Dariusz Rogowski

Institute of Innovative Technologies EMAG

APPLYING COMPUTER TOOL TO COMMON CRITERIA METHODOLOGY

Introduction

The vast majority of users of software and hardware IT (Information Technology) products demand for some kind of assurance that security functionality of these products are reliable and dependable. Developers try to fulfill these demands by following the rules of different security standards which impose requirements on development, documentation and evaluation processes. Next these products can be evaluated by an independent licensed laboratory which can issue a certificate stating that the given IT device fulfills its security specification. In this way the user can get more confidence that the certified product will work securely as it is expected.

One of the security standards developers can use in their practice is the Common Criteria for Information Technology Security Evaluation standard (referred to as "Common Criteria" or "CC" throughout this paper), also known as ISO/IEC 15408 [CC12]. The CC standard provides a set of stringent development and evaluation rules for the security functionality of IT products. The first part of the CC standard is a general introduction to the methodology with the explanation of terms and definitions. The second and the third part describe respectively security functional and security assurance requirements for IT products and evidence documentation which is needed during the CC evaluation process.

There are many additional documents supporting developers in using the standard. Till now researchers prepared such documents like technical reports [TR09], and user's guides [PPST07, Dev07, Eval10]. These documents provides methodologies, techniques, documents patterns and practical hints, and recommendations which can be used by developers during writing the CC evidence documentation of their IT products. But this guides-based solution is not enough developers-friendly because they still have to do a lot of exhausting work manually what can be the source of many mistakes [RogNow12]. That is why some computer tools were next applied to improve the work with the guidelines and documents patterns.

Some of these software tools were described in [Kane08, Horie09, Higaki10, Trust13]. In [Rog13] I concluded that these tools were mainly dedicated to building only functional specification document of IT product and some of them were not longer supported by producers. Since these publications the situation has not changed much for better – the developers have not still got one integrated software tool with built-in documents patterns according to the CC methodology. Thus far applying the CC standard into the daily developers' practice is very difficult task so there is a huge demand for consulting CC services on the market. But these services are very expensive and developers seek for other cheaper alternatives in the form of supporting software.

The current paper presents the solution to the developers' problems mentioned above. This is the computer aiding system which was worked out in the research and development project titled "Common Criteria compliant, Modular, Open IT security Development Environment" (project acronym – CCMODE) [www1]. The computer system called CCMODE Tools automates evidence elaboration, security analysis and facilitates verification of documentation. Another result of the project was the set of patterns which can be used by developers in making evidence documentation necessary for CC evaluation and certification processes. These results can solve the common problem of lack of computer tools and patterns supporting developers in designing and manufacturing security-enhanced IT products according to the CC rules. This solution improves the quality of IT products development and documentation, and also facilitates the work of developers who are not familiar with the CC standard.

The paper is organized as follows. Section 1 presents the background and three main processes of the Common Criteria methodology. Section 2 delivers the short description of the CCMODE project research stages leading to the elaboration of documentation patterns and software tools. Section 3 gives an overview of the computer tool main modules. The last section contains discussion and conclusions, and states future work focused on further improvements of the tool.

1. The Common Criteria methodology – the primer

The CC standard assures that if IT products are developed and evaluated according to the given requirements then users will place more confidence to these products. This confidence is measured and quantified in CC by EALs

(Evaluation Assurance Levels) within the range from EAL1 to EAL7. If a higher level is chosen then the more stringent development process of an IT product should be used by a developer.

The CC methodology comprises three main steps [Rog13, Biał11]: 1) IT security development which is focused on security analysis and working out a special document called ST (Security Target); 2) IT product development whose aim is to make the product (called in CC language as TOE – Target of Evaluation) and to elaborate evidence documentation; 3) IT security evaluation which is conducted by an independent certified evaluation laboratory in order to assess the evidence documentation and security features of the product.

