Mobile Information Systems 7 (2011) 217–239
DOI 10.3233/MIS-2011-0119
IOS Press

Automatic security assessment for next generation wireless mobile networks

Francesco Palmieri[a], Ugo Fiore[b] and Aniello Castiglione[c,*]
[a]*Dipartimento di Ingegneria dell'Informazione, Seconda Università degli Studi di Napoli, Aversa* (*CE*)*, Italy*
[b]*Università degli Studi di Napoli "Federico II", Napoli, Italy*
[c]*Dipartimento di Informatica "R. M. Capocelli", Università degli Studi di Salerno, Via Ponte don Melillo, I-84084 Fisciano* (*SA*)*, Italy*

Abstract. Wireless networks are more and more popular in our life, but their increasing pervasiveness and widespread coverage raises serious security concerns. Mobile client devices potentially migrate, usually passing through very light access control policies, between numerous and heterogeneous wireless environments, bringing with them software vulnerabilities as well as possibly malicious code. To cope with these new security threats the paper proposes a new active third party authentication, authorization and security assessment strategy in which, once a device enters a new Wi-Fi environment, it is subjected to analysis by the infrastructure, and if it is found to be dangerously insecure, it is immediately taken out from the network and denied further access until its vulnerabilities have been fixed. The security assessment module, that is the fundamental component of the aforementioned strategy, takes advantage from a reliable knowledge base containing semantically-rich information about the mobile node under examination, dynamically provided by network mapping and configuration assessment facilities. It implements a fully automatic security analysis framework, based on AHP, which has been conceived to be flexible and customizable, to provide automated support for real-time execution of complex security/risk evaluation tasks which depends on the results obtained from different kind of analysis tools and methodologies. Encouraging results have been achieved utilizing a proof-of-concept model based on current technology and standard open-source networking tools.

Keywords: Dynamic access control, active networks, analytic hierarchy process, multiple criteria decision analysis, nomadic computing, security audit, security assessment, ubiquitous networking

1. Introduction

The networking world is going mobile and the heterogeneous wireless communication infrastructures enabling ubiquitous connectivity are now an important part of our daily life and their fields of application are rapidly increasing. Most of us take for granted the ability to be connected from anywhere, at any time and at a reasonable cost, so that the widespread adoption of low-cost wireless technologies, such as Wi-Fi and 3G, makes nomadic networking more and more common. Simultaneously, the increasing number of possible malicious/hostile behaviour, exploiting available known vulnerabilities or design flaws may lead to dramatic consequences. In fact, as mobile users migrate from a wireless access point to another, by moving between home, office, airport and hotel, they take with them a large number

*Corresponding author: Aniello Castiglione, Dipartimento di Informatica "R.M. Capocelli" – Università degli Studi di Salerno, Via Ponte don Melillo, I-84084 Fisciano (SA), Italy. Tel.: +39 089 969594/+39 089 969594; Fax: +39 089 969821/+39 089 969821; E-mail: castiglione@ieee.org.

of networked devices such as mobile phones, tablet PCs, handhelds and other "electronic hitchhikers", probably with unpatched or misconfigured applications. The continual afflux of vulnerable/compromised nodes threatens the integrity of the environments, as well as that of other mobile nodes operating within the same environments. As corrupted machines move from network to network, they will be able to quickly spread offending code to more network resources and to new users. In situations where transit routing is used [27], internal traffic between mobile nodes may even cross the network border.

The traditional paradigm based on the separation between a secure area "inside" and a hostile environment "outside" can no longer be effective, because authenticated users can bring uncontrolled machines in the network. A user may unwittingly bring in active threats such as viruses, Trojan Horses, Denial-of-Service daemons, or even create a hole for a human intruder. Alternatively, they may bring in passive threats such as vulnerable packages or poorly configured software. Particularly, worms could use nomadic trends to augment their spreading activity in dense urban centers or large campuses, without promptly spread to the Internet. This is a fundamental security threat that must be addressed. To contain or at least mitigate the impact and spread of these attack "vectors", wireless access control environments must be able to examine and evaluate clients on their first access to the network for potential threats or vulnerabilities.

Accordingly, a new third party authentication, authorization and audit/assessment paradigm in which once a device enters an environment it is subjected to active analysis by the infrastructure (*Admission Control*) is proposed. Moreover, if its security level is found to be not adequate it is immediately taken out from the network and denied for any further access until its problems and vulnerabilities have been fixed. The essence of the proposed architecture is allow or deny access to the network to systems based on their "perceived" threat level, i.e., before they can become infected or attack other systems. Hence, to evaluate their "health" status, an active *vulnerability assessment* is carried out, followed by a continuous passive monitoring against further suspect activities such as port scans, broadcasts or other hostile activities. The above assessment is based on a fully automated methodology for deterministically evaluating the "security level" of a networked object/node by observing it from the outside without requiring any information about its equipment, structure, security policies and services offered. The associated security analysis process takes advantage from a reliable dynamically built knowledge base containing semantically-rich information about the node under examination. This information is provided by several network mapping and security assessment tools relying on different service/application discovery, vulnerability analysis, operating system and protocol fingerprinting techniques, integrated within a common framework. It provides a complete set of metrics aiming at representing in a concise and reliable way the most significant security-related properties characterizing the mobile node itself. The above knowledge base is then exploited by using a structured decision-making methodology such as AHP (Analytic Hierarchy Process) [29] to properly combine the specific metrics and evidences collected and reliably guess the fundamental properties that are needed to explicitly rank, according to standardized risk assessment schemes and methodologies, its overall security degree. This results in a set of quantitative security indicators that could be used as a reference for the numerical evaluation of the node security. In order to take the decision on the node access admission or refusal, a numerical comparison between the resulting security levels and a set of preconfigured thresholds defining the "minimal" acceptance security profile can be performed.

The proposed approach seems to be promising for several reasons. Firstly, it does not rely only on global network traffic monitoring. Secondly, it does not require the use of specific agents to be installed on the mobile hosts, implying an undesired strong administrative control on the networked systems, so it is very suitable for the untrusted mobile networking environment. Thirdly, the whole security

arrangement is easily applicable at any network location and does not depend from specific software packages. In fact, the underlying assessment methodology is flexible, customizable, extensible, and can easily integrate other off-the-shelf analysis tools to provides automated support for the execution of complex risk evaluation tasks that require the results obtained from several different analysis processes. Finally, this type of infrastructure would strongly encourage active and timely patching of vulnerable and exploited systems, increasing the overall network security. It would also benefit users by protecting their systems, as well as keeping them up to date, and benefit local providers by protecting their infrastructure and reducing theft of service. Its deployment would also protect the Internet as a whole by slowing the spread of worms, viruses, and dramatically reducing the available population of denial-of-service agents.

2. Related work

Self-defending networks, supporting dynamic access control mechanisms have received a significant attention in literature to face the continuously evolving security challenges that require high degrees of autonomy, scalability, adaptability, and robustness. Generally, in all the environments where no reliance can be made on the presence of an experienced administrator, automated methods are called for in order to detect and block malfunctioning or malicious nodes. Nomadic wireless environments, as well as Mobile Ad-hoc NETworks (MANETs) and sensor networks, clearly belong to this class. Dynamic access control can be traced back to an early work by Thomas and Sandhu [36]. Knorr [15] used Petri net workflows to achieve dynamic access control. In [12] a vulnerability scanner is used to keep an updated record of their current vulnerabilities. The authors in [1] use the historical record of vulnerabilities and their frequency to assess the overall security for each service, by estimating the probability that new vulnerabilities will be discovered and form the basis for new attacks. In [35] a system to manage network level access based on threat information is presented. To each node and service is associated an access threshold, which is checked against the threat level of a potential access requestor to grant or deny access to it.

In [8] the authors assess the risk associated with granting a given access request and derive a corresponding level of trust required. In parallel, the trust level of the actual requesting subject is calculated and compared against the established level of admission. The work in [17] presents a Budget-Based Access Control Model aiming at mitigating the threats from insiders whose risk level may be unknown to the involved organization. A more innovative approach in self-defending networks is related to the research area of Artificial Immune Systems (AIS) [24], covering the development of biologically-inspired security systems that focus on the monitoring and detection of improper or dangerous behaviour [2] to trigger the artificial immune system reaction. The majority of research works on AIS-based intrusion/Anomaly Detection in wireless networks [14] is centered on passive detection of danger signals. Recent works have been focused on fault reconnaissance agents [31], while other research directions have integrated in this context several nonlinear techniques such as nonlinear classifiers [21] and phase-space reconstruction [32]. Similar approaches have also led to improvements in the modeling of mobility in wireless networks [9,10]. These ideas have also been adapted to Anomaly Detection [25,33] and offer a promising perspective for the identification of signals that linear systems fail to detect. In contrast, the work in [23] emphasized the role of preventive auditing and access control, coupled with active monitoring. This proposal builds upon the previous one and extends it, by adding a more robust and versatile self-defending framework and by detailing the key vulnerability assessment issues that constitute its foundation.

