

Copyright ©2009 IEEE. Reprinted from (Empirical Software Engineering and Measurement, 2009. ESEM 2009. 3rd International Symposium on Publication Date: 15-16 Oct. 2009 On page(s): 485 - 489).

This material is posted here with permission of the IEEE. Such permission of the IEEE does not in any way imply IEEE endorsement of any of SHIELDS's products or services. Internal or personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution must be obtained from the IEEE by writing to pubs-permissions@ieee.org.

By choosing to view this document, you agree to all provisions of the copyright laws protecting it.

Progress Report on the Experimental Evaluation of Security Inspection Guidance

Frank Elberzhager, Marek Jawurek, Christian Jung, Alexander Klaus
*Fraunhofer Institute for Experimental Software Engineering
Kaiserslautern, Germany*

{frank.elberzhager, marek.jawurek, christian.jung, alexander.klaus}@iese.fraunhofer.de

Abstract

Although security inspections have proven to be a very efficient means for assuring software security early in the software development lifecycle, they are not used extensively because they usually need to be performed by security experts, who are few and thus expensive. Adoption of security inspections could be facilitated if one could encapsulate the expertise and experience of security experts as guidance for security inspections performed by software developers. Our approach to addressing this challenge consists of two different kinds of reading support that provide the required guidance to software developers: Vulnerability Inspection Diagram (VID) and Security Inspection Scenario (SIS). In this article, we sketch our

initial experimental evaluation of VIDs and SIS with a group of software developers of an industrial project partner. We present the setup and the experiment's results. In addition, we describe the implications of our results on future work regarding the approach and further evaluation.

1. Introduction

Handling software security is becoming harder as software complexity increases. In order to assure that the cost for software security remains as low as possible, it is essential to apply appropriate means early in the software development lifecycle (SDLC), as this has been shown to reduce costs significantly in the long run [6]. One such quality assurance technique is

software inspection, a manual task applicable to different artifacts of the SDLC, ranging from a requirements document to source code. With respect to software security, inspections have so far been performed by security experts, as the required expertise and experience is profound. The facts that software systems are becoming more complex and bigger and that security experts are few and thus expensive constitute the major reasons why acceptance of security inspections is low. One approach to dealing with the amount of software to inspect is to determine critical parts of the software and only perform inspections on those parts, e.g., by threat modeling.

Our complementary approach to dealing with the number of personnel capable of performing security inspections enables software developers to do inspections like security experts, to some extent. In the European Research Project SHIELDS [1], we have developed inspection guidance that encapsulates the expertise and implicit strategy security experts apply in inspections for use by non-security-experts: VIDs are flow-chart like diagrams that direct inspectors, whereas SIS are textual descriptions of vulnerability classes that support defect detection [2][3][7].

In order to obtain initial feedback concerning the applicability of our approach and to guide our future work, we designed and conducted an experiment.

The remainder of this paper proceeds as follows: In Section 2, we describe the experimental setup and the context of its execution. In Section 3, we analyze the results of our experiment and their implications for our approach as well as for further work. Finally, in Section 4, we present conclusions drawn and sketch future work in Section 5.

2. Experiment Design

We formulated several research questions regarding VIDs and SIS:

- RQ1:** Do VIDs and SIS provide support for non-experts and guide them successfully through the security inspection?
- RQ2:** How do users rate the ease of use of VIDs and SIS?
- RQ3:** How do users rate the usefulness of VIDs and SIS?
- RQ4:** Is the preference of VIDs and SIS dependent on the expertise of the user?

To get answers regarding our research questions, we set up an experiment which then was conducted by three people from our industrial project partner TXT e-solutions S. p. A.

In this experiment, before the inspections were started, we captured the individual levels of experience of the participants. This first questionnaire concentrated on their professional experience in general and was related to topics like inspections, security, Java, or flow charts.

For the inspection, we developed a web application called "RentABook". The software, written in Java, has several vulnerabilities built in on purpose, among which we focus on SQL Injection, Cross-Site Scripting (XSS), badly implemented or incomplete error-checking, and weak password recovery mechanisms. We extracted two code fragments, each with about 1,300 lines of code, for the inspection. The participants had not seen any related development artifact.

For the experiment, we split up the participants into two groups. Both had to perform two iterations of the inspection. Each iteration consisted of an initial tutorial, the inspection itself, and a questionnaire.

