INTERNATIONAL JOURNAL OF COMMUNICATION SYSTEMS
Int. J. Commun. Syst. 2005; **18**:763–793
Published online 18 July 2005 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/dac.729

Computer and network security risk management: Theory, challenges, and countermeasures

Mohamed Hamdi[‡] and Noureddine Boudriga[*,†]

School of Communication Engineering, Computer Network and Security (CNAS) Laboratory, University 7th of November of Carthage, Tunisia

SUMMARY

The need to secure information systems and networked infrastructures is now commonplace in most enterprises. The use of novel communication technologies has became a crucial factor that can considerably improve or affect productivity. This is essentially due to the importance of the information transmitted across communication networks and stored in servers. As a consequence, strong ties are being built between security and the enterprise business activity. Risk management, which is the discipline that deals with this aspect, integrates a litany of architectures, techniques, and models that are described in this paper. A global view is proposed to the reader through a presentation of the research activity that has been directed towards this field. Copyright © 2005 John Wiley & Sons, Ltd.

KEY WORDS: risk management; risk analysis; security countermeasures

1. INTRODUCTION

Since the beginning of modern network technology, especially the Internet, enterprises have renovated their communication infrastructure in order to take benefit of this factor. Advertising, electronic banking, electronic mail, to name just few, have contributed substantially to the success of many companies. Nevertheless, the concurrent expansion of digital attacks has made stakeholders lose confidence in Information Technology (IT). In fact, security incidents that occurred recently (see Reference [1] for a detailed analysis) have emphasized two main facts: (1) computer network attacks can lead to huge amount of loss, (2) many of the attacked enterprises were possessing a security infrastructure at the moment the incident occurred. While the first statement is a natural consequence of IT involvement into the production process, the second point is very intriguing as it may suggest that security equipments have no efficiency in thwarting

*Correspondence to: Noureddine Boudriga, SUPCOMP, Ecole Supérieure des Communications de Tunis Cité Technologique des Communications—2083 El Ghazala, Ariana, Tunisia.
†E-mail: nab@supcom.rnu.tn
‡E-mail: mmh@certification.tn

Contract/grant sponsor: Publishing Arts Research Council; contract/grant number: 98-1846389

Received 9 October 2004
Revised 1 February 2005
Accepted 19 February 2005

sophisticated malicious acts. Such assertion is obviously wrong as the real weakness resides at the security planning level. Henceforth, structured methodologies that identify, analyse, and mitigate computer security risks have been evolved to help enterprises integrate security in their strategic plans. In fact, experience has shown that the appropriate application of a risk management (RM) approach would considerably limit the aforementioned losses.

A *lato sensu* definition of RM includes 'the ideas, models, abstractions, methods, and techniques that are employed to control risk'. Typically, a RM approach is intended to (1) identify the risks threatening the target information system, (2) select the appropriate security solutions to reduce this risk to an acceptable level, and (3) continue to control the system in order to preserve its security level. More precisely, RM relies heavily on risk analysis, which in turn consists in studying the state of a system in order to select the appropriate actions that should be done to enhance its defence capabilities. This improves consistently the efficiency of security solutions as they are selected according to a defined strategy and not on the basis of human intuition. When an enterprise applies a RM approach, security is integrated *ipso facto* in the managerial process.

In spite of its importance at the enterprise level, very few research studies have discussed RM in substance. Existing works mainly focused on several ramifications of RM without considering the whole problem. An abundant research activity has been directed towards vulnerability analysis, risk analysis, and network state monitoring while RM approaches have been rarely cited in the literature. This paper is an attempt to bridge the gap between enterprise and research RM views. A review of these disparate research activities is performed in order to present a concealed view to the reader. Of course, as some the considered issues may be used in other fields, their discussion will be restrained to the RM frame.

The remainder of the paper is organized as follows. The second section sets the requirements that must be respected by a risk management framework. Section 3 reviews the most renowned risk management approaches. Some shortcuts related to these methodologies are presented in Section 4. Section 5 discusses aspects related to vulnerability, attack and decision libraries. Section 6 addresses the risk analysis problem and describes some related techniques. Section 5 shows how preventive and reactive risk assessment should be conducted. Section 7 examines various system monitoring issues. Section 8 presents a generic view of the risk management process as well as a relational model for security attacks and responses. Finally, Section 9 concludes the paper.

2. RISK MANAGEMENT REQUIREMENTS

Since the development of the first risk management method [2], many attempts to develop new approaches and to strengthen the existing ones have arisen. Nonetheless, the major factor that limits the application of those methods in the real world is the lack of suitable software tools. Effectively, risk analysis is so complicated that performing it manually would be quasi unfeasible. A glance at the existing automated risk management tools shows that they do not keep up with the evolution of the related theoretical models. More importantly, all of these software are incomplete, in the sense they do not allow their users to perform a complete risk management cycle. This section investigates the characteristics that a RM framework should possess. The most important issues, in our sense, that should be covered at this level are listed in

the following. Additional obstacles to the widespread application of risk management may stem from several intrinsic weaknesses of the existing approaches will be underlined in Section 4.

1. *Cost estimation*: In view of the fact that security planning itself has a cost to the enterprise that conducts it, the amount of effort related to a risk management project should be conveniently predicted. Naturally, if the cost of security planning overtake the allowed budget, risk management operations have no reason to start. The absence of techniques estimating the required effort with regard to the characteristics of the analysed system is one of the key reasons that alter the confidence of IT users, especially high-level managers, in risk management.
2. *Attack modelling*: Representing attacks against computer networks is among the most challenging tasks the risk analyst should do. This stems from the intrinsic complexity of these attacks, which can be reduced to the following points:
   (a) *Primo*, as network attacks are conducted by humans, a corresponding predictive model would be hard to evolve. On the opposite of various events, human thoughts cannot be easily translated into a formal representation. Most of the existing methods assume the existence of a static attack database without addressing this issue.
   (b) *Secundo*, attacks are seldom performed on a single step. Attackers often follow a structured methodology to achieve malicious acts. Hence, the risk analyst should establish various kinds of links between elementary threats (e.g. causality, time precedence). The major shortcoming of the traditional approaches is that they do not integrate security countermeasures and enterprise resources as a part of the expert knowledge related to attack scenarios.
3. *Control selection*: In traditional risk management approaches, it is customary to map statically security decisions to digital attacks. In other terms, a set of countermeasures correspond to each attack, in the sense that they mitigate it. We found that this static mapping does not satisfy totally the needs of the risk analyst because it can be the source of many problems, especially when evaluating the efficiency of a decision. Consequently, several conditions should be integrated within the security solution selection module to make the choice of the optimal countermeasure alternative depend not only on the possible attacks but also on the environment of the studied system (e.g. network topology). As a result of this reasoning, the security solutions that are optimal for an enterprise are not necessarily the same as for another enterprise even if both of them face the same threats.
4. *Monitoring*: This task is often reduced to a continuous control of the system state. However, given that it belongs to a risk management cycle, it seems inadvisable to confine monitoring to its pure security facets. It should be fully integrated in the business activity of the enterprise. Incident response, for instance, should be enriched by a set of models allowing the accurate measurement of an incident impact, and the selection of the appropriate reactions.

When applied inside an organization, RM should be structured into a complete framework including an approach, a set of techniques, and a set of software tools. Obviously, all these components should fully interoperate to guarantee the efficiency of the process. Throughout the following sections, we show that this point constitute one of the most important weaknesses of the existing frameworks.

3. RISK MANAGEMENT METHODS

3.1. Historical overview

The rapid growth of attacks against computer network systems and the high cost of the available security countermeasures have favoured the development of structured high-level methodologies aiming at evaluating the security state of such systems and selecting the most convenient defence measures. Therefore, integrating security issues into the business activity of enterprises has been an important challenge since several decades.

One of the earliest risk management approaches has been proposed on 1979 by Campbell [2] who developed a structured methodology based on a set of concepts that subsisted in the later approaches such as vulnerability analysis, threat analysis, risk analysis, and control implementation. Summers [3] proposed a similar method based on four steps:

1. Asset analysis: Identifying assets and assigning values to them,
2. Threat and vulnerability identification,
3. Annual Loss Expectancy[§] (ALE) calculation for each threat,
4. Safeguard selection.

Effectively, these approaches came to enhance earlier efforts within this field. Since the mid-1970s, the National Institute of Standards and Technology (NIST, previously the National Bureau of Standards) began to address some aspects of risk management, especially risk assessment. Two standards, FIPSPUB-31 Automated Data Processing Physical Security and Risk Management, and FIPSPUB-31 Guidelines for Automated Data Processing Risk Assessment were, respectively, published in 1974 and 1979. This pioneering work proposed some mechanisms to evaluate security risks quantitatively. Throughout the difficulties that were faced when applying these methods, it has been realized that quantitative risk assessment is very hard to conduct. This was the result of two major factors. First, as security models were, at this moment, still in infancy, building a representation of the analysed environment was a very complex task. The huge amount of data collected from various components of the information systems could not be treated accurately. Second, the lack of automated tools supporting the risk management activity rendered its manual execution more difficult. Consequently, easier subjective methods, called quantitative approaches, have been introduced. The qualitative valuation of the basic risk analysis variables (e.g. asset importance, threat probability, threat impact) consists in a simple scaling. For instance, the probability of occurrence for a specific threat can be either 'high', 'medium', or 'low'. Obviously, the collection of knowledge becomes much more easier than in the quantitative case. In fact, instead of relying directly on digital security data such as audit log records, interviews and questionnaires are used in the qualitative case.

In addition, RM methodologies can be divided into two categories: bottom–up approaches and top–down approaches. The bottom–up approach consists in selecting *a priori* the residual risk, i.e. the degree of protection of the system, and implementing the countermeasures that allow to reach it. Top–down risk estimation defines scheduled tasks that are intended to reduce the identified threats. Examples of these tasks include, but

[§]The ALE is a concept that allows to assess and rank security threats. It will be discussed in Section 7.2.

are not limited to, risk identification, risk mitigation, system state monitoring, and risk assessment.

Another pertinent aspect that has arisen due to the need of cost-effectiveness is the development of automated software tools that support the risk management activity. In this paper, we do not focus on such tools but we rather discuss the influence of the automation requirement on the constituency of several methodologies. In other terms, it should be underlined that the ease of implementation is among the most relevant factors to assess a RM approach.

