Available online at www.sciencedirect.com

SCIENCE DIRECT®

Computer Communications 27 (2004) 1569–1584

computer communications

www.elsevier.com/locate/comcom

Anomaly detection methods in wired networks: a survey and taxonomy

Juan M. Estevez-Tapiador*, Pedro Garcia-Teodoro, Jesus E. Diaz-Verdejo

Research Group on Signals, Telematics, and Communications, Department of Electronics and Computer Technology, University of Granada, E.T.S. Ingenieria Informatica, C/Daniel Saucedo Aranda, S/N 18071 Granada, Spain

Received 16 July 2003; revised 24 June 2004; accepted 6 July 2004

Available online 28 July 2004

Abstract

Despite the advances reached along the last 20 years, anomaly detection in network behavior is still an immature technology, and the shortage of commercial tools thus corroborates it. Nevertheless, the benefits which could be obtained from a better understanding of the problem itself as well as the improvement of these mechanisms, especially in network security, justify the demand for more research efforts in this direction.

This article presents a survey on current anomaly detection methods for network intrusion detection in classical wired environments. After introducing the problem and elucidating its interest, a taxonomy of current solutions is presented. The outlined scheme allows us to systematically classify current detection methods as well as to study the different facets of the problem. The more relevant paradigms are subsequently discussed and illustrated through several case studies of selected systems developed in the field. The problems addressed by each of them as well as their weakest points are thus explained. Finally, this work concludes with an analysis of the problems that still remain open. Based on this discussion, some research lines are identified.

Keywords: Anomaly detection; Network intrusion detection; Computer and network security; Network management

1. Introduction

The automatic identification of anomalies in the behavior of network infrastructures is a concern that has aroused interest since the increasingly greater complexity and speed of operation of current technologies are complicating the system monitoring functions until reach unmanageable limits. Nowadays networks are complex systems, in which tasks like correct configuration of components, control of change over time, or the very verification of proper operation are extremely arduous chores that require highly skilled personnel as well as a large amount of time and effort.

In the field of system management, network faults are typically classified into two categories: *hard* and *soft* failures [39]. Strictly speaking, the difference between them is not given by the nature of the phenomenon that caused the malfunction (there are other classifications according to this criterion), but by the level of seriousness of the state reached. Thus, soft failures correspond to performance degradation in any parameter of a network element, like an increase in the delay of a service response, or a reduction of the available bandwidth. Alternatively, the term hard failure is commonly used to designate situations in which the whole network or some of its elements cease completely to work. Undoubtedly, in a typical work environment any of the two types of situations commented above conduce to an anomalous behavior of the network that is sooner or later experienced by users. A variety of methods have been proposed and developed to assist in the process of recognizing this kind of situations, albeit current best tools are almost all ad hoc and require qualified administrators.

This challenging scene is aggravated by the immeasurable growth of security incidents in the last years. Reports from CERT® Coordination Center on incidents, vulnerabilities, security alerts, etc. [30] corroborate this fact.

* Corresponding author. Tel.: +34-958-24-23-05; fax: +34-958-24-08-31.

E-mail addresses: tapiador@ugr.es (J.M. Estevez-Tapiador), pgteodor@ugr.es (P. Garcia-Teodoro), jedv@ugr.es (J.E. Diaz-Verdejo).

0140-3664/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2004.07.002

Thus, the number of incidents reported has evolved from only 6 in 1988, to 82,094 in 2002 (and more than 42,000 during the first quarter of 2003). Regarding vulnerabilities, 171 alerts were pointed out in 1995, while in 2002 this number growth until 4129. Furthermore, nobody calls into question that these statistics do not comprise all the incidents occurred, but merely those reported. Companies have often their own reasons to keep secret the security incidents suffered.