In the first step the ST document is prepared which says what is to be evaluated in the product. This document describes SPD (Security Problem Definition) – the security problem which is to be solved by the security features built into the product. The security problem describes threats to valuable user's assets. Next, in the same document, the security objectives are specified which have to counter the identified threats. These security objectives, expressed in a natural language, are further translated into the language of the CC standard by using SFRs (Security Functional Requirements) components. The ST document describes a specific product and is written by the developer, and it also claims conformance with the declared EAL. The chosen level determines all the requirements which have to be fulfilled during the product development. These requirements are grouped into the packages consisting of SARs (Security Assurance Requirements) components. These all SFRs and SARs are very difficult to follow by developers inexperienced in using the CC rules. This is why documents patterns were built up in the CCMODE project. The patterns have a fixed structure of chapters and sections and have footnotes with hints concerning the CC requirements.

In the second step, according to the written ST and the chosen EAL, the product itself and the rest of the documentation are made. The documentation must follow all the SARs taken from the given EAL package and include issues concerning among others: development of the product – security architecture, product design, functional specification; preparative and operational users guides; aspects of establishing discipline and control in the product development and maintenance during its whole life-cycle; testing the product; vulnerabilities analysis. In this step several documents have to be created and this is why it is a very complex and demanding task for a developer.

In the last step, an independent laboratory makes evaluation tasks according to the Common Evaluation Methodology (CEM) [CEM12]. CEM determines the minimum actions that have to be performed by the evaluator during an evaluation process where subsystems and modules of the product with built-in security functions are tested and their documentation is verified. After the positive result of the evaluation the product can be further certified by a certification body. The current list of certified products can be found at [www2].

2. Methodology – building the software tool

Little attention has been paid so far to the development of software tools which can support developers in preparing evidence documents. That is why one of the CCMODE project targets was to model and create such a tool which can be based on the design patterns. The details about the building of the CCMODE Tools system and its usage can be found at [Biał12]. At the very first stage of the project the model of a development environment of security-enhanced IT products was worked out. This model was worked out on the basis of such research methods like: security standards documents studies, the surveys and observations of IT products development environments, literature review. This model comprises pattern modules, methods and software which can be used for building similar environments for different types of IT products (software, hardware, etc.). The documents patterns and software models were elaborated by using analysis and logical modeling method.

In the project there were created design patterns for documents. The patterns were made for all security assurance components. Next the patterns were verified and validated by independent experts and developers chosen from the software and hardware industry. The validation was made upon the use cases research method. The developers were to use selected design patterns to make evidence documentation of their software and hardware IT products. As a result of the validation, necessary changes and amendments were incorporated into the patterns. Furthermore, the developers concluded that some automation features should be implemented into the patterns. This enabled to make functional assumptions for the software tool.

In the next project stages a prototype of the software tool was developed. The prototype was next validated in selected development environments in two fields (software and hardware) using the case study method. A few evidence materials were prepared by the testers. These case studies showed what can additionally be implemented in the tool to make the work with documents more effective and simpler. Mistakes in the patterns and the tool were eliminated and some functionality was improved.

As a result the CCMODE Tools system was worked out which integrates modules of development environment management, design patterns, knowledge base, evaluation methodology, and external supporting software for security analysis, testing and flaw remediation. Next section describe the modules of the system and their functionality.

3. Results – CCMODE Tools system main modules

The system consist of the following modules: Environment Management Tool (EMT), Documents Generator (GenDoc), Knowledge Base, Evaluation module, external supporting systems, optional security systems (Business Continuity Management – BCM or Information Security Management – ISM) which can be used as an additional source of assurance to the whole CCMODE Tools system and projects data. A general model of the system is depicted in Figure 1.

