

# Extracting Relations among Security Patterns

Atsuto Kubo<sup>1</sup>, Hironori Washizaki<sup>2</sup> and Yoshiaki Fukazawa<sup>1</sup>

<sup>1</sup>Department of Computer Science and Engineering,  
Waseda University  
3-4-1 Ohkubo, Shinjuku-ku, Tokyo 169-8555, Japan  
{ a.kubo, fukazawa }@fuka.info.waseda.ac.jp

<sup>2</sup>Information Systems Architecture Research Division,  
National Institute of Informatics  
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan  
washizaki@nii.ac.jp

## Abstract

*The activity of the secure system development can be supported by reusing extensive knowledge accumulated about security in the form of security patterns. There are a number of catalogs of security patterns available on WWW and literatures; however all of relations among security patterns are closed in each pattern catalog. Moreover even in each catalog, the author of the catalog might have overlooked useful relations among patterns belonging to the same catalog. This situation makes the selection and application of the right pattern for each security development activity a daunting task. To acquire such useful but overlooked relations in each catalog and cross-cutting relations over different catalogs, we have applied our technique for the automatic pattern relation analysis to a set of security patterns. Our technique utilizes existing text processing techniques to extract patterns from documents and to calculate the strength of pattern relations. As a result of experimental evaluations, it is found that our technique can extract appropriate relations in each security pattern catalog and over different catalogs, without information on relations described in original pattern documents. These newly found relations will be useful for retrieving, selecting, and combining security patterns.*

## 1 Introduction

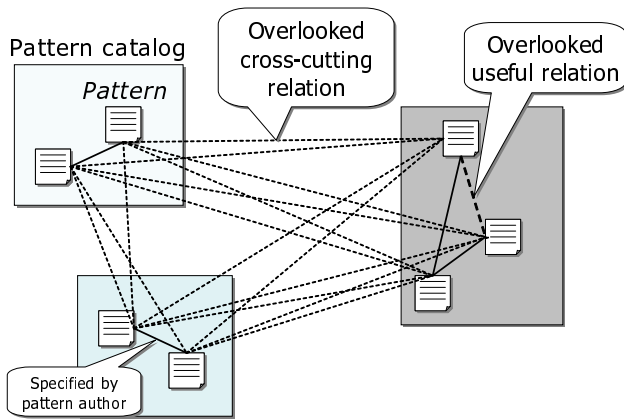
A security pattern is a well-understood solution to a recurring security problem. Each security pattern encapsulates security expertise in the form of worked solutions to

these recurring problems, presenting issues and trade-offs in the usage of the pattern[1]. A good number of catalogs that have collected related security patterns are available in public on the World Wide Web (WWW) and in other resources, such as [1, 2, 3, 4, 5]. However, the large amount of security patterns make the selection of the right pattern for the job a daunting task[6]; we need a guidance for the developers about how to select appropriate patterns.

To build a good guidance for security pattern selection, it is necessary to clarify relations among patterns. For example, there might be two or more patterns that provide different solutions dealing with different constraints for the same or similar security problem. In that case, it is preferable to show such a "similar problem relation" to help developers know the existence of patterns dealing with the same or similar problem and select an appropriate pattern dealing with a constraint faced in the target development. However, all of pattern relations specified by pattern authors are closed in each pattern catalog. Moreover even in each catalog, the author of the catalog might have overlooked useful relations among patterns belonging to the same catalog (shown in Figure 1).

There are several approaches for analyzing relations among patterns by hand, such as [7, 8, 9]; however, conventional approaches have only used a small number of patterns. There are difficulties in the following activities associated with the manual analysis (i.e. analyzing by hand).

- Analyzing the relations among a large number of patterns.
- Directly comparing patterns in different pattern forms with each other.



**Figure 1. Overlooked relations among patterns**

- Directly comparing patterns published in different catalogs with each other.

The relation analysis among a large number of patterns by hand is not realistic. An automatic approach that can be applied to a large number of patterns is required; however, to the best of our knowledge, there is no approach for automatic relation analysis. Moreover, none of the conventional manual approaches has been applied to security patterns.

