed a classification based on the relationships between patterns by using an automatic relationship extraction and analysis technique. This classification is abstract and only regards a small number of security patterns.

Washizaki et al. point out that the previously introduced classifications have only a few dimensions and do not embrace the relations between patterns [19]. Therefore, they introduce a meta-model to express the patterns' properties and relations uniformly. The base is an excerpt of the multidimensional classification dimensions presented by VanHilst et al. [17]. They picked out the dimensions as follows: *Lifecycle stage*, *Architectural level*, *Concern*, *Domain*, *Type of pattern* and *Constraint*. In addition, they used the three UML standard relationship types *association*, *generalization*, and *aggregation* to model relationships between security patterns, for example, a *Firewall* pattern [11] is the generalization of an *Address Filter Firewall* [20] and an *Application Firewall* [21] pattern.

They also propose two instances for the meta-model which represents two points of view, namely pattern-to-pattern relations, represented as a pattern graph, and pattern-to-dimension relations modelled as a dimension graph. They tested their approach with only eight different security patterns, which are close to implementation patterns.

To summarize, the published classifications take only a small number of security patterns into account. The used security patterns are often very similar to the first introduced patterns by Yoder and Barcalow [3]. Thus they do not consider

 ISBN: 978-1-61208-158-8

other types of security patterns such as enterprise patterns and mostly imply that only programming issues are covered by security patterns. Due to the fact that many people, who are involved in the software life-cycle, are rarely security experts and not aware of security issues [22], the classification should be easy to use. Based on these points, we will form a new classification for this purpose.

IV. New Classification for Security Patterns

Our goal is to present a classification that covers the heterogeneity of the published security patterns till today. It unifies the existing patterns in a common scheme. In addition, not every task needs information about attack surfaces or vulnerability classification properties like STRIDE or other facets that are introduced in Section III. On that account we omit specialized criteria like STRIDE and focus on obvious differences among the security patterns. With this in mind, we develop a new classification with a more general perspective based on a domain criterion (see Section II) and the security patterns we collected in our systematic literature review.

A. Systematic Literature Review

This section describes how the literature survey was conducted systematically [23], [24]. We start our literature research with the surveys by Laverdiere et al. [25], Heyman et al. [26] and Yoshioka et al. [27]. Additionally, we considered common pattern-related conferences and found 1150 articles (see Table I). In addition, we looked for security patterns in the IEEE Digital Library [28] and the ACM Digital Library [29] and took two books about security patterns [11] [14] into account.

We skimmed the conferences and electronic publications of the years 1997 to 2010 for several keywords such as cryptographic, security, software or secure. At first we picked out all publications that contain these keywords, secondly we read the abstract if it described the presentation of a security pattern. Finally, we read the publications, which are not filtered out previously to verify that they describe security patterns. Moreover, we scanned and collected the referenced publication of each identified pattern for further readings.

In summary, we identified 63 different publications describing security patterns, including books, journals, proceedings, and technical reports. Most of them were found by looking at the Hillside Group [30] pattern conferences such as PLoP and EuroPLoP (see Figure 2 and Table I) and books. Another publication type containing many security patterns were technical reports discovered by cross-references. New conferences that have only few security pattern publications are also discovered by cross-references such as International Conference on Emerging Security Information, Systems and Technologies (SECURWARE), Working Conference on Data and Applications Security (IFIP WC 11.3), International Conference on Internet Computing (ICOMP), and International Workshop on Security, Trust and Privacy in Grid Systems (GRID-STP). The search at the ACM and IEEE Digital Library produced many false-positive articles that were at a closer look

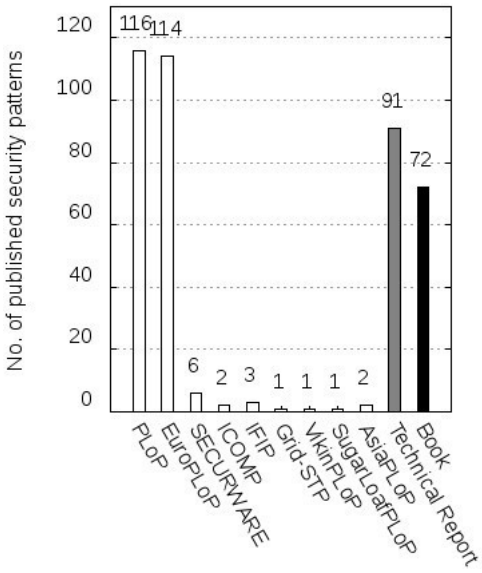

Fig. 2. Distribution of security patterns across different venues; white bars denote conferences, the grey bar technical reports, and the black bar books.

no security pattern descriptions but deal with them in other ways like discussing secure software design in practice [31].