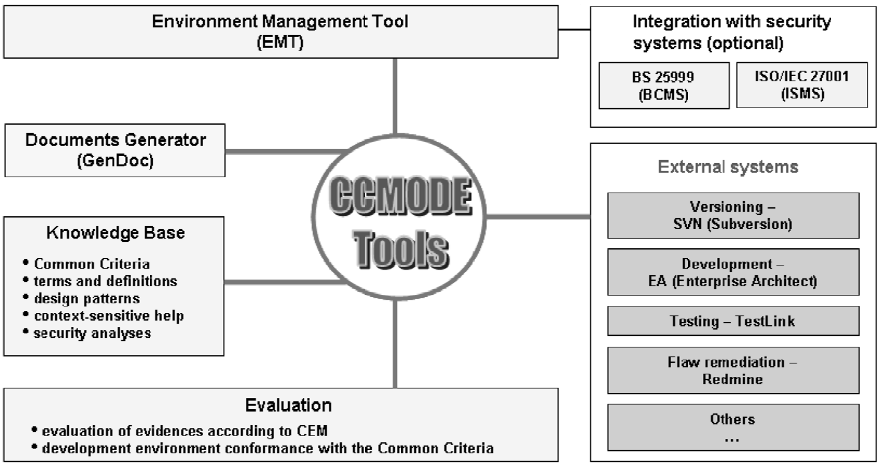

Environment Management Tool (EMT)
Integration with security systems (optional)
BS 25999 (BCMS)
ISO/IEC 27001 (ISMS)
Documents Generator (GenDoc)
CCMODE Tools
Knowledge Base
• Common Criteria
• terms and definitions
• design patterns
• context-sensitive help
• security analyses
External systems
Versioning – SVN (Subversion)
Development – EA (Enterprise Architect)
Testing – TestLink
Flaw remediation – Redmine
Others ...
Evaluation
• evaluation of evidences according to CEM
• development environment conformance with the Common Criteria

Fig. 1. The general model of the CCMODE Tools system

Source: Own elaboration based on [Rog13].

The EMT module supports the configuration of the development environment of IT products. Here it is possible to define the desired EAL level and the TOE lifecycle model. A list of evidence documents is defined automatically according to the chosen EAL. Every assurance component from the list is connected with the design pattern which can be later edited in the GenDoc application.

The Knowledge Base is a source of context-sensitive help about the CC requirements and accompanying documents like CEM. It comprises design patterns, terms and definitions which are also accessible in other modules. It also includes some notions of the project researchers concerning the ways how to resolve typical security problems with predefined security objectives, threats, assumptions, and security policies.

The external systems support versioning of files and documents – Subversion (SVN) application; modeling, development and analyses which are made with the usage of UML (Unified Modeling Language) – Enterprise Architect (EA); flaws reporting and flaws remediation – Redmine; management, plans and scenarios of tests – TestLink.

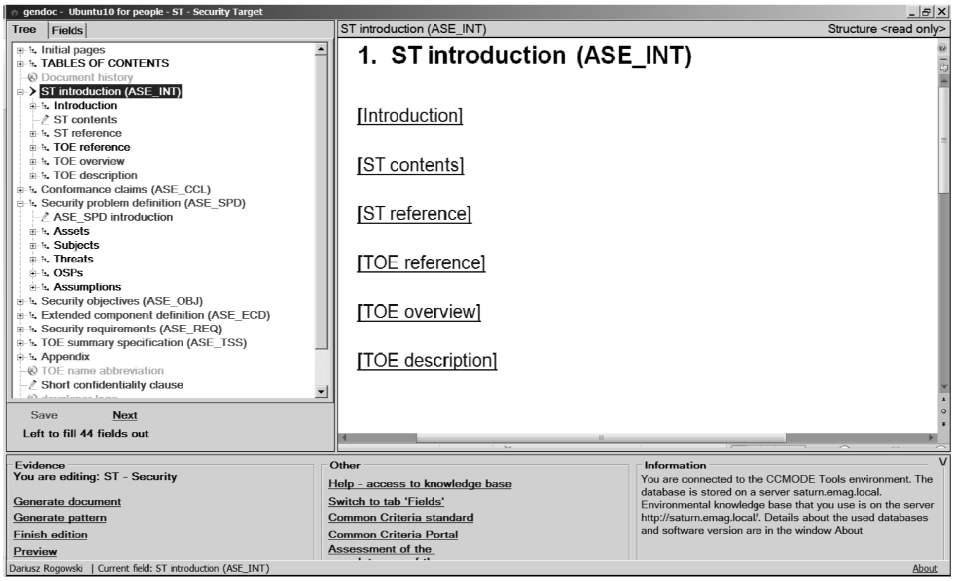
The evaluation module is used to verify the development environment against the CC requirements and to evaluate evidence documents according to the CEM methodology.