The reading support used covered the four vulnerabilities mentioned above; the first two examples were covered by VIDs, the latter two by SIS. Table 1 shows the experiment schedule and the distribution of vulnerabilities.

Table 1. Experiment schedule

Iteration 1			
	Code fragment	Reading support	Vulnerabilities
Group 1	1	VID	9
Group 2	1	SIS	3
Iteration 2			
	Code fragment	Reading support	Vulnerabilities
Group 1	2	SIS	6
Group 2	2	VID	7

For an initial estimate of the efficiency and effectiveness of our approach, we collected quantitative data, such as the vulnerabilities reported by the participants, in issue lists as well as the time they needed for the inspection. Thus, we can draw first conclusions as far RQ1 is concerned.

Immediately after an inspection, the participants had to judge the ease of use ("The degree to which an individual believes that using a particular system would be free of effort" [4]) and usefulness ("The degree to which an individual believes that using a particular system would enhance his or her job performance" [4]) of the approach just applied. Regarding RQ2 and RQ3, we included several questions in the final questionnaire. This questionnaire was designed according to [4].

We had to combine data from the questionnaires to analyze possible relations between the experience of a

person and the subjective judgment of usefulness for RQ4.

Participants could rate statements on a seven-point bipolar interval scale. The scale ranged from “strongly disagree” to “strongly agree”. Figure 1 displays an exemplary scale for the statement "SIS are complex" and the quantification we used in the analysis.

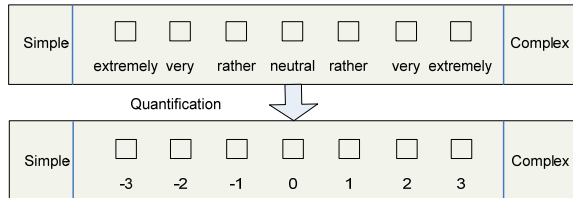


Figure 1. Exemplary scale and quantification

By structuring the experiment the way we did, several threats to validity could be minimized: Due to the counterbalanced intra-subject design we used, learning effects and sequence effects could be avoided. However, some threats to internal and external validity [5] could not be mitigated, as shown in Table 2.

Table 2. Threats to validity

Internal validity	
	Threat
Selection	Group is too small
External validity	
	Threat
Sampling error	Group is not representative, and results cannot be generalized.
	Possible threat
	Reading support for inspections is from a real-world scenario. The inspection process is not fully run through, but the application of methods and models is not influenced.
Environment not representative	Problem: the intentional distribution of errors in the code. This is mitigated because the reading support was unknown to the developers.

3. Results and Experiences from the Experiment

While conducting the described experiment and investigating the two different kinds of inspection reading support, we wanted to gain quantitative and qualitative results from the participants. Due to the low number of attendees, the qualitative results from the completed questionnaires are more relevant than the quantitative data, and we will focus on these qualitative results in the following. The complete rating of the VID and SIS questionnaires is summarized in Table 5 and Table 6. Initial answers and results from the experiment regarding our research questions will be emphasized and explained next.

The preparation time used for the reading support, which was supported by a five- and an eight-page exemplary description, respectively, was between 8 and 20 minutes. Collecting information from our

general questionnaire, we were able to categorize the participants as follows:

- **Participant A** is a software developer with one year of practical experience and some knowledge in inspections but no knowledge in security.
- **Participant B** is a senior software developer with 10 years of practical development experience and security knowledge.
- **Participant C** is a junior software developer with one month of development experience, who has no security experience.

RQ1: Table 3 and Table 4 show an excerpt of security problems found by participants B and C. For those two participants who used the same SIS on the same piece of code, we found that participant C with very low development experience and no security knowledge was able to find the same vulnerabilities as a senior software developer with a lot of knowledge in the security area (found security problem with ID WPR-B1 corresponds to WPR-C1 and WPR-B2 corresponds to WPR-C2 – line numbers differ because vulnerabilities usually stretch over multiple lines and cannot be pinpointed to one line). However, participant B found two additional security problems (see Table 3). When using VIDs, both participants found some of the same security problems. Additionally, participant C even found some more security problems compared to the skilled participant B (see Table 4). We assume that schematic and very detailed nature of the XSS detection instructions helped the unskilled inspector but made the skilled inspector feel patronized. Finally, participant A, who used the VIDs and SIS on different pieces of code, could also identify certain security problems.