Some publications describe more than one pattern. In total, we got 409 security patterns. This list depicted that some of these patterns have been described twice. Hence, we filtered out duplicates and reduced the number of patterns to 360. These duplicates were identified by the use of similar names and then comparing their descriptions. Because of the abundance of patterns, we were not able to check in depth whether two patterns with different names relate to the same concept. Based on this heterogeneous security pattern collection we formed our classification bottom-up in contrast to the aforementioned classifications in Section III.

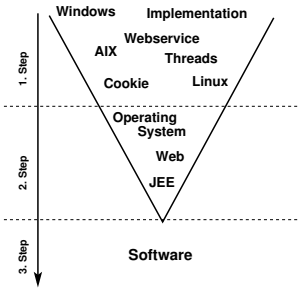

Fig. 3. Proceeding steps in our classification modelling.

B. Organizing by Application Domain

First of all, we skimmed over the data and collected keywords for the security patterns such as user, password, operating system, enterprise or process. These keywords were inspired by information we found in the pattern descriptions.

In the next iteration, we went through the pattern list and extracted keywords for the patterns. On further reading these

Acronym	Description	Total no. of articles 1997 to 2010	Articles describing security patterns
PLoP	Conference on Pattern Languages of Programs	427	26
EuroPLoP	European Conference on Pattern Languages of Programs	395	18
VikingPlop	The Nordic Conference on Pattern Languages of Programs	57	1
KoalaPLoP	Australian Conference on Pattern Languages of Programs	17	0
AsianPLoP	Asian Conference on Pattern Languages of Programs	14	1
SugarLoafPLoP	American Conference on Pattern Languages of Programming	156	1
PATTERNS	International Conferences on Pervasive Patterns and Applications	14	0

TABLE I
CONFERENCES INVOLVED IN THE INITIAL ARTICLE SELECTION.

keywords were unified into common groups. For instance, we united the keywords *AIX*, *Linux* and *Preforking* to the group *Operating System*. The result contains a mixture of purpose and domain criterion. We formed 13 different groups this way. To further simplify the classification along the lines like described in Section II, these keywords were further condensed to form an application-domain based distinction, which is easy to understand and intuitively applicable (see Figure 3). Finally, Figure 4 depicts the five target application domains which were discovered: *Enterprise*, *Software*, *Cryptographic*, *User*, and *Network*. They are described in the following in more detail.

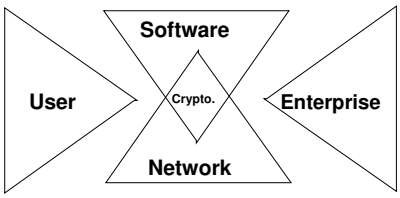

Fig. 4. Application-domain based classification.

Enterprise security patterns deal with aspects that are important for enterprises to ensure security in several enterprise segments like third party communication with suppliers. This means security in processes, physical authentication to several areas, risk mining or securing communication in inter- and external businesses. A good example of this pattern type is the *Manage Risk* pattern introduced by Elsinga and Hofman [32]. The problem addressed by this pattern is as follows "What is the right (combination of) paradigm(s) to formulate the corporate security strategy in order to select and implement the appropriate set of security safeguards?" The pattern suggests to instruct people and units to pay attention on known and unknown risks to develop prevention and roll-back strategies.

Network security patterns picture network infrastructures and their ideal composition, for example, using a *Packet Filter Firewall* to shield an internal network from Internet attacks or just tunnelling the communication traffic though a single controllable instance [11].

User security patterns are focused on user behaviour or awareness of security issues, for example, the *password lock box* pattern, which encourages the user to protect master passwords with the highest level of security [38]. It stresses the significance of protecting master password files and depict situations where such a file can be useful.