Recently, more sophisticated methods have been evolved. Even though the security objectives slightly differ from an approach to another, a lot of interesting subtleties have been introduced at different levels (e.g. architectural modelling, implementation design, decision-making algorithms). More details about these considerations are given in the sequel.

3.2. The OCTAVE method

Operationally critical threat, asset, and vulnerability evaluation (OCTAVE) [4] has been evolved jointly by the Carnegie Mellon University and the Software Engineering Institute. It is a self-directed method relying on a interdisciplinary team. The OCTAVE process has been structured with respect to the various categories of results that could be reached by this team. Mainly, three types of outputs have been considered: organizational data, technological data, and risk analysis and mitigation data.

Unlike the typical technology-focused assessment, which is targeted at technological risk and focused on tactical issues, OCTAVE is targeted at organizational risk and focused on strategic, practice-related issues. The OCTAVE approach is driven by two of the three aspects: operational risk and security practices. Technology is examined only in relation to security practices, enabling an organization to refine the view of its current security practices. OCTAVE is organized around these three basic aspects (illustrated in Figure 1), enabling organizational personnel to assemble a comprehensive picture of the organizations information security needs. The phases are

- *Phase* 1: *Build asset-based threat profiles*: This is an organizational evaluation. The analysis team determines what is important to the organization (information-related assets) and what is currently being done to protect those assets. The team then selects those assets that are most important to the organization (critical assets) and describes security requirements for each critical asset. Finally, it identifies threats to each critical asset, creating a threat profile for that asset. It is worth mentioning that threat profiles are derived from event-trees.
- *Phase* 2: *Identify infrastructure vulnerabilities*: This is an evaluation of the information infrastructure. The analysis team examines network access paths, identifying classes of information technology components related to each critical asset. The team then determines the extent to which each class of component is resistant to network attacks.
- *Phase* 3: *Develop security strategy and plans*: During this part of the evaluation, the analysis team identifies risks to the organizations critical assets and decides what to do about them. The team creates a protection strategy for the organization and mitigation plans to address the risks to the critical assets, based upon an analysis of the information gathered.

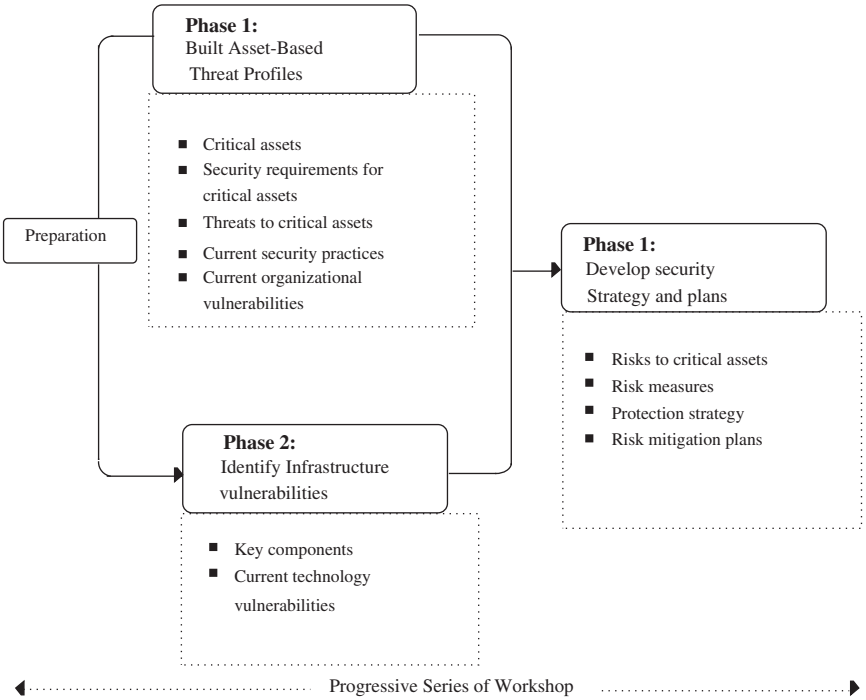

Figure 1. The OCTAVE method [4].

3.3. The CORAS framework

CORAS [5] is a EU-funded RM project that has been conducted between 2001 and 2003. A wide spectrum of risk-related issues have been addressed in the frame of this research. This has resulted, among others, in the definition of a global model-based approach for security risks relying on several aspects that have been borrowed from existing risk assessment techniques. One of the key components of the CORAS framework is an automated tool supporting the methodology. It integrates a UML-based specification language that can be used for three major purposes consisting in (1) representing accurately the system state (from a security point of view) and the interaction between the various entities, (2) standardizing the communication between the risk assessment team members through the use of a uniform language, and (3) documenting the risk management activities. Moreover, this method highlights the library concept

The processes of the CORAS RM method are illustrated by Figure 2. They are briefly discussed in the following:

- Context identification aims at determining the security-critical assets as well as the related security requirements. CORAS relies on the initial phase of CRAMM to address this need. A questionnaire is submitted to system users to determine the important 'data groups' (i.e. the relevant assets that should be analysed).
- Risk identification consists in identifying the weaknesses of the crucial components and the potential threats that may harm them. Different methods are used at this level:

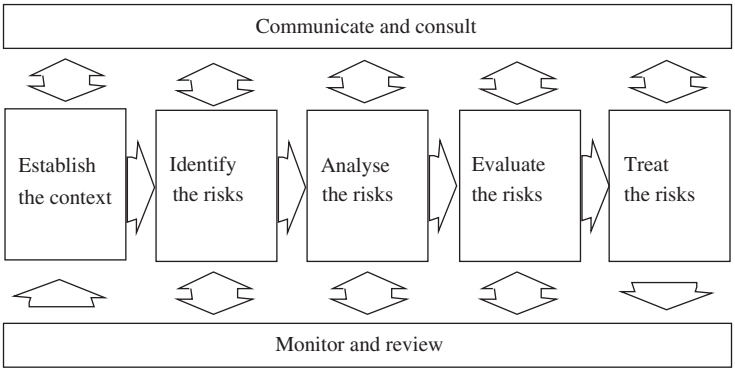

Figure 2. The CORAS RM method [5].

  - Fault-tree analysis (FTA) identifies causes for an unwanted outcome event (top–down approach)
  - Failure mode and effect analysis (FMECA) focuses on single failures of components (bottom–up approach)
  - The UK Government's risk analysis and management method (CRAMM) addresses threats/vulnerabilities of an asset by means of predefined questionnaires
  - Goal means task analysis (GMTA) identifies tasks and pre-conditions needed to fulfill an identified security goal

- Risk analysis has the objective to investigate possible consequences of unwanted outcomes, as well as the likelihood of their occurrences. FMECA supports well this activity if the system description is sufficiently detailed. Moreover, HozOp can be useful when there is no detailed knowledge about the analysed system. Those methods are complementary in the sense that the former is quantitative while the latter is qualitative. FTA and Markov analysis can be used to support hazard and operability analysis (HazOp) as they, respectively, assign probabilities to the basic events in the fault trees (to compute the ones of the top events) and evaluate the likelihood of sequences.
- Risk evaluation ranks the identified risk events on the basis of the likelihood of occurrence and on the impact. This process is based on FMECA to handle quantitative parameters and on CRAMM to combine qualitative weights. A cause–effect analysis is also performed to determine relationships between risk events.
- Risk treatment defines the strategy that should be applied to thwart the potential attacks. Multiple options are available at this level such as risk avoidance, reduction of likelihood, reduction of consequences, risk transference, and risk retention.

The execution of CORAS-based RM project can be viewed as an iterative process where the initial input is specified using UML (or similar object-oriented) state diagrams. Therefore, the iterations follow the RM-ODP process viewpoints. It is noteworthy that two processes are run in parallel to the process of Figure 3. 'Monitoring and review' re-evaluates the results of each iteration while 'Communication and consultation' implies a strong cooperation inside the RM team.

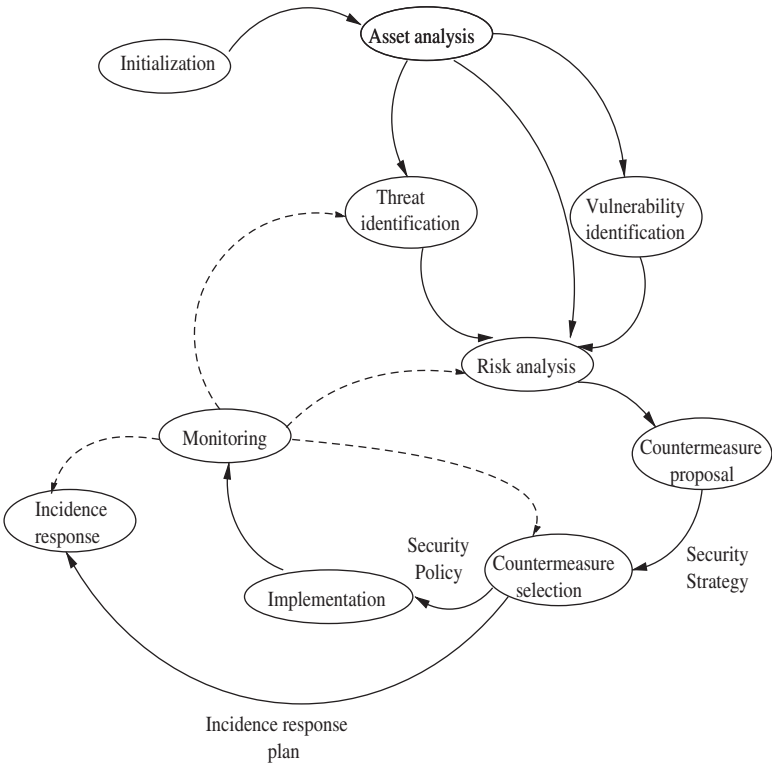

Figure 3. The NetRAM framework for risk management.

3.4. The NetRAM framework

To cover the shortcomings that have been mentioned in the foregoing discussion, a risk management framework, called Network Risk Analysis Method (NetRAM) has been developed by the authors in Reference [6]. A key characteristic of NetRAM is that it is heterogeneous, meaning that it integrates three different components: (1) software tools, (2) architectural design methodologies, and (3) theoretical models.

