JOURNAL OF OBJECT TECHNOLOGY

Online at http://www.jot.fm. Published by ETH Zurich, Chair of Software Engineering ©**JOT, 2005**

Vol. 4, No. 3, 2005
Special issue: 6th GPCE Young Researchers Workshop 2004

A Common Criteria Based Approach for COTS Component Selection

Wes J. Lloyd, Colorado State University

Abstract

Component-based software engineering (CBSE) endeavors to enable software developers to develop quality software systems with less time and resources than traditional development approaches. Software components must be identified and evaluated in order to determine if they provide required functionality for systems being developed. Consideration of security requirements for component selection is of interest. This research considers how the Common Criteria (CC), an internationally recognized standard for security requirements definition and security assessment of IT systems, can be applied towards the development of component-based systems. A CC-based COTS component selection process is proposed which integrates activities of the CC for security requirements specification and evaluation. Research questions are presented for the evaluation of the process to establish its value for COTS component selection as well as to identify areas for improvement.

1 INTRODUCTION

Rising costs and schedule constraints have forced many organizations, including the U.S. government to use Commercial off-the-shelf (COTS) components for developing applications with security concerns [Devanbu00]. By using preexisting components to implement security requirements, software development organizations seek to reduce development costs while still producing quality software in a timely manner. Components that provide implementation of security functions such as encryption, digital signing, access control, and authentication are often needed. Implementing security mechanisms such as cryptographic algorithms is complex and can result in the accidental introduction of serious flaws into the system [Lindqvist98]. Consequently using COTS components to implement complex security functions is an attractive option for developers.

Component selection decisions are often made in an ad-hoc manner. Component selection processes have been proposed to improve upon the efficiency and effectiveness of informal methods. Existing selection processes do not fully address the specification and evaluation of functional and non-functional security requirements. Ruhe points out that many existing COTS component selection processes do not provide procedures for

Cite this article as follows: Wes J. Lloyd: "A Common Criteria Based Approach for COTS Component Selection", in *Journal of Object Technology*, vol. 4, no. 4, April 2005, Special issue: 6th GPCE Young Researchers Workshop 2004, pp. 27-34,
http://www.jot.fm/issues/issue_2005_04/article7

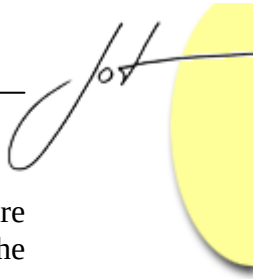
the evaluation of non-functional requirements [Ruhe03]. Security vulnerabilities in components can result from development oversights including unanticipated attacks, component integration issues, software bugs, and unanticipated changes in COTS component product updates. Lindqvist identifies insufficient component validation as a security risk when using commercially available COTS components [Lindqvist98].

To support the development of secure component-based systems, component selection processes need to address the problem of security evaluation of COTS components. This research proposes a component selection process, which is based on the Common Criteria (CC), an internationally recognized standard for security requirements definition and evaluation for IT systems [ISO99]. The CC-based process proposes how the Common Criteria can be used to enhance the quality and efficiency of security specification and evaluation of COTS components. By using the CC-based component selection process developers can gain confidence using COTS components to implement security functions when developing component-based systems.

2 BACKGROUND

Selecting the appropriate COTS component to meet specified security requirements is a security assessment problem. Myers states that written and measurable objectives (requirements) are required in order to validate their compliance in a software system [Myers76]. It is widely agreed that requirements must be defined and quantifiable in order for testing to be effective. For the assessment of COTS component security, functional security requirements must be specified and delegated to particular components in the software design. Existing component selection processes specify methods to elicit software requirements in the general sense, but they do not explicitly address how to specify security requirements. Security requirements specification is challenging because requirements encompass both functional and non-functional aspects and many developers may not be familiar with the breadth of security issues needing to be addressed.