Software security patterns describe mostly how to structure parts of software to ensure security requirements. Sometimes they also describe a specific behaviour or way to manage or control a data flow in a secure way. On one hand, patterns in this domain can be very specific like JEE patterns, which can be applied only at Java enterprise applications [14]. On the other hand, patterns in this domain can be more general like the *Single Access Point* pattern, which models a kind of login structure that can be be found in several software systems like UNIX, ICQ or Twitter [3].

Cryptographic security patterns depict secure communication between two applications over a network. They are often described abstractly. Therefore, it is not clear whether these patterns reside in the *Network* or *Software* domain. Their implementation or application is possible in both domains. On that account, we see them as a part of network and software in our classification (see Figure 4). An example is the *Sender Authentication* pattern. It presents the problem and solution how to guarantee that a received message has been sent by a person one expected [41]. Obviously, such a pattern can be applied at network level (Level 3 and 4) or at application level and depending on that it resides on the *Network* or *Software* application domain.

The aforementioned classifications in Section III cover only

Application Domain	Publications Describing Security Patterns	Total no. of Security Patterns
Enterprise	[33], [34], [32], [35], [36], [37], [11]	84
User	[38], [39], [40]	23
Cryptographic	[41], [42], [43], [44]	35
Network	[45], [46], [21], [47], [48], [49], [50], [51], [52], [20], [53], [54], [55], [56], [57], [11], [58], [59], [60]	56
Software	[45], [50], [61], [62], [63], [64], [65], [66], [67], [68], [69], [70], [71], [72], [73], [74], [75], [76], [77], [78], [37], [79], [40], [11], [80], [14], [81], [3], [82], [62], [83]	162

TABLE II
PUBLICATIONS AND NUMBER OF SECURITY PATTERNS PER APPLICATION-DOMAIN

parts of the fields we discovered. The *Network* domain is partly touched by the classification of Konrad et al. [10]. Schumacher et al. factor *Enterprise* requirements customizable with view points in their classification, but they do not distinguish other domains as our approach does. The domains *User* and *Cryptographic* are not mentioned in the existing classification approaches, although they represent approximately one sixth of the patterns.

Our classification scheme can be tailored further to practical or research interests by employing view points as recommended by Fernandez et al. [18]. For software engineering in particular, applicable patterns are located in the category *Software*, which can be further divided into specific purposes such as pattern detection by using the existing pattern classifications by Gamma et al. [2] or Shi and Olsson [84]. Developing new view points or finer grained classifications to cover new needs in terms of special purposes for one of the application domains is also conceivable.

V. CONCLUSION AND OUTLOOK

In this paper, we presented our systematically literature review on security patterns and introduced a new classification scheme. It embraces 360 published security patterns and exceeds existing classifications by far. The presented classification scheme fulfils the requirements of classifications in the terms of expandability, intuitive use, and is applicable for security laymen.

We hope that this classification will support our ongoing research toward security patterns and reverse engineering. In particular, we plan to investigate software architectures in terms of security pattern usage. Further, we expect that this classification helps other researchers and practitioners with specific application goals focussing on security patterns.

VI. ACKNOWLEDGMENTS

This work was supported by the German Federal Ministry of Education and Research (BMBF) under the grant 01IS10015B (ASKS project).