In this paper, we have applied our technique for the automatic pattern relation analysis[10] to several security patterns that are collected manually from WWW, in order to acquire useful but overlooked relations in each catalog and cross-cutting relations over different catalogs. Our analysis technique can treat major pattern forms and various security patterns belonging to different catalogs, by using a common pattern model and several text processing techniques (such as stop-word removal[11], stemming[12], the TF-IDF term weighting method[13], and vector space model[11]).

## 2 Analysis procedure

Figure 2 shows an overview of the analysis procedure in our technique. Many pattern documents that exist on WWW are described using HTML. Therefore, our technique targets pattern documents described with HTML. The outline of the proposed analysis procedure is as follows.

### 2.1 HTML analysis

Input pattern document is analyzed, and sections (such as "Name: Firewall" and "Problem: ...") are extracted from the document in the *HTML Analysis* block in Figure 2.

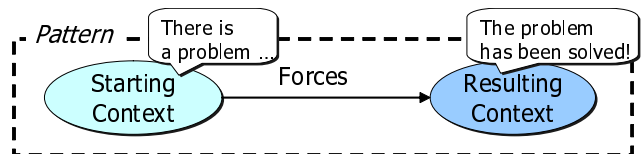
### 2.2 Pattern form judgment

The form of the input pattern document is judged in the *Pattern Form Judgment* block in Figure 2, such as GoF form[7].

### 2.3 Pattern extraction

A pattern is obtained from the sections according to the judged pattern form in the *Pattern Extraction* block in Figure 2. In our technique, we model each pattern as a labeled directed graph that illustrates a flow of the pattern application (shown in Figure 3). Here, we call the context before the pattern application "starting context". Similarly, we also call the context after the pattern application "resulting context". Therefore, we assume that a pattern application is a context transition from a starting context to a resulting context. In addition, we include a force (i.e. constraints that should be considered when the pattern is applied) in the model because two patterns that differ only in terms of forces are considered different patterns.

*Pattern Extraction* block map each section to one of elements (starting context, resulting context, and force) in this pattern model (shown in Figure 4). For example, the section "Problem" of GoF form can be mapped to the starting context.



**Figure 3. Our pattern model**

### 2.4 Relation analysis

Relations between patterns are analyzed in the *Relation Analysis* block in Figure 2 by calculating similarity among elements according to the pattern model. The degree of similarity between two elements is calculated by using the vector space model with the weighting by the TF-IDF method. *Relation Analysis* block calculate the similarity of any pair of pattern elements, and extract the pairs whose similarity is more than pre-designed threshold as related patterns (shown in Figure 5).

We classify relations into the following seven types based on the similarity among pattern elements:

- *Same*: If two patterns are similar in both starting and resulting contexts, they are almost the same pattern.