Once security requirements are specified, available COTS components must be evaluated to determine their suitability for use in the system being developed. Bachman et al. identify the problem of evaluating the security characteristics of COTS components [Bachman00]. Bertoa proposed a quality model that defines a set of quality attributes and metrics for evaluating COTS component quality. Bertoa identifies just (3) security characteristics: data encryption, controllability and auditability in [Bertoa02]. A presence metric is suggested to assess support for these security characteristics. A presence metric as defined only validates the existence of the function. Bertoa's quality model only begins to address the topic of COTS component security evaluation. The consideration of many other functional and non-functional security characteristics is desired. Emergent security characteristics are security properties that are realized as a result of the system's components working together to provide security. Although an interesting problem, this research focuses on security requirements specification and evaluation at the component



level to aid component selection. The evaluation of security properties of the software system after component selection and integration is an evaluation problem beyond the scope of this research.

A case study conducted by Bertoa discovered that of 164 COTS components only (4%) provided information in product documentation (help-files, manuals, demonstrations, and UML diagrams), which identified support for security quality characteristics [Berota03]. This absence of documentation detailing security properties of COTS components combined with the questionable reliability of security claims in available documentation creates the need for developers to perform their own component evaluations to assess security.

Functional security requirements identify security mechanisms such as user authentication, access control, encryption, data integrity, and intrusion detection systems. The goal of these mechanisms is to provide ample security to meet the system's non-functional security requirements. Non-functional security requirements identify a system's resilience and level of immunity to attack. In order to be highly resilient a system needs to minimize the fallout resulting from attackers (hackers, crackers, cyber-terrorists) exercising system vulnerabilities such as software bugs through viruses, impersonation, sniffing, etc. In a practical sense proving that software is secure is likened to proving it correct. The validation of a software system's non-functional security is a NP-hard assessment problem which existing component selection processes cannot address [Ruhe03]. To validate security, the software would need to be formally proved correct against a functional specification. This is an unrealistic problem to solve. Instead a generalized level of security can be established using the Common Criteria for Information Technology Evaluation (CC). The CC is a multi-part standard for specification and evaluation of functional and non-functional security properties for IT systems [ISO99]. The CC includes a general process model for system security evaluation, a comprehensive set of security functional requirements which can be used as a baseline to express security requirements of all IT systems, and a thorough set of security assurance activities for evaluating security provided by IT systems.

The CC defines a comprehensive set of security requirements composed in (11) classes, which group together families of related security requirements. Security classes include: Security Audit, Communication, Cryptographic Support, User Data Protection, Identification and Authentication, Security Management, Privacy, Protection of the system security functions, Resource Utilization, System Access, and Trusted path/channels. The security requirements identified in these classes can be used as a basis to create security requirements specification documents.

The CC defines seven different evaluation assurance levels (EALs), which define evaluation activities to assess security. These EAL levels can be used to establish a generalized "level of security" provided by a component. EAL levels include: functionally tested (EAL1), structurally tested (EAL2), methodically tested and checked (EAL3), methodically designed, tested and reviewed (EAL4), semi formally designed and tested (EAL5), semi formally verified designed and tested (EAL6), and formally verified designed and tested (EAL7). Evaluations beyond EAL3 require intervention at the design

phase of software development for security assessment. Such evaluations may not be helpful for COTS component evaluation since COTS components are typically preexisting. The CC identifies (40), (83), and (110) assessment activities for levels EAL1, EAL2, and EAL3 respectively. The CC certifies higher levels of security assurance through applying higher levels of evaluation effort. Evaluation effort is described as the scope, depth, and rigor of the evaluation activities. Scope considers the percentage of the system being evaluated, depth identifies the level of design and implementation analysis, and rigor describes the level of formality of the assessment. No guarantee of security exists regardless of the testing effort just as there is no way to solve the oracle problem of software testing. We assume that by applying higher evaluation assurance levels additional security faults will be identified. The evaluation of the CC's effectiveness to establish security through evaluation is interesting and important, but beyond the scope of this research.

3 HARNESSING THE COMMON CRITERIA

We propose a Common Criteria based COTS component selection process based on common steps of existing component selection processes: requirements identification, COTS component search, COTS component evaluation, and component selection. Our process harnesses the Common Criteria (CC) for security specification and evaluation. The process includes six steps: (1) System High Level Design, (2) Component Requirements Definition, (3) Component Search, (4) Component Evaluation, (5) Component Selection, and (6) Component Integration and Operation.

