

A Semiformal Approach to the Security Problem of the Target of Evaluation (TOE) Modeling

Andrzej Białas

Institute of Control Systems, 41-506 Chorzów, Długa 1-3, Poland

e-mail: abialas@iss.pl tel. +48 32 3492-900

Abstract – The paper deals with the IT security development process according to the Common Criteria, particularly the security problem definition of the target of evaluation (TOE), expressing basic security concerns and needs of the developed IT product or system. The conciseness and preciseness of these concerns and needs influence the security objectives specifications that, in turn, influence the design quality and, finally, the TOE assurance. The UML-based approach and the predefined set of enhanced generics are proposed. The paper shows the used high level ontology, its representation by generics and the operations defined on these generics. The paper deals with more extensive works concerning IT security modeling and the development of computer-aided tools. To conclude the issue, current results, experiences and plans were presented. The emphasis was put on UML-based designs which can be better understood by a wide community of UML users.

Keywords: Common Criteria, IT security development, security engineering, UML, modeling.

1.0 Introduction

The development of the e-business, e-government or e-health applications and the critical information infrastructure protection will not be possible if trust and confidence technologies do not grow. These technologies, in turn, will have no chances to develop if there is no assurance basis for them. Generally, assurance is understood as the confidence that an entity, i.e. IT product or system, called the TOE (target of evaluation), meets the security objectives specified for it. The Common Criteria (CC), i.e. ISO/IEC 15408 [1] approach is one of the well established methodologies concerned with the creation of assurance. This international standard, currently in version 3.0, provides criteria for evaluating IT products or systems with focus on their assurance. It was assumed that the assurance foundation is created during a rigorous IT development process, while verification is done during independent evaluation and operation of a certified IT product or system. The users, entrusting their information and other business assets to IT systems, ought to believe that IT systems with their safeguards guarantee the right assurance, i.e. they offer intended functionality and eliminate unexpected, malicious behavior.

The TOE can be expressed with the use of one of the following kinds of specifications that have well-defined structures (additionally, the CC v. 3.0 introduces their simplified, “low assurance” versions):

- security target (ST), which contains an implementation-dependent statement of security needs, for the specific IT product or system;
- protection profile (PP), which includes an implementation-independent statement of security needs; it characterizes a family of IT products or systems.

These specifications are created during the IT security development process and serve as main documents on entry of the evaluation process. The information they include concerns IT security needs, requirements, features and functions, evidence to back up the assurance, reference documents – this allows to judge whether the TOE meets the declared evaluation assurance level (EAL1 to EAL7).

The paper concentrates on the means and tools for the TOE security environment specification, called “security problem definition” in the latest CC v. 3.0. Both names will be used, because the previous one meets the [2] and is better known from numerous ST or PP documents issued for certified products. The elaboration of the TOE security environment is the key step in the IT security development that influences directly the security objectives specification. Meeting the security objectives by the IT product or system, elaborated further to satisfy requirements and different design specifications, is the essence of the assurance term and the subject of the evaluation process. Providing means and tools that allow to reach adequate, concise and precise specifications, influences the cost and quality of the IT security development process and the issued designs of IT products or systems. The paper deals with the UML approach concerning IT security development according to Common Criteria [1] and the related standards [2], [3]. Although the methodology presented there encompasses the whole development process, the paper focuses on the security problem definition. After short introduction to the IT security development process, the security environment high level ontology will be presented. Then formalized definition of generics will be introduced and the enhanced generics set will be shortly characterized. It should be noted that due to parameterization and defined operations on generics, they have comparable features to the functional and assurance components defined by the [1]. The example of an IT product environment will be shown and illustrated by the supporting tool. Generally, the UML approach allows to achieve:

- more consistency of the modeled features, systems and their processes,
- their better understanding by developers, evaluators, administrators, auditors and other users,
- possibility to develop computer-aided tools on this basis.

The paper presents an example of using the UML language within the domain of the security systems development. The UMLsec [4], being the UML extension, provides a unified approach to security features and behaviors description. The paper [5] presents advanced and Common Criteria based composition methods. The UML specification of the generics and components libraries used in the computer-aided Common Criteria IT development process was discussed in [6]. The UML use for the information security management systems in the [7], [8], [9], [10] was presented. There are also works concerning formal methods and tools for the advanced Java smartcards development [11], [12].