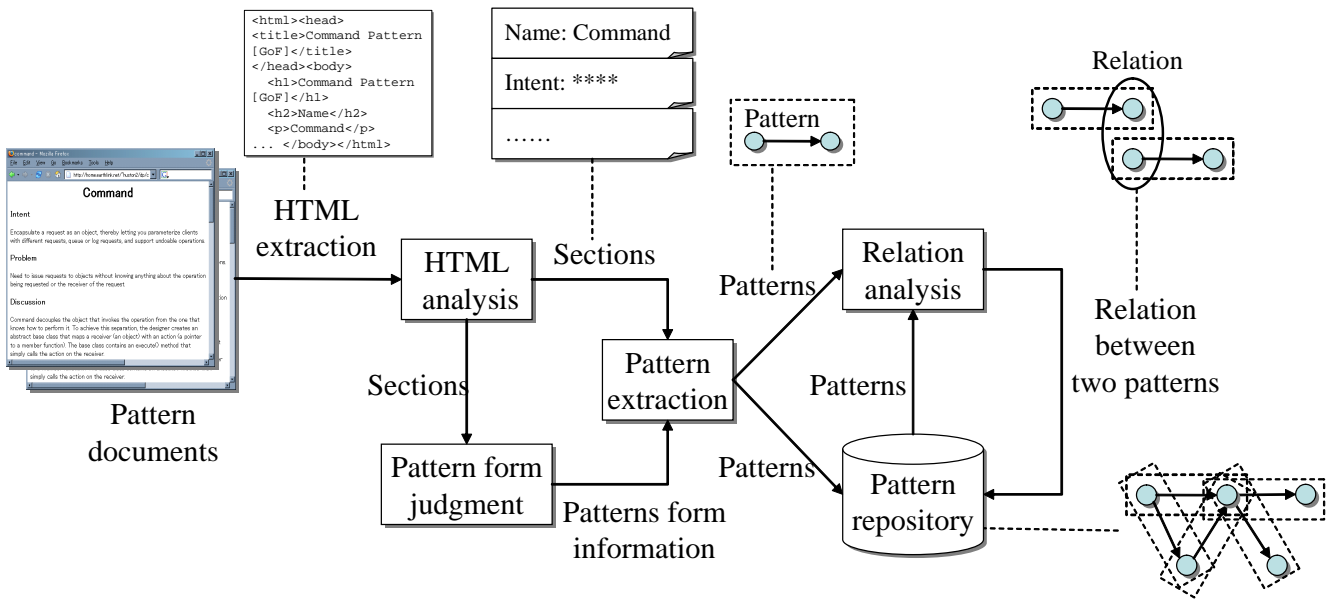


Figure 2. Overview of analysis procedure

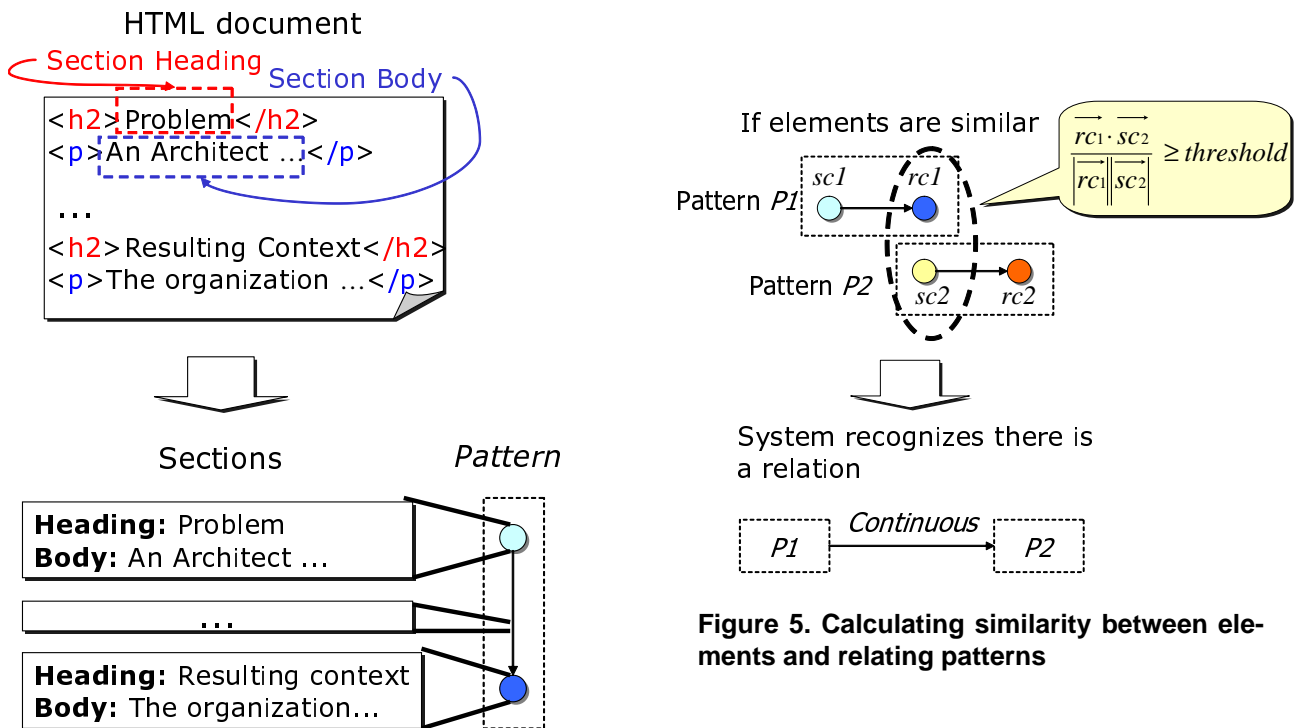


Figure 4. Mapping sections on pattern model

Figure 5. Calculating similarity between elements and relating patterns

- *SubInStarting*: If pattern  $p_1$ 's starting and resulting contexts are similar to another pattern  $p_2$ 's starting context,  $p_1$  can be a subpattern included in  $p_2$ 's starting context.
- *SubInResulting*: If  $p_1$ 's starting and resulting contexts are similar to  $p_2$ 's resulting context,  $p_1$  can be a subpattern included in  $p_2$ 's resulting context.
- *SimilarProblem*: If two patterns are similar in starting contexts and forces but not in resulting contexts, they deal with the same or similar problem.
- *SimilarResult*: If two patterns are similar in resulting contexts and forces but not in starting contexts, applications of those patterns lead to similar results.
- *Continuous*: If  $p_1$ 's resulting context is similar to  $p_2$ 's starting context and their forces are similar, they can be applied continuously in the order of  $p_1 \rightarrow p_2$ .
- *SimilarForce*: If  $p_1$ 's force and pattern  $p_2$ 's force are similar, it might be possible to use those patterns in the same development even if their contexts are different.

### 3 Experiments and discussion in security patterns

We implemented a system that automatically executes the above-mentioned analysis process, and performed an experimental evaluation for nine security patterns belonging to four catalogs: Schumacher's firewall patterns[5], Fernandez's legal cases pattern[3], Fernandez's attack patterns[4], and Yoder's architectural patterns[2]. The system calculated the strength of the relation for all of combinations of nine patterns, sorted all pairs of patterns in order of the relation's strength, and used the strongest relation per combination as the representative relation about the combination.

Table 1 shows the strength and type for each representative relation extracted in the experiment. Described relations are sorted in order from the largest of each relation strength.

