Integrating Security in the Development Process with UML

Folker den Braber
SINTEF ICT, Norway

Mass Soldal Lund
SINTEF ICT, Norway

Ketil Stølen
SINTEF ICT, Norway

Fredrik Vraalsen
SINTEF ICT, Norway

INTRODUCTION

Today, most business processes and communications as well as a lot of everyday life situations involve IT technology. Apart from requirements on functionality, this development of IT systems has increased the need for security. Security issues are reaching the main headlines on a regular basis. Virus, worms, *mis*configuration and program bugs are common problems in a world where new releases and updates are almost as frequently announced as spam e-mail pop-ups in our inboxes.

Having a closer look at the world of IT technology through security-colored glasses, we observe that security is often one step behind functionality. Security issues are mainly addressed as a reaction to existing problems. Like firemen and emergency units security measures do not come into action before it is too late.

The concept of IT security contains a lot of different aspects. One of the IT security aspects is security risk analysis, in the sequel referred to as security analysis. Security analysis is an inevitable and crucial activity for every system developer, system user or system owner, in order to get control over and knowledge about the security level of the actual system.

Security analyses are costly and time consuming and cannot be carried out from scratch every time a system is updated or modified. This motivates the need for specific methodology addressing the integration of security analysis and system development, providing access to, storage of, and maintenance of analysis results.

CORAS (2000) provides such a methodology in the form of so-called "model based security analysis". The CORAS methodology combines traditional risk analysis techniques like HazOp (Redmill, Chudleigh, & Catmur, 1999), FTA (IEC 1025, 1990) and FMEA (Bouti & Ait Kadi, 1994) with system development techniques like UML (OMG, 2003b) and UP (Jacobson, Rumbaugh, & Booch, 1999). It builds on international standards for risk management: the Australian/New Zealand AS/NZS 4360 (1999), "Risk Management"; the ISO/IEC 17799 (2000), "Code of Practice for Information Security Management"; the ISO/IEC 13335 (2001), "Guidelines for the management of IT Security"; and system documentation in the form of the Reference Model for Open Distributed Processing (RM-ODP) (ISO/IEC 10746, 1995).

BACKGROUND

The CORAS methodology incorporates a documentation framework, a number of closely integrated security analysis techniques, and a risk management process based upon widely accepted standards. It gives detailed recommendations for the use of modeling with UML and similar languages in conjunction with security analysis in the form of guidelines and specified diagrams. Security analysis requires a firm, but nevertheless easily understandable, basis for communication between different groups of stakeholders. Graphical, object-oriented modeling techniques have proven well suited in this respect for requirements capture and analysis. We claim they are as equally suited as part of a language for communication in the case of security analysis. Class diagrams, use case diagrams, sequence diagrams, activity diagrams, dataflow diagrams, and state diagrams represent mature paradigms used daily in the IT industry throughout the world. They are supported by a wide set of sophisticated case tools, are to a large extent complementary, and together support all stages of system development.

The CORAS methodology may be separated into three different components: tools, processes and languages. This is shown in Figure 1. The language part defines

common languages to support the methodology. The process part includes instructions for how to "execute" the methodology, that is, descriptions of what should be done, and how and when it should be done. The CORAS methodology for model based security analysis (MBSA) integrates aspects from partly complementary risk analysis methods and state-of-the-art modeling methodology. During execution of the methodology, one may require the use of different tools, which are specified under the tool part. The CORAS methodology includes a computerized platform that can be integrated with third party modeling tools and risk analysis tools. It also includes two languages: a UML-based specification language, the UML profile for security analysis (OMG, 2003c), targeting security risk analysis, and an XML markup for exchange of risk analysis data (World Wide Web Consortium, 2000).

In the section called *Model-Based Security Analysis*, the core of model-based security analysis, based on integration of risk management and system development, is explained.

Related Approaches

The CORAS methodology, with its unique approach to security analysis from a modeling point of view, addresses a problem area in which also other methods and technologies exist. Some important ones are mentioned here.

