Component Selection Using the Common Criteria for Security Evaluation

Wes J. Lloyd
Computer Science
Colorado State University
Fort Collins, Colorado 80523
wlloyd@cs.colostate.edu

Abstract

Component based software-engineering (CBSE) promises to enable software developers to develop quality software systems with less time and resources that traditional development approaches. Software components must be identified and evaluated in order to determine if they provide required functionality for systems being developed. Component selection requires the assessment of functional and non-functional properties in order to have proper information to make component choices. Currently non-functional requirements, and particularly security requirements, are of interest for many software systems. This paper considers how the Common Criteria (CC), an internationally recognized standard for security requirements definition and security assessment, can be applied towards the development of component-based systems. The results of a case study of COTS components are presented suggesting that the CC is useful for CBSE. A CC based component selection process is then proposed including an example of its application.

1. Introduction

Component Based Software Engineering (CBSE) involves the building of component-based software systems (CBS) through the use of preexisting software components to realize the functional requirements. By using components, software development organizations hope to reduce the time and cost of development while improving software quality. In order to realize benefits from using CBSE practices, developers must identify candidate components to provide system functionality [9]. Developers then assess if the candidate components implement the functional requirements of the CBS under development. Developers collect data about component attributes during the evaluation, which then drives the component selection decision. The problem of identifying available components and selecting the most appropriate one is known as the component selection problem.

Rising costs and schedule constraints have forced many organizations, including the U.S. government to use Commercial off-the-shelf (COTS) components in the development of applications with security concerns [4]. Components that provide for the implementation of common security functions such as encryption, digital signing, access control, and authentication are desired [14]. Using COTS components to provide security functions is generally seen as less expensive and time-consuming than using pre-built components. The difficulty of coding complicated security mechanisms such as cryptographic algorithms involves the risk of introducing serious flaws into the CBS. By using COTS components to implement security requirements developers hope to realize the benefits of CBSE when developing systems with security concerns.

Component selection decisions are often made in an ad-hoc manner [11,12]. Component selection processes are proposed to improve upon the efficiency and effectiveness of ad-hoc methods. However many do not explicitly describe procedures for assessment and selection when non-functional requirements are of concern [18]. Security requirements are generally considered to be non-functional requirements [6]. In order to more effectively support the development of secure component-based systems, a component selection process that addresses the assessment and selection of components providing security functions for component-based systems is desired. This paper introduces the Common Criteria a standard methodology that helps developers specify and assess security for IT systems. The applicability of the Common Criteria to CBSE is discussed. The results of a case study which investigates the existence of CC security requirements among a diverse set of COTS components is presented. A Common Criteria based component-selection process is then proposed which specifies a process for component assessment and selection for selection decisions with security concerns. An example application of the process is then presented, followed by conclusions and directions for future research.

2. Background

The Common Criteria for Information Technology Security Evaluation (CC) is a multi-part standard for evaluating the security properties of IT products and systems [9]. The CC includes a general model for system security evaluation, a set of standard security functional requirements for expressing the functional requirements of IT systems, and a set of security assurance requirements, which identify activities to evaluate the level of security assurance for systems.

The CC identifies several key terms for identifying artifacts of the security assessment effort. A target of evaluation (TOE) is the IT system and its associated administrator and user guidance documentation that is the subject of the evaluation. A security target (ST) is a set of security requirements presented as a specification that are used as the basis for the evaluation of a TOE. A protection profile (PP) is an implementation independent set of security requirements for a category (domain) of TOEs that meet specific consumer needs. A protection profile is essentially a reusable predefined set of security requirements that can be used to help evaluate various IT systems that have common functional applications. The value of reusable sets of security requirements is further discussed in [5]. An example of a protection profile is PP-2001-07, which is the U.S. Department of Defense's (DoD) security profile for firewalls for use in basic environments [15]. This protection profile defines security requirements required by the DoD for firewall products. Various commercial firewall products have been certified to provide security requirements as specified through the use of protection profiles [2]. A developer knowing that a particular product has been certified to provide the security requirements specified by a protection profile is assured that the product has been tested and should provide security in relation to those requirements. This assurance helps the developer when choosing an appropriate IT product when security is a concern.