Table 3. Excerpt of found security problems by two participants of the experiment who used SIS

SIS	Person	ID	Class	Line	Description
weak password recovery	B	WPR-B1	password Recovery.java	124	user is not forced to change password immediately
		WPR-B2		105	favorite colours is the only secret asked
		WPR-B3	login.jsp	105	secret is inserted once and no locking mechanism is implemented
		WPR-B4		94	no counter for password recovery attempt
	C	WPR-C1	password Recovery.java	110	the password isn't changed immediately
		WPR-C2		105	there is only one question - the secret color - it's better if there will be more than one.

In summary, although the results are far from being significant, we got an indication that our reading support successfully guides the participants through the security inspection.

RQ2: Regarding our second research question – analyzing the questionnaires with a focus on the ease of use of VIDs and SIS – we made the following observations: The participants rated the understanding of both kinds of reading support with positive values. In more detail, they were easily able to understand the syntax, respectively the structure, the language used, and the instructions. All ratings except one are

between one and three, which is between rather and extremely positive. Furthermore, the VIDs were rated as more complex than the SIS. An easier representation could lead to a less complex structure.

Table 4. Excerpt of found security problems by two participants of the experiment who used VIDs

VID	Person	ID	Class	Line	Description
Cross-Site-Scripting (XSS)	B	XSS-B1	addressAdd.jsp	82	validation to be checked more
		XSS-B2		90	validation to be checked more
		XSS-B3		97	validation to be checked more
		XSS-B4		99	validation to be checked more
	C	XSS-C1	addressAdd.jsp	82	value is not checked
		XSS-C2		88	value is not checked
		XSS-C3		90	value is not checked
		XSS-C4		97	value is not checked
		XSS-C5		99	value is not checked
		XSS-C6		93	value is not checked
		XSS-C7		95	value is not checked
		XSS-C8		104	value is not checked
C	XSS-C9	creditCardEdit.jsp	86	value is not checked	
	XSS-C10		94	value is not checked	
	XSS-C11		102	value is not checked	
	XSS-C12		110	value is not checked	

The adaptation to a new context was rated more difficult for the SIS than for the VIDs. This could stem from the long text passages in the SIS, which have to be restructured and customized for adaption. In summary, both models were rated positive. The results imply that a flow-chart like diagram for reading support is easier for the inspectors than following a textual description focusing on defect classes. However, the low complexity of the SIS indicates an advantage compared to the VIDs.

Table 5. Questionnaire results form participants using SIS

Statements / Questions		Participant		
		A	B	C
Minutes for preparation		15	10	10
Ease of use	Structure is easy to understand	2	1	1
	Questions or obscurities occurred / structure	-1	0	-2
	Instructions are clear	2	1	0
	Explanations are clear	2	1	1
	Used language is easily understandable	3	1	2
	SIS are complex	-2	0	-3
	Learning and applying in practice: easy	1	-1	0
Adapting SIS to needs: easy	1	-1	0	
Usefulness	Using SIS: do inspections quicker	2	-1	0
	Using SIS: enhanced confidence in results	2	0	2
	Using SIS: do inspections more easily	2	0	2
	Would like to use SIS in practice	2	-1	0

RQ3: The third research question focused on the usefulness of our models. The values regarding the statement "Using SIS / VIDs would enable me to do the inspection more easily" were (2, 0, 2) and (1, 1, 2), which is between neutral and very probable. We assumed that more experienced persons need less guidance in the reading support. This was backed up by just taking into account the ratings from the two low-experienced participants (A, C). Furthermore, using our reading support, the confidence of the participants in their inspection results was enhanced. Based on the structured guidance of VIDs and SIS, it was easier for the participants to estimate how secure the code was based on their inspection results. Another positive estimation is the assumption that using VIDs

for reading support will speed up the inspection process. In contrast, the rating for SIS regarding this question is much lower. One reason could be a missing schematic overview or procedure in the SIS. Though these are only subjective impressions, it seems that it helps the participants more than performing no or only an ad-hoc (i.e., experienced-based) inspection. Based on the questionnaires' results displayed in Table 5 and Table 6, the VIDs and SIS were assessed positive, whereas the VIDs were rated slightly more useful.