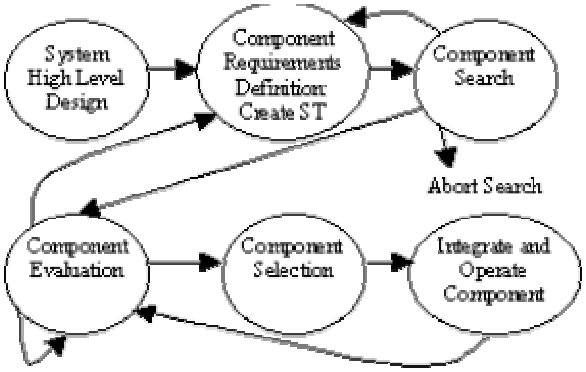
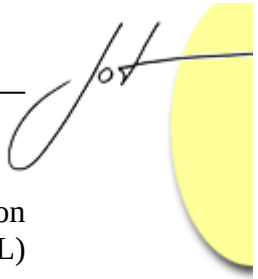


Fig. 1. CC Based Component Selection Process

In step (1), a high level design is produced for the component-based system. The high level design specifies the underlying component-based architecture and development platform(s). When selecting a component architecture, security functions provided by the architecture need to be considered.

Security requirements for the desired COTS component are elicited in a functional specification document known as the Security Target (ST) document in step (2). The (11) classes of common security requirements identified by the CC are used to develop the ST document. The use of Protection Profiles (PP's), documents that elicit reusable sets of CC



requirements, can help speed the process of creating ST documentation. A protection profile which identifies security requirements for secure socket layer (SSL) communication components can help speed the process of generating ST documentation for a component search seeking SSL-based communication components.

During step (3) an initial search for candidate COTS components is conducted using the security requirements identified in the ST document as a filter. Product documentation (brochures, help files, manuals, etc.) is searched to identify support for requirements identified in the ST. In the event no components are found during the initial search, the ST document can be refined so that COTS components can be identified which state support for a base level of security requirements. An alternative to revising the ST is to abandon the search and develop custom component(s) to meet security requirements.

In step (4) components are evaluated to eliminate inadequate candidates from consideration. Variations of the activities in step (4) can be considered to adapt the selection process to operate under time and cost constraints. A CC based evaluation is conducted on the candidate components. Initially an EAL level 1 evaluation is performed. An EAL level 1 evaluation identifies (40) evaluation activities. It may only be necessary to perform a subset of the evaluation activities. Evaluation activities are halted when an appropriate component is identified. The following ordering of evaluation activities is suggested: ADV_FSP Functional Specification and documentation evaluation, ADV_RCR evaluation of the component's correspondence to functional requirements, ATE_IND independent testing, AGD_ADM Administrator Guidance, AGD_USR User Guidance, ACM_CAP CM Capabilities, and ADO_IGS Installation, integration, and start-up. This evaluation suggests performing function verification and testing activities first since they are most likely to identify functional inadequacies. If multiple candidates meet all ST security requirements after the EAL level 1 evaluation the evaluator can choose to modify the ST to specify more rigorous security requirements, or to proceed with the next EAL level of evaluation. This process of modifying the ST or advancing to the next EAL level for evaluation is repeated until all candidate components except for one is eliminated from consideration. In the event not all COTS components are eliminated, or when the evaluation has exhausted time and financial resources a formal decision making technique such as the weighted sum method (WSM) or analytic hierarchy process (AHP) can applied based to aid decision-making.

In step (5) the component that best meets the criteria stated in the ST is selected as a result of the CC evaluation activities performed in step (4). Once selected the component is integrated for use into the component-based system under development during step (6). Once a component is in operation previously unknown errors or vulnerabilities may surface. Required corrections are identified forcing a revision of the COTS component. Depending on the cooperation of the component vendor changes may or may not be possible. If the component is updated then the security functionality may need to be reevaluated. The reevaluation may require a complete revaluation or only a partial evaluation of the changes.

4 RESEARCH APPROACH

By treating components as software systems as in [Orso00] we can use the CC to specify, and evaluate the security provided by individual components. Kahn states that all of the requirements of the CC may not be directly applicable to individual COTS components because of their smaller functional scope as compared to IT systems [Khan00]. We conducted a survey of COTS components to discover the classes of CC security requirements that were supported by commercially available products. Forty-two COTS components, which claimed to provide security, were located using the Internet component vendor ComponentSource.com. Components were identified through searches with security related keywords. On average each COTS component provided security functions from two Common Criteria classes (~2.2). Support for security requirements of eight of the eleven CC classes were directly identified through evaluation of product documentation.