3. The admission control subsystem

Network administrators are highly interested in controlling which devices are allowed onto their networks, and the associated access admission decision is always driven by security concerns. In mobile wireless LAN environments this is even more important, because these networks are usually deployed to grant ubiquitous access to the Internet and are therefore popular targets for hackers. In addition, many systems/devices requiring connection to these environments are owned by people nearly completely unaware of security chores like keeping their operating systems or anti-virus definitions up-to-date. The fundamental concept of the proposed active access control strategy is that once a client mobile device enters a new environment it has to be analyzed by the infrastructure to check for potential vulnerabilities and contaminants and its security degree is evaluated in order to decide if network access can be granted without compromising the target environment. This enhanced security facility can be implemented as an additional access control phase immediately following the traditional IEEE 802.11 authentication and authorization paradigm. Here, the involved wireless access point (playing the *Authenticator* role) performing successful IEEE 802.1x authentication of the supplicant against the RADIUS or DIAMETER server (which is called *Authenticator* or *AS*) when the mobile node receives full network access and is reachable through an IP address, requires its complete security analysis by a new network entity called the *Auditor*. To ensure the proper model scalability in huge and complex network environments there may be many *Auditors* associated to the different access points, each one (eventually more than one) dedicated to the coverage of a specific area. This also provides fault tolerance and implicit load distribution where necessary. There is a large set of possible types of analyses that could be performed by the *Auditor*, including external network scans and probes, virus scans, or behaviour monitoring. For instance, a simple examination might determine the versions of installed OS and software along with appropriate security patches, verifying the existence of open vulnerabilities where applicable. During the different examination phases, the *Auditor* computes a complex multi-dimensional score, which represents the resulting client insecurity degree, that is a vector of different indexes associated to each security category, where each index is calculated as the weighted combination of individual scores assigned to the different analysis results.

The task of specifying the individual scores to be assigned to the different categories (or security criteria) is handled through a complex decision-making strategy, known as AHP, that structures the available choices concerning the different criteria into a hierarchy and assigns a relative importance weight to each one of these criteria by comparing and ranking all the possible alternatives. That is, a numerical weight or priority is derived for each element of the hierarchy, allowing diverse and often incommensurable elements to be compared to one another in a rational and consistent way. Since the target access network environments and their security requirements may vary widely, the above comparison process is driven by the experience and knowledge of several security experts (maybe involved in the specific administration task), with the role of decision makers, that systematically evaluate the various elements by comparing them to one another, two at a time, with respect to their impact on an element above them in the hierarchy. This results in a judgments matrix, defined at the initial system configuration time, expressing in terms of pairwise comparisons, the relative values of a set of security attributes. For instance, it may define the relative importance to the node operating system security degree as opposed to the presence of a certain type of vulnerability or to the lack of a required security update/patch. The choice is whether the former security criterion is extremely more important, rather more important, as important, and so on down to extremely less important, than the other one.

Once the security analysis terminates, the resulting degree is compared with an acceptance threshold configured at the *Auditor* level and, if the computed security score does not fall below the threshold,

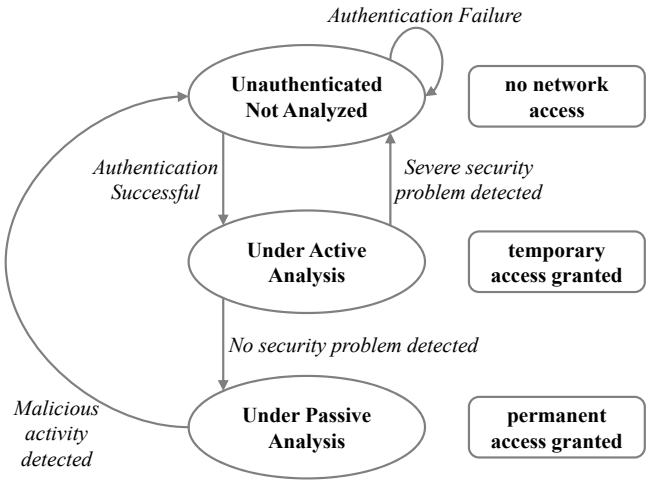

Fig. 1. The admission control process status diagram.

the *Auditor* returns a reject response to the access points that immediately denies any further network access to the examined device. Otherwise, the mobile client does not lose its network access, but the examination for the involved node will transition from the active to the passive state, which means that using standard intrusion detection techniques, the *Auditor*, on behalf of the local infrastructure, could continuously examine network traffic to determine if any entity is trying to launch a port scan of any other form of attack or take over other machines. In this case the *Auditor* quickly notifies the above event to the access point and consequently the node is immediately taken off from the network. The whole process is depicted in Fig. 1.

The issue of mobile clients possibly equipped with anti-scan mechanisms deserves a specific discussion. These clients will resist the scanning, thus not triggering any vulnerability detection alert. The presence of anti-scan systems can result from a legitimate intent of raising the security of the client, but it can also be part of a malicious plan to evade surveillance and enter the network. If the second case holds, the client would likely be granted access, but subsequent active behavior infringing the security policy would be detected by the Auditor though passive monitoring. There is, though, the threat of a malicious system crafted to appear as a secure client, whose purpose is to enter the network and grab information via passive monitoring. The gathered data could then be released after disconnection.

The proposed architecture can scale quite easily. It can be adapted to a large network by simply replicating its components. All that is needed is to properly set up relationships between each *Auditor* and its controlled access points. The resulting system is flexible, customizable, extensible, and can easily integrate other common off-the-shelf tools. In addition, it provides automated support for the execution of complex tasks which require the results obtained from several different tools.

3.1. Implementing access control

Available technologies implement wireless security and authentication/access control mechanisms such as IEEE 802.11i and IEEE 802.1x to prevent the possibility of unauthorized users gaining undue access to a protected network. Unfortunately, this arrangement falls short as a tool to avoid the inside spread of hostile agents, since in the modern nomadic networking environment authentication and authorization are no longer sufficient and need to be completed with a strong security assessment aiming at clearly

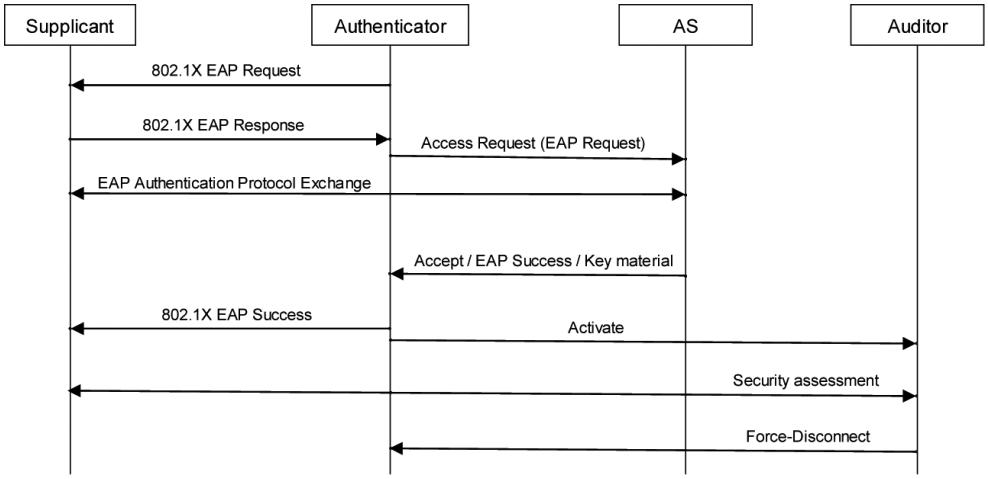

Fig. 2. Simplified access control scenario.

	Octet Number
Protocol Version	1
Packet Type	2
Packet Body Length	3-4
Identifier	5-8
Client MAC Address	9-14
Client IP Address	15-18

Fig. 3. The *Auditor* <-> access point message interface.

highlighting any potential menace. Therefore our proposed security strategy operates according to the scenario described in Fig. 2.