Table 6. Questionnaire results from participants using VIDs

Statements / Questions		Participant		
		A	B	C
Minutes for preparation		20	10	8
Ease of use	Syntax is easy to understand	1	2	2
	Questions or obscurities occurred / syntax	1	0	-3
	Components are logical	2	1	2
	VIDs are logically structured	3	1	2
	Instructions are clear	1	2	2
	Used language is easily understandable	2	1	2
	VIDs are complex	1	1	-2
	Learning and applying in practice: easy	1	1	1
	Adapting VIDs to needs: easy	2	1	2
	Using VIDs: do inspections quicker	2	1	1
Usefulness	Using VIDs: enhanced confidence in results	3	1	2
	Using VIDs: do inspections more easily	1	1	2
	Would like to use VIDs in practice	2	1	1

RQ4: Another related observation in this context, answering our fourth and last research question, is the preference of experienced software developers for using VIDs. They performed the inspection much easier by following the instructions given by the flowchart instead of reading long text passages to get the instructions. This can be seen in Table 5, where the experienced person B rates the statements "Using SIS would enable me to do inspections quicker" and "I would like to use SIS in practice" rather low, whereas the rating is more positive for VIDs. The remaining two participants with low experience rate both kinds of reading support positive, which can be explained by a general need for reading support in order to improve their security inspection.

In addition to asking the participants to express their feelings towards statements with pre-defined answer scales, we also provided the opportunity to clarify some answers with comments. Regarding the statement "SIS are complex" one participant commented: "...sometimes a more schematic approach would help". Regarding VIDs, the participants demanded more explanations or programming-language dependent examples. One of the participants also expressed the idea to use VIDs or SIS constructively during software development as guidance for developers or architects. Another comment suggested tool support for handling guidance material during the inspection.

4. Conclusion

We performed an initial evaluation of our proposed security inspection guidance material for software developers. Even though we had a low number of participants, we gathered information that will have a significant impact on the further evolution of our approach. The data from this experiment, but especially the participants' comments, have led us to the following conclusions:

- Inspectors find our approach useful and have found vulnerabilities using our approach, but we need further experimentation to obtain statistically significant results.
- The inspectors' demands for more information in VIDs and more structure in SIS indicate that a synthesis of both would be sensible.
- We feel supported by the participants' demand for a tool, as this is something we have anticipated and are already developing.

In summary, we feel confident that our approach will eventually support software developers in performing security inspections, but we have also identified potential for improvement in our future work.

5. Future Work

First of all, we will try to back up our results with more statistically sound experiment setups. This will allow us to quantify the benefit of our approach.

Furthermore, we will research the creation of a synthesis between SIS and VIDs. The resulting guidance should combine the best of both designs and should provide a clear structure with easily accessible background information on the specific vulnerability.

The proposed constructive usage of VIDs and SIS in the implementation and design of software is something we do not feel comfortable with at the moment. On the one hand, our guidance is specifically designed to be used in analysis: The flow of execution imposed on the inspector does not support the creation of software very well. On the other hand, there are already constructive descriptions for developers, such as best practices and security patterns.

As already mentioned, we are developing a tool that will support inspectors in using our guidance material. We will evaluate, whether this will facilitate the usage of VIDs and SIS and therefore lead to higher acceptance among users.

Acknowledgment

The research leading to these results has received funding from the European Community Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 215995.

We would also like to acknowledge all the members of the SHIELDS Consortium for their valuable help, and especially Alessandra Bagnato and other participants from TXT e-solutions S. p. A. for participating in the experiment.

References

- [1] SHIELDS: Detecting known security vulnerabilities from within design and development tools. Research project within the European Community's Seventh Framework Programme. <http://www.shields-project.eu>.
- [2] D2.1: Formalism definitions and representation schemata. Deliverable report within [1].
- [3] A. Klaus, F. Elberzhager, "Security Inspection Scenarios – A Facet of Security: Conducting Vulnerability-based Code Inspections", International Conference on Advances in System Testing and Validation Lifecycle, Porto, 2009, in press.
- [4] F. Davis, "User acceptance of information technology: system characteristics, user perceptions and behavioral impacts", Intl. Journal of Man-Machine Studies, 38, pp. 475-487, 1993.
- [5] R. E. Kirk, Experimental Design. Procedures for the Behavioral Sciences, 3rd ed. Pacific Grove, Brooks/Cole Publishing Company, 1995.
- [6] W. Boehm, "Industrial software metrics top 10 list", IEEE Software, pp. 84-85, 1987.
- [7] D4.1: Initial specifications of the security methods and tools. Deliverable report within [1].