The proposed CC-based COTS component selection process seeks to make two significant contributions for component selection. The (11) classes of Common Criteria security requirements can be used as a baseline for specifying component security requirements. Developers can develop more complete security specifications for COTS components before beginning a product search. The CC can also help developers evaluate the security provided by candidate components. By applying the evaluation activities specified by each CC evaluation assurance level a documented and repeatable assessment of security can be achieved. Empirical evaluation of software development processes is generally considered difficult. Consider evaluations which would measure the success of various software proceses such as the waterfall model, the spiral model, or eXtreme Programming. The next step in this research is to conduct an exploratory investigation to apply the CC-based selection process to investigate existing research questions. An exploratory investigation will likely help to identify how to further direct efforts to evaluate and improve the proposed CC-based process. Empirical studies as well as case studies may be desired to fully understand how software developers benefit (or do not benefit) from using the process. These investigations will evaluate the CC-based selection processes' applicability for COTS component selection, as well identify future process improvements. Investigations should seek to quantify if the CC-based component selection process provide improvements in component specification and security evaluation. Future research can seek to answer: How useful is the CC-based approach in predicting suitability of components? What difficulties are encountered: specifying security requirements, evaluating component security? Does the CC-based selection process provide advantages over existing processes or the ad hoc approach? Which evaluation activities are most helpful for component selection? Which activities are time consuming? Which provide the least/most information for component selection decisions? What are the confusing and difficult parts of the process? What effort is required to train developers on the use of the process?

REFERENCES

[Bachman00] Bachman, F., Bass, L., Buhman, C., Comella-Dorda, S., Long, F., Robert, J., Seacord, R., Wallnau, K.: Volume II: Technical Concepts of Component-Based Software Engineering, Technical Report, CMU/SEI-2000-TR-008, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA., 2000.

[Bertoa02] Bertoa, M.F., and Vallecillo, A.: "Quality Attributes for COTS Components." In Proc. of the 6th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE 2002). Malaga, Spain, June 2002.

[Bertoa03] Bertoa, M.F., Troya, J., and Vallecilo, A.: "A Survey on the Quality Information Provided by Software Component Vendors." In Proc. Of the 7th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE 2003). Darmstadt, Germany, July 2003.

[Devanbu00] Devanbu, P. and Stubblebine, S.: Software Engineering for Security: a Roadmap. In The Future of Software Engineering. Special volume of the proceedings of the 22nd International Conference on Software Engineering - ICSE 2000, June 2000.

[ISO99] ISO/IEC-15408 (1999) Common Criteria for Information Technology Security Evaluation, v 2.1, Nat'l Inst. Standards and Technology, Washington, DC, August 1999, http://csrc.nist.gov/cc

[Khan00] Khan, K.M., Han, J., Zheng, Y., Characterizing User Data Protection of Software Components. In Proceedings of the 2000 Australian Software Engineering Conference, Gold Coast, Queensland, Australia, April 2000.

[Lindqvist98] Lindqvist, U., Jonsson, E. A Map of Security Risks Associated with Using COTS, IEEE Computer, June, 1998, pp. 60-66.

[Myers76] Myers, G., Software reliability: principles and practices: New York: John Wiley & Sons, 1976.

[Orso00] Orso, A., Harrold, M.J., Rosenblum, D., Component Metadata for Software Engineering Tasks, in Proceedings of EDO 2000, LNCS Vol. 1999, Springer-Verlag, November 2000.

[Ruhe03] Ruhe, G., Intelligent Support for Selection of COTS Products, in Proceedings of the Net.Object Days 2002, Erfurt, Springer, 2003, pp. 34-45.

About the author(s)

Wes J. Lloyd is presently a PHD Student in the Department of Computer Science at Colorado State University. His research interests in software engineering include component based software development, empirical investigations of software engineering practices, software processes and requirements engineering. E-Mail: wlloyd@acm.org