1. NetRAM is adapted to different modern enterprise structures at different levels. In fact, it is possible to handle various network architectures and topologies. Moreover, NetRAM can be customized so that the business activity of the enterprise is taken into consideration. Therefore, it instills a big importance to the risk management process at the enterprise level as it would appear as a focal activity.
2. NetRAM ensures an optimal handling of the risk management triad: analysis, decision, and response. Many sources provide necessary data and multiple tools can be used to execute various operations. For instance, information contained in intrusion detection systems (IDSs) and automated vulnerability scanners can be imported and analysed in a single uniform environment. Likewise, a group of experts might be involved at some stages of the decision making process. Consequently, NetRAM can be viewed as a collaborative and a

collective framework that requires the contribution of several automated and human components.

3. NetRAM uses appropriate formalisms allowing to handle appropriately the uncertainty introduced by different factors. Both computer-aided risk analysis (CARA) and expert-based decision making are performed through the use of structured methodologies that reduce the error-rate and prevent it from propagating across the risk management steps. Obviously, this approach permits to control efficiently the residual risk and to make the enterprise aware of the actual threats related to its environment.
4. Monitoring the security of the analysed Information Technology (IT) infrastructure is a crucial task. A set of modules ensuring a continuous control of the system state has been integrated with the NetRAM framework. Detecting deviations from the normal behaviour and conducting convenient reactions are the main objectives of these modules.
5. NetRAM is highly adapted to the rapid changes of the IT world that might affect the analysed system. A learning process is thus developed to limit human intervention during the update of several quantitative parameters and semantic links. In fact, automating the learning process makes its output more accurate because the sources from which the necessary data is gathered are tightly related to the system environment. For instance, log files coming from routers, firewalls and IDSs are highly useful at this level. However, human intervention remains necessary at this level to avoid effectively, this process is semi-automatic as human experts must continue to participate to it.

From an architectural point of view, NetRAM can be represented by a ten-process lifecycle as depicted by Figure 4. The interested reader may remark that it consists of two concurrent graphs. The graph designed with solid lines represents the workflow of the risk management activities. The arrows model the precedence relation between the ten processes of NetRAM, which follows the progress of the risk management activities. On the other hand, the second graph, represented by dashed lines, illustrates the retroactivity existing between some processes of NetRAM. It is worth noting that this characteristic is specific to NetRAM. In fact, existing risk assessment methods always ignore the importance of re-executing some risk management activities further to the emergence of some vulnerabilities or attacks threatening the information systems.

A brief description of the constituency of each module is given in the following:

1. *Initialization*: This process aims to prepare the risk management project by providing the managers with the cost of the project and the schedule of the different risk management activities. Moreover, an estimation of the effort required to conduct the risk management project is performed. The underlying activities described in the following are then scheduled according to this estimation.

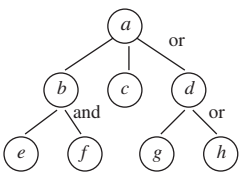

Figure 4. Typical attack-tree.

2. *Asset analysis*: The aim of this process is to gather information about the analysed system. In addition to the valuation of the inventoried resources, some dependency links between them should be established. Such relationships might be needed when building attack scenarios.
3. *Vulnerability identification*: Weaknesses and security breaches of the analysed system are identified at this level. A combination of automated tools (vulnerability scanners) and questionnaires is used to this purpose.
4. *Threat identification*: This process aims to identify the attacks that threaten the analysed system. A list of potential threats is deduced using an exploit relationship with the identified vulnerabilities.
5. *Risk analysis*: The aim of this process is to identify the risks that may threaten the assets of analysed system based on the identified vulnerabilities and threats. These risks are then ranked and prioritized with respect to their influence on the system.
6. *Countermeasure proposal*: A security strategy mitigating the identified risks is proposed at this level. It consists in high-level statements that ensure the respect of the security properties.
7. *Countermeasure selection*: This process aims to define the security policy that will be adapted by the analysed system to mitigate security risks. A set of countermeasures is selected from a decision library according to the strategy established in the former step. A multi-attribute algorithm is set out to cope with this need.
8. *Implementation*: At this level, selected security countermeasures are implemented according to the security policy. A schedule for the deployment of these solutions is performed depending on the available budget and on the total period.
9. *Monitoring*: The aim of this process is to maintain the analysed system in an acceptable security level. A set of relevant security metrics should be continuously measured and addressed to detect security incidents at the right time. Monitoring activity can result in the re-execution of some processes if needed.
10. *Incident response*: This process aims to react to security intrusions according to the incidence response plan. A risk analysis process is conducted to select the most cost-effective reactions and to ensure the continuity of the business-critical processes.

NetRAM has been tested and applied at the National Digital Certification Agency which is the root certification authority in Tunisia. The interested reader can learn more about the advantages in [6].

4. LIMITS OF THE EXISTING METHODOLOGIES

The approaches that have been presented above exhibit several weaknesses that may affect the efficiency of the risk management process. These limits are categorized into architectural limits and technical limits.

4.1. Architectural limits

The following aspects constitute the most important architectural shortcuts of the existing RM methods:

- *There is no separation between preventive and reactive RA*: A glance at Figures 1 and 2 shows that the related methodologies focus on preventive risk analysis and subordinate reactive risk analysis. As it will be explained in Section 6, preventive and reactive risk analysis exhibit many differences. The former needs an *a priori* reasoning about security (before the occurrence of a security incident) while the latter requires *a posteriori* intervention after a security incident has occurred. Moreover, preventive risk assessment relies on probabilities of occurrence of the identified threats. Reactive risk assessment relies on alerts and therefore introduces probabilities which are principally related to the efficiency of the detection mechanism. As a result, it is advisable to give a bigger importance to reactivity in the RM architecture. Unfortunately, existing RM approaches have not addressed this point.
- *Real-time reactivity is quite absent*: When speaking about reactive countermeasures, time is fundamentally important. According to the nature of the attack, security solutions might be implemented immediately (real-time response) or after a collaborative reasoning process involving both stakeholders and security specialists. This differentiation is not made by the methods discussed above. In fact, this depends essentially on how the attack impact varies according to time. In some cases, the decision to block a TCP or USP port must be made in a very short period of time (a few seconds). Other countermeasures may require the approval of high-level management before being implemented, especially when they cause an important loss to the enterprise. This means that a compromise between security solutions cost and benefit should be ensured. With regard to the rapidity of most of the known attacks, a high proportion of this decision-making framework should be automated. Only crucial decisions should be subjected to a human approval.
- *Libraries are not described as they should be*: Putting aside the structure of RM libraries, the security analyst should focus on how to ensure these functions. Due to the large number of vulnerabilities, attacks, alerts (that can be evaluated by thousands), semantic links cannot be built manually (i.e. by the security analyst himself). Suppose that we just want to associate to every attack the vulnerabilities that it exploits. If N_A attacks and N_V vulnerabilities are considered; and giving that the risk analyst has to inspect, for every attack, the whole set of vulnerabilities to select those that can be exploited by the attack of interest, then the total number of semantic links that should be addressed equals $N_A \times N_V$. This is equivalent to 10 000 links if $N_A = N_V = 1000$. Obviously, one may argue that databases provided by official security organisms (e.g. SANS Institute, CERT, NITS) can be used to cope with this problem. Unfortunately, these databases can be used only in certain simple situations. In fact, they are restricted to system weaknesses and do not consider human or documentation-related vulnerabilities. Consequently, one of the key components of a RM methodology is a language which uniformly treats vulnerabilities, attacks, and alerts. Of course, this language should support several automated inference operations such as mechanically determining which vulnerabilities can be exploited by a given threat, which security alerts correspond to a specific attack, or which countermeasures can thwart a harmful action.

Another important lack of the existing RM methods consists in the absence of decision libraries. A careful consideration of these approaches shows that an accurate structure is needed to represent countermeasures. This can substantially facilitate operations such as searching or

sorting. Furthermore, an appropriate syntax should be defined to allow the automated association with attacks, assets, and vulnerabilities.

4.2. Technical limits

Some technical limits have also been noticed:

- *Absence of homogeneity*: One of the most important limitations of OCTAVE and CORAS stems from their lack of uniformity. The techniques that have been proposed to implement RM processes seem disparate and hard to integrate with each other. For instance, combining the myriad of risk analysis techniques proposed within the frame of the CORAS approach requires an important adaptation effort, even though a uniform language (UML) is used to specify security. Naturally, a RM cycle consists of various activities that cannot be described nor implemented through the use of a single technique. Henceforth, what is required is to take into account the effort that should be made to set up a toolbox that adapts and integrates many techniques. In this context, the use of specification tools that provide an abstraction view of RM processes and tasks may be very useful.
- *Lack of powerful techniques*: As it will be underlined in the following sections, many sophisticated tools and theories have been evolved around most of RM tasks, especially vulnerability detection and intrusion detection. Nonetheless, RM methodologies have not kept up with those research advances. Techniques that are proposed by OCTAVE or CORAS are certainly easy enough for being quickly understood and applied but they also present some severe limitations. For instance, HAZOP (used by CORAS to identify security threats) is not compatible with automated vulnerability tools.

5. HANDLING RISK MANAGEMENT LIBRARIES

5.1. The vulnerability library

A careful consideration of networked computer systems shows that three main factors determine whether it is secure or not: (1) the state of hardware and software infrastructure, (2) the behaviour of the system's users, and (3) the correctness of the security documentation (e.g. policies, standards, procedures). This means that a security breach can be related to one or more of those factors. Conceptually, it is obvious that vulnerabilities corresponding to each of those classes have different natures. The hardware and software components of the IT system are by far the most complex entity. Assessing their security should take into account OSs, applications, protocols, and network topologies. Consequently, a huge amount of data must be accurately collected and scanned. Of course, such task cannot be performed manually unless it will last infinitely. Thus, automated tools are necessary to accomplish it. On the other hand, studying the interaction of human subjects with sensitive components involves much less information. Nonetheless, it can be argued that the difficulty of detecting this type of weaknesses stems from the nature of human behaviour, which is hard to model. Besides, the research directed towards this aspect is insignificant with respect to the other categories. Practically, security questionnaires are often used to detect user-related vulnerabilities. Finally, analysing security policies for breaches requires methods and mechanisms allowing to state whether a given policy

fulfills a set of specific requirements. To this end, a policy should be translated to formal expression in order to be validated rigorously.