Figure 6 visualizes the obtained results of representative relations. Though all relationships between each pattern are obtained as mentioned above, Figure 6 shows several highest relationships and corresponding patterns. The resulting graph suggests several useful relationships including some which are not explicitly specified in the original pattern documents.

- For example in Figure 6, the pattern "Secure Handling of Legal Cases"[3] is recognized as a subpattern included in the resulting context of another pattern

"Packet Filter Firewall"[5]. This relation is not specified in [3, 5]; however it is reasonable because domain models provided by "Secure Handling of Legal Cases" assume the usage of some reference monitors such as the packet filter firewall. In other words, the usage/application of "Packet Filter Firewall" leads to a resulting context in which legal cases (such as some clients are sued by another party) can be handled in secure way.

- As another example, there is a *SimilarForce* relation between "Roles"[2] and "Secure Handling of Legal Cases"[3]. This cross-cutting relation over different catalogs is reasonable because those patterns share similar constraints such as the existence of a number of users for the target system and the impossibility of customizing security for each person.

These newly found relations are useful for retrieving, selecting, and combining security patterns belonging to several catalogs. Moreover in Figure 6, we can see that "Denial Of Service (DoS) in VoIP"[4] relates to none of other patterns. This result is reasonable because it is a kind of attack patterns under specific contexts such as using VoIP, and others are security (defense) patterns under somewhat general contexts; viewpoints/contexts are quite different.

Since our analysis technique is automated, the technique is superior to other manual classification techniques (such as [8, 9]) in terms of scalability. However, there is a trade-off between scalability and validity; manual classification techniques are thought to be superior in terms of result's validity because those analysis activities are based on experts' considerations.