The Common Criteria (CC)

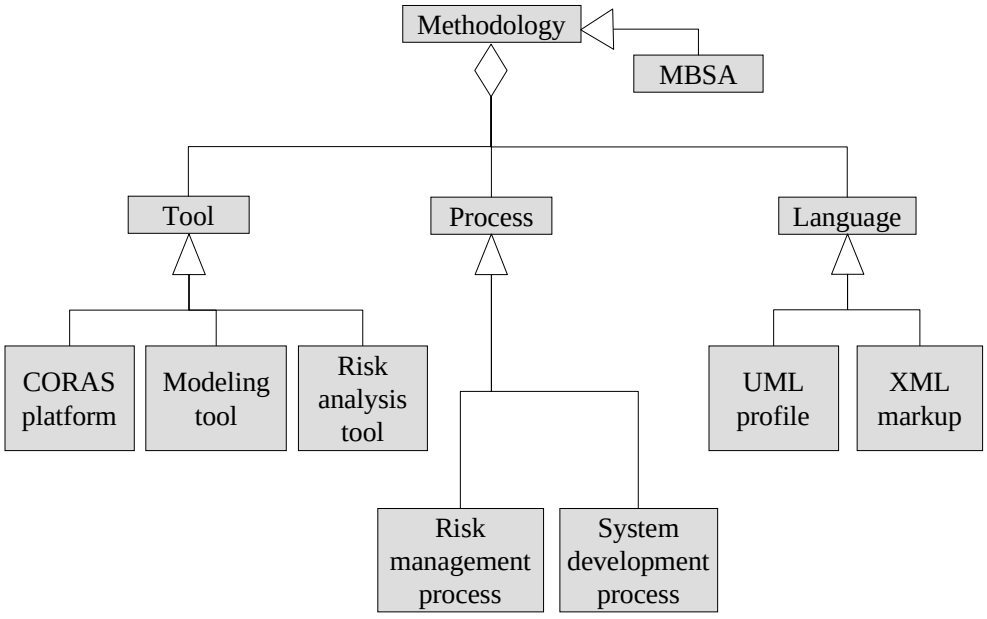
Since 1990, work has been going on to align and develop existing national and international schemes into one, mutually accepted framework, for testing IT security functionality. The Common Criteria (CC) (CCO, 2002) represents the outcome of this work. The Common Criteria project harmonizes the European "Information Technology Security Evaluation Criteria (ITSEC)" (Communications-Electronics Security Group, 2002), the "Canadian Trusted Computer Product Evaluation Criteria (CTCPEC)", and the American "Trusted Computer System Evaluation Criteria (TCSEC)" and "Federal Criteria (FC)". Increasingly, it is replacing national and regional criteria with a worldwide set accepted by the International Standards Organization (ISO15408) (ISO/IEC, 1999).

The CC and CORAS are orthogonal approaches. The CC provides a common set of requirements for the security functions of IT products and systems as well as a common set of requirements for assurance measures that are applied to the IT functions of IT products and systems during a security evaluation. CORAS provides concrete methodology for model based security analysis.

Surety Analysis (SA)

Surety Analysis (SA), developed in Sandia National Laboratories (2003), is a methodology based on the creation of

Figure 1. Structure of the CORAS methodology



an explicit model that covers several aspects of the system's behavior. The modeling framework in SA is proprietary, whereas in CORAS the standardized RM-ODP is used as a common basis. SA supports modeling by means of basic techniques such as interaction and state and dataflow diagrams. CORAS uses the full descriptive power of UML/OCL (Object Constraint Language) (OMG, 2003a) enhanced with aspects of other modeling paradigms specific to security modeling.

COBIT

The control objectives for information and related technology (COBIT) (2003) addresses the management of IT. COBIT and CORAS are orthogonal approaches. COBIT focuses on control objectives defined in a process-oriented manner following the principles of business reengineering.