The CC defines seven different evaluation levels known as evaluation assurance levels (EALs), which define sets of evaluation activities that can be applied to assess security at particular levels. 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). The CC asserts that greater assurance results from the applicant of greater 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. Figure 2-1 shows the TOE process for establishing and evaluating security for IT systems [9].

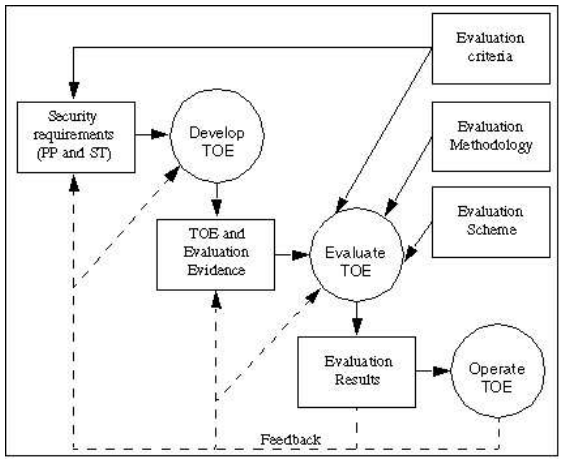

Figure 2-1 - TOE Evaluation Process

The TOE evaluation process begins by generating the ST document, which defines system security requirements. Protection profiles can be used to aid in the creation of the ST. The artifacts for evaluation are the PP and ST documentation and the TOE system itself. The CC evaluation proceeds to conduct the assurance activities specified by the level of the EAL the system is being assessed at. The evaluation process attempts to confirm whether the TOE satisfies the security requirements as stated in the ST. If the TOE fails to provide security then the system can be corrected, or the security requirements adjusted. Once the evaluation process confirms that the TOE provides required security the TOE can be operated with a degree of confidence equal to that of the evaluation assurance level met.

In addition to providing a security evaluation process the CC also defines a standard set of security requirements composed in (11) classes as listed in table 2-1. Each class groups together families of related security requirements. Families then group together components, which define sets of individual requirements that are called component elements. 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. Protection profiles define reusable sets of security requirements from these families specific to particular types of IT systems.

Common Criteria Security Classes	Number Of Requirements
(FAU) - Security Audit	27
(FCO) – Communication	12
(FCS) – Cryptographic Support	5
(FDP) - User Data Protection	67
(FIA) – Identification and Authentication	20
(FMT) – Security Management	19
(FPR) – Privacy	20
(FPT) – Protection of the System's Security Functions	50
(FRU) Resource Utilization	9
(FTA) System Access	15
(FTP) Trusted Path/Channels	6

Table 2-1 - Security Requirements of Common Criteria Classes

3. Common Criteria Applicability to CBSE

The CC has provided a standard for the evaluation of security of IT products and systems. The CC can also be applicable to CBSE. By treating components as software systems as in [16] we can use the common criteria to specify, and evaluate the security provided by individual components.

This paper considers CBSE in an architecture first context [2]. In developing the software system a particular architecture such as .NET, or Java is chosen and then appropriate components are sought to implement system functionality. Protection profiles already exist and are readily used for the security assessment for COTS products. In this context COTS products can be considered as stand alone products rather than plug-in development-level components. A web browser, or operating system is considered a COTS product, but not a development-level component. Although it can be argued that such systems can be treated as components, this research considers evaluating security for small architecture and function specific plug-in components, not systems.

In order to specify the security properties of COTS components the applicability of the CC to components treated as IT systems should be considered. What are the differences between IT systems and individual COTS components? Where a typical IT system is likely to posses security requirements from many of the Common Criteria security classes, components which are essentially small systems are more likely to be focused on requirements from only a few of the security classes. An exception to this case is that of a security framework, where a set of components works together to provide common security functionality to the entire system. Such a security framework may implement many security functions. A full analysis of the differences between IT systems and individual COTS components is beyond the scope of this work. It is interesting however to consider the scope of security functions of component types and the CC security requirement classes together in order to gain confidence that the CC is applicable to evaluating component security.