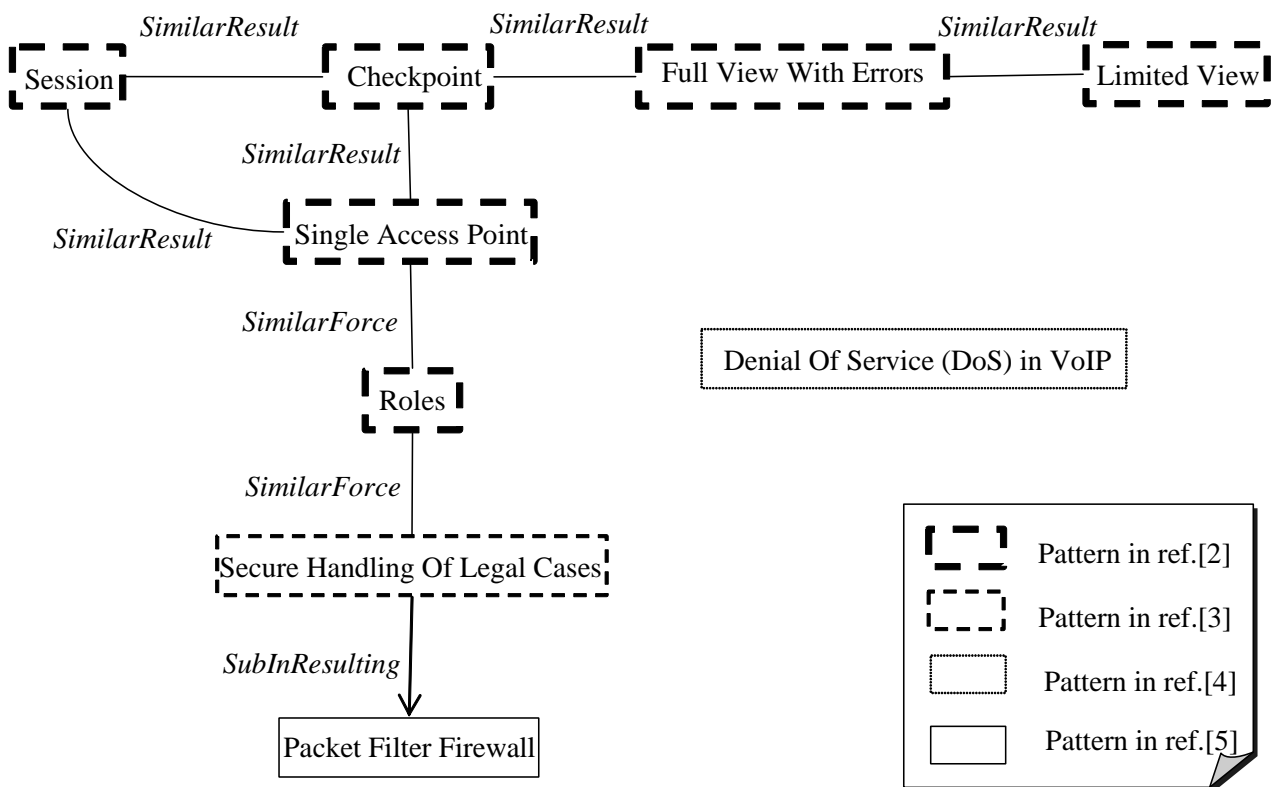
For example in Figure 6, "Session" and "Checkpoint" are tightly coupled together by a *SimilarResult* relation (i.e. relation in which two patterns give similar resulting contexts); however these patterns are thought to be quite different. In the future, we will check this tradeoff and clarify the applicability of each classification technique.

### 4 Conclusion and future work

We have applied the automatic pattern relation analysis to several security patterns to acquire useful but overlooked relations in each catalog and cross-cutting relations over different catalogs. As a result of experiments, we succeeded in the analysis of the appropriate relations among security patterns without using explicit information in the pattern document. The system that implements our technique can suggest the relations that original pattern authors have not noticed. Using our system, developers can compare and select security patterns easily and precisely. Our technique is expected to contribute the activity of retrieving security patterns.

