

SRRS: A Recommendation System for Security Requirements

Jose Romero-Mariona
University of California, Irvine

Donald Bren School of Information
and Computer Sciences

jromerom@uci.edu

Hadar Ziv

University of California, Irvine

Donald Bren School of Information
and Computer Sciences

ziv@ics.uci.edu

Debra J. Richardson

University of California, Irvine

Donald Bren School of Information
and Computer Sciences

djr@ics.uci.edu

ABSTRACT

Despite the availability of approaches to specifying security requirements, we have identified a lack of comparative studies of those approaches and, subsequently, lack of guidance and useful tools to determine the most appropriate approach for a specific project. In this paper we propose SRRS (Security Requirements Recommendation System), which takes user input about the most desirable characteristics for their project and recommends the most appropriate approach. We provide an example of applying our system, and show how the SRRS process works in general.

Categories and Subject Descriptors

D.2.1 [Requirements/Specifications]: *Elicitation Methods, Methodologies, Tools.*

General Terms

Reliability, Security.

Keywords

Security, Requirements, Software Engineering, Requirements Engineering, Recommendation System.

1. INTRODUCTION

Security is an essential and critical non-functional requirement (NFR) of present and future software systems; a system's security requirements must therefore be elicited and specified. A variety of security requirements approaches have been advocated, some derived from existing requirements-specification methods while others designed specifically for security requirements engineering. Interestingly, there is little guidance available to software developers attempting to select an approach to specifying security requirements that better suits a particular project.

This paper proposes a recommendation system for helping developers choose among security requirements approach(es) so as to better suit a specific project given a set of characteristics most desirable for that project. In addition, the proposed tool also takes as input the minimum level of support expected for each

specific desirable characteristic.

2. BACKGROUND AND RELATED WORK

As Alexander indicates, "security is unlike all other areas in a specification, as someone is consciously and deliberately trying to break the system" [1]. The security NFR is increasingly critical in its importance, unique in stakeholder needs, yet still must be integrated with all other functional and non-functional requirements and mapped into architectures, designs, and implementations [2].

Security needs arise when stakeholders determine that some resource belonging to a software system, tangible (e.g. money) or intangible (e.g. confidential information), is valuable to the organization. Resources are called assets [3, 4], and stakeholders wish to protect these assets from damage or attack. Security requirements engineering is thus focused on the protection of these valuable assets from the requirements perspective.

As part of our literature study, we were unable to identify detailed surveys of requirements specification methods in general over the last decade, nor specifically of security requirements engineering. We consider our survey to be the first evaluation and comparison of security requirements specification methods. Some work has been done in the specific area of selecting a requirements approach based on user input, primarily by Mylopoulos et. al. [5,6]. While this work is not necessarily a recommendation system, it is similar to ours in the importance placed on user input in order to make a decision and/or select from various choices based on specific parameters.

3. SYSTEM'S LIMITATIONS

We believe that the contribution of our system lies not in its technical innovation but rather in the information that is used as the basis for the recommendation. The computations that our proposed system performs are straightforward, but depend on the results obtained from our survey in order to be useful.

4. SRRS

As part of ongoing research in security requirements engineering, we evaluated and compared 12 published approaches along 34 different dimensions. For the purpose of our Security Requirements Recommendation System (SRRS), we identified ten key characteristics, which we can use to recommend which of the 12 approaches surveyed better suits a specific project.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

RSSE '08, November 10, Atlanta, Georgia, USA

Copyright 2008 ACM 978-1-60558-228-3 ...\$5.00.

The 12 approaches surveyed were chosen as major representatives of currently available choices in security requirements engineering. The candidate approaches for recommendation by SRRS are listed below:

- Misuse Cases (MC) [7]
- Abuser Stories (AS) [8]
- Secure TROPOS (ST) [9]
- Security Problem Frames (SP) [10]
- Anti-Models (AM) [11]
- i* Security Requirements (i*)[12]
- Common Criteria (CC) [13]
- SQUARE (SQ)[14]
- OCTAVE (OC) [15]
- Attack Trees (AT) [16]
- USEr Method (US) [17]
- CLASP (CL) [18]

SRRS uses ten desirable characteristics identified for security requirements approaches. A developer using SRRS would rank the relative importance of the ten characteristics in order to determine the most suitable approach(es) for the specific project at hand. The ten characteristics used in SRRS include, for each approach to security requirements its,

- 1- Elicitation support
- 2- Level of customer involvement
- 3- Unambiguity level
- 4- Completeness level
- 5- Clarity level
- 6- Traceability level
- 7- Overall security level
- 8- Update difficulty of the security specifications
- 9- Automation support level
- 10- Scalability level

4.1 Applying SRRS

SRRS depends on developer input about the specific characteristics of the project at hand, as well as the minimum level of support that they expect from the approach for each one of those ten different characteristics. In a nutshell, SRRS runs through the 12 approaches looking for the specific characteristics as ranked by the developer while keeping track of only those approaches that meet the minimum level of support for each characteristic.