This diversity of the vulnerability sources heightens the complexity of the vulnerability analysis process even though the existence of a myriad of automated scanning tools has considerably distorted the meaning of this word. Effectively, most of the existing RM methods reduce vulnerability analysis to vulnerability identification and thus subordinate many key issues. For example, OCTAVE defines this activity as 'running vulnerability evaluation tools on selected infrastructure components'. A similar definition has been given by Ozier [7]. 'This task includes the qualitative identification of vulnerabilities that could increase the frequency or impact of threat event(s) affecting the target environment'. It appears from these definitions that the goal of vulnerability analysis has been commonly viewed as ascertaining the existence of a set of vulnerabilities in several components of a given system. In other terms, the risk analyst has to map some elements of the vulnerability library to each asset of the studied system. Nevertheless, this task can be more skillfully applied by considering the following concepts:

1. *Vulnerability detection strategies*: Security specialists should be aware of the various testing methodologies that can be applied to detect vulnerabilities. The choice of a testing strategy depends upon the nature of the weakness and the characteristics of the target system.
2. *Vulnerability identification techniques*: Many vulnerability detection mechanisms are available. Thus, the most convenient ones should be selected according to the need of the risk analyst.
3. *Vulnerability library structure*: The vulnerability library can contain thousands of security weaknesses. Therefore, it must be structured in a manner to support various types of queries.

We take this approach to describe vulnerability analysis as we believe it is more amenable to reason about this task. In the sequel, these ideas are detailed to show how the richness of the vulnerability library enhances the efficiency of the risk management process.

5.1.1. Vulnerability detection strategies. Automated software vulnerability testing strategies break into two categories: black-box testing and white-box testing. The former consists in evaluating the behaviour of the system without having a precise idea about its constituency while the latter presupposes that the source code of the tested entity (e.g. piece of software, OS, communication protocol) is available. Most of the existing vulnerability detection methods belong to the black-box class. Three principal sub-classes can be defined:

1. *Rule-based testing*: The vulnerability detection engine relies on a set of rules that represent known patterns corresponding to security breaches. For instance, a password can be checked to see if it fulfills the requirements given in FIPS Pub 112.
2. *Penetration-based testing*: Intrusion scenarios are actually carried out on the analysed system to assert if the corresponding vulnerabilities exist or not. In the case where only hashed passwords are available (e.g. UNIX based systems), the vulnerability detection procedure should obviously differ from the aforegiven one. Various character combinations have to be generated and their hash be computed sequentially and compared with the original hash. In this case, the robustness of the password is measured according to the period of time necessary to guess it. The longer this period, the stronger is the password.

3. *Inference-based testing*: Many events that do not correspond to security weaknesses may lead, if combined, to an insecure state. Hence, the vulnerability detection engine should take into consideration the relationship between the components of the target environment. Consider the case where the hash of a strong password is put in a file that does not possess a suitable access rights. For example, if a user has a write access to the password file, he can substitute the password of a given account with an arbitrary password that he would use for a more privileged access to the system. Neither the rule-based nor the penetration-based strategies permit to detect such vulnerabilities as the detection engine should perform some inferences on the state of the information system.

The second possibility to detect software security breaches is to perform white-box testing by analysing a specific source code. This analysis can be either static or dynamic. Static security scanning encompasses the cases where the security analyst searches for pre-defined patterns in the scanned code. On the other hand, dynamic testing means that the code should be modified, compiled, and executed to know whether it is vulnerable. This vulnerability scanning strategy (i.e. white-box) is clearly more efficient than black-box testing especially to identify buffer overflow vulnerabilities, which are among the most numerous and harmful. However, they cannot be always applied. They are particularly useful to scan system-specific software (i.e. software that is used within the frame of a particular application) or open-source applications that can be freely enriched and customized by users. In addition, white-box testing can be used to analyse the security of software during its development.

As a result, the risk analyst should identify the tests to run in order to build an accurate view of the security level of the analysed infrastructure. Often, the testing strategy is hybrid in the sense that it is constituted by several black-box techniques and some white-box methods.

5.1.2. Vulnerability identification techniques. Many static white-box testing techniques have been evolved and discussed. The more basic one is related to the use of the `gets()` function within C programs. This function normally captures strings from the standard input but it has been demonstrated to be vulnerable to buffer overflow errors because it does not test the size of this input [8]. Likewise, Bishop and Dilger [9] focused on a specific type of software errors called Time-Of-Check-To-Time-Of-Use (TOCTTOU). These race condition errors occur when a function or a piece of program is executed without checking whether it yields a secure state or not. A known example resides in SUID-root programs in UNIX based environments that check the access permissions to a file and modify it directly without checking whether this security property still holds after the modification. In Reference [10], Viega *et al.* give 12 rules that prevent several errors in JAVA code.

Gosh *et al.* [11] proposed a white-box dynamic security testing technique that relies on two main principles: random input generation and fault injection. Input generation gives the risk analyst the ability to generate a set of test values that will be submitted to the analysed program as inputs in order to study its behaviour. Fault injection is the process of perturbing the execution of a program by introducing several insecure states.

5.1.3. Vulnerability library structure. Vulnerability information is customary structured in various manners including traditional (i.e. relational) databases, mailing lists, and newsletters. The most important vulnerability databases (VDBs) are given in the following points:

- *The Computer Emergency Response Team* (*CERT*) *Advisories*: CERTs publish warnings related to the most important, and most recent, vulnerabilities. They collect information about different types of security incidents. After analysing these events, reports including the possible exploits and the related countermeasures are established.
- *The MITRE Common Vulnerability and Exposures* (*CVE*) *list*: CVE is defined as a 'standardized list of names for vulnerabilities and other information security exposures'. Substantially, CVE is not a database; nevertheless, it is very helpful when the risk analyst uses data from different VDBs. Practically, within the risk management framework, multiple vulnerability scanners are used for a better completeness of the detected weaknesses. In this case, CVE can serve to fuse the data emanating from those tools into a unified list.
- *The Bugtraq mailing list*: This archive basically consists in a mailing list which allows security specialists to exchange information and to share their knowledge. Its main limit consists in the fact that the focused vulnerabilities and exploits concern only windows-based operating systems.
- *The NITS ICAT metabase*: ICAT is a CVE-based VDB which offers interesting search possibilities. It may link the user to other vulnerability databases where he can find more detailed information, or it can even redirect him towards web resources where he can find accurate defence solutions. ICAT is probably the most complete VDB as it 'indexes the information available in CERT advisories, ISS X-Force, Security Focus, NT Bugtraq, Bugtraq, and a variety of vendor security and patch bulletins'. Hence, it would be considered as a metabase that can be used as a search engine rather than a VDB itself.

These vulnerability databases often include advanced research and browsing functionalities. The security analyst often needs precise information to process accurately the security weaknesses. Especially, when mapping vulnerabilities to threats (in order to identify the potential attack scenarios), an appropriate data structure should be used. Schumacher *et al*. [12] proposed a data-mining approach to build VDBs. The main focus of this work was the identification of the faulty patterns. It has been argued that the use of data-mining within this context has been motivated by the fact that this technique can be used to discover existing patterns in data sets or to classify the new elements (i.e. when the base is being built). Of course, training constitutes the main factor upon which depends the efficiency of this approach. A plethora of concrete cases can be used at the training phase so that the mining engine acquires enough knowledge to manage vulnerability description. To increase the efficiency of the training process, unsupervised learning algorithms should be used because they are more suitable to build VDBs. In fact, the system does not initially have a classification scheme; vulnerability classes are rather defined progressively according to the similarities between the input descriptions. Furthermore, one of the most important aspects is the language that describes the vulnerabilities. As the use of a natural vocabulary may, at the mining stage, lead to various problems such as homonyms and identical words, logic-based languages are preferred because their properties can be more easily controlled.

Moreover, as for traditional databases, VDBs should be structured in such a way that they can be accessed easily and securely. In Reference [12], four models have been proposed to adapt with different situations. The *central* architecture, as its name indicates, defines a single VDB which is used by all the community. The *federated* model requires the participation of multiple operators that can manage distinct (in the case of a partitioned architecture) or redundant data

(in the case of a replicated architecture). The *balkanized* organization corresponds to the context where no co-ordination is set in place between the different operators. Schumacher *et al*. have remarked that this is actually the case. Finally, the *open-source* principle can be applied to VDB by making available full access to all data. Users can even get a copy of a whole VDB and customize it with respect to their needs.

5.2. The attack library

Attack modelling is one of the most crucial tasks of the risk management cycle. In fact, representing the behaviour of various types of adversaries before or after they conducted harmful actions is often affected by an important uncertainty degree. The risk analyst can never reach absolute statements about the probability or the impact of a given attack. Moreover, as attackers become more and more clever, analysing their activities is increasingly complicated. To this purpose, attack libraries structure should be sufficiently sophisticated to take into consideration both of those aspects.

One of the most important issues is to model the gradual evolution of the attacker's goal to reach the main objective. It has been remarked that attackers often proceed through multiple steps in order to fulfill their malicious intentions. Tree structures have therefore been introduced to capture those steps. Indeed, fault trees, that have been used for a long time to analyse the failure conditions of complex systems, have been adapted to the computer network security context. As it has been underlined above, the CORAS RM method relies on fault tree analysis (FTA) to evaluate and prioritize risks. Two concepts have been especially focused: minimal cut sets and top event frequencies. The former identifies attack scenarios (or composite attacks) whilst the latter allows to compute the frequency of the main attack with respect to the frequencies of the elementary threats. Helmer *et al*. also proposed to use FTA to model attacks within the frame of an intrusion detection approach.

Attack trees that were first introduced by Schneier in Reference [13] and then extended in Reference [14] can be seen as a transposition of FTA into the computer security field. Their purpose is to determine and assess the possible sequences of events that would lead to the occurrence of main attacks.