**Table 1. Extracted representative relations among security patterns (Top 10)**

Rank	Pattern $p_1$	Pattern $p_2$	Strength	Type
1	Limited View	Full View With Errors	0.183	<i>SimilarResult</i>
2	Session	Check Point	0.141	<i>SimilarResult</i>
3	Session	Full View With Errors	0.138	<i>SimilarResult</i>
4	Single Access Point	Roles	0.125	<i>SimilarForce</i>
5	Secure Handling Of Legal Cases	Packet Filter Firewall	0.124	<i>SubPatternInResulting</i>
6	Single Access Point	Session	0.122	<i>SimilarResult</i>
7	Secure Handling Of Legal Cases	Roles	0.119	<i>SimilarForce</i>
8	Single Access Point	Check Point	0.118	<i>SimilarResult</i>
9	Secure Handling Of Legal Cases	Full View With Errors	0.115	<i>SimilarResult</i>
10	Secure Handling Of Legal Cases	Session	0.115	<i>SimilarResult</i>



**Figure 6. Analysis results for security patterns in different catalogs**

Contrastingly, our technique has several limitations in itself. First, since the meanings of the relationships obtained from our technique are all based on similarity, the “negative” relationships would not be obtained. Second, the current version of our technique can not deal the variation of authors’ description, such as synonyms and domain-specific usage of words.

In the future, we will conduct experiments for evaluating usability and validity of our relation analysis technique using a large number of security patterns and application results. We are planning to introduce thesauri to mitigate the second limitation above.

## Acknowledgement

We would like to thank Prof. Eduardo B. Fernandez for giving us detailed comments on section 3 of an earlier version of this paper.

## References

- [1] D.M. Kienzle and M.C. Elder: Security Patterns for Web Application Development, Final Technical Report, Univ. of Virginia., 2002.
- [2] J. Yoder and J. Barcalow: Architectural Patterns for Enabling Application Security, 4th Conference on Pattern Languages of Programs, 1997.
- [3] Legal E.B. Fernandez, D.L. la Red Martinez, J. Forneron, V.E. Uribe and G. Rodriguez: A secure analysis pattern for handling legal cases, Proc. 6th Latin America Conference on Pattern Languages of Programming, 2007.
- [4] E.B. Fernandez, J.C. Pelaez and M.M. Larrondo-Petrie: Attack patterns, a new forensic and design tool, Proc. Third Annual International Conference on Digital Forensics (IFIP WG 11.9), 2007.
- [5] M. Schumacher, E.B. Fernandez, D. Hybertson, F. Buschmann, and P. Sommerlad: Security Patterns, J. Wiley & Sons, 2006.
- [6] T. Heyman, K. Yskout, R. Scandariato and W. Joosen: An Analysis of the Security Patterns Landscape, 29th International Conference on Software Engineering Workshops (ICSEW’07), 2007.
- [7] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1994.
- [8] Walter Zimmer. Relationships between Design Patterns. In Pattern Languages of Program Design, Vol.1, Addison-Wesley, pp. 345–364, 1995.
- [9] Han-Yuen Ong, Michael Weiss, and Ivan Araujo. Rewriting a Pattern Language to Make it More Expressive. Proc. 6th Southwestern Conference on Pattern Languages of Programs (ChiliPLoP2003), 2003.
- [10] A. Kubo, H. Washizaki, A. Takasu, and Y. Fukazawa: Extracting relations among embedded software design patterns, Journal of Design and Process Science, vol.9, No.3, 2005.
- [11] G. Salton, and M.J. McGill. Introduction to Modern Information Retrieval. McGraw-Hill, Inc., 1983.
- [12] Chris D. Paice. Another Stemmer, SIGIR Forum, Vol. 24, No. 3, pp. 56–61, 1990.
- [13] G. Salton, and C. S. Yang. On the Specification of Term Values in Automatic Indexing. Journal of Documentation, Vol.29, 1973.