Within the context of network security, anomaly detection is one of two fundamental approaches used in intrusion detection (ID) technology [4,9], together with misuse-based (also called signature-based) techniques. Despite their proved benefits, one of the major limitations of current misuse-based mechanisms consists in their inability to detect patterns not previously added in the attack signatures library. For instance, Fig. 1 illustrates the best detection results for different intrusion types used in the 1999 DARPA ID Evaluation performed at MIT Lincoln Labs. In the case of Denial-of-Service (DOS) and Remote-to-Local (R2L) attacks, the detection rate with not previously seen attacks is quite poor. As in other fields like that of virus/worm detection, to rely on a system whose effectiveness is rooted on the knowledge of patterns is not a bad approach, although it is obviously not expected a good operation with unknown activities. Alternatively, the anomaly detection approach has been typically conceived as a more powerful method due to its theoretical potential for addressing novel or unforeseen attacks.

1.1. Overview of the article

Methods of anomaly detection as a component of early warning systems for network security are the focal point of this article. Section 2 introduces the problem through the presentation of several aspects: theoretical considerations, general architecture of anomaly detection systems, and a brief historical background. With the aim of facilitating the reading as well as to provide a sound framework, a taxonomy of network-based anomaly detection methods is proposed and discussed in Section 3. This classification scheme gathers the most relevant features related to both the problem itself and the proposed solutions. With the help provided by a taxonomy, it is possible to perform an analysis in which a better understanding of the problem is achieved and solutions are easily placed according to the facets that they address.

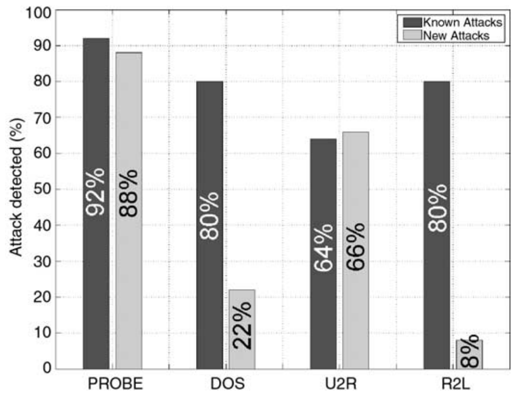

Fig. 1. Detection results of the top three IDS for the four different intrusion classes used in the 1999 DARPA/MIT Lincoln Labs evaluation [18].

In Sections 4–6, the main paradigms of traffic analysis for anomaly detection are examined. Flows analysis is presented and commented in Section 4, while Section 5 introduces and illustrates by means of several examples systems based on protocol analysis. Because of its special relevance, anomaly detection at the application layer is treated separately in Section 6. These sections do not intend to be an exhaustive list of developed systems, but an exposition of different paradigms and approaches to the problem. Nevertheless, several case studies are commented in detail in each section, providing thus examples of prototype designs as well as real systems. Most of the techniques proposed and surveyed in this work lacks of a term to designate them because their authors have not chosen a name for them. In these cases, the criterion followed has been that of referring to each model by using the authors' initials.

As was stated before, anomaly detection is still an immature technology which does not end up implanting. In Section 7, the main problems concerning the current state-of-the-art of anomaly detection technology are discussed, indicating by this way possible directions for future research. Finally, Section 8 summarizes this work by presenting the main conclusions.

2. Problem statement and scope

Evidently, to pose the problem of anomaly detection in any system implies the existence of a subjacent concept of normality. The notion of 'normal' is usually provided by a formal model that expresses relations between the fundamental variables involved in the system dynamics. Consequently, an event is catalogued as anomalous because its degree of deviation in relation to the profile of characteristic behavior of the system, specified by the model of normality, is high enough.

Formally, an anomaly detection system λ can be defined as a pair $\lambda=\langle M,D\rangle$, where M is the model of normal behavior of the system and D is a similarity measure that allows obtaining, given an activity record, the degree of deviation (or likeness) that such activities have with regard to the model M. This similarity function, D, usually depends strongly on the specific used model, in such a way that the measure process (commonly referred to as *detection*), as well as the very mechanisms used to monitor and register activities, are highly related to the modeling technique.