The root of a tree represents the main objective of an attacker while the subordinate nodes represent the elementary attacks which are necessary to perform for achieving the global goal. A *qualitative analysis* yields to a logical representation of the tree by reducing it to the form

$$t_0 = S_1 \vee S_2 \vee \cdots \vee S_N \tag{1}$$

where t_0 is the root of the tree and S_i, for $i \in \{1, \ldots, N\}$, is the ith attack scenario corresponding to t_0 and having the following structure:

$$S_i = t_1^{S_i} \wedge t_2^{S_i} \wedge \cdots \wedge t_{N_{S_i}}^{S_i} \tag{2}$$

where $(t_j^{S_i})_{j \in \{1,\ldots,N_{S_i}\}}$ are the elementary threats belonging to S_i.

For instance, the tree shown in Figure 4 can be reduced to the following expression:

$$a = (b \wedge e \wedge f) \vee c \vee (d \wedge g) \vee (d \wedge h) \tag{3}$$

Writing a global threat as the conjunction of multiple threat scenarios offers a useful tool to the risk analyst who can henceforth describe the various manners for achieving the main attack.

Then, a *quantitative analysis* is carried out to compute the security attributes, such as the probability of occurrence or the outcome, of the composite attacks (i.e. scenarios). Often, the

attribute of the main attack can be expressed as a function of the elementary attacks' attributes through the use of simple algebraic operators. For example, the probability of the attack a defined in Equation (6) is equal to

$$P_a = \frac{P_a P_e P_f + P_c + P_d P_g + P_d P_h}{4}$$

A substantial attack library has been proposed in Reference [15]. This approach relies on attack patterns, which are essentially used to build generic representations of attack scenarios. The constituents of an attack pattern are mainly: (1) the goal of the attack, (2) a list of preconditions (i.e. hypotheses that should be verified by the system so that the attack succeeds), (3) the attack mechanism (i.e. the steps followed by the attacker to reach the objective), and (4) a list of postconditions (changes to the system state resulting from the occurrence of the attack).

Attack patterns are in turn organized into attack profiles that introduce variants (i.e. abstract parameters which can be instantiated with concrete values).

Moreover, Reference [15] develops the concept of attack tree refinement to enrich the library of a particular organization. Generic attack profiles are used for this purpose. If the analysed system presents a set of values that match with several variants parameters, then the attack profile is said to be consistent with the system architecture. Then, the instantiation principle is applied to determine which attack pattern has the same goal as the identified attack. Based on this reasoning, three enriching mechanisms have been defined: (1) leaf node application, (2) non-leaf node application to OR-decomposition, and (3) non-leaf node application to AND-decomposition.

Mc Dermott [16] presented *Attack Net*, an attack modelling approach relying on Petri nets. A Petri net is customary composed of places, transitions, arcs, and tokens. Places are the equivalent of graph nodes while a transition is a token move between two places. According to this analogy, attack steps are represented by places which are similar to nodes in attack trees. Formally, an attack net consists of a set of places $P = \{p_0, \ldots, p_n\}$ and a set of transitions $T = \{t_0, \ldots, t_m\}$. P represents the various states of the target system whilst T models events that may trigger state changes. Moreover, tokens are used to evaluate the progression of a given attack through the study of a set of moves within the attack net.

Attack nets combine the advantages of hypothetical flaws and attack trees. The main difference with the latter is that attack events (i.e. transitions) are described separately from the system states (i.e. places). On the contrary, attack trees merge the preconditions, the postconditions, and the running steps of a given attack into a single node. Mc Dermott [16] asserts that 'attack nets are better than attack trees for top–down testing of poorly documented operational systems'. Furthermore, transitions can catch a vast variety of relationships between system states. As a passing remark, transitions turn out to be more efficient than AND/OR operators to build attack libraries. In fact, these traditional logical operators exhibit some important limits especially when enriching an existing state. For instance, adding an AND branch to a set of alternatives (OR node) cannot be achieved without restructuring the child nodes.

Recently, more sophisticated models have been developed. In Reference [17], attack nets have been combined with the WikiWeb system to enable knowledge sharing. Moreover, Wiki pages can contain customized descriptions of the formal objects (i.e. places, transitions, tokens). This gives more flexibility with respect to the rigorous attack nets. Particularly, this allows to build attack libraries through a collaborative process that involves many security experts. Of course,

this means that managing attack libraries using this framework (called AtiKi) makes them more rich.

Another key consideration consists in co-ordinated attacks involving multiple malicious users which have been addressed in Reference [18]. A formal model has been proposed to model this concept. It defines a co-ordinated attack plan as the union of *co-ordinated* individual attack plans belonging to the members of a given group. A theoretical framework based on graph theory and system state transitions has been developed to implement this reasoning.

6. RISK ANALYSIS

Risk analysis basically stands for setting up relationships between the main risk attributes: assets, vulnerabilities, threats, and countermeasures. Obviously, the main objective of building this quad is to achieve a precise representation of events that can affect the security of the information system and to deduce the corresponding security countermeasures. In other terms, risk analysis aims at (1) identifying the risk events threatening a particular system, (2) assessing their magnitudes, and (3) maintaining this magnitude below an upper threshold. Risk analysis is among the most complex components of the risk management framework. It generally involves many processes of the risk management cycle. In the following, we present the ingredients of a generic risk analysis activity and we describe several techniques that are commonly used in this context.

6.1. The risk analysis process

Generally, risk analysis breaks into three elementary activities which are

- Threat analysis, which consists in determining the potential attacks and modelling them accurately. In Reference [19], it has been affirmed that threat analysis should involve the examination of three basic elements: (1) the agent, which is the entity that carries out the attack, (2) the motive, which is the reason that causes the agent to act, and (3) the results, which are the outcome of the occurrence of the attack. Another element that can be added to the aforementioned ones is the attack mechanism. This key factor can serve to differentiate between threats that share the same agent, motive, and results. In addition, analysing the steps to carry out a harmful action is helpful for automating several tasks, such as the selection of appropriate defence solutions.
- Business impact analysis (BIA), which consists in evaluating the effect of the occurrence of an attack against a given asset (or group of assets). The major outcome of CBA is the RoI. This gives high-level managers the ability to define a strategic view of the efficiency of the potential countermeasures. For instance, balancing the initial cost of a security decision against the value of the assets that it protects might be insufficient. An idea about the period necessary to recoup the cost of the concerned safeguards is often required as some protection equipments depreciates more rapidly than others.
- Cost benefit analysis (CBA), which consists in comparing the cost and the benefit of the candidate security countermeasures to select the more appropriate ones.

It should be remarked that asset identification (or analysis) has been considered as an own task, which means that it is a focal issue. In the RM jargon, asset identification and valuation

corresponds to BIA. Its goal is to estimate the replacement cost of the security critical assets, as well as the values of their security attributes (i.e. confidentiality, integrity, availability). Depending on whether the risk analysis method is quantitative or qualitative, these values can be expressed in monetary or non-monetary terms.

The purpose of a BIA is to identify the impacts of the unavailability of some specific assets. This is indirectly needed to assess the efficiency of the potential countermeasures. Therefore, BIA is strongly linked to CBA as it will appear from the following discussion. Likewise, BIA has an important influence on the business continuity plan (BCP) development, which consists in determining the impact of security incidents on the vital business processes in order to ensure their continuous availability. However, when being directed towards the establishment of a BCP, the BIA should slightly differ from the normal case as some parameters, such as the maximum tolerated downtime (MTD), should be considered during the asset valuation.

6.2. Classifying risk analysis techniques

A first idea to classify risk analysis approaches consists in reasoning about their continuity. In fact, when looking for locating the risk analysis activity in the RM cycle, it turns out that two possibilities are available: (1) prior to the implementation of the security solutions, and (2) during the monitoring (or the incident response) process. The first alternative allows to think of risky events before their occurrence in order to make the infrastructure more convenient with the enterprise context. Therefore, this type of risk analysis approaches is called preventive as its main concern is to reduce the risk magnitude so that the system state becomes as close as possible to the ideal state (where no attack is possible to carry out). The second possibility is to conduct *post facto* risk analysis, meaning that the selected countermeasures should limit the effect of malicious acts after they substantially occur. This category is called reactive risk analysis. Although this topic has been plentifully addressed by researchers, it has seldom been put in the frame of RM approaches. It has been rather considered as a functionality of several IDSs. In fact, an interesting point might be highlighted at this level. Even though the goal of a risk analysis activity does not depend upon if it is preventive or reactive, its constituency might be heavily affected by this factor.

The major difference between preventive and reactive risk analyses resides in the uncertainty formalisms. More concretely, when reasoning about preventive countermeasures, the risk analyst has not a full idea about the occurrence of the attacks. Henceforth, a probability of occurrence is assigned to every threat event. On the other hand, reactive incident response relies on the actual occurrence of the harmful event. What is uncertain in this case is whether the attack is effectively related to the detected signs of intrusion. Traditionally, alerts generated by IDSs form the major indicator of the occurrence of an attack. Two factors dictate the introduction of uncertainty formalisms at this level. First, security alerts do not necessarily imply the actual presence of an intrusion as they can correspond to false positives (FPs). Second, more than one attack can correspond to a single alert, meaning that a weighted link can be built between the potential attacks and the generated alerts. Consequently, it appears that the attack–alert paradigm is at the centre of the reactive risk analysis. Moreover, the security attributes differ from the preventive to the reactive case. For instance, the probability of occurrence, which is used to assess attacks prior to their occurrence, cannot be considered for incident response. Even though it can be substituted by the probability of a true positive (TP), this latter is global as it does not vary from one attack to another.

7. RISK ASSESSMENT

The risk assessment module is, in some sense, the core of the computational framework in a RM process. It is customarily done according to two factors: impact and probability which represent, respectively, the damage that would result from the concretizing of a potential threat and the chance (likelihood) that the risk becomes actual. As three concepts can be modelled in various ways, a plethora of risk assessment techniques have been developed. This section explains how these techniques can be categorized and explores the most relevant models that associate formal representations to computer security risks. Finally, to complete the decision support framework, methods allowing the selection of the best countermeasures are reviewed.

7.1. Quantitative vs qualitative approaches

As it has been underlined in the foregoing discussion, risk assessment can be performed either quantitatively or qualitatively. In this subsection, we highlight briefly the advantages and the shortcuts that are associated with each of these approaches.