2.0 IT security development process

There are four stages in the security target elaboration process [2]:

- establishing the so-called security problem definition (security environment) specification;
- setting security objectives – for the TOE and its operational environment;
- using CC components catalogues [1] and analyzing the above objectives, working out the sets of functional and assurance requirements for the TOE and for the operational environment;
- using functional and assurance requirements, preparing the TOE summary specification (TSS) – deals with ST specification only.

It is necessary to provide rationales for transitions between stages. One can use the evaluated PP to elaborate different, but compliant with the identified PP, security targets, on whose basis the TOE is developed. Please note that thanks to IT security development processes it is possible to conduct step-by-step IT security specification refinement – until the implementation-dependent TSS specification, used on entry in the TOE development, is completed. During the TOE development the design specifications are refined, starting from security requirements and TSS, through functional specification, high-level design specification, implementation, to low-level specification and others, required by the assurance development class [1] (ADV) according to the declared EAL level.

3.0 Security problem definition and its specification

The TOE security environment specification discussed there (Figure 1), defining the nature and scope of security needs to be addressed by the TOE, is the first main development stage. Please note three basic elements of this specification, i.e. threats, security policy rules (OSPs) and assumptions, and

some auxiliaries. Basic relations between all of them are presented as the high level ontology. The first step is to identify and optionally valueate the TOE assets (internal, e.g. user data file, plain text within the smartcard microprocessor register) and/or the assets protected by the TOE (external, e.g. database system behind the firewall being the TOE). The TOE as such and the realm outside the TOE, i.e. within its operational environment, require to be identified. Please note that subjects may represent both legal users (e.g. in OSPs) and different types of threat agents.

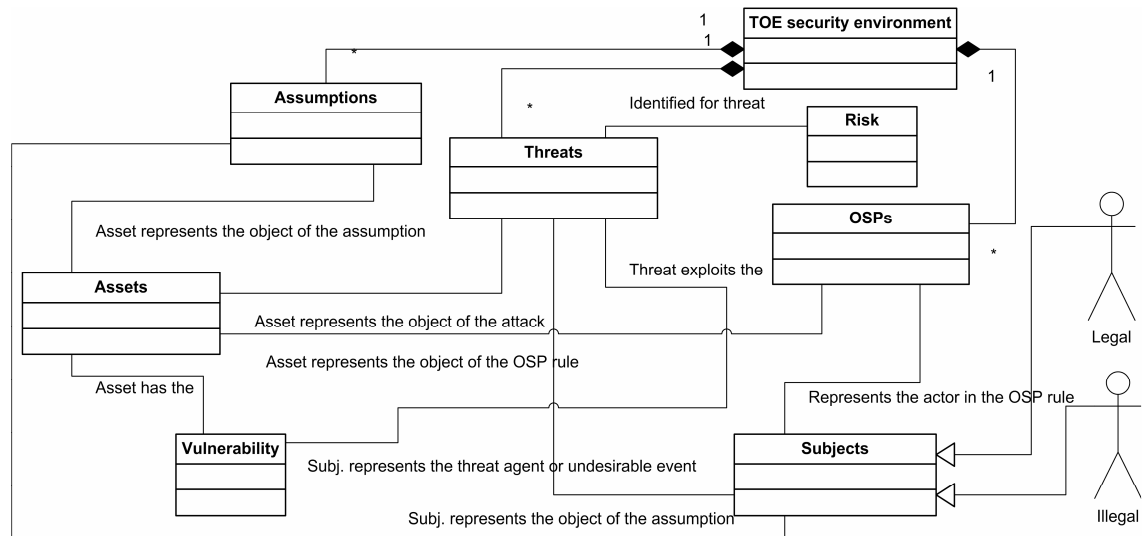


Figure 1. TOE environment high level ontology as the UML class diagram

The goal is to identify all possible threats to the assets, exploited vulnerabilities and optional risk (likelihood, consequence) existing in the TOE and its environment. These threats define the scope of IT security needs as they will be the main target of all actions undertaken in the next stages. Some specific security needs can be expressed by security policy rules: those specified in the code of practice, those which enforce appropriate behavior and operation; and those which restrict TOE environment. The developer analyzes the TOE, its operational environment, threats, and required policies, and usually specifies some assumptions which ought to be satisfied obligatorily. These assumptions can exclude the occurrence of some threats within the TOE boundary or its environment. They also deal with the TOE intentional usage, the proper behavior of the personnel during the use of the TOE within its environment, the environment responsibility in the TOE protection, and connectivity aspects. Assumptions implying specific assurance requirements are assigned if the TOE is designated for a special purpose, e.g. to protect assets of considerable value.