Four entities are involved, called the Supplicant (the wireless station), the Authenticator (the access point), the Authentication Server, and the *Auditor*. We assume, as in IEEE 802.11i, that also the access point and *Auditor* have a trustworthy UDP channel between them that can be used to exchange information and action triggering messages, ensuring authenticity, integrity and non-repudiation. After successful authentication, when a wireless device is granted access to the network, it usually issues a "DHCP Request" for an address. The local DHCP server then hands an IP address to the wireless device passing through the access point which has been modified to detect the DHCP response message (message type "DHCP ACK") and acquire knowledge about the layer-3 availability of a new device together with its IP address. Now the access point has to ask the *Auditor* for the needed security analysis.

Basically, the communication between the *Auditor* and the access point is twofold: the access point signals the *Auditor* that a new client has associated and that the assessment should begin; the Auditor may request that the access point disconnect the client in case the computed score exceed the admissibility threshold. So only two messages are necessary, respectively the "Activate" message and the "Force-Disconnect" message, whose layout is shown in Fig. 3. Port UDP/7171 is used on both the *Auditor* and

the access point to exchange messages with the other party. The message interface, as shown in Fig. 3, has been carefully designed by leaving room for further extensions. For instance, message authentication may necessary to avoid potential DoS attacks based on forging such messages. The detailed message structure is reported in Fig. 3, whose involved fields are described below:

- *Protocol Version*. This field is one octet in length and represents an unsigned binary number. Its value identifies the version of protocol supported by the sender.
- *Packet Type*. This field is one octet in length and represents an unsigned binary number. Its value determines the type of packet being transmitted. The following types are defined:
  1. *Activate*. A value of 0000 0000 indicates that the message carries an Activate packet.
  2. *Force-Disconnect*. A value of 0000 0001 indicates that the message is a Force-Disconnect packet.
- *Packet Body Length*. This field is two octets in length and represents an unsigned binary number. The value of this field defines the length in octets of the packet body: a value of 0 indicates that there is no packet body.
- *Identifier*. The Identifier field is one octet in length and allows matching of responses with requests.
- *Client MAC Address*. This field is six octets in length and specifies the MAC address of the client.
- *Client IP Address*. This field is four octets in length and indicates the IP address of the client.

No acknowledgement messages have been provided because the communication between the access point and the *Auditor* is asynchronous and no one of the involved entities need information about the completion of task by the other. In addition, synchronous communication would have introduced unacceptable latency.

3.2. Passive monitoring

The *Auditor* also acts as a passive malicious activity detector. This is a necessary facility, because in some cases the analysis can take too much time for mobile users, and so the decision adopted has been to grant immediate access to users, without waiting for the assessment completion. This may open a security hole, because a mobile client can immediately be active to propagate malware, but this activity would be immediately noticed by the passive detector. In fact, if, at any time, some hostile activity such as scanning, flooding or known attacks is discovered from an associated client, the *Auditor* requests the access point to disconnect it. To perform passive monitoring the *Auditor* must be equipped with two network interfaces. The first one is used to communicate with the connected access points, while the other will be dedicated to passively listen (to discover illegitimate activity) all the traffic generated by its associated Authenticators/access points. This mechanism, that is implemented through port mirroring on the underlying network infrastructure, allows the *Auditor* to continuously analyze, for each connected node, some properly chosen traffic health parameters, directly reflecting the network behaviour in presence of various malicious/hostile activities (massive scans, or DoS attacks), outbreaks (viruses and worms propagation) or other botnet-related services, and checking them against a "sanity" per-time limit threshold.

The most significant parameters used in the prototypal passive monitoring facility are the outgoing flow and the connection failure rate. The outgoing flow of each node represents the number of outbound nodes that an internal machine may contact per time interval. This choice is motivated from the observation that during normal operation the volume of outbound flows to unique machines is relatively small, and that this volume generally greatly increases in presence of an outgoing DoS attack or when a machines is performing an aggressive port or address space scan by seeking for susceptible nodes to be exploited

for security vulnerabilities. By choosing a good threshold value and using some well-crafted heuristics to discriminate some classical hostile behaviour, such as trying access, eventually according to a locality principle, to the same set of ports on many hosts on a target network, the above detection technique may reach a satisfactory level of success by minimizing the number of false positives that may cause guiltless nodes to be abruptly taken out from the network. The rate of failed connection requests is another significant traffic health parameter that can be measured by monitoring the failure replies that are sent to each specific node. The failure rate measured for a normal host is likely to be low. For most Internet applications (www, telnet, ftp, etc.), a user normally types domain names instead of raw IP addresses to identify the servers. Domain names are resolved by Domain Name System (DNS) for IP addresses. If DNS cannot find the address of a given name, the application will not issue a connection request. Hence, mistyping or stale web links do not result in failed connection requests. An ICMP host-unreachable packet is returned only when the server is off-line or the DNS record is stale, which are both uncommon for popular or regularly-maintained sites (e.g., Yahoo, Google, E-bay, CNN, universities, governments, enterprises, etc.) that attract most of Internet traffic. Moreover, a frequent user typically has a list of favorite sites (servers) to which most of his/her connections are made. Since those sites are known to work most of the time, the failure rate for such a user is likely to be low. If a connection fails due to network congestion, it does not affect the measurement of the failure rate because no "ICMP host-unreachable" or "RST" packet is returned. On the other hand, the failure rate measured for a compromised host, performing scanning activities, is likely to be high. Unlike normal traffic, most connection requests, initiated in such a situation, are doomed to fail because the destination addresses are randomly picked, and in addition, the associated ports may not be in use, without any service listening on it. Accordingly, by passively monitoring the outgoing flow and outgoing failure rate from the traffic flow counters associated to all the admitted nodes, the *Auditor* can easily identify potential offending hosts, such those belonging to BotNets and participating to Distributed DoS attacks, also after successfully passing the security assessment phase. Thus, if any of the measured activity indicators for one of these host exceeds a pre-configured alarm threshold (set at the system configuration time), the active *Auditor* immediately sends the "Force-Disconnect" message to the Authenticator, causing the involved mobile node to be immediately dropped from the wireless network.

The passive monitoring facility can be extended by integrating within the *Auditor* a very simple and flexible network intrusion detection system (NIDS) such as **snort** (open-source intrusion detection system) [28] to perform in a more sophisticated way real-time traffic and protocol analysis, signature matching against an up-to-date signature database and packet logging. Anyway, since passive monitoring is essentially based on the on-line analysis of outgoing traffic patterns, it can also be effective in identifying any "noisy" (i.e. volume-based) unknown (a.k.a. "zero-day") attacks originated by the mobile nodes under observation.

4. The security assessment process

Security assessment is a complicated, expensive and time-consuming process that consists in many steps and provides input to the *Auditor*'s risk evaluation needs in form of valid and reliable data about several security aspects of the involved node. Because of the impossibility to directly measure the risk associated to the admission of an unknown mobile node, different factors that may affect its security degree have to be measured, combined and correlated in an effective security assessment methodology. Thus, in order to examine the adequacy of the current security degree of a specific host, several different aspects and security-relevant properties within the scope of the assessment process have to be considered.

All the following tasks produce several results that are often measured on different scales, which makes them difficult to be compared. Depending on the targets of security measurements and the needed security level granularity several different approaches to security assessment can be identified:

- *Observation*: implies that the target to be analyzed is only viewed from the outside, while its internal characteristics are not taken into consideration at all.
- *Security testing*: describes a security assessment approach based on the extensive use of vulnerability scanners and penetration testing. The number of detected vulnerabilities in a target system under test is used to determine several security metrics. While could be many questions about the relevance of these metrics, this approach is really effective in capturing valuable information on the security strength of the examined nodes [16]. However, this methodology is highly dependent on the tools used in the penetration testing process and on the skills of the IT experts choosing the specific tests to define the needed metrics. Hence, security testing alone is a useful and effective technique for identification of vulnerabilities but lacks the "total picture" view of formal evaluation [4].
- *Security functionality structure*: in order for the assessment to stress security relevant characteristics of the particular node under examination, the meaning of the security functionalities and defense countermeasures implemented on the target network has to be well defined. This applies to the verification of the compliance to some specific security policies and controls through the correlation of specific security metrics [34]. Such security metrics monitor the accomplishments of the policies goals and objectives by quantifying their possible degree of violation by the target host. Magnitude, scale and interpretation are three important elements of a security metric that need to be decided for a successful correlation [26]. Most of the methods developed to approach security assessment have enormous difficulties to capture the complexity of the process because of the quantity of different measurable properties that are involved.