Fig. 2 graphically illustrates the general architecture of a network-based anomaly detection system. Although the detection is basically an analysis process, every system

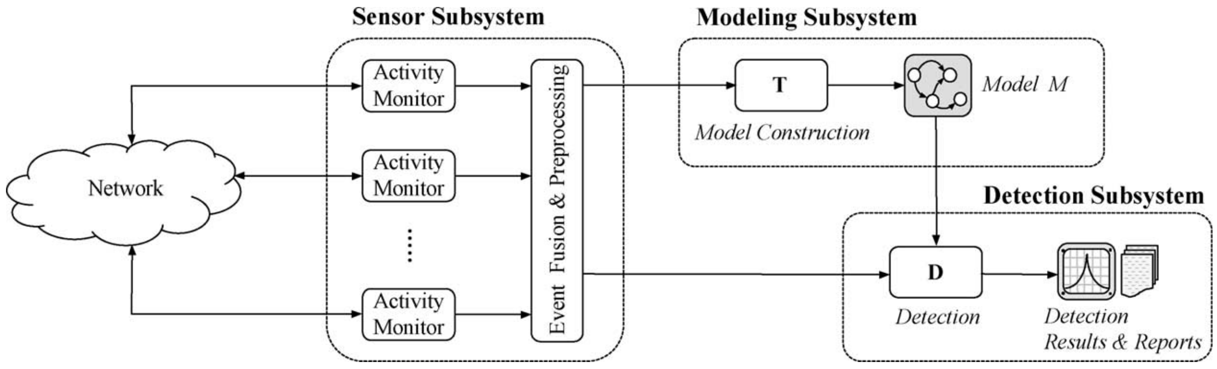

Fig. 2. Architecture of a generic anomaly detection system.

relies upon a sensor net that continuously monitors the supervised objects and records interesting activities. In the case of a distributed capture of events, a module for mixing them up is required ('Event Fusion and Preprocessing' block in the figure). Problems like deciding which activities are interesting, where to place the sensors, or how to properly manage and store the enormous amount of events which could eventually be present in a system, are out of the scope of this work.

The core of the system is constituted by two main modules: the modeling subsystem and the detection subsystem. The first of them works during a training stage (module 'T' in Fig. 2), and performs an event processing in order to obtain the model M of the normal behavior of the system. It is drastically important the completeness of the training data during this period, since the modeling subsystem will infer what is normal from them. The obtained model is subsequently used by the detection engine to evaluate new events. As was stated before, this evaluation is a measurement of the degree of deviation that such events present in relation to the model of the system.

These two modes of operation (construction and detection) are usually carried out separately. However, it is important to note that systems evolve and, therefore, the model should be reconstructed periodically in order to provide a way of adaptation to the new environment.

2.1. The suspicion hypothesis and the anomaly detection problem in network security

The key to the application of anomaly detection methods to the field known as intrusion detection consists in a simple but crucial hypothesis: *to assume that anomalous events are suspicious from a security point of view*. The acceptance of this conjecture, which can be designated as the Suspicion Hypothesis, is supported by the analysis of a large amount of network attacks. A careful study of hostile traffic reveals the existence of peculiar characteristics that could differentiate it from those usual communications across the network. With the aim of clarifying the assumption of this hypothesis, a few examples are mentioned below.

2.1.1. Example 1: denial of service (DoS) attacks by network flooding

It is a well-known fact that certain variables related to the amount of resources consumed by network traffic (for instance, the total number of packets/bytes sent or received) exhibit a regular waveform when they are considered as temporal series. The specific behavior shown is characteristic of each network system, and it depends on a large number of factors like the number of network elements, the user patterns of activity, the time of the day, and so on. While the network configuration is not altered and the users stay within their daily routine, these patterns of network activity remain unaltered along time.

On the other hand, one of the unfortunately better known forms of DoS attacks are those based on resource saturation. In the case that occupy us, traffic flooding has been surely the most used technique to carry out these kinds of actions. During a DoS attack based on traffic flooding, the attacker continuously sends a large amount of information to the target system, consuming bandwidth and degrading thus the performance of the victim. Several variants of this principle can be mentioned, from the simple TCP/UDP flooding to the devastating distributed DoS (DDoS) tools, which floods simultaneously the target from many sources [28].