After the project is completed and properly configured the edition of the evidence documents can be started by using the GenDoc module.

3.1. Documents generator (GenDoc)

GenDoc is used for editing evidence documents based on the design patterns. In order to evaluate the TOE, an ST document and accompanying documents must be prepared. These additional documents are determined by the chosen EAL. This section shows by the example of ST how the software tool is used for filling in the patterns. The structure of the patterns, context-sensitive help and data links to modules of the CCMODE Tools were described.

The main editor window is depicted in Figure 2. Every pattern was prepared as a tree of data fields which represent chapters, sections and subsections of the output document. The tree is based on the requirements of the given CC component. Some of the fields are automatically filled in with the information taken from the Knowledge Base and other external software modules such as EA, EMT, SVN, TestLink. In order to create the document, the user must follow all the tree branches and find out which fields have to be completed.



gendoc - Ubuntu10 for people - ST - Security Target

Tree
Fields
Initial pages
TABLES OF CONTENTS
Document history
ST introduction (ASE_INT)
Introduction
ST contents
ST reference
TOE reference
TOE overview
TOE description
Conformance claims (ASE_CCL)
Security problem definition (ASE_SPD)
ASE_SPD introduction
Assets
Subjects
Threats
OSPs
Assumptions
Security objectives (ASE_OBJ)
Extended component definition (ASE_ECD)
Security requirements (ASE_REQ)
TOE summary specification (ASE_TSS)
Appendix
TOE name abbreviation
Short confidentiality clause
Save
Next

Left to fill 44 fields out

ST introduction (ASE_INT) Structure <read only>

1. ST introduction (ASE_INT)

[Introduction]

[ST contents]

[ST reference]

[TOE reference]

[TOE overview]

[TOE description]

Evidence

You are editing: ST - Security

You are connected to the CCMODE Tools environment. The database is stored on a server saturn.emag.local. Environmental knowledge base that you use is on the server http://saturn.emag.local/'. Details about the used databases and software version are in the window About

Dariusz Rogowski | Current field: ST introduction (ASE_INT)

Fig. 2. The main window of the GenDoc application with the ST pattern

Source: Own elaboration based on [Rog13].

Every field has its own context help which gives guidelines and hints about the information to be delivered. The following types of help are distinguished: ready to use – it comprises the text which is ready to use by the user without the necessity to change any information in it; Common Criteria help – comprises all the information and requirements taken from the CC standard according to the selected and currently edited branch of the pattern; hints – these are interpretations, tips and guidelines which help the user to write down proper information in the data field; examples – sample of information for the given data field; data sources – the external system from which the data were downloaded.

After completing all the information in the pattern, the output evidence document can be generated by a user and saved in the SVN repository of the project.

The most important part of the ST pattern is the security problem definition section which is outside the scope of the CC methodology. That is why the special plug-in software was developed in the CCMODE project. The plug-in is the part of the CCMODE Tools system and uses basic features of the EA external system. The plug-in supports developers by applying wizards and inferring features which help in defining the security problem that is to be solved by the IT product. The security problem definition (SPD) made in the plug-in is automatically uploaded to the proper sections of the ST document. In the next subsection main features of the plug-in tool were described.

3.2. Plug-in tool for defining security problem

The plug-in has the form of a special toolbox with objects needed for defining such elements of SPD as (Figure 3): assets, threats, assumptions, organisational security polices, security objectives for TOE, security objectives for environment, security functional requirements, security functions. These are the main drag-and-drop objects with special wizards which facilitate creation of security problem definition and its solution.