Suppose developer Dave is assigned to a new software project where traceability and overall security are of utmost importance. Dave decides to use SRRS to help choose the best approach for the new project from among the 12 approaches to security requirements listed in section 3.1. Developer Dave identifies the relative ranking of the ten characteristics described above as well as the Minimum Support (MS) level for each of those characteristics, shown in Table 1. This table serves as input to SRRS.

Table 1. Developer Input Sample

PROPERTY	RANK	MINIMUM SUPPORT
Elicitation	7	25%
Customer Involvement	3	50%
Unambiguity	9	100%
Completeness	10	75%
Clarity	4	50%
Traceability	1	100%
Overall Security	2	75%
Update Difficulty	6	50%
Automation	8	25%
Scalability	5	25%

SRRS uses a predetermined Ratings Matrix (RM), shown in Table 2, rating each of the 12 methods against the ten characteristics identified above. Each of the ten characteristics were rated using a Likert scale [19]; the scale ranges from 0 points, signifying no support at all for that specific characteristic by the approach at hand, to 4 points, signifying high/maximum support provided.

Table 2. Ratings Matrix

	Elicit	Customer	Unambiguity	Completeness	Clarity	Trace	Security	Update	Auto	Scalability
MC	4	3	1	2	1	1	2	4	1	1
AS	3	3	1	1	2	4	1	4	0	1
ST	1	1	1	2	2	1	2	2	3	4
SP	0	0	3	4	2	1	2	2	0	3
AM	2	2	3	1	3	2	2	2	2	4
i*	2	3	1	2	3	2	4	1	0	2
CC	1	1	4	3	2	2	4	1	1	4
SQ	1	4	3	2	4	1	3	2	1	4
OC	4	4	0	2	3	2	2	1	0	4
AT	3	1	2	0	2	3	1	3	1	1
US	4	4	3	4	2	4	2	3	0	2
CL	2	1	1	2	2	1	4	1	3	4

SRRS then takes the following steps:

Step 1- SRRS creates a new Minimum Support vector (MS) using the minimum support column of the user input. MS contains values between 0 and 4 for each of the ten characteristics. Figure 1 shows what MS would look like based on the sample input in Table 1 (0%=0, 25%=1, 50%=2, 75%=3, 100%=4).

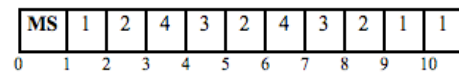


Figure 1. Minimum Support Vector

Once MS is created, SRRS traverses RM, determining for each value whether it is greater than or equal to the value in MS for that characteristic. If the value in RM is equal to or greater than the one in MS, then a value of 1 is assigned, if not then a value of 0 is assigned, this creates matrix RM*. Figure 2 shows the calculation of RM* for one of its rows, corresponding to the Common Criteria approach.

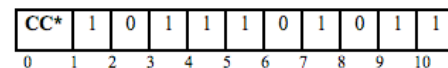


Figure 2. RM* for Common Criteria row

Step 2- SRRS creates a new vector (R), which contains points associated with the ranking of the ten characteristics as provided

by the developer. While the developer can choose the number of points for each rank, SRRS starts with a default set of values: Rank 1 (20 points), Rank 2 (15 points), Rank 3 (10 points), Rank 4 (8 points), Rank 5 (6 points), Rank 6 (5 points), Rank 7 (4 points), Rank 8 (3 points), Rank 9 (2 points), and Rank 10 (1 point). Based on Dave's input combined with the default values, R would look like Figure 3. It is ordered based on the characteristics, meaning Elicitation is first, then Customer Involvement, etc. In the example of Table 1, Elicitation received a ranking of 7, meaning it gets 4 points; Customer Involvement received a ranking of 3, which gets it 10 points, etc

R	4	10	2	1	8	20	15	5	3	6
0	1	2	3	4	5	6	7	8	9	10

Figure 3. Ranking Vector

Step 3- SRRS performs a matrix multiplication of RM^* and R. $RM^* \times R$ effectively produces a 12×1 vector (F) corresponding to the sum total of support provided by each approach for all ten characteristics. In this example, if we take CC^* and multiply it by R, the result for Common Criteria is 39. The resulting F vector is sorted so that approaches with the highest scores appear at the top. For the example project described in Table 1, the top 3 approaches are Abuser Stories with 55 points, USER with 54 points, and SQUARE with 51 points.