Advantages of quantitative risk assessment:

- A rich variety of metrics can be used to represent and evaluate the various risk parameters. This allows a more granular analysis of the risk events.
- The values of the risk parameters are expressed according to their nature (e.g. monetary units for asset importance, threat impact). As no translation to a different scale is necessary, those values are more accurate.
- Sophisticated decision-making techniques can be used as the quantitative assessment provides a credible set of parameters. This aspect is useful to fulfill the cost-effectiveness requirement.
- The results of the risk analysis process can be expressed in management's language. This makes it more efficient to help the enterprise in reaching its business objectives.

Limits of quantitative risk assessment:

- The computational methodologies are often complex. Managers may face several difficulties to understand some advanced aspects.
- An important hardware and software infrastructure is needed to conduct quantitative risk assessment approaches. Automated tools are then required to support the manual effort in order to accelerate the process and to make it more precise by avoiding computation errors. Nonetheless, such software are not always available. Developing an appropriate tool for an enterprise may be so expensive that the managers become reluctant to proceed to a risk management activity.
- Data collection requires a substantial interest. In fact, the gathered information must be precise enough to ensure the required analysis efficiency. On the other hand, the data collection mechanisms should be themselves secured to prevent the unauthorized access to sensitive data or the violation of privacy policies.

Advantages of qualitative risk assessment:

- The process does not involve a complex reasoning and it can be understood by high-level managers. This makes it easier to convince them about the importance of the risk assessment process.

- Qualitative risk assessment can often be performed manually. Computational steps are restricted to simple arithmetic operations on a reduced set of integer numbers.
- The data collection process is less complex than in the quantitative case. Questionnaire-based techniques are often used to this end.
- The use of several parameters that cannot effectively be measured but that can be subjectively evaluated gives more freedom to the risk analyst to choose the assessment parameter basis.

Limits of qualitative risk assessment:

- The lack of theoretical bases increases the uncertainty rates that affect the risk assessment results. Those latters do not give a precise knowledge about the identified risk events.
- The efficiency of the qualitative reasoning relies on the expertise of the risk assessment team (or the population that responded to the questionnaire). This is an important weakness as security specialists are not always available at reasonable costs.
- The results of the qualitative risk assessment process cannot be substantially tracked because their subjective nature renders them hard to evaluate.

7.2. Risk assessment for preventive risk analysis

Almost all of the computer security risk assessment methods rely on the ALE, which is derived from the annualized rate of occurrence (ARO) and the single loss expectancy (SLE). Formally, let V (resp. E) be the value (resp. the exposure) assigned to the studied asset. E is the percentage of the asset that would be lost should the attack A be realized against it. Then, the single loss expectancy (SLE), is equal to $A \times E$ and

$$\text{ALE} = \text{ARO} \times \text{SLE} = \text{ARO} \times A \times E \tag{4}$$

In the following we discuss briefly some relevant aspects related to each of these variables.

- The ALE is expressed in annualized terms in order to be easily integrated within the business planning activity.
- The ARO stands for the frequency with which a given threat is expected to occur during a year. This parameter is global in the sense that it is not concerning a specific asset.

7.3. Risk assessment for reactive risk analysis

In Reference [20], the authors presented a cost model for reactive countermeasures relying on the following prominent decision criteria:

- *Detection cost*: Prior to the generation of a security alert by an analyser, the corresponding sensor should have captured appropriately the required parameters (e.g. metric value, packet header fields). Then, the analyser inspects this data to decide whether an intrusion occurred or not. Both of these operations have a cost that must be taken into account when computing the total cost of the incident. Hereinafter, we suppose that this cost is intrinsic to the elementary IDS (consisting of an analyser and a sensor) that it does not depend on the nature of the generated alert. A way to express this detection cost, denoted by $(\gamma_{A_i})_{i\in\{1,\ldots,n_A\}}$.
- *Cost of reaction*: Each of the potential reaction r_k ($k \in \{1, \ldots, n_r\}$, where n_r is the total number of security countermeasures) has a cost γ_{r_k} that depends heavily on the reaction

itself. It can be expressed in terms of different attributes such as monetary units or processing resources.

- *Attack impact*: Carrying out a malicious act on an information system causes various kinds of undesirable effects. To this purpose, the impact ι_{a_j}, which represent these effects, will be represented by multiple attributes. Furthermore, a novel concept, called progression factor and denoted λ_{a_j}, is introduced to have a more suitable representation of the benefit of a given reaction. In fact, the system can react before or after the attack occurs. Each of these reactions has a different benefit. The former aims at stopping the complete execution of the attack after receiving signals that reveal it while the objective of the latter is to make the system recover at the right time. To this purpose, the impact is modelled as a function of the progression factor as it will be illustrated below.
- *IDS efficiency*: Efficiency represents a fundamental topic in IDS research. It should detect a substantial percentage of intrusion into the supervised system while still keeping the false alarm rate at an acceptable level. In our case, it relies on two factors: the alert and the related attack. To this purpose, two random variables are defined. D is related to the detection activity and equals 1 if an alarm is generated and 0 if not, while A states if an attack occurred. Similarly, it is equal to 1 if an attack took place and 0 if not. According to this reasoning, the efficiency of the IDS can be measured using the following conditional probabilities:

  1. $P(D|A)$: represents true positive rate. Can be estimated by submitting a packet flow which is contaminated of 'known' attacks,
  2. $P(D|\neg A)$: represents false positive rate. Corresponds to the case where the detector issues an alarm while the corresponding attack did not occur,
  3. $P(\neg D|A)$: represents false negative rate. Measures the probability of non-generation of an alert in case of occurrence of a security threat,
  4. $P(\neg D|\neg A)$: represents true negative rate. Corresponds to the absence of security alerts knowing that no attack has been conducted against the system.

The alert reader may have remarked that part of these criteria has been proposed in References [21, 22]. Effectively, the main novelty of the approach proposed in Reference [20], is the introduction of the progression factor at the level of the computation of threat events. As it has been pointed out above, the impact of an attack depends on its type.

Concerning the impact distribution of security threats, numerous models can be developed. Three examples are given below, they can be obviously enriched by other functions.

1. *Constant impact*: The impact function has the following expression in this case:

$$I(t) = I_0, \quad \text{if } 0 \leqslant t \leqslant t_0 \tag{5}$$

2. *Linear impact*: In this case, the maximum tolerated downtime (MTD) is introduced in the impact function. Practically, the MTD corresponds to the maximum time interval in which the service of interest can be stopped. In the following expression, the MTD is denoted t_0:

$$I(t) = \frac{I_0}{t_0} t, \quad \text{if } 0 \leqslant t \leqslant t_0 \tag{6}$$

3. *Exponential impact*: This model corresponds to attacks that have an impact that increases rapidly across time.

$$I(t) = I_0 \, e^{(t/t_0)-1}, \quad \text{if } 0 \leqslant t \leqslant t_0 \tag{7}$$

These functions model the variation of the impact according to time. More sophisticated approaches can be conducted. For example, the rank of an attack in a scenario (i.e. composite attack) can be significant in some contexts.

Furthermore, we consider the evolution of the attack only between its occurrence and the MTD. In fact, we suppose that the IDS can react only during this interval. Applying reactions after the MTD should be subjected to a more complex decision framework, which is the disaster recovery plan (DRP). This process is effectively collective and requires the intervention of incident response team (IRT) members. Therefore, the decision analysis presented in this paper cannot be applied in this context.

Consider an analyser A_i that has generated an alert corresponding to an attack a_j, the cost of performing a response r_k to stop the attack a_j is given by

$$\gamma(r_k, \lambda_{a_j}) = (\gamma_{A_i} + \gamma_{r_k} + \iota_{a_j}(\lambda_{a_j}))P(D|A) + (\gamma_{A_i} + \gamma_{r_k})P(D|\neg A) \tag{8}$$

while the corresponding benefit of such reaction is expressed by

$$\beta(r_k, \lambda_{a_j}) = (\iota_{a_j}(1) - \iota_{a_j}(\lambda_{a_j}))P(D|A) \tag{9}$$

In fact, as the benefit results from the fact that the attack is stopped at the progression degree λ_{a_j}, the positive effect of this reaction can be approximated by $\iota_{a_j}(1) - \iota_{a_j}(\lambda_{a_j})$ where $\iota_{a_j}(1)$ is obviously the maximum damage of the attack a_j. In addition, $\iota_{a_j}(\lambda_{a_j})$ appears as a cumulative sum of the elementary impacts of the attack a_j in the interval $[0, \lambda_{a_j}]$. Formally, this is expressed as follows:

$$\iota_{a_j}(\lambda_{a_j}) = \int_0^{\lambda_{a_j}} I(\lambda t_0) \, d\lambda \tag{10}$$

meaning that the ratio t/t_0 is considered as the basis for the estimation of the progression factor λ_{a_j} in this case.

8. MONITORING THE SYSTEM STATE

Upto this point, security policy violations have been shown to be more or less related to the vulnerabilities of the security critical component of an information systems. In the previous section, we gave an idea about how to react if a security violation is detected. However, we did not address the detection process itself. Indeed, deploying cost-effective security mechanisms to reduce the loss resulting from the occurrence of harmful actions results into a certain outcome at the business level. Nevertheless, experience has shown that such mechanisms are never sufficient as they cannot be asserted to be fully efficient. To this purpose, the security state of the analysed system should be continuously controlled. At least three reasons confirm this idea:

1. New vulnerabilities, exploits, and attack mechanisms can be discovered,
2. Violations can be detected at the right time,

3. Effective countermeasures can be reviewed to state whether they are as sufficient as they were thought to be at the design stage.

Security monitoring constitutes a tricky activity since it aims to control the security level in the analysed system. Security monitoring is customary performed through the use of IDSs. The latter use two detection techniques: pattern-based monitoring and behaviour-based monitoring.

8.1. Pattern-based monitoring

This category of monitoring techniques, also referred to as misuse detection, attempts to characterize known patterns of security violations. This basically requires a sensor network, a signature database, and an analyser. Sensors are deployed at the *strategic* segments of the monitored network to collect the needed data. Obviously, an engineering activity should be conducted before setting those sensors to ensure that the target information can be gathered without introducing a considerable overhead at the communication flow level (i.e. without affecting the quality of service). Signatures consist in descriptions of the known intrusions. An appropriate language is always needed for this representation. Such languages should possess some characteristics such as the capability of allowing multiple types of reasoning about alerts and scenarios [23, 24]. Finally, the analyser matches the collected data with the existing signatures to decide whether an attack did occurs.