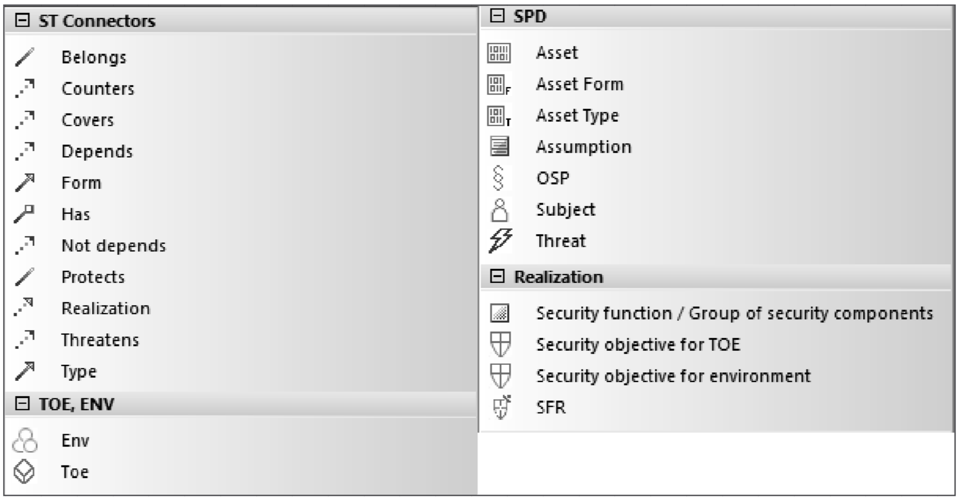


Fig. 3. Basic elements of the plug-in toolbox

Source: Own elaboration.

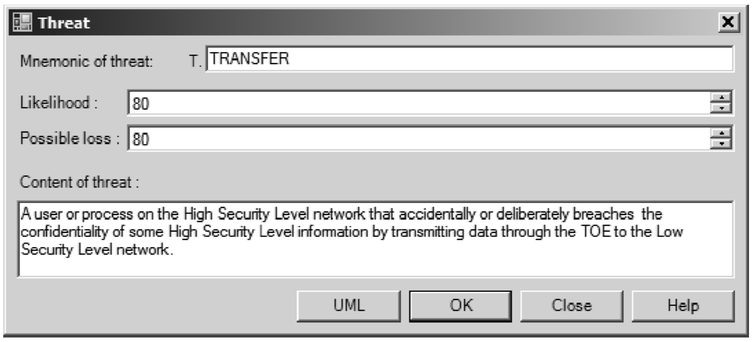
In Figure 3 there are the following main objects of SPD and Realization sections of the toolbox [CC12]:

- Asset – the user's goods that have to be protected by the TOE, e.g.: information, process, system data, database. The assets can have the form (Asset form): transferred, processed, stored, displayed, etc., the assets can have the type (Asset type): business process, cryptographic key, software application, hardware device.
- Assumption – it is made on the operational environment in order to be able to provide security functionality of the TOE. If the TOE is placed in an environment that does not meet these assumptions, the TOE may not be able to provide all of its security functionality anymore.
- OSP (Organisational Security Policy) – this describes security rules, procedures, or guidelines imposed by organization in the operational environment.

- Subject – active entity in the TOE (users, threat agents, processes, hardware elements) that performs operations on objects (passive entities in the TOE, that contain or receive information).
- Threat – it consist of an adverse action performed by a threat agent on an asset.
- Security objective for the TOE – it consists of a set of objectives that the TOE should achieve in order to solve its part of the security problem.
- Security objective for environment – the operational environment of the TOE implements technical and procedural measures to assist the TOE in correctly providing its security functionality (which is defined by the security objectives for the TOE).
- SFRs (Security Functional Requirements) – these are a translation of the security objectives for the TOE in the form of the CC requirements.
- Security function – it shows the general implementation of the SFRs and description of how the TOE satisfies all the SFRs. It provides the general technical mechanisms that the TOE uses for this purpose.