The automatic assessment performed by the *Auditor* has the objective of characterizing the security of a nomadic node when requiring access to the network, by using several metrics whose combined observation results in different clearly quantified security indexes, each one associated to a specific security perspective. Such indexes, combined into a multi-dimensional score in the form of a vector representing the specific security profile of the involved node, can be easily compared against the minimal security requirements of the target infrastructure to perform admission control. The underlying methodology takes into account the specific security properties and criteria associated to the individual evaluation metrics and combines them within a structured hierarchical decision process which assess the relative importance of these criteria, comparing the alternatives for each of them, and determining an overall ranking associated to the confidence according to which each criterion will provide correct information about the security index describing a particular security aspect. The information about the most important properties will be put together to form a relevance matrix which is then used to generate a vector of "fundamental" security indexes that is the final multi-dimensional security score. In order to automatically estimate and analyze the associated metrics values, the aforementioned security criteria must be described in a rigorous, controllable way, according to a formal model containing a possibly complete information about the node Operating System, connection properties and deployed services (identified through port or vulnerability detection scans, or assumptions based on some configuration evidences). Such a security analysis model has to be designed by taking into account the models used by the existing remote management and vulnerability scanning tools.

The key for a successful and effective assessment is the availability of quantitative data, as complete and accurate as possible, for the generation of the above security analysis model. This implies the

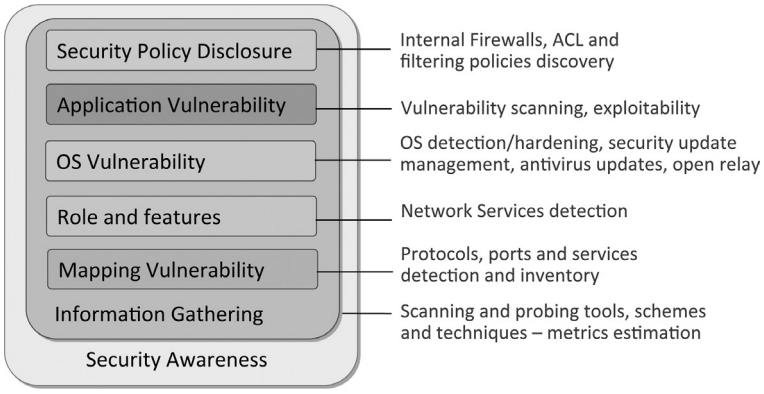

Fig. 4. Basic metrics definition model.

assignment of quantitative values to the individual security criteria and the mathematical combination of them in a way that demonstrates their relative or absolute effects on the overall system security. These effects can be generic numbers that only have bearing on each other or they can be converted into specific cost or risk values. Such a quantitative analysis can be viewed as just a refining or a partitioning of a more complex qualitative security analysis problem. It breaks down the qualitative issues into smaller factors that can have easily obtainable quantities ascribed to them. The inherent hierarchical structure of the above analysis, starting from the principle of breaking down the process into individual measurable components, requires a method to accurately estimate the weight, and hence the relevance, of each component for the determination of an accurate and quantified network security degree. AHP is an ideal decision making methodology to do this, since it allows an easier and efficient identification of the security evaluation criteria, their weighting and analysis.

4.1. Information gathering

The assessment process has been designed to be carried out according to a two phase scheme. The first phase, needed to generate a quantifiable scoring index for each fundamental network security aspect/component, requires the definition and the evaluation of a set of metrics associated with the fundamental security criteria to be considered in the context of the analysis process. The set of metrics used has been determined according to a layered approach which combines several well-known techniques and evaluation methods into a general security awareness consolidation model (see Fig. 4).

The above approach starts from the basic information gathering phase focusing on the determination of the fundamental properties of the target node in terms of service/application availability and role. It also aims at acquiring a basic visibility about the security characteristics of the node, specifically searching for the presence of potential vulnerabilities, lack of controls, weaknesses or misconfiguration problems. The above task takes advantage from a reliable and possibly complete knowledge base, which contains semantically-rich information provided by network mapping and configuration assessment facilities. Accordingly, client profiling will be accomplished through the use of specialized network analysis tools that can be used to examine the mobile node and the services running on it in a fairly non-intrusive manner. For example, the detailed information about the services active on the hosts could be determined by scanning the ports of such hosts. In addition, the results obtained from the execution of one tool are often used as the basis for additional analysis and possibly as input for the execution of other tools. That is, starting from the obtained information on the known hosts and services, more

sophisticated vulnerability assessment tools can automatically perform checks on them by looking for vulnerable applications, misconfigured services, and flawed operating system versions. This analysis can thus result in the detection of services that are unnecessary or undesirable in the target environment or in the identification of explicit anomalies and system vulnerabilities. For example, if a port scan was run and if it is determined that a normally unused port, e.g., TCP port 31337, was open on a scanned host, a penalty would be added to the security score indicating that the machine had been potentially exploited and may represent a security problem. In more detail, all the information gathering activities are accomplished by applying the following techniques:

- *OS Detection*. It is of fundamental importance to assess the OS security degree and to continuously monitor its evolution with respect to system configuration or operation changes, along with application and environment evolution. For this sake, both the OS type and its current version and patching level must be reliably identified by remotely assessing the new node requiring network access. Most of the available techniques for remotely detecting the OS running on a system rely on implementation differences between OSes to identify the specific software variant. They work by using a set of network queries and a classification model. Detection is performed by issuing the queries against the networked host, typically by sending carefully crafted network packets, collecting response packets, and feeding these responses into the classification model. If the different implementations of the software predictably generate different responses to the queries, the classification model can use them to reliably identify the remote OS. Several very common OS detection techniques have been used. The first one, called "*IP stack fingerprinting*", allows the determination of remote OS type by comparison of variations in OS IP stack implementation behaviour. Ambiguities in the RFC definitions of core Internet protocols coupled with the complexity involved in implementing a functional IP stack, enable multiple OS types (and often revisions between OS releases) to be identified remotely by generating specifically constructed packets that will invoke differentiable but repeatable behaviour between OS types, e.g. to distinguish between Linux RedHat and Microsoft Windows 7. Additionally, the pattern of listening ports discovered using service detection techniques may also indicate a specific OS type; this method is particularly applicable to "out of the box" OS installations. By determining the presence of older and possibly unpatched OS releases the *Auditor* can increase the overall insecurity score according to the locally configured policy. For example, if network administrators know that all of the legitimate clients are running Windows boxes, they will likely raise the score of an OS fingerprint indicating Linux, or the reverse. In those cases when OS fingerprinting is able to determine the operating system version, besides the type, higher scores can be assigned to the older versions. The available fingerprinting tools provide a database of thousands of reference summary data structures for known OSs that are continuously kept up-to-date by finding new probes and re-examining the existing ones.
- *Service Detection*. Also known as "*port scanning*" performs the availability detection of a TCP, UDP, or RPC service, e.g. HTTP, DNS, NIS, etc. Listening ports often imply the existence of associated applications/services running, e.g. a listening port 80/tcp often implies an active apache web server or a listening 53/udp can flag the presence of an active DNS server. There are certain vulnerabilities which can be reliably inferred from the status of a port. Many of these vulnerabilities relate to policy violations, i.e. the presence of deprecated services, from the security point of view, or, at worst ports associated to possible worm infection. Here, for each dangerous service or suspicious port detected the insecurity score is increased according to a configurable policy. Service detection will be accomplished with the nmap tool that can provide a very fast and complete assessment of the running services and in some cases their versions. Several different port scanning techniques (i.e.

traditional, sweep, half, xmas and stealth scan [7]) have been used in order to bypass internal security controls and hence to obtain a more and accurate result regarding the ports which are opened, closed, or blocked. More sophisticated techniques inherited from the *Traceroute-Like Analysis* of "Time Exceeded", "Host Unreachable" and "Network Unreachable" packet responses can be used to easily determine the list of local controls/rule sets implementing the internal access control policies of the host.

- *Vulnerability Detection*. Based on the results of the previous probes, that attempt to determine what services are running, more sophisticated scanners (e.g. [5]) with semantic knowledge about up-to-date security issues, try to exploit known vulnerabilities of each service to test the overall system security. Here, the concept of vulnerability is used in a loose sense which includes not only software design errors and implementation flaws but also misconfigurations and questionable user decisions such as using a weak password. More specifically, vulnerability detection techniques work by searching for the presence of default accounts, form validation errors, insecure **CGI-BIN** files, test Web pages, lack of bound checking in service implementation code and other known vulnerabilities. This can be accomplished through a variety of means operating both at the application-level (service behaviour and protocol compliance probing, banner grabbing or exploiting stack smashing buffer overflow for malicious code execution), or at the network-level (packet forgery, hijacking TCP connections, port diversion and ARP or IP spoofing). Testing by exploit involves using a script or program designed to take advantage of a specific vulnerability. However, there is a well-known problem associated with such kind of vulnerability assessment that often prevents it from being used extensively as a basic security practice: the safety of the scanning tools used for information gathering. That is, many scanners can cause adverse effects on the network systems being tested by crashing the involved systems/services (e.g. due to the unpredictable results of exploiting a buffer overflow vulnerability), or even worse leaving permanent damaging side effects and/or undesirable modifications to the system state (e.g., by putting a plus in the "**rhosts**" file). Consequently, the analysis must be accomplished by using a "weakened" exploit code with the only sake of probing the target system to demonstrate the presence of a vulnerability, that should not leave the system itself in a vulnerable or damaged state.
- *Security Policy Discovery*. Some useful information about the implementation of internal security restrictions through personal firewalls and/or packet filters (e.g. IP tables) can be obtained by using several sophisticated ICMP and SNMP-based probing techniques. These techniques, combined with massive scanning activities (e.g. idle scanning) are really effective in guessing the list of controls/rule sets implementing the above access control policies.