REFERENCES

[1] The H Security, "Number of critical, but unpatched, vulnerabilities is rising," 2010, (28.08.2011). [Online]. Available: http://www.h-online.com/security/news/item/Number-of-critical-but-unpatched-vulnerabilities-is-rising-1067495.html
[2] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Object-Oriented Software*. Addison Wesley, 1994.
[3] J. Yoder and J. Barcalow, "Architectural patterns for enabling application security," in *Proc. of PLOP*, 1997.
[4] S. Haldikis, A. Chatzigeorigou, and G. Stephanides, "A practical evaluation of security patterns," in *Proceedings of AIDC*, 2006.
[5] M. Hafiz and R. E. Johnson, "Evolution of the mta architecture: the impact of security," *Softw. Pract. Exper.*, vol. 38, no. 15, 2008.
[6] M. Hafiz, P. Adamczyk, and R. E. Johnson, "Organizing security patterns," *IEEE Software*, vol. 24, 2007.
[7] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, *Pattern-Oriented Software Architecture: A System of Patterns*. Wiley, 1996.
[8] E. B. Fernández, N. Yoshioka, H. Washizaki, and J. Juerjens, "Using security patterns to build secure systems," in *Proc. of 1st SPAQu*, 2007.
[9] D. M. Kienzle and M. C. Elder, "Final technical report: Security pattern for web application development," Tech. Rep., 2002, 27.04.2011. [Online]. Available: http://www.scrypt.net/~celer/securitypatterns/final%20report.pdf
[10] S. Konrad, B. Cheng, L. Campbell, and R. Wassermann, "Using security patterns to model and analyze security requirements," in *Proc. of RHAS*, 2003.
[11] M. Schumacher, E. B. Fernandez, D. Hybertson, and F. Buschmann, *Security Patterns: Integrating Security and Systems Engineering*. John Wiley & Sons, 2005.
[12] "The zachmann framework for enterprise architecture," 2011, (28.08.2011). [Online]. Available: http://zachmaninternational.com/2/Zachman_Framework.asp
[13] D. Alur, D. Malks, and J. Crupi, *Core J2EE Patterns: Best Practices and Design Strategies*. Prentice Hall PTR, 2001.
[14] C. Steel, R. Nagappan, and R. Lai, *Core Security Patterns: Best Practices and Strategies for J2EE(TM), Web Services, and Identity Management*. Prentice Hall International, 2005.
[15] D. G. Rosado, C. Gutiérrez, E. Fernández-Medina, and M. Piattini, "Security patterns related to security requirements," in *Proc. of WOSIS*, 2006, pp. 163–173.
[16] F. Swiderski and W. Snyder, *Threat Modeling (Microsoft Professional)*. Microsoft Press, 2004.
[17] M. VanHilst, E. B. Fernandez, and F. A. Braz, "A multi-dimensional classification for users of security patterns," in *Proc. of WOSIS*, 2008, pp. 89–98.
[18] E. B. Fernandez, H. Washizaki, N. Yoshioka, A. Kubo, and Y. Fukazawa, "Classifying security patterns," in *Proc. of Asian-Pacific Web Conference*, Apr. 2008, pp. 342–347.

 ISBN: 978-1-61208-158-8

[19] H. Washizaki, E. B. Fernandez, K. Maruyama, A. Kubo, and N. Yoshioka, "Improving the classification of security patterns," in *Proc. of DEXA*. Los Alamitos, CA, USA: IEEE Computer Society, 2009, pp. 165–170.

[20] E. B. Fernandez, M. M. Larrondo-petrie, N. Seliya, N. Delessy, and A. Herzberg, "A pattern language for firewalls," in *Proc. of PLoP*, September 2003.

[21] S. R. Nelly Delessy-Gassant, Eduardo B. Fernandez and M. M. Larrondo-Petrie, "Patterns for application firewalls," in *Proc. of PLoP*, 2004.

[22] K. van Wyk and G. McGraw, "Bridging the gap between software development and information security," *Security Privacy, IEEE*, 2005.

[23] B. Kitchenham, "Procedures for performing systematic reviews," Keele University, Keele, UK, Technical Report TR/SE-0401, 2004.

[24] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," Keele University, Keele, UK, Technical Report EBSE-2007-001, 2007.

[25] M. Laverdiere, A. Mourad, A. Hanna, and M. Debbabi, "Security design patterns: Survey and evaluation," in *Procs. of CCECE*, 2006.

[26] T. Heyman, K. Yskout, R. Scandariato, and W. Joosen, "An analysis of the security patterns landscape," in *Proc. 3rd SESS*. Washington, DC, USA: IEEE Computer Society, 2007, p. 3.

[27] N. Yoshioka, H. Washizaki, and K. Maruyma, "A survey on security patterns," *Progress in Informatics*, vol. 5, pp. 35–47, 2008.

[28] IEEE Digital Library, 2011, (28.08.2011). [Online]. Available: http://www.computer.org/portal/

[29] ACM Digital Library, 2011, (28.08.2011). [Online]. Available: http://portal.acm.org/

[30] The Hillside Group, 2011, (28.08.2011). [Online]. Available: http://hillside.net

[31] P. Meland and J. Jensen, "Secure software design in practice," in *Proc. of ARES*, march 2008, pp. 1164 –1171.