The elaboration of more refined security objectives for the TOE can be done with the use of the security environment specification. These objectives provide a concise statement of the intended response to the security problems (being mainly threats, in fewer cases OSPs and assumptions). This is the focal point of the whole ST or PP, a linkage between security concerns and precise security requirements specification based on CC components.

4.0 Enhanced generics to specify the TOE security issue

The Common Criteria contains semiformal specification means, i.e. functional and assurance components that are used only on the requirements specification level. To provide other stages, such as the security environment stage, with the means of comparable possibilities, some developers tried to use more formalized means, i.e. generics. The paper [6] introduces a unified definition of enhanced generics. The generics can be considered as mnemonic names that express common features, behaviors or actions of different aspect or elements of IT security. The following general format of a generic was assumed ([] stands for an optional field in the generic or a parameter within the generic, while "?" substitutes one character to distinguish generic subtypes, if they exist):

Generic = [*Domain*].[*Type*].[*Mnemonic*].[*Version*].[*Description*].[*Refinement*].[*Attributes*], where:
„*Domain*” refers to the area of applications, e.g. GNR concerns common aspects, COM – network equipment, CRP – cryptographic applications, SCR – smart cards;

„*Type*” deals with the group of IT security aspects (concerns), like:

- Data and other assets – DA?, e.g. DAD represents data asset, DAS – service,
- Actors (legal subjects, intruders, sources of undesirable events) – S??. e.g. SAU – authorized users, SNA – unauthorized users, SNH – undesirable events, non-human actions,
- Different kinds of threats T??. e.g. TDA – direct attacks, TFM – force majeure,
- Organizational security policies (OSPs) – P??. e.g. PACC – access policy, PIDA – identification and authentication, PCON – confidentiality, PEIT – IT aspects in the operational environment, PSMN – security management, PEPH – physical protection,
- Assumptions for the environment – A?. e.g. AU- deals with intended usage of the TOE, AP – personnel, AE – to be satisfied by the operational environment,
- Security objectives for the TOE or its IT environment – O??. OCON – deals with confidentiality, OACC – access control, OEIT – must be satisfied by IT means of the operational environment, OINT – concerns integrity, OSMN – security management aspects,
- General security requirements for the environment: general IT aspects (REIT) dealing with technical infrastructure or physical security (REPH) and non-IT aspects (RENIT),
- TOE security functions – F;

„*Mnemonic*” – a concise expression of a feature, behavior or action, useful in analyses and visualization of relationships;

„*Version*” remains “empty” in the case of the basic version, while “Dn” stands for the version derived from the basic one, according to developers’ needs, where *n* represents its successive number;

„*Description*” – a full description that expresses the mnemonic meaning;

„*Refinement*” – details and interpretations regarding the “Description”; this field is attached by the developer and matches the meaning of a generic to the TOE reality;

„*Attributes*” – a list of attributes may have values assigned, e.g. AV (asset value), EV (exploited vulnerability), EL (event likelihood), AVL (asset value loss), RV (risk value), I (influence of the security objective, i.e. “preventive”, “corrective” or “detective”).

Example 1. Deals with an alternative [13] definition of the smart card controller security problem using the above introduced generics (Figure 2). Environment generics suggest specific security objectives. These have default requirements assigned and the requirements have the predefined security functions assigned by default. These assignments make the entire IT security development process easier. The *TDA.DiffPowAnal* is a 2-parameter generic example (“<=” means substitution).

TDA.DiffPowAnal. Card intruder [S??_param<= SNA.HighPotenIntrud] may compromise cryptographic keys [DAD_param<= DAD.EncKey] by analyzing power consumption of the smart card chip during the cryptographic operation (i.e. Differential Power Analysis Attack).

SNA.HighPotenIntrud. Intruder having high level skills, enough resources and deep motivation to perform a deliberate attack.