The result of the aforementioned operations are retained for a configurable time interval, in order to prevent a client continuously trying to get access from consuming inordinate amounts of resources. The optimal value of this amount of time is a tradeoff between the quest for accurate and up-to-date security assessment as well as the compelling need to save resources.

The adopted philosophy is to implement all the above techniques with the tool best suited for the specific task. This choice is more effective if a tool performs one specific task instead of implementing many different functionalities as a monolithic application. Also, tools that can perform many tasks are, normally, able to limit their operation to exactly what is needed. For example, **nmap** [18] is able to perform ping scans, port scans, OS fingerprinting, and RPC scans in a flexible and very effective way, so that it can be finely tuned to suit the specific needs of each initial discovery activity. On the other hand the Metasploit framework [22], which is an advanced open-source platform providing a set of application programming interfaces (APIs) for packaging vulnerability exploit code in a fully automated

fashion, has been used, along with the **OpenVAS** [5] open-source remote security scanner, for detecting vulnerable applications and services running on the available hosts and providing a warning level (to be used for quantitative assessment) for each possible vulnerability. Both the above tools provide very comprehensive and up-to-date vulnerability exploit databases, covering a large variety of architectures, operating systems and services. As a general rule, only open-source applications will be used for that purposes, in order to avoid licensing costs and legalities.

4.2. Security metrics and indexes

Another fundamental step in the security assessment of an host entering the network is to define significant variables that can be used to quantitatively characterize and evaluate its security. However network and security management objectives may substantially change between different target access infrastructures, as well as their basic security requirements and usage policies. Therefore, the security metrics considered by the proposed scheme are associated to a customizable set of security indexes, intended to be easily modified. The choice of such a set of indexes should be based on plausible assumptions in order to obtain significant results. Any choice should be an approximation of the real security degree and associated risk level, with some unavoidable trade-offs between metric plausibility and usability. As a demonstrative example, that might also be used as a starting point, what follows are the basic security indexes:

- *Network Mapping Vulnerability* Reports the host responsiveness to different known scanning techniques.
- *Operating System Vulnerability* Reports the degree of vulnerability associated to the OS used together with its patching/security upgrade status.
- *Application Disclosure, Vulnerability and Threat* These indexed are associated to the number of different vulnerable applications/services running on the hosts as well as to the type (i.e. leading to confidentiality, integrity, availability impacts) and associated risk level of different vulnerabilities.
- *Security Policies Disclosure* Number and types of security and filtering rules (e.g. access controls, anti-spoofing practices etc.).
- *Unexpected Findings* Results from the preceding analyses indicating the presence of unexpected/unauthorized OS or services.

To estimate the *Network Mapping* vulnerability index is necessary to model the basic scan responsiveness features of a node by using a vectors of binary variables $M = \{m_1, \ldots, m_7\}$ where the individual elements are defined as:

- $m_1 = 1$ if the node responds to ICMP echo requests;
- $m_2 = 1$ if the node is responsive to UDP port scans;
- $m_3 = 1$ if the node is responsive to "half scan" techniques;
- $m_4 = 1$ if the node is responsive to "stealth scan" techniques;
- $m_5 = 1$ if the node responds to TCP port scans;
- $m_6 = 1$ if the node shows its full network path when solicited with a traceroute (TTL-expiration triggered) technique;
- $m_7 = 1$ if the node shows an incomplete path when solicited as above.

Starting from the above variables the *Network mapping* vulnerability index Ψ_{NM} can be determined from the linear combination of the above variables m_i, each one weighed with its relative risk level, or

security relevance μ_i to be determined through AHP as described in Section 4.3.

$$\Psi_{NM} = \sum_{i=1}^{7} \mu_i m_i \tag{1}$$

After performing a portscan, some open ports will be found to belong to well-known applications, while some others will be unknown. The *Application Disclosure* vulnerability index Ψ_{AD} can be determined by taking the weighted sum of the relative number of the well-known and of the unknown listening ports. The index can thus be defined as follows:

$$\Psi_{AD} = \lambda_1 \frac{W}{P_T + P_U} + \lambda_2 \frac{(P_T + P_U) - W}{P_T + P_U} \tag{2}$$

where W is the count of well-known applications and P_T and P_U are respectively the total number of listening ports resulting from TCP and UDP portscans, and λ_1 and λ_2 are, as usual, the ranking values associated to the relative security relevance of the metric.

As far as *OS Vulnerability* together with the *Application Vulnerability and Threat* categories are concerned, the auditing process will refer to a standardized list of known vulnerability tests such as those provided by **OpenVAS**. This approach will ensure that the vulnerability checklist will be kept up to date as new vulnerabilities are discovered, their details are published, and the corresponding tests are designed and added to the **OpenVAS** archive. Additional, personalized checks may also be added if needed, provided that the actual scanner is instructed on how to perform them.

Analogously, instead of specifying the scores manually for these classes of vulnerabilities, the associated scores (or at least a good starting value) can be advantageously derived by referencing the CVSS (Common Vulnerability Scoring System) [20] scoring for the vulnerability under consideration. CVSS is an open, jointly developed, standard providing a universal method for rating vulnerabilities and quantifying the associated risk. OpenVAS supports the CVSS scoring, that is included in the new vulnerability tests, which are dispatched from the OpenVAS database to the application instances. Most vulnerability checks in OpenVAS are associated to a CVE (Common Vulnerabilities and Exposures) [6] code and any of such codes has a CVSS. CVE is a list or dictionary that provides common names for publicly known information security vulnerabilities. CVSS consists of 3 groups: Base, Temporal and Environmental. For each group, a numeric value in the range 0–10 is given, complemented by a compressed textual representation reflecting the values used to derive the score. While the Base group is associated to intrinsic and unchanging characteristics of a vulnerability, the Temporal group reflects those qualities that change over time, and the Environmental group describes the characteristics that are unique to any given environment. The calculation of Base metric involves many aspects including the complexity of a successful exploit, the possibility of a remote exploit, details about authentication and the security impact, both in terms of confidential data exposure and in terms of reduced system integrity or availability. These details describe intrinsic qualities that do not change over time. The threats generated by a vulnerability, instead do change over time, as they depend on the confidence in the vulnerability description, and on the availability of exploits and remediation solutions.

The mapping from CVSS to the internal assessment score is an important parameter that should be customized to reflect the differences and the priorities of the various environments. In this example, a weighted combination of the Base B_j and the Temporal T_j metrics (ρ_B and ρ_T are the normalized weights for the Base and Temporal metrics, respectively) has been used to obtain a score for the j-th

vulnerability belonging to the set of analyzed vulnerabilities V. Since a chain is as strong as its weakest link, the highest value is then selected.

$$\Psi_V = \max_{j \in V} (\rho_B B_j + \rho_T T_j) \tag{3}$$

Here, multiple sets V may be associated to several different indexes referring to explicit classes, e.g. *Application Vulnerabilities* (Ψ_{AV}) or *OS Vulnerabilities* (Ψ_{OV}), etc.

To calculate the *Security Policies Disclosure* index Ψ_{SD} it is necessary to combine several filtering rules-related metrics calculated from the following counters, that are updated during the initial scan operations, and reflect the number of active access controls presumably configured on internal firewalls or IP filters to protect the host:

- the number of incoming ICMP controls c_1 is incremented for each scanning action resulting in an "Admin prohibited filter" response message to an "ICMP Echo Request" or an "ICMP Info Request";
- the number of generic ICMP controls c_2 is incremented for each scanning action not resulting in any response to an "ICMP Echo Request" solicitation;
- the number of incoming TCP controls c_3 is incremented for each scanning action resulting in an "Admin prohibited filter" response message to a TCP port scan with the "SYN" or "ACK" TCP flags set;
- the number of generic TCP controls c_4 is incremented for each scanning action not resulting in any response to any TCP port scan solicitation;
- the number of incoming UDP controls c_5 is incremented for each scanning action resulting in an "Admin prohibited filter" response message to an UDP port scan solicitation;
- the number of generic UDP controls c_6 is incremented for each scanning action not resulting in any response to any UDP port scan solicitation.

