A Practical Model For Rating Software Security

Haiyun Xu*, Jeroen Heijmans*, Joost Visser*†
*Software Improvement Group, Amstelplein 1, 1096 HA Amsterdam, The Netherlands
†Radboud University Nijmegen, P.O. Box 9102, 6500 HC Nijmegen, The Netherlands
{h.xu, j.heijmans, j.visser}@sig.eu

I. Introduction

To bring software security to a higher level, international standards and models have been developed to address security issues for software product quality. Well-known models are the Common Criteria [1] and the OWASP Application Security Verification Standard [2]. Application of the former model typically involves a major effort over a protracted time period, while the latter is limited in scope to web applications.

In 2011 ISO issued the updated standard for software product quality ISO/IEC 25010 [3]. One important change in ISO 25010 is that *Security* appeared as one of the main software product quality characteristics. Although ISO 25010 defines software product quality characteristics, ISO has not yet provided concrete models and measurements to evaluate software product security.

We propose a new security product quality model that makes ISO 25010 operational. We specify four requirements for the model: (1) the model shall be applicable for all types of software products; (2) the model shall be applicable from the early development phase; (3) the model shall be lightweight, concrete and repeatable; (4) the model shall lead to ratings that allow for comparison between software products.

Heitlager et al. [4] provide a practical model for measuring software product *Maintainability*, which is one quality characteristic defined in the ISO 25010 software quality model. This security model will provide a broader operational solution based on ISO 25010.

II. Security Quality Model

A. Software Product Security in ISO 25010

ISO 25010 defines eight quality characteristics in its product quality model: functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability and portability. The *security* characteristic is defined as “the degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization”. The security characteristic is composed of five subcharacteristics:

- **Confidentiality**: ensures that data are accessible only to those authorized.
- **Integrity**: prevents unauthorized access or modifications.
- **Non-repudiation**: actions or events can be proven to have taken place.
- **Accountability**: actions of an entity can be traced uniquely to the entity.
- **Authenticity**: the identity of a subject or resource can be proved to be the one claimed.

B. System properties

To rate the software product security we define eleven system properties, which reflect how a typical software product addresses the desired security subcharacteristics. Their mapping to ISO 25010 is presented in Figure 1. In our model, we combine the characteristics *Confidentiality* and *Integrity*, *Non-repudiation* and *Accountability* since they share many attributes in their definitions in ISO 25010. In practice, they are addressed together during software development.

1) Confidentiality and Integrity: In order to ensure that data is only accessed by authorized users, most systems provide a form of protection for data flowing into or out of the system. If data is stored by the system, the data store is also protected. For every attempt to access data within the system, it should be verified that the user is authorized. The authorization mechanism itself must be robust and impossible to circumvent. Finally, in order to prevent data going in to or out of the system from harming either the system or its users, only desired content should be allowed. This typical approach is reflected in the five system properties *secure data transport*, *secure data storage*, *authorised data access*, *secure authorisation* and *input and output verification*.

2) Non-repudiation and Accountability: In order to make it impossible to repudiate actions, most systems employ a way to obtain ‘proof’ of that action. As part of this proof, information should be logged to be able to connect the evidence with the action. Furthermore, it should be possible to uniquely trace back proof to a particular user. This typical approach is reflected in the system properties *strength of proof*, *logging completeness* and *unique identification*.

3) Authenticity: Most systems provide access management: a user has to authenticate prior to access. After gaining access, it is typically not required to fully re-authenticate for every subsequent action: a user session is started within which access is granted. To know which users are allowed to enter and what their credentials are, user management is necessary. This typical approach is reflected in the system properties *access management strength*, *session management strength* and *secure user management*.

	Secure data transport	Secure data storage	Authorised data access	Secure authorisation	In/output verification	Strength of proof	Logging completeness	Unique identification	Access management strength	Session management strength	Secure user management
Confidentiality Integrity	X	X	X	X	X						
Non-repudiation Accountability						X	X	X			
Authenticity									X	X	X

Fig. 1. Mapping eleven system properties to ISO/IEC 25010 security subcharacteristics.

C. Rating and Aggregation

To determine if a system possesses all the desired system properties, we break it down into logical parts. For example, to determine *secure data storage*, we need to determine all the places where and what data is stored in the system. This covers items such as databases, files, logs, configuration and binaries. With the system broken into logical parts, we need to determine if these parts possess the desired properties. To do so, we have decomposed all of the eleven system properties into subproperties. As with the properties themselves, these are based on best-practice approaches in typical systems. For example, *access management strength* is broken down into three subproperties. Access management is achieved by picking one (or more) authentication methods. Such a method has an inherent strength (*authentication method strength*), but also depends on correct implementation (*authentication method implementation*) and prevention of circumvention (*authentication enforcement*).

For each of the relevant items, a trained evaluator will assign ratings to the subproperties by following a set of detailed evaluation guidelines. For example, a biometric authentication method could be scored *strong* (rating 3) on authentication method strength since it is intrinsically strong, *medium* (rating 2) on authentication method implementation since the false acceptance rate is not low enough, *weak* (rating 1) on authentication enforcement since the login screen can easily be circumvented.

These ratings are then aggregated, from items to subproperties, then to property level, to subcharacteristic, and finally to the security characteristic level. A typical aggregation approach is to use the arithmetic mean, but that does not reflect the *weakest link principle*. Using the minimum function for aggregation would reflect this principle, but it would not be very distinctive, as every system with a single serious flaw would be rated equally. This makes the minimum unsuitable for comparing multiple systems. We have therefore chosen the *Power Mean* [5] (with parameter $p = -2$) as the aggregation method. This function weighs lower values more heavily, but does not entirely discard the higher scores.

III. Discussion

We apply this model to assess software product security quality. The use of ISO 25010 as frame of reference implies that the model is grounded in a consensual terminology for software product quality. The eleven system properties provide broad coverage to identify software security vulnerabilities.

In ISO 25010, the security principle *Availability* is categorized as one of the sub-characteristics of *Reliability*. Currently, we are working on a model to assess software product reliability quality. Therefore, although *Availability* is one of the classic CIA principles (Confidentiality, Integrity, Availability), we do not include it in our model in order to remain consistent with ISO 25010.

The rating that comes out of the model provides input for security risk assessments and allows for progress monitoring and comparison between software products. From our software assessment experiences, a rating can effectively draw attention to security and put it in the agenda of decision makers. Subsequently, the technical findings can be used for effective product security improvement.

IV. Conclusions and future work

This paper presents a security quality model based on ISO 25010 that fulfills several practical requirements. The model targets at assisting evaluators assessing the security quality of a software product. The model can be applied to different types of software products and for assessments from the early development phase. We tested the model on several software systems and the assessments took several days on average. The ratings we obtained matched intuition of system experts and concrete vulnerabilities were identified in the process.

Our future work will focus on model validation and partially automatizing the assessment. Furthermore, since the security quality ratings allow for comparison between software products, we will work on model calibration based on benchmark data.

References

[1] “Common Criteria - Common Methodology for Information Technology Security Evaluation,” 2009.

[2] OWASP, “OWASP Application Security Verification Standard 2009 - Web Application Standard,” 2009.

[3] ISO, “ISO/IEC 25010:2011 Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models,” 2011.

[4] I. Heitlager, T. Kuipers, and J. Visser, “A practical model for measuring maintainability,” in *6th Int. Conf. on the Quality of Information and Communications Technology (QUATIC 2007)*, 2007, pp. 30–39.

[5] P. Bullen, *Handbook of Means and Their Inequalities*. Springer, 2010.