The toolbox also includes connectors and realization instances for making relations between the objects. More details about the plug-in and the example of its usage can be found in my paper [Rog14].

The objects are dragged-and-dropped symbols which have dialog windows (Figure 4 and 5) with detailed descriptions. Figure 4 depicts the example of a threat definition which consists of: mnemonic threat name; the likelihood of threat occurrence and the value of possible asset losses caused by a threat; description of the threat which shows adverse actions performed by a threat agent on an asset.



Mnemonic of threat: T. TRANSFER

Likelihood : 80

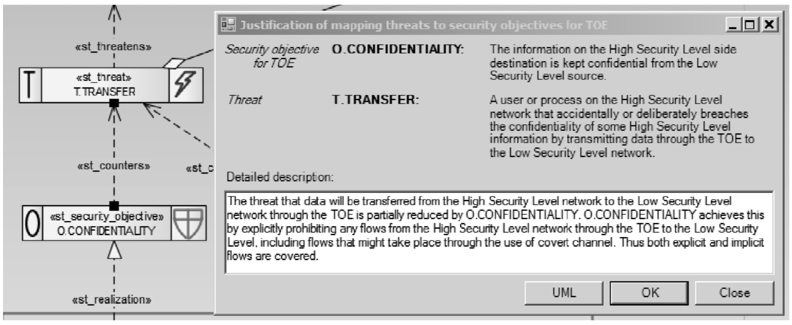
Possible loss : 80

Content of threat :

A user or process on the High Security Level network that accidentally or deliberately breaches the confidentiality of some High Security Level information by transmitting data through the TOE to the Low Security Level network.

Fig. 4. Threat details window

Source: Own elaboration.



Security objective for TOE **O.CONFIDENTIALITY:** The information on the High Security Level side destination is kept confidential from the Low Security Level source.

Threat **T.TRANSFER:** A user or process on the High Security Level network that accidentally or deliberately breaches the confidentiality of some High Security Level information by transmitting data through the TOE to the Low Security Level network.

Detailed description:

The threat that data will be transferred from the High Security Level network to the Low Security Level network through the TOE is partially reduced by O.CONFIDENTIALITY. O.CONFIDENTIALITY achieves this by explicitly prohibiting any flows from the High Security Level network through the TOE to the Low Security Level, including flows that might take place through the use of covert channel. Thus both explicit and implicit flows are covered.

Fig. 5. Rationale window for the chosen threat and security objective

Source: Own elaboration.

In Figure 5 the security objective rationale detailed description demonstrates how this security objective is achieved. If the security objective is met then the tracing to the threat is effective (it means that the given threat is countered).

A user selects all the necessary symbols in order to complete the security problem definition of the TOE. More over the plug-in offers wizards with inferring functions which support users with choosing security objectives and security functional requirements. The deductions are based on the weights parameters of the objects and on the predefined security objectives, and threats stored in the knowledge base.

In the result, the graphical form of SPD and its solution are worked out by the user (Figure 6).

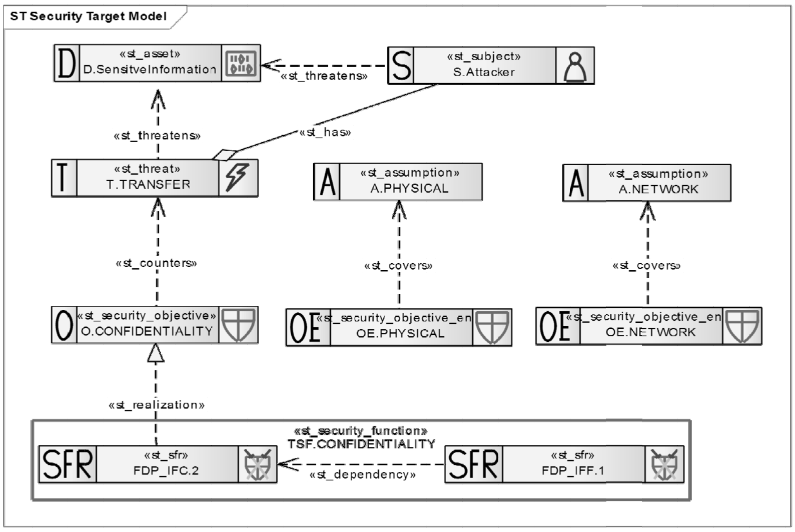


Fig. 6. The SPD graphical model

Source: Own elaboration based on [Rog14].