Let $C = \sum_{i=1}^{6} c_i$ be the total number of detected controls, as in the previous cases, the linear combination of the individual rules/controls counters is weighted by using their relative vulnerability/risk priorities ω_i, resulting in the following equation:

$$\Psi_{SD} = \frac{1}{C} \sum_{i=1}^{6} \omega_i c_i \tag{4}$$

Finally, the *Unexpected Findings* index is computed by comparing the results from the previous analyses against a security policy specifying what the expected parameters (e.g. OS or open ports/accessible applications) are like in the environment under consideration. This metric is heavily dependent on the operating environment. For it to be effective, network administrators are requested to compile a list of OSes, applications, and open ports which they consider appropriate in their environment. Anything outside that list will be considered suspicious or harmful, that is each detected problem heavily taxes the computed insecurity score since it typically may be a very dangerous symptom of an open vulnerability. However, the presence of open ports outside an admissible range specified by the network administrator could be evaluated differently, in accordance with locally defined policies. In this context, the metrics chosen are the following:

- the u_1 metric is computed by evaluating, upon detection of an unexpected OS, the reliability of the detection itself. OS fingerprinting tools usually return a percentage which expresses the confidence in their findings;

- the number of unexpected fingerprinted applications u_2 is incremented for each detected application not found in the list of allowed/expected applications;
- the average reliability score of the unexpected fingerprinted applications u_3 is computed by taking the average of the single reliability percentages involved;
- the number of unexpected TCP open ports u_4 is incremented for each TCP port found to be responsive;
- the number of unexpected UDP open ports u_5 is incremented for each TCP port found to be responsive.

The overall index is computed by taking a linear weighted combination (with weights being γ_i) of the above metrics:

$$\Psi_{UF} = \sum_{i=1}^{5} \gamma_i u_i \tag{5}$$

4.3. The ranking methodology

In order to measure the network security indexes correctly and accurately, it is essential to explicitly quantify the importance of each component metric associated to the previously defined fundamental security criteria for the sake of the overall node security. Furthermore, when scoring the node security by combining several individual criteria, care must be taken to ensure that the method used is both simple and consistent. The chosen methodology decomposes the overall security scoring problem into simpler and more manageable sub-problems, each one specialized in ranking the criteria/metrics associated to each individual index, to form a multi-level, multi-target and multi-factors structure. The AHP allows these factors to be compared, with the importance of individual factors being relative to their effect on the problem solution. Ranking decisions, represented by priorities between individual factors, are established using pairwise comparisons determining the relative weights of the basic security evaluation metrics. Different weights will lead to different evaluation results. Such decisions, to be taken at the initial system configuration time (or "training" phase), should rely on the experience and knowledge of several competent security professional or decision makers performing pairwise comparisons between the various performance metrics associated to each criterion, and between the various criteria. Each decision maker has to indicate, according to the security objectives and policies of the target network, his/her own pairwise comparison matrices and the resulting opinions are then aggregated, i.e. for each pairwise comparison the average value is computed. The essence is to construct a matrix expressing the relative values of a set of attributes. For instance, an individual element may represent the relative importance of the presence of exploitable vulnerabilities ensuring super-user access as opposed to the possibility of remotely crashing the host. Each of these judgments is assigned a number on a scale. A commonly used scale (see [3]) is given in Table 1.

In detail, the aforementioned pairwise comparisons matrix E, where the generic element E_{ij} represents the quantitative judgment e_{ij} between the security criterion (or metric) X_i compared with the criterion X_j, is developed as follows:

$$E = \begin{array}{c} \\ X_1 \\ X_2 \\ \vdots \\ X_n \end{array} \begin{array}{c} \begin{array}{cccc} X_1 & X_2 & \cdots & X_n \end{array} \\ \begin{pmatrix} 1 & e_{12} & \cdots & e_{1n} \\ \frac{1}{e_{12}} & 1 & \cdots & e_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{1}{e_{1n}} & \frac{1}{e_{2n}} & \cdots & 1 \end{pmatrix} \end{array} \quad \begin{cases} E_{ii} = 1 \quad \forall i \\ E_{ji} = \frac{1}{E_{ij}} \quad \forall E_{ij} \neq 0, j > i \end{cases} \tag{6}$$

Table 1
Individual judgments scale

Judgment	Rating
Extremely preferred	9
Very strongly to extremely	8
Very strongly preferred	7
Strongly to very strongly	6
Strongly preferred	5
Moderately to strongly	4
Moderately preferred	3
Equally to moderately	2
Equally preferred	1

Starting from the pairwise comparisons matrix, the relative weight ω_i of each security criterion X_i can be easily estimated according to a stepwise process. At first, the total of each column in the matrix E has to be calculated. Then each element in E must be divided by its column total. The resulting matrix $\widehat{E}$ is the normalized pairwise comparison matrix, defined as:

$$\widehat{E} = \begin{array}{c} \\ X_1 \\ X_2 \\ \vdots \\ X_n \end{array} \begin{array}{c} \begin{array}{cccc} X_1 & X_2 & \cdots & X_n \end{array} \\ \begin{pmatrix} \frac{1}{\sum_{i=1}^{n} E_{i1}} & \frac{E_{12}}{\sum_{i=1}^{n} E_{i2}} & \cdots & \frac{E_{1n}}{\sum_{i=1}^{n} E_{in}} \\ \frac{E_{21}}{\sum_{i=1}^{n} E_{i1}} & \frac{1}{\sum_{i=1}^{n} E_{i2}} & \cdots & \frac{E_{2n}}{\sum_{i=1}^{n} E_{in}} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{E_{n1}}{\sum_{i=1}^{n} E_{i1}} & \frac{E_{n2}}{\sum_{i=1}^{n} E_{i2}} & \cdots & \frac{1}{\sum_{i=1}^{n} E_{in}} \end{pmatrix} \end{array} \tag{7}$$

The next step is to carry out the calculation of the individual weights ω_i. A priority vector $\vec{\omega}^T = (\omega_1, \ldots, \omega_n)$, that represents the relative weight of each performance indicator, can be directly determined from the average of the elements in each row of the normalized pairwise comparison matrix $\widehat{E}$:

$$\omega_i = \frac{\sum_{j=1}^{n} \widehat{E}_{ij}}{n} \tag{8}$$

Using these weights, it is possible to estimate the value of each security index S as a linear combination of the associated security criteria/metric (X_i), each contributing with its relative importance to the overall evaluation (ω_i):

$$S = \sum_{i=1}^{n} \omega_i X_i \tag{9}$$

Once obtained the needed ranking values the last step consists in measuring the consistency of the pairwise comparison judgments used as inputs of the whole ranking process (the so called *Consistency Ratio* or shortly C_R). In other words, this step evaluates the quality of the judgments established in the training phase and reported within the pairwise comparison matrix. Since the final weights strongly depend on the pairwise comparison matrix, it is important to guarantee that the degree of

inconsistency among the pairwise comparisons is relatively small. Typically a ratio exceeding 0.10 indicates inconsistent judgments, whereas, C_R values lower than 0.10 are considered reasonable. When the C_R is above 0.10, the pairwise comparisons matrix needs to be re-evaluated to reduce the degree of inconsistency.

To determine the C_R, the weighted sum vector $W = \{w_1, \ldots, w_n\}$ has to be estimated by multiplying each column element of the pairwise comparison matrix E by the relative weight of each item reported in $\vec{\omega}^T$ and summing the resulting values across the rows to obtain:

$$\vec{W} = E \times \vec{\omega} \tag{10}$$

The calculation method is based on Eigenvalue Analysis [13] [11]. More precisely, the inconsistency for a judgment matrix can be estimated as a function of its largest eigenvalue $\lambda_{\max}$ and the order n of the matrix. According to [29] the maximum eigenvalue $\lambda_{\max}$ would be

$$\lambda_{\max} = \frac{\sum_{j=1}^{n} \left(\frac{w_j}{\omega_j}\right)}{n} \tag{11}$$

The aforementioned relative weight (or priority) vectors $\vec{\omega}^T$ can be also viewed as the most relevant eigenvectors of the normalized pairwise comparison matrix. They are represented in their distributive form, that is, they must be normalized by dividing each element of the eigenvector by the sum of its elements so that they sum to 1. The $\vec{\omega}^T$ vectors can be transformed to their idealized form by selecting the largest element in the vector and dividing all the elements by it, so that the largest element assumes value 1, whereas the others assume proportionally lower values. The above relevant eigenvectors $\vec{V}$ can be determined by resolving:

$$(E - \lambda_{\max} I) \times \vec{V} = 0 \tag{12}$$

The consistency ratio C_R is then calculated from another quantity known as the *Consistency Index* C_I [30] defined as:

$$C_I = \frac{(\lambda_{\max} - n)}{n - 1} \tag{13}$$

by dividing it by the value R_I, or *Random Index*, that is the consistency index of a randomly generated pairwise comparison matrix, depending on the number of elements being compared.

$$C_R = \frac{C_I}{R_I} \tag{14}$$

The overall hierarchical structure of the AHP decision process is depicted in Fig. 5, where some example ranking values for the metrics associated to the aforementioned security indexes have been reported.