Kahn states that all of the requirements of the CC may not be directly applicable to individual software components because of the distributed nature of components and their compositional complexity [10]. Considering Kahn we should expect that no single component should require all of the security functions defined by the CC to be implemented within. Rather each component is likely to provide a small set of requirements as appropriate for the overall system. The range of security requirements implemented across the components in a component based system is likely to be diverse because of diverse domains of the component based systems and their internal architectures.

Volter defines (2) categories of components in his component taxonomy [19]. Volter defines logical components as a category of components composed of domain, data, and user components. Domain components provide the business logic often referred to as middleware. Domain components represent the controller in the model – view - controller (MVC) aggregate design pattern. Data components provide data access services including validation, and conversion. Data components represent the model in the MVC aggregate design pattern. User components provide access to domain and data components through a user interface. User components represent the view in the MVC aggregate design pattern.

Technical components act as containers that provide a runtime environment for the components of a system. Thus technical components act as frameworks that provide a central runtime environment for components to handle technical

concerns. The precise technical concerns are dependent on the application domain but they could include things such as transactions, security, fault tolerance, and load balancing. Component frameworks can address security concerns centrally. Examples of component frameworks include the familiar commercial frameworks J2EE, DCOM/COM, CORBA, and .NET. These frameworks are not specific to a particular application domain. Frameworks with specific application domains include the Java-based Struts web application framework [1] and the DataObjects .NET framework [3], which can speed the development of 3-tier relational database business applications.

4. COTS Component Survey

Domain components perform the work of the system. They could implement any security requirement relevant to the system interacting with the outside environment. Whenever a user interacts with the system the domain component provides the functionality to the user component, which then provides the human-computer interface and the data component maintains the state of the system. A domain component can provide specific business functions or unique security functions. For example a domain component might provide network services to the application. For a server-based network application a secure sockets layer may be required. Secure Tunnel is a Java based component, which provides SSL support to any socket-based server [8]. This component implements security functions from three security classes. Trusted path and channels (FTP) requirements include the establishment of a trusted communication channel between communicating clients. This component logs activity that can be used to implement requirements of the Security Audit (FAU) class. Clients using the secure tunnel SSL support must provide secure authentication using digital certificates. Authentication requirements are addressed in the Identification and Authentication (FIA) CC security class.

To understand the breadth of COTS components available, which provide security functions identified by the CC, a survey was conducted. Through conducting the component survey a better understanding of the applicability of the Common Criteria for security requirements definition and assessment for CBSE is sought. One question of interest is what is the breadth of security functions provided by COTS components in relation to the CC security classes? The component website www.componentsource.com was used in addition to the www.google.com search engine. The following security related keywords corresponding to CC security requirements were used in the search: "cryptography (FCS), log (FAU), logging (FAU), access control (FDP), repudiation (FCO), authentication (FIA), anonymity (FPR), and psuedonymity (FPR), access limits (FTA), connection limits (FTA), secure socket (FTP), secure channel (FTP), trusted channel (FTP), trusted socket (FTP), system security (FMT/FPT), system security management (FMT/FPT), utilization (FRU), resource utilization (FRU)." Selection of components for evaluation was done randomly after discovery by matching with the search keywords listed above.

Component	Security
Domain: Secure Tunnel	FAU, FCO, FCS, FIA, FTP
PDFlib	FCS, FIA
Jsockets	FIA, FTP
3-D Secure MPI	FCO, FCS, FIA, FTP,
SSL Secure Internet Sockets (18 components)	FCO, FCS, FIA, FTP,
Common Logging	FAU
Mcipher	FCS
Chronalyzer	FAU, FPT
IAIK-JCE	FCS
RSA Bsafe	FCS
IS Networks JCE provider	FCS
User Manager Professional	FAU, FCS, FDP, FIA, FMT, FPT (implied)
Filter Plus	FAU, FDP, FIA
Polar Crypto	FCO, FCS
Logicrypto	FCS
Xsign	FCO
Data: DbUtils	FDP, FRU
Castor	FAU, FDP, FRU
Dbcp	FRU
Commons Validator	FDP
JSQL	FAU, FCS, FDP, FIA, FTP
Technical: DataObjects .NET	FDP, FMT, FPT (implied)
Struts	FAU, FDP
User: Jsuite	FDP, FIA