CRAMM

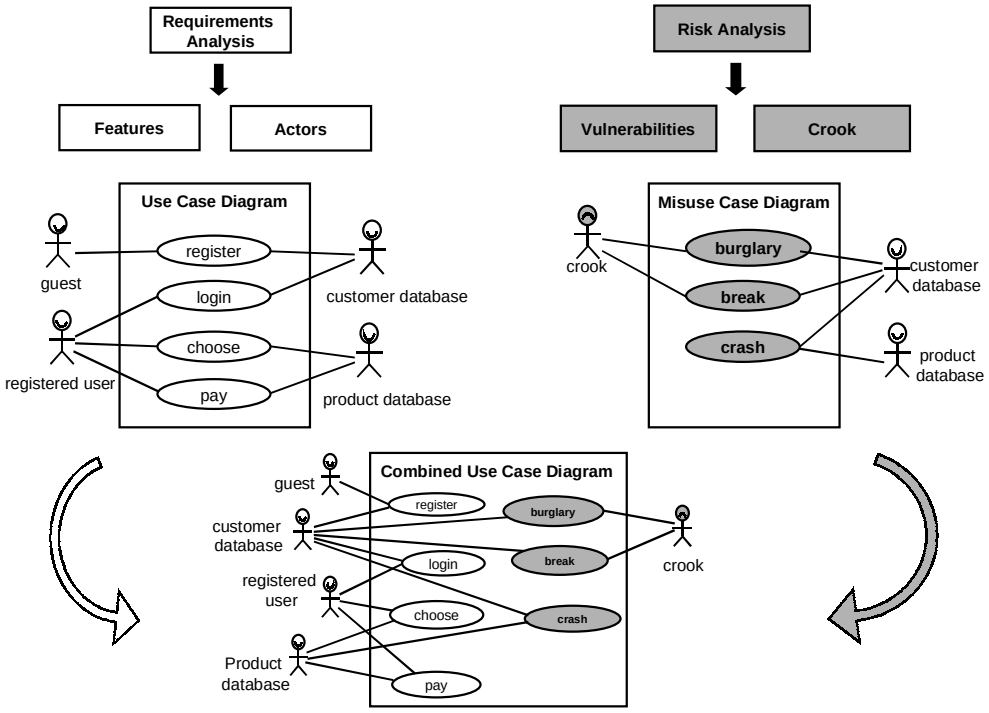
CCTA security analysis and management methodology (CRAMM) (Barber & Davey, 1992) was developed by the British Government's Central Computer and Telecommunications Agency (CCTA) with the aim of providing a structured and consistent approach to computer security management for all systems. The UK National Health Service considers CRAMM to be the standard for security analysis of information systems within health care establishments. CRAMM has been an important source of inspiration for CORAS, and aspects of CRAMM have been incorporated in CORAS. Contrary to CRAMM, CORAS provides a security analysis process in which state-of-the-art modeling methodology in the form of UML is tightly integrated.

MODEL-BASED SECURITY ANALYSIS

As mentioned before, "model-based security analysis" is a core concept in the CORAS methodology. This section explains what "model-based security analysis" is, and gives a glimpse on how it is to be applied.

The basis of model-based security analysis is about identifying negative functionality and making this *mis*functionality (Sindre & Opdahl, 2000) visible and known in order to protect the system. This misfunctionality can be everything from vulnerabilities that open for hacker attacks to bad user interface design increasing the chance for crucial mistakes. The idea is that the well-accepted techniques for specification and design of functionality also suit to model misfunctionality. This is illustrated by Figure 2.

Figure 2. Model-based security analysis



The requirements analysis forms the basis for the traditional kind of modeling (Cockburn, 1997) with focus on desired behavior. Security analysis has a similar role but focuses on unwanted behavior. Every designed system carries its own risks, and it is important that they are known. It is therefore not enough to model only the desired behavior but also the unwanted behavior. In addition, the malicious actors, crooks, need to be identified just like the normal actors. Model-based security analysis is about documenting the results of traditional risk analysis techniques in the same way as we are used to doing for system requirements. The design process needs to take into account both wanted and unwanted behavior and designed actors and malicious actors. As shown here, UML like graphical techniques can be used for both aspects providing the complete documentation of the system design from both good and bad angles.

UML Profile

The CORAS methodology defines its own UML stereotypes and methods for describing UML models related to security analysis. These specific security related UML aspects are caught in the CORAS UML profile (OMG, 2003c). This UML profile for security analysis, introduces a meta model that defines an abstract language for supporting model-based security analysis. Furthermore, the profile provides a mapping of classes in the meta model to UML modeling elements by defining so-called stereotypes, and introduces special symbols (icons) for representing the stereotypes in UML diagrams.

The motivation for the profile is the practical use of UML to support security management in general, and security analysis, in particular. In the model-based security analysis methodology of CORAS, UML models are used for three different purposes:

- Target of evaluation " Describing the target of evaluation at the right level of abstraction.
- Communication " Facilitating communication and interaction between different groups of stakeholders involved in a security assessment.
- Documentation " Documenting security assessment results and the assumptions on which these results depend to support reuse and maintenance.