DAD.EncKey. Cryptographic Keys used as input parameter for encryption or decryption stored in a register of the block cipher algorithm of the TOE.

Please note the proposed OCON.DiffPowAn objectives to solve this problem:

OCON.DiffPowAn. The TOE will ensure the confidentiality of keys during cryptographic function performed by the TOE.

The *TDA.CrpAnal* generic contains more details called refinements, and this generic is also an example of iteration. The iteration is the use of the same generic (e.g. threat) in the specification many times (instances are numbered) with different parameters assigned (e.g. assets). The iteration allows more consistency and has the same possibility as the iteration for the CC components.

TDA.CrpAnal. Card attacker [SNA_param] may compromise [DAD_param] user data being encrypted by the TOE or he/she may compromise the key needed to calculate the plain text from

cipher text. Refinement: To perform this attack the intruder has to know the cipher text but is neither able to use the decryption function of the TOE nor to observe the behavior of the TOE during the cryptographic operation.

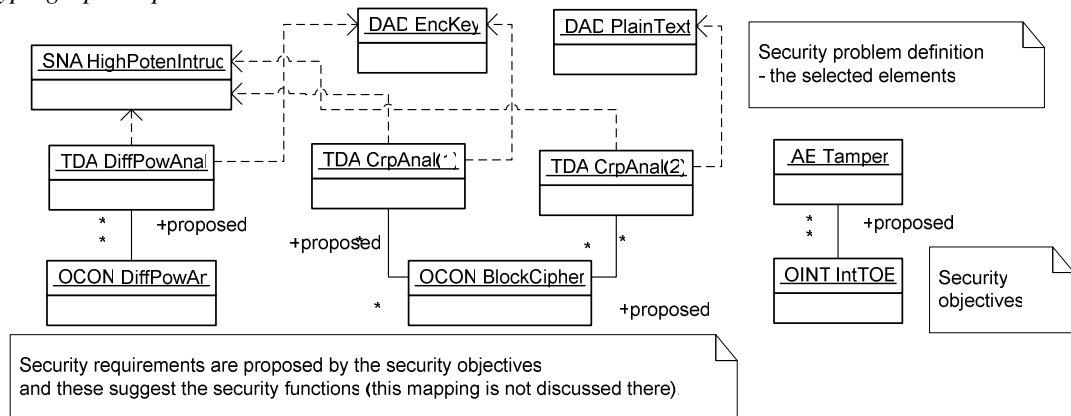


Figure 2. The selected elements of the smart card TOE environment specification. The UML objects represent generics

DAD.PlainText. Plain document to be encrypted. Refinement: placed in the smartcard register.

DAD.EncKey. Cryptographic keys used as input parameter for encryption or decryption.

SNA.HighPotenIntrud. Intruder having high level skills, enough resources and deep motivation to perform a deliberate attack.

OCON.BlockCipher. The TOE will implement a cryptographic strong symmetric block cipher algorithm to ensure the confidentiality of the plain text by encryption and to support secure authentication protocols.

The Figure 2 presents an example of a non-parameterized assumption generic describing the operation environment responsibility regarding tamper resistance. It implies *OINT.IntTOE* objective.

AE.Tamper. The environment in which the smart card (plastic card with an embedded chip) is used guarantees the physical integrity of the TOE embedded in the smart card and the usage of the TOE under the defined working conditions (which are described in the user documentation).

OINT.IntTOE. The TOE will protect itself against external interference or tampering by untrusted subjects, or attempts by untrusted subjects to bypass the TOE security functions.

For the mentioned smart card no OSPs were needed, though the OSPs are the basic security problem statement as well.

Example 2. Presents the restricted TOE user access to the *DAD.DataFile* asset owned by the *SAU.AssetOwner* basing on the discretionary access policy rule expressed by the *PACC.DAC* generic.

PACC.DAC. The right to access specific data objects [DA?_param <= DAD.DataFile] is determined on the basis of: the owner of the object [SAU_param <= SAU.AssetOwner], the identity of the subject attempting the access [SAU_param <= SAU.TOEUser] and the implicit and explicit access rights to the object granted to the subject by the object owner.

SAU.AssetOwner. Distinguished asset owner.

SAU.TOEUser. TOE user with restricted access rights.