Table 4-1 - Security Classes Supported by Identified COTS Components

Forty-two individual COTS components were identified, including a package of eighteen SSL-based secure socket components. Table 4-1 shows the CC security class mappings to the components evaluated in the case study. Although the ratios of component-types are not significant due to random selection and COTS component availability (Domain 16: ,Data: 5, User: 1, Technical: 2), from observation this ordinal ranking is suggestive of the predominance of component types exhibiting CC security requirements. There seems to be significantly more Domain COTS components than data, user, or technical ones that implement CC security requirements. Analysis consisted of reviewing product overviews, feature lists, and documentation in search of claims for support of CC security requirements. On average each component provided implementation for security requirements from about two CC security classes. (~2.167 classes per component)

Of the eleven classes of Common Criteria security requirements, COTS components were found which claimed support of security requirements from eight classes. Security requirements from Security Audit (FAU), Cryptographic Support (FCS), User Data Protection (access control) (FDP), and the User Identification and Authentication (FIA) security classes appeared to be the most common provided by the components surveyed. Table 4-2 shows a mapping of the implementation of CC security class requirements among the Volter component types.

No COTS components identified directly implemented security requirements from three classes of the Common Criteria: Privacy (FPR), Protection of the system's security system (FPT), and system access (FTA). However because of components support of related security classes it is likely that security requirements from these security classes could be significant for COTS components.

The Privacy (FPR) class contains requirements relating to user anonymity. Anonymity can appear in two forms: Anonymity to the world, as in anonymous access, and anonymity from hackers/attackers. Components providing cryptography to encrypt user identification and authentication communication could achieve some degree of anonymity from outsiders. So although the product documentation did not explicitly elicit support for anonymity it could be implied.

The protection of the system's security functions (FPT) classes includes requirements for the protection of a security system and its required data storage. A COTS component must provide a security system in order to be concerned with protecting it. Only two components surveyed provided a security system: User manager professional, and DataObjects .NET. Although the existence of FPT requirements were not confirmed from analysis, they are likely to exist. FPT_ITT and FPT_TDC are requirements related to internal security system data use within an application. It can be assumed that an application performing user authentication and access control functions is likely to need to transfer data internally. FPT_RPL replay detection and FPT_STM time stamps are FPT functions also related to security auditing (FAU). If a security system provides logging and analysis then these FPT requirements could be considered.

System access (FTA) requirements identify security requirements related to user connection quotas, session establishment rules, reports to the user about access history, and session locking criteria. FTA requirements are security functions related to User authentication and Identification (FIA). COTS components providing user authentication and identification functions could consider and implement these requirements as well.

CC Class	Domain (16)	Data (5)	User (1)	Technical (2)	Total (24)
FAU	(5) 31%	(2) 40%		(1) 50%	(8) 33%
FCO	(5) 31%				(5) 21%
FCS	(11) 69%	(1) 20%			(12) 50%
FDP	(2) 13%	(4) 80%	(1) 100%	(2) 100%	(9) 38%
FIA	(7) 44%	(1) 20%	(1) 100%		(9) 38%
FMT	(1) 6%			(1) 50%	(2) 8%
FPR					0
FPT					0
FRU		(3) 60%			(3) 13%
FTA					0
FTP	(3) 19%	(1) 20%			(4) 17%
Totals	34	12	2	4	52

Table 4-2 – Component type mapping to CC security classes

From the case study it is shown that COTS components could implement requirements from any of the CC security classes. This cast study suggests that the CC is likely to be useful for the specification

and evaluation of security for COTS components. The observations of this study support the claim in [7] that COTS components can be treated as IT systems for purposes of applying the CC. The CC can provide help to generate security requirements documentation for components being developed or evaluated. The CC evaluation process presented in figure 2-1 should be adaptable to the development of components. Section 5 proceeds with the presentation of a Common Criteria based process for component selection when security requirements are of concern.