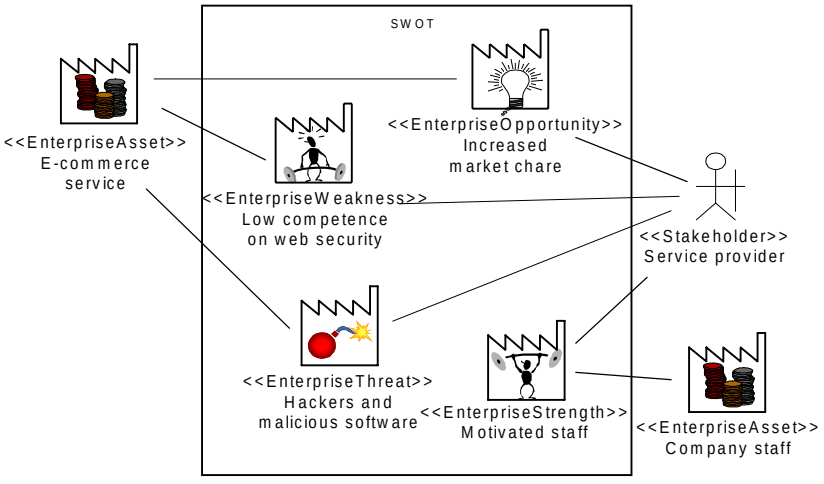
The target of evaluation in a security analysis is a system or part of a system to be assessed, but also the context and assumptions under which the system is assessed. The CORAS profile supports describing the target of evaluation by allowing explicit modeling of the context of assessments, like stakeholders and assets. An example of this is shown in the SWOT diagram in Figure 3, where the assets and stakeholders are directly related to main characteristics of the system to be assessed.

The effectiveness and success of security analysis depends on the extent to which the involved stakeholders and analysts understand and is understood by each other. Users, system developers, decision makers and system managers are different examples of such stakeholders. These will have different backgrounds and competencies and misunderstandings are likely to occur. The CORAS profile supports communication by introducing easy to understand icons representing the various concepts of the ontology, as shown in the SWOT diagram previously shown.

Imagine the following security analysis results generated by a security analysis session as defined in the CORAS methodology.

"Regarding the availability of service as an important asset of the system, it was identified that some persons with the right knowledge and the wrong intentions could threaten this availability of service by flooding the

Figure 3. SWOT (Strength-Weakness-Opportunity-Threat) diagram



system. The indicated flooding threat might generate a Denial-of-Service attack which directly affects the availability of service. One way of securing the system against this risk is to apply authentication on requests."

The same results represented in UML using the CORAS profile are shown in Figure 4. Clearly, this representation is easier to communicate and leaves less room for misunderstanding. It also shows how the CORAS methodology allows for specifying discovered risks and treatments related to threats. One might argue that the result in text is just as clear, but this will typically only be the case for relatively simple cases. The CORAS methodology advices to use the given diagrams and modeling techniques, but does not restrict one from using normal text in addition.

In CORAS, security analysis results are documented by the means of elements (diagrams, tables, etc.). Each of these elements belongs to an activity of the risk management process. The CORAS profile supports documentation by introducing types of diagrams supporting various activities of the risk management process being founded in a meta model conforming to the underlying data structure.

The diagrams mentioned here, become part of the system documentation and are also stored like that. This documentation process is supported by the CORAS platform, a tool containing both guidance and storage functionality. Recent security analysis results are stored while the process is supported by results from past security analysis experiences. This opens for running the CORAS methodology parallel to the traditional system development process in addition to being able to follow the maintenance and upgrading phases by similarly maintaining and upgrading the security analysis results.