Another important aspect, in addition to the description language, that affects heavily the efficiency of pattern-based monitoring consists in identifying the parameters that should be controlled by the sensors. As it is not possible to specify the behaviour of the system with full confidence, the security analyst should define exactly what should be monitored. Among the most relevant issues that should be taken into account when defining attack signatures, we cite the following three:

- *Access to the information system objects*: The access policy to the system assets (e.g. files, hosts) should be first expressed in a machine language. Then, the access events to critical resources should be logged and checked. For instance, in UNIX-based operating systems, the execution of the system command ls from a user account into/root/directory should be detected. Moreover, the access policy corresponding to malicious actions can be specified. A relevant example is to associate the existence of a Trojan horse to an open port.
- *Operations sequencing*: The order in which some operations are executed is sometimes a good indicator about the legitimacy of an access. For example, the execution of some critical commands after being logged in as a normal user is prohibited. This shows how a signature can be based on a sequence of two, or more, events.
- *Protocol violation*: Many signatures rely on detecting a communication flow that do not conform with the rules of a given protocol. For example, looking for IP packets where the source and the destination addresses are equal permits to detect the Land attack. Likewise, checking the conformance of the fragmentation fields with the corresponding requests for comments (RFCs) allows to thwart fragmentation attacks such as Teardrop.

Techniques belonging to this monitoring class are categorized by two major shortcuts. First, they are not able to detect previously unknown attacks because it is often impossible to map them to any existing signature. Second, as different manners can exist to carry out a specific attack, it might be difficult to build signatures that cover all the variants of such intrusions.

8.2. Behaviour-based monitoring

Anomaly based intrusion detection is a crucial issue as it permits to identify attacks that do not necessary have known signatures. Detection results in this mechanism depend on the values of several measurable features called metrics. Two techniques are used in anomaly detection: network profiling and thresholding. A crucial problem that needs to be addressed within the second technique is the threshold setting mechanism. In fact, the rates of false positives and false negatives depend highly on the threshold value. What's more, approaches using anomalies are not built upon a strong relationship between anomalies and attacks. This can lead to a high false alarm rate, as it is the case of the existing anomaly based intrusion detection systems. In addition, these approaches often consume more resources than those based on misuse detection.

The first task in the anomaly representation is to define the metrics that have to be monitored. Consider three families of functions denoted by $(\upsilon_i(t))_{i\in\{1,\ldots,\Upsilon\}}$, $(\eta_j(t))_{j\in\{1,\ldots,\mathbf{H}\}}$ and $(v_k(t))_{k\in\{1,\ldots,\mathbf{N}\}}$ that model the following activities, respectively:

- *User-level activities*: These include attributes, such as most used commands, typing frequency, and login/logout period, that help develop the profiles of behaviour patterns of users.
- *Host-level activities*: These include attributes (such as file structure, consumed CPU, and consumed memory) that provide indication of resource usage.
- *Network-level activities*: These use attributes (such as total packet count, packets with specific source or destination ports, and packets with specific source or destination addresses) to provide information that are gathered on network usage and engineering.

The above functions are the metrics that represent efficiently the information system state. They should be measured and monitored continuously. Information system administrators have to determine what attributes to record to ensure efficient representations of their systems. Moreover, for each of these metrics, a set of decision rules have to be defined in order to decide whether a value taken by a given function corresponds to a misuse or a legitimate use of the system. To this purpose, we introduce the following sets:

- $(N_{\upsilon_i})_{i\in\{1,\ldots,\Upsilon\}}$ (resp. $(A_{\upsilon_i})_{i\in\{1,\ldots,\Upsilon\}}$): the sets of normal (resp. abnormal) values that could be taken by the functions $(\upsilon_i(t))_{i\in\{1,\ldots,\Upsilon\}}$,
- $(N_{\eta_j})_{j\in\{1,\ldots,\mathbf{H}\}}$ (resp. $(A_{\eta_j})_{j\in\{1,\ldots,\mathbf{H}\}}$): the sets of normal (resp. abnormal) values that could be taken by the functions $(\eta_j(t))_{j\in\{1,\ldots,\mathbf{H}\}}$,
- $(N_{v_k})_{k\in\{1,\ldots,\mathbf{N}\}}$ (resp. $(A_{v_k})_{k\in\{1,\ldots,\mathbf{N}\}}$): the sets of normal (resp. abnormal) values that could be taken by the functions $(v_k(t))_{k\in\{1,\ldots,\mathbf{N}\}}$.

An action at an instant t_0, denoted $\tau(t_0)$, can force the modification of the values of the aforementioned functions at t_0. In other terms

$$\tau(t_0) = (\upsilon_1(t_0), \ldots, \upsilon_\Upsilon(t_0), \eta_1(t_0), \ldots, \eta_{\mathbf{H}}(t_0), v_1(t_0), \ldots, v_{\mathbf{N}}(t_0))$$

Definition

An action $\tau(t_0)$ is said to be *anomalous* if one of the following conditions holds:

1. $\exists i \in \{1, \ldots, \Upsilon\}$ such that $\upsilon_i(t_0) \in A_{\upsilon_i}$,

2. $\exists j \in \{1, \ldots, \mathbf{H}\}$ such that $\eta_j(t_0) \in A_{\eta_j}$,
3. $\exists k \in \{1, \ldots, \mathbf{N}\}$ such that $v_k(t_0) \in A_{v_k}$.

This means that if, during a action, a metric takes an abnormal value, then the whole transaction is abnormal. Therefore, the efficiency of this decision rule depends mainly on the efficiency of the elementary decision rules stating whether the value of a single metric is normal or not. The characterization of network anomalies is then easy as it consists simply in checking if the values of the monitored metrics belong, at a given instant, to several predefined sets. A more sensitive problem is to state whether an anomaly is related to an attack or not.

9. ADVANCED ISSUES

This section deals with several important aspects that have been subordinated by the existing methods. It has been remarked that over the years, researchers focused on evolving modular approaches for analysing security risks. Nonetheless, these approaches have not been supported by specification tools that allow to check their correctness. In addition, the relationship between attacks and countermeasures have often been reduced to simple static links, while it is substantially more complex. Finally, security solution libraries have been rarely evolved as a concept. They have rather been viewed as traditional databases. In the following, we investigate these three points and we propose some ideas to cover these lacks.

9.1. Generic risk management representation

A glance at the various RM methods shows that they share the same architectural concepts. Therefore, a generic unified representation of the RM activities can be developed. Such representation would be particularly helpful to evaluate the efficiency of RM approaches.

In Reference [25], the authors developed an algebraic model that allows to represent each RM module using its properties. This model relies on many-sorted signatures and rewriting logic. Basically, a many-sorted signature Σ is characterized by

- a set S of sort names,
- an $S^\star \times S$-sorted set of operation names (denoted Ω in the following),
- an $S^\star$-sorted set of predicate symbols (denoted Π).

In Reference [23], the authors have proposed the signature Σ_0 depicted in Figure 5 to model the risk analysis process. The richness of this algebraic structure (i.e. the many-sorted risk analysis signature) allows the representation of attack scenarios, which are attacks performed on multiple steps. The introduction of this concept affects considerably the decision making process as the efficiency of a decision differs for two scenarios having two distinct semantic structures. This stems from the fact that the quantitative comparison attributes (e.g. probability of success, impact) of the main attack is computed using the attributes of the elementary attacks.

It can be demonstrated that security countermeasures can be viewed as pseudo-inverses of potential threats with respect to the composition law $\star$. More precisely, decisions aim at making the system recover from the effect of the various possible attacks. The major interest of this reasoning resides in the fact that the decision making process, which is the essence of risk

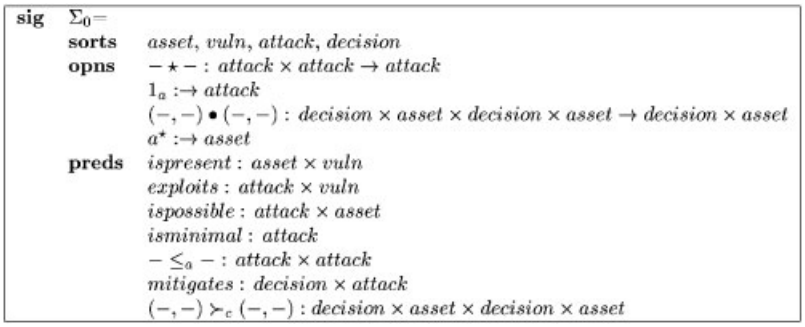

Figure 5. Many-sorted signature representing the risk analysis process.

analysis, is performed through the resolution of the equation $a \star d = 1_a$, where a models the possible attacks, d represents the potential countermeasures and 1_a is the neutral element of $\star$.

The algebraic approach defines a uniform environment for the description of assets, vulnerabilities, attacks and security decisions. An approach based on set theory and relational algebra has been used to evaluate security risks. The advantage of this approach is that it can be easily applicable in real situations since it can be implemented through the use of relational databases. Algebraic specification helps also in the decision making process where many objectives have to be optimized such as the implementation cost and the influence on the impact and the probability of the possible attacks. In fact, applying NetRAM to an enterprise can be seen as the definition of an algebra that conforms with the aforementioned specification.

Furthermore, one major benefit of this unified view is that it points out the key issues that should be focused when developing a RM approach and that has a considerable impact on its effectiveness. More precisely, a RM method should mandatory rely on:

- A set of libraries supporting the necessary knowledge to conduct the RM activities (e.g. vulnerabilities, attacks, countermeasures).
- An appropriate cost model that suits the need of the enterprise owning the assessed IT system.
- A decision support system allowing to select the *best* security solutions.
- A monitoring component ensuring a continuous control of the system state.