5. Component Selection Process

The applicability of the Common Criteria to specifying and evaluating security is suggested by the component survey in section 4. This section presents a Common Criteria based process for component selection for selecting COTS components when security issues are of concern.

Step 0 – System High Level Design

A high level design is generated for the component-based system under development. The high level design should specify the underlying component-based architecture. Security concerns should be considered when considering the choice of component architectures. A CC-based evaluation of component architectures versus a specification of security concerns could be conducted. The output of step 0 should be to select and specify the underlying component technology. Since pluggable COTS components are primarily architecture dependent, the selection of the component-based architecture is necessary in order to scope the component search.

Step 1 – Component Requirements Definition

A Security Target (ST) document is generated to elicit the security requirements for the desired COTS component. The use of protection profiles (PP's), reusable sets of CC security requirements relevant to particular types of applications, could help to generate the ST document. For components, PP's could be generated which specify security requirements for sub-types of the Volter component types.

Step 2 – Component Search

Using the requirements identified in the ST an initial search for components is conducted. Searching product brochures, feature lists, and documentation, as stated in the ST, could identify COTS components that meet the minimum predefined requirements. In the event that no components are found during the search, the ST document should be refined to reduce the security requirements such that existing components can be identified claiming support for the requirements stated in the ST. An alternative to revising the ST and searching again is to abandon the search and develop a custom component to meet requirements.

Step 3 – Component Evaluation

The purpose of step 3 is to perform component evaluation to effectively eliminate inadequate candidate components from selection. Variations of the activities in step 3 should be considered to adapt the selection process to operate under the time and cost constraints of the evaluation. The Common Criteria based evaluation is conducted on the candidate components. Initially an EAL level 1 evaluation is performed. An EAL level 1 evaluation identifies forty evaluation activities. In many cases it

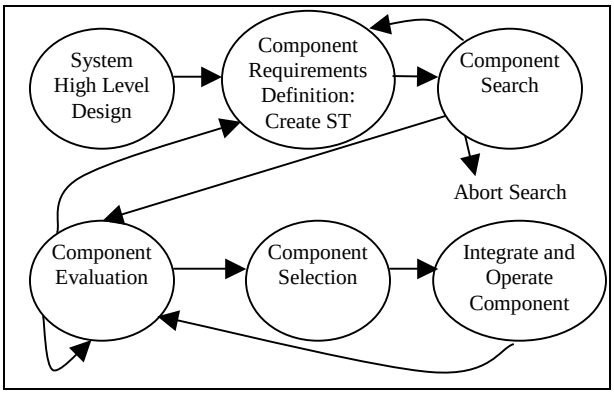

Figure 5-1 - CC based selection process

will not be necessary to perform all activities to arrive at a conclusion as to which component is the most appropriate based on the ST. Evaluation activities should be halted when an appropriate component is identified, or the evaluator can choose to pre-select a subset of evaluation activities to perform before considering the next action. The following ordering of evaluation activities is suggested: ADV_FSP Functional Specification and documentation evaluation, ADV_RCR evaluation of the representation correspondence to functional

requirements, ATE_IND independent testing, AGD_ADM Administrator Guidance, AGD_USR User Guidance, ACM_CAP CM Capabilities, and ADO_IGS Installation, generation, and start-up. This evaluation suggests performing function verification and testing activities first since they are most likely to identify functional inadequacies. In case there are multiple candidates that meet all ST security requirements after the EAL level 1 evaluation, then the evaluator must choose to either modify the ST to include more rigorous security requirements, or proceed with the next EAL level of evaluation. This process of modifying the ST or applying the next EAL level should be repeated until an appropriate component is identified.

Step 4 – Component Selection

The component that best meets the criteria stated in the ST is selected as a result of the CC evaluation activities performed in step 3.

Step 5 – Integrate and Operate the Component

Upon selection of the component, it should be integrated for use into the component-based system being developed. Once a component is in operation it is possible that previously unknown errors or vulnerabilities may surface. As a result of component operation required corrections could be identified. Once corrected such changes could require the component to be re-evaluated. The re-evaluation may require a complete revaluation or only an evaluation of the changes.