The developer can: derive other generics using the existing ones, refine them by adding some specific details, assign parameters and perform the iteration. These features make the development process easier, allow to reach more concise and precise specifications, and support reusability.

5.0 Implementation in the computer supporting tool

The generics (about 400) are implemented as the library of the computer tool [14] supporting IT security development and evaluation (Figure 3). All developer's activities are wizard-driven (the right

part of the window). The upper part of the main application window presents the specified security environment elements with the use of tree structures. Please note the basic threat specification elements:

- assigning the threats to the TOE, to its operational environment or to both, which decides where the security objectives will be assigned; this indirectly influences the TOE shape and cost,
- threat agent and threatened asset specified by appropriate generics,
- exploited vulnerability,
- risk parameters of the threat (in the predefined scale), i.e. occurrence rate, consequence, risk value can be considered, while the security objectives are selected to encounter a given security problem.

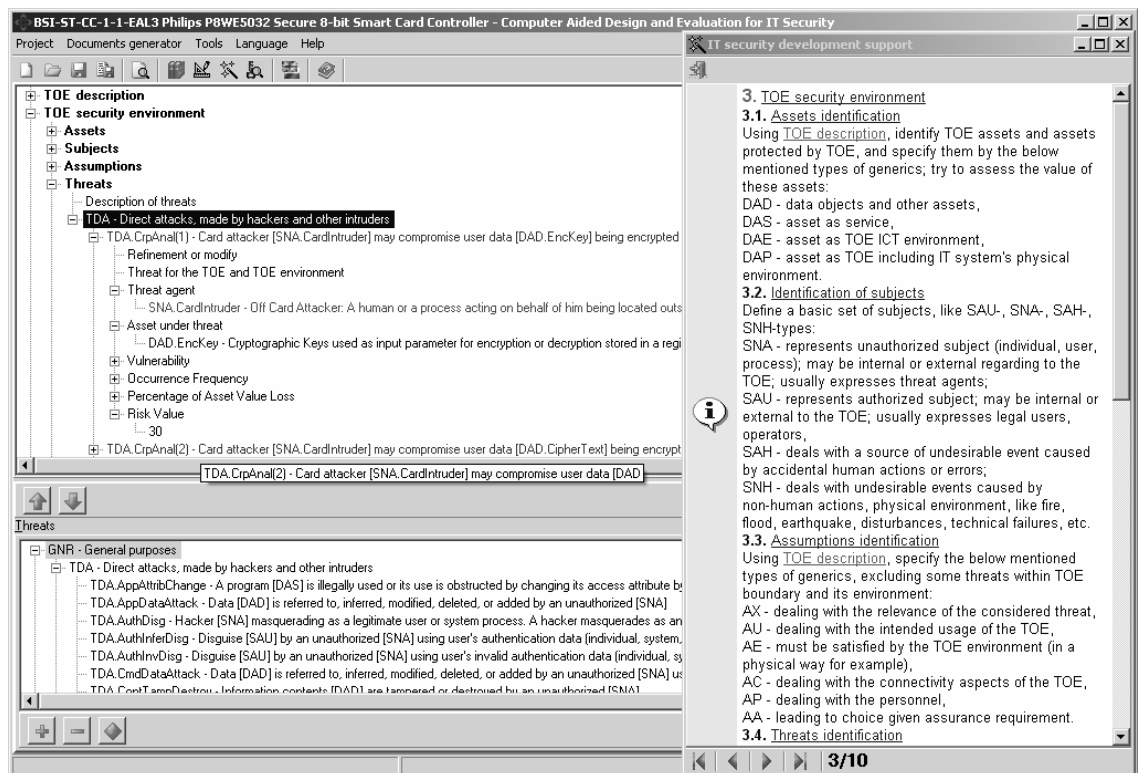


Figure 3. The security problem definition using supporting SecCert tool [14]

The lower part of the main window presents generics or components to be selected by the developer, parts of automatically generated reports or graphical visualization of the relationships between generics and/or components.

The developed tool better supports the development process with respect to its different aspects, like: risk analysis, rationale process, composability, reusability, visualization of relationships, preparing evidence, designing trade-offs problems solutions, specification completeness and correctness. Developers can focus on problem solving (better design entry, integration with the UML modeling tools). The tool allows to issue designs that are more precise and compliant with the IT security standards. The tool reduces the costs and improves the effectiveness of the evaluation process.