FUTURE TRENDS

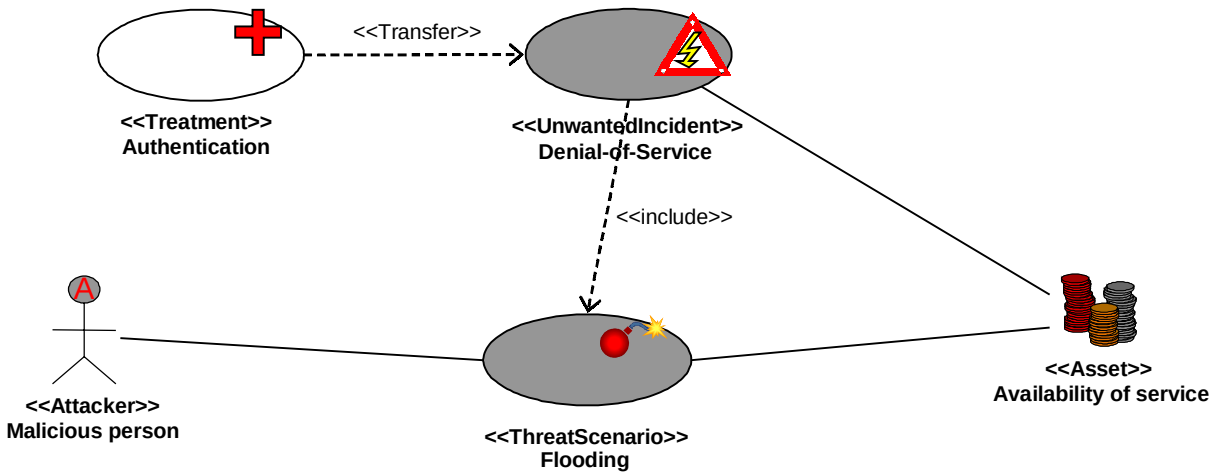
A main future trend regarding security analysis is the integration of security in the development process of IT systems. Developers will have to start to think security at the very start of the development process. Today, security and dealing with risks do not always get the attention they deserve: It is often forgotten or simply ignored. Apart from door locks and bank safes, most people let security "live its own life" and often just hope for the best. This attitude is probably necessary in our everyday life: There are so many things that can go wrong, and we cannot worry about all of them. When it comes to IT systems, and especially security critical IT systems, such an attitude could be fatal. We, therefore, need to break the habit of living on hope and turn it into informed, calculated, and constructive actions. To be able to do this, methods and tools are needed to guide us in our security analyses.

CONCLUSIONS

The CORAS methodology and the corresponding developed CORAS platform help developers in keeping focus on security and provide guidelines on how security analysis can become a natural part of the system development process. The combination of security analysis with system development methodologies like the Unified Process makes security related adjustments possible during all phases of the development process.

A security analysis identifies and documents the undesirable system behavior. The CORAS UML profile

Figure 4. Threat diagram



employs the same kind of modeling techniques to document the undesirable behavior that leading system development methodologies employ to capture requirements to the desirable system behavior. For example, as pointed out previously, in the same way as use cases are employed to capture desired functionality, misuse cases may be employed to capture threat scenarios. The fact that the CORAS methodology employs techniques well known to system engineers simplifies security analysis during system development.

The experiences from applying the CORAS methodology in major field trials within e-commerce and telemedicine are promising. The CORAS project involved seven major field trials, each of which made use of the earlier versions of the CORAS methodology and tool to analyze the security of an already existing system or application. The field trials were all of industrial size. Two field trials within telemedicine are documented in Stathiakis et al. (2003) and Stamatiou et al. (2003), while Dimitrakos et al. (2002) and Raptis, Dimitrakos, Gran and Stølen (2002) document field trials within e-commerce. More documentation on the CORAS methodology can be found on the CORAS Web site (CORAS, 2000).

The research on which this article reports has partly been funded by the Research Council of Norway projects COBRA (152209/431) and SECURIS (152839/220) and partly by the 5th Framework EU project CORAS (IST-2000-25031). The CORAS consortium consisted of 11 partners from four countries: CTI (Greece), FORTH (Greece), IFE (Norway), Intracom (Greece), NCT (Norway), NR (Norway), QMUL (UK), RAL (UK), SINTEF (Norway), Solinet (Germany) and Telenor (Norway). The results reported in this article have benefited from the joint efforts of the CORAS consortium.

REFERENCES

Australian/New Zealand Standard AS/NZS 4360 (1999). Risk management.

Barber, B., & Davey, J. (1992). The use of the CCTA risk analysis and management methodology CRAMM. *Proc. MEDINFO92*, North Holland (pp. 1589-1593).

Bouti, A., & Ait Kadi, D. (1994). A state-of-the-art review of FMEA/FMECA. *International Journal of Reliability, Quality and Safety Engineering, 1,* 515-543.

Cockburn, A. (1997). Structuring use cases with goals. *Journal of Object-oriented Programming*, Sep/Oct, 35-40, Nov/Dec, 56-62.