9.2. *A uniform specification of the RM variables*

One issue remained untreated by the researchers that addressed the risk analysis problem. It consists in the association between a potential threat and a set of decisions that thwart it. This selection is mainly, even totally in most cases, based on the knowledge of the risk analyst. As the number of attacks and decisions are increasing dramatically, this approach is no more practical. The solution for this problem is to develop a unified language to express all of the risk management variables so that they can be automatically linked. Relational specifications can, for instance, play this role.

Representing the system state relies on three main principles:

1. The system consists of a finite set of hosts (or nodes) denoted $H = \{h_1, \ldots, h_H\}$.
2. Each network node can be represented using three sets of attributes consisting in ports (P), applications (Ap), and accounts (Ac).

$$P = \{p_1, \ldots, p_P\}$$

$$Ap = \{ap_1, \ldots, ap_{Ap}\}$$

$$Ac = \{ac_1, \ldots, ac_{Ac}\}$$

   An attribute $x \in P \cup Ap \cup Ac$ related to a node h_i is denoted $h_i.x$.
3. The state of the system is modelled by a conjunction of atomic first-order predicate formulae that should be verified by several attributes. For instance, a state s such that isopen $([h_i.p_j]_s)$ corresponds to the case where the port $p_j \in P$ of the host (network node $h_i \in H$) is open.

This means that the set S of system states is theoretically infinite despite the fact that the set of attributes is finite. In other terms, any precision degree can be reached for a given system state. The state s verifying isopen $([h_i.p_j]_s)$ is less precise than s' for which isopen $([h_i.p_j]_{s'}) \wedge \exists h_k$: host, isknown $([h_i.p_j, h_k]_{s'})$. More information is given about the state by adding a predicate indicating that there exists a host h_k that knows the state of the port p_j of the host h_i. Therefore, we suppose that the system state can be represented in $\mathbb{R}^p$, one dimension is reserved for qualitative attributes while the remaining ones model the quantitative criteria.

Furthermore, we suppose that the set of system states, denoted S, is partitioned into two subsets S_I and S_S, which represents insecure and secure states. Insecure states can be seen as negations of secure states.

The most important features of the proposed approach are highlighted in the following points:

- Binary relations can be used to model preventive and reactive risk analysis processes. This means that relational decision selection mechanisms can be adapted to be applied before or after the occurrence of attacks.
- An important issue with regard to reactive risk analysis is the introduction of relational specifications to model security alerts. Particularly, the efficiency of a security alert can be assessed in the sense that the amount of false alerts (i.e. false positives and false negatives) can be estimated from the specifications of a given alert and those of the corresponding attacks.
- Attack scenarios can be managed by the risk analysis relational specification framework. The interest of this issue stands in the possibility of thwarting attacks chains instead of elementary threats. This is a more natural way to think of the security of a information system as attackers often proceed by steps to achieve their major goals.
- Complex attacks that cannot be exactly represented, or that are hard to treat, can be approached by more simple binary relations, called regular relations. We will prove mathematically that an arbitrary relation can, if it fulfills some hypotheses, be bounded by two sequences of relations that converge to it.

- Binary relations can be used to compare candidate security decisions. More than one criterion can be defined to this purpose. Essentially, the number of secure states that a countermeasure allow to reach can be taken into account when ranking security solutions.

9.3. On decision libraries

Decision archives represent a focal component of the RM framework. They allow the incident response team (IRT) members to reason about security threats. From another angle, they should be compatible with the automated IDSs used to detect the occurrence of computer network attacks. Thus, this collaborative process that allows to build and enrich countermeasure libraries imparts a complex structure to the latter. As the IRT is often multi-disciplinary, the description of the decision repository components can take various forms (e.g. logical sentences, natural language). The importance of these databases increases the complexity of the defence measure management. Consequently, appropriate structures should be set in place to provide efficient search mechanisms, as well as the reuse of library components. In fact, these two requirements are the most important among the IRT needs. Search functionalities allow IRT members to match the library elements with a query. Obviously, this query can be arbitrary and therefore, exact match would not be useful in this case. To this end, mechanisms that select the approached solutions to such matching problems have to be evolved. In addition, the reasoning held by the security specialists within a definite concept could often serve to deal with later analogous situations. Algorithms to determine the best reusable components with respect to an identified attack scenario should also be developed.

10. CONCLUSION

Throughout this paper, the most renowned risk management methodologies have been presented and analysed. Moreover, it comes from the discussion of the existing risk management methods that the existing methodologies are still insufficient to fulfil security requirements. Some architectural and technical limits have been underlined. The main conclusion of this work is that the important research literature, if exploited accurately, can considerably thwart the shortcomings.

REFERENCES

1. Gordon LA, Loeb MP, Lucyshyn W, Richardson R. 2004 CSI/FBI Computer Crime and Security Survey. Computer Security Institute/Federal Bureau of Investigation (CSI/FBI), available at: http://www.gocsi.com/awareness/fbi.jhtml
2. Campbell RP. A modular approach to computer security risk management. *Proceedings of the AFIPS Conference*, vol. 48. 1979; 293–304.
3. Summers R. *Secure Computing*. McGraw-Hill: New York, 1997.
4. Alberts CJ, Dorofee AJ. *Managing Information Security Risks: the OCTAVE Approach*. Addison Wesley Professional: Reading, MA, July 2002, ISBN: 0321118863.
5. Stolen K, den Braber F, Dimitrakos T, Fredriksen R, Gran BA, Houmb S-H, Stamatiou YC, Aagedal JO. Model-based risk assessment in a component-based software engineering process: the CORAS approach to identify security risks. In *Business Component-Based Software Engineering*. Franck Barbier (ed.). Kluwer Academic Publishers: Dordrecht, 2003; 189–207.
6. Hamdi M, Boudriga N, Kriche J, Tounsi M. NetRAM: a novel method for network security risk management. *Nordic Workshop on Secure IT Systems* (*NordSec*), Gjovik, Norway, 2003.

7. Ozier W. Risk analysis and assessment. *Handbook of Information Security*, Chapter 15 (4th edn). Auerbach Publications, October 1999; 247–285, ISBN: 0849398290.
8. Krsul I, Spafford E, Tripunitara M. *Computer Vulnerability Analysis*. Purdue University, COAST TR 98-07, 1998.
9. Bishop M, Dilger M. Checking for race conditions in file accesses. The Usenix Association, Computing Systems, Spring 1996; 131–152.
10. Viega J, Mutdosch T, McGraw G. Statically scanning Java code: finding security vulnerabilities. *IEEE Software* 2000; **17**(5):68–74.
11. Ghosh AK, McGraw GE, Charron FH, Schatz MA. Towards analyzing security-critical software during development. *Technical Report RSTR-96-023-01*, RST Corporation, December 1996.
12. Schumacher M, Hall C, Hurler M, Buchmann A. Data mining in vulnerability databases, March 2000.
13. Schneier B. *Secrets and Lies*: *Digital Security in a Networked World*. Wiley: New York, 2001, ISBN: 0471253111.
14. Tidwell T, Larson R, Fitch K, Hale J. Modeling internet attacks. *Proceedings of the IEEE Workshop on Information Assurance and Security*, U.S.A., June 2001.
15. Moore AP, Ellison RJ, Linger RC. Attack modeling for information security and survivability. *CMU/SEI Technical Report, CMU/SEI-2001-TN-01*, March 2001, available at: http://www.sei.cmu.edu/about/website/indexes/siteIndex/siteIndexTR.html
16. Mc Dermott J. Attack net penetration testing. *The 2000 New Security Paradigms Workshop*, Ballycotton, Country Cork, Ireland, September 2000.
17. Steffan J, Schumacher M. Collaborative attack modeling. *Proceedings of the 2002 ACM Symposium on Applied Computing*, Madrid, Spain, 2002; 253–259, ISBN:1-58113-445-2.
18. Braynov S, Jadiwala M. Representation and analysis of coordinated attacks. *Workshop on Formal Methods in Security Engineering*, ACM CCS, Washington, DC, U.S.A., October 2003.
19. Peltier TR. *Information Security Risk Analysis*. Auerbach Editions, Philadelphia, PA, 2001, ISBN: 0-8493-0880-1.
20. Fessi BA, Hamdi M, Benabdallah S, Boudriga N. A decisional framework system for computer network intrusion detection. *Conference on Multi-Objective Programming and Goal Programming*, Hammamet, Tunisia, 2004.
21. Lee W, Miller M, Stolfo S, Jallad K, Park C, Zadok E, Prabhakar V. Toward cost-sensitive modeling for intrusion detection. *Journal of Computer Security* 2002; **10**(1–2):5–22, ISSN: 0926-227X.
22. Wei H, Frinke D, Carter O, Ritter C. Cost benefit analysis for network intrusion detection systems. *CSI 28th Annual Computer Security Conference*, Washington, DC, October 2001.
23. Hamdi M, Boudriga N. Algebraic specification of network security risk management. *First ACM Workshop on Formal Methods in Security Engineering*, Washington, DC, 2003.
24. Stoneburner G, Goguen A, Feringa A. *Risk Management Guide for Information Technology Systems*. National Institute of Standards and Technology, Special Publication 800-30, 2001.
25. Hamdi M, Boudriga N. An abstract reduction model for computer security risk. *IFIP World Computer Congress*, WCC-SEC, Toulouse, France, 2004.

AUTHORS' BIOGRAPHIES

Mohamed Hamdi received the engineering degree (in 2000) and the master degree in communications (in 2002) from the Engineering School of Communications (SUP'COM, Tunisia). He is the co-ordinator of the Risk Analysis and Formal Validation Team at the Communication Networks and Security (CN&S) research Laboratory (University of November 7th, Carthage, Tunisia). Since 2000, he is with the Tunisian Digital Certification Agency where he is currently Head of the Risk Analysis Unit.

Noureddine Boudriga is a professor of telecommunications at the school of Communication (SUP'COM, Tunisia) and Director of the Communication Networks and Security (CN&S) research Lab. at the University of November 7th at Carthage, Tunisia. The research interests of Professor Boudriga have covered several topics including communication networks, network engineering, optical networks, wireless networks, internetwork and network security, and security engineering. He received his PhD in Mathematics from the University of Paris XI and his PhD in Computer Science from the University of Tunis. Prof. Noureddine Boudriga is the recipient of the presidential award for science and research in communication technologies (Tunisia, 2004). He also served as the founder general director of the Tunisian National digital certification agency.