This graphical form of SPD can be easily analyzed by the developer because all the elements of SPD with its solution together with the relations between them are gathered in one place. The solution implementation is showed by SFRs and security functions.

Next this SPD is automatically uploaded to the ST document and can be edited in the documents generator (GenDoc) application. There is no place here to describe the whole methodology of the SPD elaborating because it is an extensive topic but the reader can find more details about the SPD issue in the following paper [Biał13].

Discussion and conclusions

The paper presented the computer-aided tool – CCMODE Tools. The tool was worked out in order to support developers in designing and manufacturing IT products according to the stringent rules of the CC standard. Using the CC methodology by users not familiar enough with the standard is very difficult and time consuming task.

The results of the described project are: the EMT application which helps in projects management, the GenDoc tool which supports creating evidence documents, and finally the plug-in application which facilitates elaboration of the SPD and its solution.

For the developers the CCMODE Tools system delivers the complete solution which facilitates writing evidence documents and security analysis of the IT product that is to be evaluated and certified. The context-sensitive help connected to every module of the CCMODE Tools allows the developers to concentrate on building the content of the documents only. Developers are not distracted by thinking about the form of the documents or by searching necessary requirements in the CC standard. The tools facilitates and speeds up the IT security development process, and improves the quality of evidences because they include all details required by the considered assurance components and best practices.

For the researchers the results of the CCMODE project can pave the way for using the CC standard in early stages of developing security features for their prospective IT security-enhanced products. The results could be also the starting point for seeking and improving the risk analysis methods of choosing the best security objectives for the given threats.

The CCMODE Tools system can be used for developing security functionality of different types of IT products (hardware, software, firmware, systems). For example the CC methodology and CCMODE Tools were used in an experimental laboratory of the EMAG Institute – SecLab [BiałFli14] for development of intelligent sensors for the mining industry and specialized software. The tool was also validated on the example of a data diode [Rog14] and a biometric system [Biał13]. Thus the tool can be used by developers for a wide range of IT products and system.

There are of course some limitations of the presented software solution. It concerns mainly the knowledge base which does not include help with all possible answers for developers problems. It does not also include all suitable security objectives which can counter all likely threats which are used in defining the security problem. The knowledge base should be constantly extended and tuned by using feedback from developers on practical usage of the tool. Further, the documents generator application GenDoc produces evidence documents which sometimes have to be manually complemented in some sections by developers. It has reasons in the limited structure of the evidence patterns and limited automation features of the GenDoc application. There will be always some activities which cannot be automated and information which have to be written down manually. But in spite of these drawbacks the CCMODE Tools system supports the developers in such a way that gives a great chance to evaluate IT products successfully in a independent Common Criteria evaluation laboratory.

Future researches will be focused on enhancing the risk analysis module of the plug-in and on extending the knowledge base. The plug-in could implement the costs of security measures, probability of threats, costs of possible assets losses, etc. These could support users in decision making according to security objectives and costs of their implementation. The knowledge base data could be extended by adding new guidelines and new relations between the SPD objects (threats, security objectives, and SFRs) used by the inferring wizards.

References

[Biał11] Białas A. (red): Zastosowanie wzorców projektowych w konstruowaniu zabezpieczeń informatycznych zgodnych ze standardem Common Criteria. Wydawnictwo Instytutu Technik Innowacyjnych EMAG, Katowice 2011.