[32] B. Elsinga and A. Hofman, "Security paradigm pattern language," in *Proc. of EuroPLoP*, 2003.

[33] G. Dallons, P. Massonet, J. f. Molderez, C. Ponsard, and A. Arenas, "An analysis of the chinese wall pattern for guaranteeing confidentiality in grid-based virtual organisations," in *Proc. of Grid-STP*. IEEE, 2007.

[34] P. Dyson and A. Longshaw, "Patterns for managing internet-technology systems," in *Proc. of EuroPLoP*, 2003.

[35] A. M. Ernst, "Enterprise architecture management patterns," in *Proc. of PLoP*. New York, NY, USA: ACM, 2008, pp. 1–20.

[36] E. B. Fernandez, J. Ballesteros, A. C. Desouza-Doucet, and M. M. Larrondo-Petrie, "Security patterns for physical access control systems," in *Proc. of IFIP WG 11.3*. Springer-Verlag, 2007, pp. 259–274.

[37] S. Romanosky, "Security design patterns part 1," in *Proc. of PLoP*, 2001.

[38] D. Riehle, W. Cunningham, J. Bergin, N. Kerth, and S. Metsker, "Password patterns," in *Proc. of EuroPLoP*, 2002.

[39] S. Romanosky, A. Acquisti, J. Hong, L. F. Cranor, and B. Friedman, "Privacy patterns for online interactions," in *Proc. of PLoP*. New York, NY, USA: ACM, 2006.

[40] M. Schumacher, "Security patterns and security standards - with selected security patterns for anonymity and privacy," in *Proc. of EuroPLoP*, 2002.

[41] A. Braga, C. Rubira, and R. Dahab, "Tropyc: A pattern language for cryptographic software," in *Proc. of PLoP*, 1998.

[42] A. Cuevas, P. E. Khoury, L. Gomez, and A. Laube, "Security patterns for capturing encryption-based access control to sensor data," *Proc. of SECURWARE*, pp. 62–67, 2008.

[43] Sami Lehtonen and Juha Pärssinen, "A pattern language for cryptographic key management," in *Proc. of EuroPLoP*, 2002.

[44] S. Lehtonen and J. Pärssinen, "A pattern language for key management," in *Proc. of PLoP*, 2001.

[45] B. Blakley and C. Heath, *Security Design Patterns*. The Open Group, Apr. 2004, 27.04.2011. [Online]. Available: www.opengroup.org/onlinepubs/9299969899/toc.pdf

[46] A. Cuevas, P. El Khoury, L. Gomez, A. Laube, and A. Sorniotti, "A security pattern for untraceable secret handshakes," in *Proc. of SECURWARE*, Jun. 2009, pp. 8 –14.

[47] N. Delessy and E. B. Fernandez, "Patterns for the extensible access control markup language," in *Proc. of PLoP*, 2005.

[48] M. Schumacher, "Firewall patterns," in *Proc. of EuroPLoP*, 2003.

[49] E. B. Fernandez, J. C. Pelaez, and M. M. Larrondo-Petrie, "Security patterns for voice over ip networks," in *Proc. of ICCGI*. IEEE Computer Society, 2007.

[50] N. Delessy, E. B. Fernandez, M. M.Larrondo-Petrie, and J. Wu, "Patterns for access control in distributed systems," in *Proc. of PLoP*, 2007.

[51] L. B. Jr, F. L. Brown, J. Divietri, G. D. D. Villegas, and E. B. Fernandez, "The authenticator pattern," in *Proc. of PLoP*, 1999, p. 6.

[52] E. B. Fernandez and R. Warrier, "Remote authenticator / authorizer," in *Proc of. PLoP*, 2003.

[53] M. Hafiz, "A collection of privacy design patterns," in *Proc. of PLoP*. New York, NY, USA: ACM, 2006, pp. 1–13.

[54] M. Schumacher and U. Roedig, "Security engineering with patterns," in *Lecture Notes in Computer Science*. Springer, 2001.

[55] T. Okubo and H. Tanaka, "Web security patterns for analysis and design," in *Proc. of PLoP*. New York, NY, USA: ACM, 2008.

[56] M. Sadicoff, M. M. Larrondo-petrie, and E. B. Fernandez, "Privacy-aware network client pattern," in *Proc. of PLoP*, 2005.