Common Criteria Organization. (2002). Common criteria for information technology security evaluation. Retrieved in 2003 from the World Wide Web at *http://www.commoncriteria.org*

Communications-Electronics Security Group. (2002). Information security evaluation criteria. Retrieved in 2003 from the World Wide Web at http://www.iwar.org.uk/cip/resources/uk/

Control objectives for information and related technology (COBIT). Retrieved in 2003 from the World Wide Web at *http://www.isaca.org/*

CORAS (2003). A platform for risk analysis of security critical systems. IST-2000-25031. Retrieved in 2003 from the World Wide Web at *http://coras.sourceforge.net/*

Dimitrakos, T., Ritchie, B., Raptis, D., Aagedal, J.Ø., den Braber, F., Stølen, K., & Houmb, S.H. (2002). Integrating model-based security risk management into e-business systems development " The CORAS approach. In *Proc. 2nd IFIP Conference on E-Commerce, E-Business, E-Government (I3E 2003)*, (pp.159-175). Kluwer.

IEC 1025. (1990). Fault tree analysis (FTA).

ISO/IEC (1999). Information technology – Security techniques – Evaluation Criteria for IT Security ISO/IEC, 15408-1.

ISO/IEC 10746. (1995). Basic reference model of open distributed processing.

ISO/IEC 17799. (2000). Information technology – Code of practice for information security management.

ISO/IEC TR 13335. (2001). Information technology – Guidelines for the management of IT security.

Jacobson, I., Rumbaugh, J., & Booch, G. (1999). *The unified software development process*. Object Technology Series. Addison-Wesley.

OMG. (2003a). Object constraint language specification. Part of the UML specification. OMG document number: ad/03-01-07.

OMG. (2003b). Unified modeling language specification. Version 2.0. OMG document number: ptc/03-09-15.

OMG. (2003c). UML for QoS & fault tolerance. OMG document number: realtime/03-08-06.

Raptis, D., Dimitrakos, T., Gran, B.A., & Stølen, K. (2002). The CORAS approach for model-based risk analysis applied to the e-commerce domain. In *Proc. Communication and Multimedia Security (CMS 2002)*, (pp.169-181). Kluwer.

Redmill, F., Chudleigh, M., & Catmur, J. (1999). *Hazop and software Hazop*. Wiley.

Sandia National Laboratories, Surety Analysis. Retrieved in 2003 from the World Wide Web at *http://www.sandia.gov*

Sindre, G., & Opdahl, A.L. (2000). Eliciting security requirements by misuse cases. In *Proc. TOOLS_PACIFIC 2000,* (pp.120-131). IEEE Computer Society Press.

Stamatiou, Y., Skipenes, E., Henriksen, E., Stathiakis, N., Sikianakis, A., Charalambous, E., Antonakis, N., Stølen, K., den Braber, F., Soldal Lund, M., Papadaki, K., & Valvis, G. (2003). The CORAS approach for model-based risk management applied to a telemedicine service. In *Proc. Medical Informatics Europe (MIE 2003),* (pp.206-211). IOS Press.

Stathiakis, N., Chronaki, C., Skipenes, E., Henriksen, E., Charalambous, E., Sykianakis, A., Vrouchos, G., Antonakis, N., Tsiknakis, M., & Orphanoudakis, S. (2003). Risk assessment of a cardiology eHealth service in HYGEIAnet. To appear in *Proc. Computers in Cardiology (CIC 2003).*

World Wide Web Consortium. (2000, October 6). Extensible Markup Language (XML) v1.0, W3C recommendation (2nd ed.).

KEY TERMS

Risk: The chance of something happening that will have an impact upon objectives. It is measured in terms of consequence and likelihood.

Risk Analysis: A systematic use of available information to determine how often specified events may occur and the magnitude of their consequences.

Risk Assessment: The overall process of risk analysis and risk evaluation.

Risk Evaluation: The process used to determine risk management priorities by comparing the level of risk against predetermined standards, target risk levels or other criteria.

Risk Treatment: Selection and implementation of appropriate options for dealing with risk.

Security Analysis: Thorough analysis of a system in order to get a complete picture of its security level. Vulnerabilities, risks, treatments and their costs are identified.

Stakeholders: Those people and organizations who may affect, be affected by, or perceive themselves to be affected by, a decision or activity.

SWOT Analysis: Strength, Weakness, Opportunity and Effect Analysis.

Target of Evaluation (TOE): An IT product or system and its associated administrator and user guidance documentation that is the subject of an evaluation.

Virus: Program that can infect other programs by modifying them; the modification includes a copy of the virus program, which can then go on to infect other programs.