[Biał12] Białas A. (red): Komputerowe wspomaganie procesu rozwoju produktów informatycznych o podwyższonych wymaganiach bezpieczeństwa. Wydawnictwo Instytutu Technik Innowacyjnych EMAG, Katowice 2012.

[Biał13] Białas A.: How to Develop a Biometric System with Claimed Assurance. Proceedings of the 2013 Federated Conference on Computer Science and Information Systems (FedCSIS), pp. 775-780.

[BiałFli14] Common Criteria Compliant It Product Development in the Experimental Seclab Laboratory. Proc. of CSS'2014, Uniwersytet Ekonomiczny, Katowice (accepted for publication).

[CC12] Common Criteria for Information Technology Security Evaluation, part 1-3, ver. 3.1, rev. 4, 2012.

[CEM12] Common Methodology for Information Technology Security Evaluation (Version 3.1, Revision 4) Evaluation Methodology. CCMB, September 2012.

[Dev07] Guidelines for Developer Documentation According to Common Criteria Version 3.1, BSI, 2007.

[Eval10] Guidelines for Evaluation Reports According to Common Criteria Version 3.1, Version 2.00, BSI, 2010.

[Higaki10] Higaki W.H.: Successful Common Criteria Evaluations. A Practical Guide for Vendors. Create Space Independent Publishing Platform, 2010.

[Horie09] Horie D., Yajima K., Azimah N., Goto Y., Cheng J.: GEST: A Generator of ISO/IEC15408 Security Target Templates. In: Computer and Information Science 2009, SCI 208. R. Lee, G. Hu, H. Miao (eds.). Springer-Verlag, Berlin, Heidelberg 2009, pp. 149-158.

[Kane08] Kane I.: Automated Tools for Supporting CC Design Evidence. 9th International Common Criteria Conference, Jeju 2008.

[PPST07] The PP/ST Guide, Version 1, Revision 6.2, BSI, 2007.

[Rog13] Rogowski D.: Computer-aided Tool Based on Common Criteria Related Design Patterns. In: Business Informatics. J. Korczak, H. Dudycz, M. Dyczkowski (eds.). Wrocław University of Economics Research Papers 2013, No. 3(29), pp. 111-127.

[Rog14] Rogowski D.: Software Support for Common Criteria Security Development Process on the Example of a Data Diode. In: Advances in Intelligent and Soft Computing. Vol. 286. W. Zamojski et al. (eds.). Springer 2014, pp. 363-372.

[RogNow12] Rogowski D., Nowak P.: Pattern Based Support for Site Certification. In: Complex Systems and Dependability AISC Vol. 170. W. Zamojski et al. (eds.). Springer-Verlag, Berlin, Heidelberg 2012, pp. 179-193.

[TR09] ISO/IEC TR 15446 – Information Technology – Security Techniques – Guide for the Production of Protection Profiles and Security Targets. JTC 1/SC27, Berlin 2009.

[Trust13] Trusted-labs. www.trusted-labs.com, accessed May 2013.

[www1] http://www.commoncriteria.pl.

[www2] http://www.commoncriteriaportal.org/, 2014.

ZASTOSOWANIE NARZĘDZI KOMPUTEROWYCH W METODOLOGII COMMON CRITERIA

Streszczenie

Artykuł obejmuje prezentację rozwiązania, jakim jest system informatyczny opracowany w ramach projektu badań i rozwoju pt. Common Criteria compliant, Modular, Open IT security Development Environment (CCMODE). System informatyczny zwany CCMODE Tools automatyzuje opracowanie dowodu, analizę bezpieczeństwa i ułatwia weryfikację dokumentacji. Proponowany system jest rozwiązaniem dla doskonalenia jakości rozwoju i dokumentacji produktów technologii informacji oraz ułatwia pracę projektantom, którzy nie znają standardu CC. W artykule przedstawiono podstawy i główne procesy metodologii Common Criteria, umieszczono krótki opis projektu CCMODE, scharakteryzowano główne moduły systemu, sformułowano wnioski i określono kierunki dalszych badań.