4.4. Performing the final comparison

The information gathering, metric estimation and ranking processes result into a number m of different security indexes $\{S_1, \ldots, S_m\}$ (that is $\{\Psi_{NM}, \Psi_{AD}, \Psi_{OV}, \Psi_{AV}, \Psi_{SD}, \Psi_{UF}\}$ in the case of the previously presented metrics framework), each one associated to a specific security perspective or point

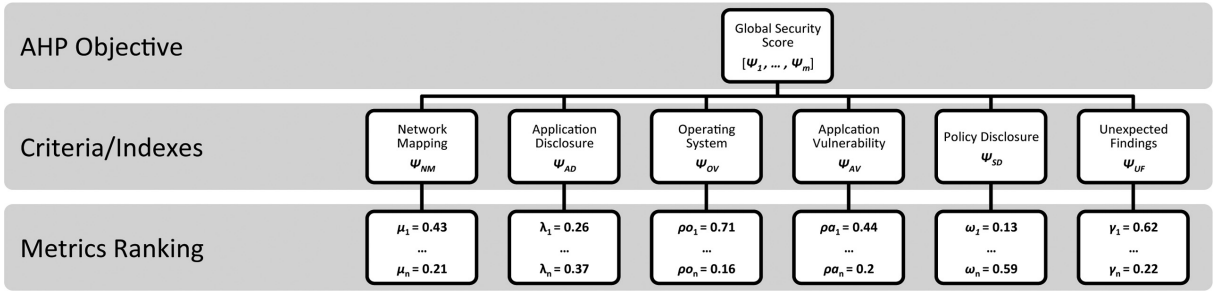

Fig. 5. The AHP decision process schema.

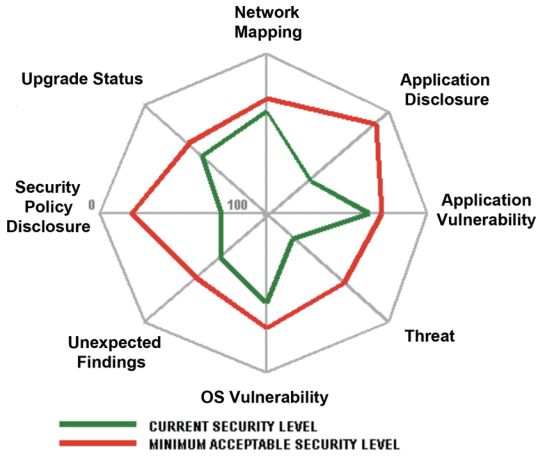

Fig. 6. Kiviat diagram representation of a comparison case.

of observation, combined into a vector $\vec{S}$ that can be viewed as the multi-dimensional score representing the actual security degree of the assessed mobile node.

In the last phase, to take the final decision for admission control, the obtained security score must be compared with the acceptance threshold configured at the *Auditor* level and, if it does not fall below the tolerable insecurity limit defined by the threshold, the node must be taken off from the network and denied further access.

However, both the security score and the threshold are multi-dimensional values and an element-by-element strict comparison against the threshold vector may be too simplistic and may lead to incorrect results. To clarify this issue, the comparison process can be modeled through the Kiviat diagram reported in Fig. 6, where the key idea is to keep the intersection between the current measurement area and the acceptance limit at its highest possible value but ensuring, however, a certain tolerance to very small noncompliance areas.

To cope with this problem, and be more confident in the comparison results, an alternative distance metric, such as the Mahalanobis distance [19] can be used to summarize the multi-dimensional characteristics into a single scale. The Mahalanobis distance is known as an effective and scale-invariant method for determining the similarity between an unknown sample vector and a known one. Such distance metric is based on an outlier detection scheme which considers the variance and covariance of the measured elements rather than just their average values. It calculates the distances, in units of standard deviation, from the group mean and weights the differences within their range of variability

accounting for natural variations within the data to be compared. It also compensates the interactions between different elements that are not guaranteed to be fully independent. This makes it more general and preferable to a traditional Euclidean distance that is more appropriate for variables that have the same variance and are uncorrelated. In detail let $\vec{x}$ and $\vec{y}$ be the vectors to be compared and C their covariance matrix, the Mahalanobis distance D_M can be expressed as:

$$D_M(\vec{x}, \vec{y}) = \sqrt{(\vec{x} - \vec{y})^T C^{-1} (\vec{x} - \vec{y})} \tag{15}$$

5. Proof of concept implementation

A simple proof-of-concept implementation of the proposed assessment-based admission control system for Wi-Fi networks has been developed to test the effectiveness of the aforementioned security enforcement strategy, with an emphasis on the use of currently available devices and open-source components. The Authenticator has been implemented on a modified Linksys access point, starting from its publicly available Linux-based firmware. Accordingly, the following new functions have been introduced:

- accept, decode and verify the new protocol messages;
- detect the "DHCP ACK" message and send the "Activate" message;
- disconnect the client upon reception of a "Force-Disconnect" message.

On the other side, the *Auditor* has been implemented on a simple Intel-based laptop PC running Linux with **nmap**, **OpenVAS**, **Metasploit** and **snort** applications installed. Although this implementation it is only a "proof of concept", the simple testbed implemented demonstrated the correct operation of the proposed security framework. In this testbed, the operations of the admission control mechanism has been verified, since the continuous vulnerability monitoring can be viewed as a subordinate case. Unless vulnerability checks are updated, the same tests are performed in both cases, so we tested them only once. An admission test has been performed with 70 wireless devices, where 54 of them were running Windows and 16 some flavors of Linux. The devices were, then, manually analyzed to verify their state of vulnerability, and the results were given to a panel of 5 experts who classified the devices according to their opinion about whether access should have been granted or not. The devices were then graded on the basis of the experts ranking, who deemed them acceptable according to the following scale:

- 5 out of 5: Safe – the device should be admitted;
- 4 out of 5: Reasonably Safe – unsure, but the device should be admitted;
- 2, 3 out of 5: Unsafe – the device should not be admitted;
- 0, 1 out of 5: Dangerous, the device should definitely be dismissed.

Of the 70 devices, 13 were graded as "Safe" and 11 as "Reasonably Safe". Thus, 24 of them received a grade $\geqslant 4$ and would thus have been admitted. The *Auditor* conceded access to 22 devices, none of them was graded "Unsafe" or "Dangerous". In addition, all 13 devices graded "Safe" were admitted. Although this experiment does not represent a conclusive test, its results are encouraging.

6. Conclusions

As millions of users migrate between home, office, airports or railway stations and bookstore, they take with them not only their computer, but also other electronic hitchhikers such as fast propagating malware

they picked up elsewhere, threatening the integrity of all the network environments they access in. This problem will only be exacerbated as wireless coverage expands and nomadic behaviour becomes more and more common. This is a fundamental security threat that must be effectively coped with. Accordingly, this work addresses the problem of authenticated users bringing vulnerable or infected machines into a secure network. In a nomadic wireless environment where machines are not under the control of the network administrator, there is no guarantee that operating system patches are correctly applied and anti-malware definitions are updated.

The paper proposes an architecture based on an automatic security assessment framework aiming at evaluating the security characteristics of the inspected host by identifying its vulnerabilities from the outside without any prerequisite information undertaken on it. The proposed architecture can scale quite easily. It can be adapted to a large network by simply replicating its components. All that is needed is to properly set up relationships between each *Auditor* and its controlled access points. The ability to dynamically estimate and compare the security of each new node at its first network access time may lead to increased awareness and improved possibility to apply specific admission control policies where they are needed. Areas for future research and development include better user interaction. Disconnection, and its reason, should be explicitly notified to users so that they would know what happened and the appropriate actions required to fix the problem. After receiving such notification, users should also be given the possibility to download a complete report about the detected vulnerabilities. If a downright disconnection is felt to be too draconian and some restricted environment should be created, a tighter integration with DHCP would also be called for.

References

[1] M.S. Ahmed, E. Al-Shaer and L. Khan, A Novel Quantitative Approach For Measuring Network Security, In *INFOCOM*, pages 1957–1965. IEEE, 2008.