[57] B. Schleinzer and N. Yoshioka, "Security pattern for data integrity in p2p systems," in *Proc. of PLoP*, 2010.

[58] P. Sommerlad, "Reverse proxy patterns," in *Proc. of EuroPLoP*, 2003.

[59] S. Romanosky, "Enterprise security patterns," in *Proc. of EuroPLoP*, 2002.

[60] A. Kumar and E. Fernandez, "A security pattern for a virtual private network," in *Proc. of SugarLoafPLoP*, 2010.

[61] C. Dougherty, K. Sayre, R. C. Seacord, D. Svoboda, and K. Togashi, "Secure design patterns," Carnegie Mellon University, Software Engineering Institute, report 2009-TR-010, Oct. 2009.

[62] B. Elsinga and A. Hofman, "Control the actor-based access rights," in *Proc. of EuroPLoP*, 2002.

[63] E. B. Fernandez and J. Sinibaldi, "More patterns for operating systems access control," in *Proc. of EuroPLoP*, 2003.

[64] E. B. Fernandez and T. Sorgente, "A pattern language for security models," in *Proc. of PLoP*, 2001.

[65] E. B. Fernandez, "Patterns for operating systems access control," in *Procs. of PLoP*, 2002.

[66] E. B. Fernandez, T. Sorgente, and M. M. Larrondo-Petrie, "Even more patterns for secure operating systems," in *Proc. of PLoP*. ACM, 2006.

[67] E. B. Fernandez and D. laRed Martinez, "Patterns for the secure and reliable execution of processes," in *Proc. of PLoP*. New York, NY, USA: ACM, 2008, pp. 1–16.

[68] E. B. Fernandez and G. Pernul, "Patterns for session-based access control," in *Proc. of PLoP*. ACM, 2006.

[69] V. Gondi, "Multiple secure observers using j2ee," in *Proc. of PLoP*, 2010.

[70] M. Hafiz, "Secure pre-forking - a pattern for performance and security," in *Proc. of PLoP*, 2005.

[71] M. Hafiz, R. E. Johnson, and R. Af, "The security architecture of qmail," in *Proc. of PLoP*, 2004.

[72] D. M. Kienzle, M. C. Elder, D. Tyree, and J. Edwards-Hewitt, "Security patterns repository, version 1.0," 2003, (28.08.2011). [Online]. Available: http://www.scrypt.net/~celer/securitypatterns/repository.pdf

[73] S. R. Kodituwakku, P. Bertok, and L. Zhao, "Aplrac: A pattern language for designing and implementing role-based access control," in *Proc. of EuroPLoP*, 2001.

[74] Q. H. Mahmoud, "Security policy: A design pattern for mobile java code," in *Proc. of PLoP*, 2000.

[75] H. Mouratidis, P. Giorgini, and M. Schumacher, "Security patterns for agent systems," in *Proc. of EuroPLoP*, 2003.

[76] P. Morrison and E. B. Fernandez, "The credentials pattern," in *Proc. of PLoP*. New York, NY, USA: ACM, 2006, pp. 1–4.

[77] Patrick Morrison and Eduardo B. Fernandez, "Securing the broker pattern," in *Proc. of EuroPLoP*, 2006.

[78] J. L. Ortega-Arjona and E. B. Fernandez, "The secure blackboard pattern," in *Proc. of PLoP*. New York, NY, USA: ACM, 2008.

[79] T. Saridakis, "Design patterns for fault containment," in *Proc. of EuroPLoP*, 2003.

[80] K. E. Sørensen, "Session patterns," in *Proc. of EuroPLoP*, 2002.

[81] M. Weiss, "Credential delegation: Towards grid security patterns," in *Proc. of The Nordic Conference on Pattern Languages of Programs*, 2006.

[82] Y. Zhou, Q. Zhao, and M. Perry, "Policy enforcement pattern," in *Procs. of PLoP*, 2002.

[83] E. B. Fernandez, S. Mujica, and F. Valenzuela, "Two security patterns," in *Proc. of Asian Conference on Pattern Languages of Programs*, 2010.

[84] N. Shi and R. A. Olsson, "Reverse engineering of design patterns from java source code," in *Procs. of 21st ASE*. IEEE Computer Society, 2006.

 ISBN: 978-1-61208-158-8