6. Process Application Example

The following is an example application of the CC-based component selection process.

Step 0: The high level design for the component-based system identifies Java as the language of implementation for the system. The component based system will consist of distributed client nodes which communicate with each other over an open network channel. Encryption must be used to protect data over the open channel. The component must provide an implementation to the Java Encryption Extension (JCE)

Step 1: A Security Target document is written to describe the security requirements for a required software component.

Sample simplified ST for a cryptographic component

A Java based component library is required to provide RSA encryption functionality. This component should provide the ability to implement RSA in a client/server based Java application. The generation of public/private keys for communication should be supported.

Based on Annex E., pp. 211-217 of the Common Criteria [9]:

Standards: PKCS #1- v2.1 RSA Encryption Std., FIPS std. Pub 140-2 – Computer Security – Cryptography – May 25, 2001

FCS_CKM _1.1. Cryptographic key generation
Key Generation algorithm: RSA
Asymmetric key generation
Cryptographic key size: 1024-bit

FCS-CKM_2.1. Cryptographic key distribution
Distribution method: A central authentication server provides public keys to clients as needed

FCS-CKM_3.1. Cryptographic key access
Cryptographic key archival on the local file storage media of the central key distribution server is used.

FCS_CKM_4.1. Cryptographic key destruction
Public keys expire after a specified period of inactivity. When a user attempts access with an expired key they will need to request a new key from the central key distribution server.

FCS_COP.1.1. Cryptographic Operation
Data encryption and decryption operations must be supported. Digital signature generation and verification must be supported.

Step 2: A search initially identifies (4) candidate components:

- RSA Bsafe
- IAIK-JCE
- Is Networks Provider
- Logi.crypto

Step 3: A partial EAL level 1 assessment evaluates the (4) candidate components eliminating candidates failing to pass the CC evaluation. The evaluation activities performed include: ADV_FSP, ADV_RCR, and ATE_IND. The ST document is adjusted as necessary by including information about the component features obtained from the initial evaluation to eliminate candidate components after the initial partial EAL evaluation. A second partial EAL level 1 evaluation proceeds to eliminate all but (1) component.

Step 4: The best component meeting the security requirements specified in the ST is selected.

Step 5: The component is integrated for use in the component-based system under development.

7. Conclusion

This paper began by identifying the component selection problem and the difficulty of component evaluation especially when security requirements are of concern. The Common Criteria was presented as a method for the assessment of security of IT systems. The CC helps to first generate security requirements documentation and then to provide an assessment methodology to validate that IT systems provide security conforming to specified evaluation assurance levels. A case study was conducted which identified the security functions provided by COTS components. Each component was classified according to Volter's component types. The CC security requirements provided by the component types were found by mapping the component types of the COTS components to the CC security classes. The most and least common areas of concern for the surveyed components were identified. In general it appears that COTS components act as IT systems with a small scope of security requirements. The survey components on average implemented security requirements from about 2 of the CC security classes. A Common Criteria based component selection process was then presented which adapts the assessment activities of the Common Criteria towards the purpose of selecting COTS components for component based systems.

The Common Criteria defines seven evaluation assurance levels. However evaluation efforts beyond EAL3 are not entirely applicable towards COTS component evaluation. Evaluations beyond level 3 require some evaluation activities to be performed during the software development phase [9] [17]. For evaluating COTS components it will be difficult to perform development-level evaluation since COTS components are preexisting before the evaluation. The first three levels are likely to provide sufficient information for component evaluation: EAL 1 (functionally tested) provides 40 evaluation activities; EAL 2 (structurally tested) defines 83 evaluation activities, while EAL 3 (methodically tested and checked) defines 110 activities.