The works aimed at the development of the supporting tool have comprised the following:

- the analysis of the development processes, identification of points whose support the developers require the most, i.e. rationale, trade-offs;
- the elaboration of general class diagrams for the security targets and protection profiles, together with their detailed sub-diagrams, referencing all main and auxiliary data and activities;

- the elaboration of a set of detailed activity diagrams that describe the IT security development process; on the basis of these activity diagrams a wizard was elaborated to help the developers;
- the preparation of a design library of Common Criteria components and an enhanced set of generics; defining a basic set of relations between them to provide the developers with better support in mapping security items (threats by security objectives, objectives by components, etc.);
- the creation of a simple risk analyzer built in the tool;
- the creation of the self-evaluation module, allowing developers to check their work in the same way as it can be done in a security lab.

6.0 Conclusions

The paper deals with the IT security development according to the Common Criteria and the related standards. It focuses on the security problem definition only, though the developed and implemented methodology encompasses all IT security development stages and the self-evaluation. The prototype of the tool, developed on the basis of this methodology, meets basic needs of IT security developers:

- it issues more precise and coherent specifications,
- it enables to gain better reusability and decision support for developers,
- it improves documentation management capabilities,
- it provides better compliance with information security management standards, e.g.[15],
- it allows to achieve the declared assurance level more efficiently.

After completing the phases of the research, modeling and case study on the existing security targets and protection profiles examples for the certified products or currently elaborated ones, it is possible to start the technology transfer phase. This phase deals with the tool development and refinement. The tool is submitted to a specialist from an IT development and evaluation lab. It is also used for training the developers. The obtained results show that the tool facilitates the IT security development process, however it needs more verification in real conditions. More user feedbacks are necessary as well. The tool prototype is being improved with some useful features developed, such as packages management, composite and complex TOE development, better reporting and better projects management. Additionally, it is worth mentioning that the tool is compliant with the recently issued Common Criteria v. 3.0 and the related standards.

References

- [1] ISO/IEC 15408, Common Criteria for IT security evaluation, Part 1-3.
- [2] ISO/IEC 15446, Guide for the production of protection profiles and security targets, 2004.
- [3] Common Evaluation Methodology for IT Security, Part 1-2.
- [4] J. Jürjens . “Secure Systems Development with UML”. Springer-Verlag, 2004.
- [5] S. Galitzer. “Introducing Engineered Composition (EC): An Approach for Extending the Common Criteria to Better Support Composing Systems”. Published in the WAEP to System Security Design Proc., 2003.
- [6] A. Białas. “IT security development – computer-aided tool supporting design and evaluation”. In: J. Kowalik, J. Górski, A. Sachenko (ed.). *Cyberspace Security and Defense*. Springer 2005, pp. 3-23.
- [7] A. Białas. “IT security modelling”. In: Arabnia, H. R., (Ed), Liwen He & Youngsong Mun (Assoc. Co-eds), *Proc. of the Int. Conference on Security and Management SAM'05*, CSREA Press 2005, pp. 502-505.
- [8] A. Białas. “A UML approach in the ISMS implementation”. In *Proc of the IFIP 11.1 & 11.5 Working Conference*, Springer Verlag 2005.
- [9] A. Białas. “The ISMS Business Environment Elaboration Using a UML Approach”. In: K. Zieliński, T. Szmuc (eds.). “*Software Engineering: Evolution and Emerging Technologies*”. IOS Press, 2005.
- [10] A. Białas. “Using ISMS concept for critical information infrastructure protection”. *Proc. of the Int. Workshop on “Complex Network and Infrastructure Protection – CNIP'06”*, Rome, March 28-29, 2006.
- [11] C. Lavatelli. “A formal framework for high level security CC evaluations”, *e-Smart Conference*, 2004.
- [12] TL FIT, <http://trusted-logic.fr>
- [13] BSI-DSZ-CC-0153. “First evaluation of Philips Secure8-bit Smart Card Controller”. Philips, 1999.
- [14] SecCert, <http://www.cbst.iss.pl>
- [15] BS-7799-2:2002 Information security management systems – Specification with guidance for use, British Standard Institution.