Using SRRS, Developer Dave now has the 12 approaches sorted in order of the most appropriate for his specific project. Figure 4 summarizes the SRRS recommendation process.

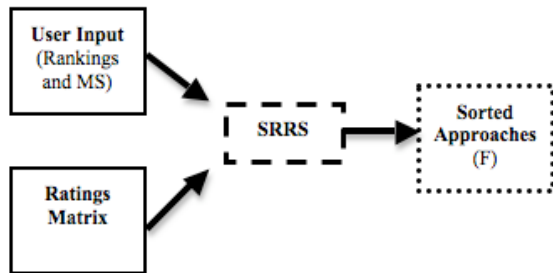


Figure 4. SRRS Process

5. CONCLUSIONS AND FUTURE WORK

Information and automated guidance for choosing an approach to specifying security requirements based on desirable characteristics have not been readily available. The system, SRRS, proposed here uses prior knowledge about security requirements approaches, combined with user preferences, to recommend the most appropriate approach for specific project characteristics, and in fact present a full-order ranking of all approaches. We believe that our system can provide the user more levers to influence the recommendation, for example changing and controlling the point values associated with each rank, or percent values associated with minimum support levels. We plan to work on those system extensions, as well as on case studies of developers using our recommendation system.

6. REFERENCES

[1] Alexander, I.: Initial Industrial Experience of Misuse Cases in Trade-Off Analysis. Proceedings of IEEE Joint International Requirements Engineering Conference. 2002

[2] Romero-Mariona, J., Ziv, H., Richardson, D.: Toward Hybrid Requirements-based and Architecture-based Testing. Proceedings of the workshop on the Role Of Software Architecture for Testing and Analysis (ROSATEA). 2007

[3] Chivers, H. and Fletcher, M.: Applying Security Design Analysis to a service based system. Software: Practice and Experience, vol. 35 no. 9. 2005

[4] ISO/IEC.: Information Technology - Security Techniques - Evaluation Criteria for IT Security - Part 1: Introduction and General Model. ISO/IEC. International Standard 15408-1. 1999

[5] Soutchanski, M., Pham, H., Mylopoulos, J.: Decision Making in Uncertain Real-World Domains Using DT-Golog. AAAI. 2006

[6] Castro, J., Kolp, M., Mylopoulos, J.: Towards requirements-driven information systems engineering: the Tropos project. Inf. Syst. 27(6): 365-389. 2002

[7] Jacobson, I. et al.: Object-Oriented Software Engineering: A Use Case Driven Approach. Addison-Wesley. 1992

[8] Peeters, J. and Dyson, P.: Cost-Effective Security. IEEE Security & Privacy. 2007

[9] Mouratidis, H., Giorgini, P., Manson G.: Integrating Security and Systems Engineering: Towards the Modelling of Secure Information Systems. Proceedings of the 15th Conference on Advance Information Systems (CAiSE). 2003

[10] Hatebur, D., Heisel, M., Schmidt, H.: Security Engineering Using Problem Frames. Proceedings of the International Conference on Emerging Trends in Information and Communication Security (ETRICS). 2006

[11] Haley, C., Laney, R., Nuseibeh, B.: Deriving Security Requirements from Crosscutting Threat Descriptions. AOSD. 2004

[12] Liu, L., Yu, E., Mylopoulos, J.: Security and Privacy Requirements Analysis within a Social Setting In. Proceedings of the International Conference on Requirements Engineering. 2003

[13] Stoneburner, G., Hayden, C., & Feringa, A.: Engineering Principles for Information Technology Security (A Baseline for Achieving Security). Computer Security Division, Information Technology Laboratory National Institute of Standards and Technology. 2001

[14] Mead, N., Hough, E., Stehney II, T.: Security Quality Requirements (SQUARE) Methodology. (CMU/SEI-2005-TR-009). Software Engineering Institute, Carnegie Mellon University. 2005

[15] Alberts, C. et. al.: Introduction to the OCTAVE Approach. CERT Coordination Center. www.cert.org/octave/approach_intro.pdf

[16] Schneier, B.: Secrets and Lies: Digital Security in a Networked World. John Wiley & Sons. 2000

[17] Hallberg, N., Hallberg, J.: The Usage-Centric Security Requirements Engineering (USEr) Method. Information Assurance Workshop. 2006

[18] Viega, J.: Building Security Requirements with CLASP. Proceedings of the Workshop on Software Engineering for Secure Systems (SESS). 2005

[19] Likert, R.: A Technique for the Measurement of Attitudes, Archives of Psychology 140: pp. 1-55. 1932