[2] U. Aickelin, P.J. Bentley, S. Cayzer, J. Kim and J. McLeod, Danger Theory: The Link between AIS and IDS? in: *ICARIS*, J. Timmis, P.J. Bentley and E. Hart, eds, volume 2787 of *Lecture Notes in Computer Science*, pages 147–155. Springer, 2003.

[3] D.R. Anderson, D.J. Sweeney and T.A. Williams, *Quantitative methods for business*, South-Western Pub, Cincinnati, Ohio, 2000.

[4] M.A. Bishop, *The Art and Science of Computer Security*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002.

[5] T. Brown, OpenVAS – The Open Vulnerability Assessment System (OpenVAS). http://www.openvas.org/.

[6] T.M. Corporation, CVE – Common Vulnerabilities and Exposures. http://cve.mitre.org.

[7] M. de Vivo, E. Carrasco, G. Isern and G.O. de Vivo, A review of port scanning techniques, *Computer Communication Review* **29**(2) (1999), 41–48.

[8] N. Dimmock, A. Belokosztolszki, D.M. Eyers, J. Bacon and K. Moody, Using trust and risk in role-based access control policies, In Trent Jaeger and Elena Ferrari, editors, *SACMAT*, pages 156–162. ACM, 2004.

[9] P. Fülöp, S. Imre, S. Szabó and T. Szálka, Accurate mobility modeling and location prediction based on pattern analysis of handover series in mobile networks, *Mobile Information Systems* **5**(3) (2009), 255–289.

[10] A. Gaddah and T. Kunz, Extending mobility to publish/subscribe systems using a pro-active caching approach, *Mobile Information Systems* **6**(4) (2010), 293–324.

[11] C. Godsil and G.F. Royle, Algebraic Graph Theory, http://www.amazon.com/exec/obidos/redirect?path=ASIN/0387952209, April 2001.

[12] A. Hess and N. Karowski, Automated protection of end-systems against known attacks. In *Proceedings of IEEE/IST Workshop on Monitoring, Attack Detection and Mitigation,(Tuebingen, Germany)*, 2006.

[13] R.A. Horn and C.R. Johnson, *Matrix analysis*, Cambridge University Press, 1990.

[14] J. Kim, P.J. Bentley, U. Aickelin, J. Greensmith, G. Tedesco and J. Twycross, Immune system approaches to intrusion detection – a review, *Natural Computing* **6**(4) (2007), 413–466.

[15] K. Knorr, Dynamic Access Control through Petri Net Workflows, In *ACSAC*, pages 159–167. IEEE Computer Society, 2000.

[16] D. Levin, Lessons learned in using live red teams in ia experiments, In *DISCEX* (*1*), pages 110–119. IEEE Computer Society, 2003.
[17] D. Liu, L.J. Camp, X. Wang and L. Wang, Using Budget-Based Access Control to Manage Operational Risks Caused by Insiders, *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications* **1**(1) (2010), 29–45.
[18] G.F. Lyon, NMAP – Free Security Scanner For Network Exploration and Hacking. http://nmap.org.
[19] P.C. Mahalanobis, On the generalised distance in statistics, In *Proceedings National Institute of Science, India*, volume 2, pages 49–55, April 1936.
[20] P. Mell, K. Scarfone and S. Romanosky, A Complete Guide to the Common Vulnerability Scoring System Version 2.0, http://www.first.org/cvss/cvss-guide.html.
[21] M.E. Mohamed, B.B. Samir and A. Abdullah, Immune Multiagent System for Network Intrusion Detection using Non-linear Classification Algorithm, *International Journal of Computer Applications* **12**(7) (2010), 7–12.
[22] H.D. Moore, Metasploit – free, open source penetration testing solution, http://www.metasploit.com/.
[23] F. Palmieri and U. Fiore, Audit-Based Access Control in Nomadic Wireless Environments, in: *ICCSA* (*3*), volume 3982 of *Lecture Notes in Computer Science*, M.L. Gavrilova, O. Gervasi, V. Kumar, C.J.K. Tan, D. Taniar, A. Laganà, Y. Mun and H. Choo, eds, Springer, 2006, pp. 537–545.
[24] F. Palmieri and U. Fiore, Automated detection and containment of worms and viruses into heterogeneous networks: a simple network immune system, *Int J Wire Mob Comput* **2** (May 2007), 47–58.
[25] F. Palmieri and U. Fiore, Network anomaly detection through nonlinear analysis, *Computers & Security* **29**(7) (2010), 737–755.
[26] S.C. Payne, A Guide to Security Metrics, Technical report, SANS Security Essentials GSEC Practical Assignment Version 1.2e, June 2006.
[27] V. Pham, E. Larsen, Ø. Kure and P. Engelstad, Routing of internal MANET traffic over external networks, *Mobile Information Systems* **5**(3) (2009), 291–311.
[28] M. Roesch, SNORT – A free lightweight network intrusion detection system for UNIX and Windows. http://www.snort.org.
[29] T.L. Saaty, *Decision Making for Leaders: The Analytic Hierarchy Process for Decisions in a Complex World*, RWS Publications, Pittsburgh, Pennsylvania, 1999.
[30] T.L. Saaty, Decision making – The Analytic Hierarchy and Network Processes (AHP/ANP), *Journal of Systems Science and Systems Engineering* **13** (2004), 1–35. 10.1007/s11518-006-0151-5.
[31] E. Shakshuki, X. Xing and T.R. Sheltami, Fault reconnaissance agent for sensor networks, *Mobile Information Systems* **6**(3) (2010), 229–247.
[32] Y.Q. Shi, T. Li, W. Chen and Y.M. Fu, Network Security Situation Prediction Using Artificial Immune System and Phase Space Reconstruction, *Applied Mechanics and Materials* **44–47** (2011), 3662–3666.
[33] J. Sun, Y. Wang, H. Si, J. Yuan and X. Shan, Aggregate Human Mobility Modeling Using Principal Component Analysis, *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications* **1**(2/3) (2010), 83–95.
[34] M. Swanson, N. Bartol, J. Sabato, J. Hash and L. Graffo, Security Metrics Guide for Information Technology Systems, Technical report, National Institute of Standards and Technology (NIST) Special Publication 800-55, 2007.
[35] L. Teo, G.-J. Ahn and Y. Zheng, Dynamic and risk-aware network access management. In *SACMAT*, pages 217–230. ACM, 2003.
[36] R.K. Thomas and R.S. Sandhu, Towards a task-based paradigm for flexible and adaptable access control in distributed applications, In *Proceedings on the 1992-1993 workshop on New security paradigms*, NSPW '92-93, pages 138–142, New York, NY, USA, 1993. ACM.

Francesco Palmieri is an assistant professor at the Engineering Faculty of the Second University of Napoli, Italy. His major research interests concern high performance and evolutionary networking protocols and architectures, routing algorithms and network security. Since 1989, he has worked for several International companies on networking-related projects and, starting from 1997, and until 2010 he has been the Director of the telecommunication and networking division of the Federico II University, in Napoli, Italy. He has been closely involved with the development of the Internet in Italy as a senior member of the Technical-Scientific Advisory Committee and of the CSIRT of the Italian NREN GARR. He has published a significant number of papers in leading technical journals and conferences and given many invited talks and keynote speeches.

Ugo Fiore leads the Network Operations Center at the Federico II University, in Naples. He began his career with Italian National Council for Research and has also more than 10 years of experience in the industry, developing software support systems for telco operators. His research interests focus on optimization techniques and algorithms aiming at improving the performance of high-speed core networks. He is also actively pursuing two other research directions: the application of nonlinear techniques to the analysis and classification of traffic; security-related algorithms and protocols.

Aniello Castiglione joined the Dipartimento di Informatica ed Applicazioni "R. M. Capocelli" of the University of Salerno in February 2006. He received a degree in Computer Science and his Ph.D. in Computer Science from the same university. He serve as a reviewer for several international journals (Elsevier, Hindawi, IEEE, Springer) and he has been a member of international conference committees. He is a Member of various associations, including: IEEE (Institute of Electrical and Electronics Engineers), of ACM (Association for Computing Machinery), of IEEE Computer Society, of IEEE Communications Society, of GRIN (Gruppo di Informatica) and of IISFA (International Information System Forensics Association, Italian Chapter). He is a Fellow of FSF (Free Software Foundation) as well as FSFE (Free Software Foundation Europe). For many years, he has been involved in forensic investigations, collaborating with several Law Enforcement agencies as a consultant. His research interests include Data Security, Communication Networks, Digital Forensics, Computer Forensics, Security and Privacy, Security Standards and Cryptography.