Unlike many component selection processes the CC-based selection process attempts to eliminate components from the selection set, rather than calculating evaluation scores through the use of a analytical decision making process [18]. This approach could increase the time required for component evaluation. Decision making techniques such as the weighted sum method (WSM) or the Analytic Hierarchy Process (AHP) could be applied in cases where the Common Criteria evaluations do not quickly eliminate candidate components from the selection set [13]. By including a formal analysis of the component assessment data into the component selection process additional assessment activities could be avoided if a selection decision can be reached more rapidly.

Security Protection Profiles (PP's) could be generated for subtypes of Volter's component types. The subtypes could encompass popular component application domains. Through the COTS component survey many components were identified which provided similar security functionality especially for the Security Audit (FAU), Cryptographic Support (FCS), User Data Protection (access control) (FDP), and the User Identification and Authentication (FIA) classes. A common application domain identified in the survey was secure network/socket components. The identification of common domains and the creation of protection profiles for them could help developers more quickly create ST documentation for the specification of security requirements for common COTS components.

The Common Criteria's evaluation assurance activities for component assessment could be evaluated and customized by conducting cast studies of the application of the CC-based component selection process. Lessons learned from the application of this process could help identify process improvements to optimize the activities for best utilization of time and testing resources. The precise distribution of resources is likely to differ from organization to organization. Each organization may

wish to consider customizing the process further based on unique conditions.

Component based software engineering is becoming an ever more popular approach for software development. Security requirements for modern IT systems seem to increase with time. Component based software engineering must address the challenges of ensuring security within the development cycle in order to provide the promised productivity and quality gains. The Common Criteria can help to improve the security of component-based software engineering by supporting component requirements definition, security evaluation and selection.

8. References

[1] The Apache Struts Web Application Framework, [online] 2004, http://jakarta.apache.org/struts/ (Accessed: April 2004)

[2] 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.

[3] DataObjects .NET , X-Tensive.com, [online] 2004, http://www.xtensive.com (Accessed: April 2004)

[4] 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.

[5] Firesmith, D.G., Specifying Reusable Security Requirements, in Journal of Object Technology, vol. 3, no. 1, January-February 2004, pp. 61-75., http://www.jot.fm/issues/issue_2004_01/column6

[6] Franz, E., Pohl, C., Towards Unified Treatment of Security and Other Non-Functional Properties, In proceedings of the 2004 AOSD Technology for Application-Level Security Workshop, AOSD 2004, Lancaster, UK, 2004.

[7] Han, J., Zheng, Y., Security Characterization and Integrity Assurance for Component-Based Software, in proceedings of the international conference on Software Methods and Tools (SMT 2000), Wollongong, NSW Australia, pp. 61-66, 2000.

[8] IP*Works! Secure Tunnel – Add SSL/TLS Security to any server, /n software inc – The Leading Provider of Internet Components, [online] 2004, http://www.nsoftware.com/products/stunnel.aspx (Accessed: April 2004)

[9] 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

[10] 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.

[11] Kontio, J. A Case Study in Applying a Systematic Method of COTS Selection, in Proceedings of the 18th International Conference on Software Engineering, Berlin, Germany, 1996.

[12] Kunda, D; Brooks, L. Applying Social-Technical Approach for COTS Selection, Proceedings of 4th UKAIS Conference, University of York, McGraw Hill, 1999.

[13] Kunda, D., Brooks, L. Identifying and Classifying Processes (traditional and soft factors) that Support COTS Component Selection: A Case Study. Proceedings of the 8th European Conference on Information Systems, Vienna, Austria, 2000

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

[15] NIAP Protection Profile & Package Registry List, CC – Common Criteria for Information Technology Security Evaluation, [online] 2002, http://csrc.nist.gov/cc/ppreg/ppreg_pp_registry.htm (Accessed: April 2004).

[16] 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.

[17] Prieto-Diaz, Ruben, The Common Criteria Evaluation Process: Process Explanation, Shortcomings, and Research Opportunities, Technical Report, CISC-TR-2002-003, Commonwealth Information Security Center, James Madison University, Harrisonburg, VA., 2002.

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

[19] Völter, M., A Taxonomy for Components, in Journal of Object Technology, vol. 2, no. 4, July-August 2003, pp. 119-125., http://www.jot.fm/issues/issue_2003_07/article3