Other paradigmatic example is the case of the attack known as *Smurf*. It is an old attack based on the use of forged ICMP echo request packets. On IP networks, a packet can be broadcasted to an entire network if routers allow to pass along that traffic. This attack includes the unintentional participation of an intermediary, to whom spoofed packets are sent. When all the machines in the intermediary's network respond to the ICMP echo request, the replies are sent to the target, which could potentially become unusable due to the congestion produced by the flood [29].

In any case, the fundamental result of such attacks is always a substantial change in the network typical pattern of activity, providing thus a way for an early detection.

2.1.2. Example 2: protocol misuses in intelligence gathering and other hostile processes

Far from being random, a network intrusion is used to being a very structured operation, carefully planned and executed. These processes are usually comprised of a series of well-identified steps. Nevertheless, in nearly all cases the gathering of intelligence is at the basis of every intrusion attempt, and its objective is seemly simple: to acquire as much knowledge about the target as you can. Information sought by an intruder concerns network, host and user mapping. In the case of a host, to find out data like the operating system or the services offered by the platform is crucial for subsequent actions. In most of cases, this information can be obtained through diverse protocol usages that fall out of the official protocol description.

Although protocols are described by means of formal specifications with the aim of dictating proper use, their description is not always complete. Different implementations of the same protocol stack exhibit different behavior facing certain situations, according to the particular interpretation performed by the engineers who have designed it. This fact generally allows to remotely identify the operating system that is running in the platform, thus facilitating the task of seeking exploitable vulnerabilities in the potential target. Likewise, it is possible to reconstruct part of the network topology by exploring it from the outside through 'especial' probe packets. Knowledge concerning the hosts reachable from a given point is extremely useful for an attacker with the aim of propagating his/her activities along the whole network infrastructure. As was stated above, the mentioned activities are usually performed through non-conventional protocol usages. Even when TCP and UDP scans are one of the most primitive forms of probing, it is easy to identify anomalous usages in several techniques based on them. In the case of TCP, the presence of rare combinations of flags as well as suspicious TTL (Time To Live) field values are almost always indications of scanning activity.

Regardless of their use for scanning purposes, protocol misuses have been typically used for other kind of attacks. An example is the usually referred to as *land* attack, in which an IP packet is launched against the target with a bogus source address, specifically with both destination and source addresses equal [42]. Several protocol stacks experienced serious troubles when such a packet was processed, and the destination host was usually disabled.

These and other related anomalies were pointed out during a study carried in 1993 by Bellovin [26]. Despite this early work, nowadays more works still mention the presence of malformed packets in networks as well as the increasingly important need to detect them. For instance, in Ref. [27] Bykova et al. identify several categories of malformations observed in TCP/IP packets that include:

☐ Packets with low TTL values
☐ Packets with the same source and destination port numbers
☐ Packets containing private IP addresses and/or other address violations
☐ Packets with invalid TCP flags
☐ Packets containing zero port number
☐ Packets with *strict source routing* option
☐ Too short packets

Despite the two examples provided above, the Suspicion Hypothesis is not always verified in the practice due to the inherent differences between the notions of *normal/anomalous* and *harmless/attack* event. This fact introduces as a main implication the apparition of an interesting casuistry according to the real nature of the event (from a security point of view) and the classification performed by the detector. Fig. 3 summarizes the four possible cases. True negatives as well as true positives correspond to a correct operation of the detector; that is, harmless events which are successfully labeled as normal, and detected attacks, respectively. False positives and negatives are the events that undermine the detection performance when the Suspicion Hypothesis is not verified. In fact, the false positive rate (also known as false alarm rate) is the true limiting factor in anomaly detectors. It is really easy to construct a detector devoid of false negatives, i.e. a detector that identify every anomaly: it only has to label *every* event as anomalous. By using this approach, every anomaly will be correctly detected, but its false alarm rate makes it absolutely unfit for use.

Therefore, the main objective of anomaly detection within this context is to detect all possible abnormal activities, minimizing the number of false alarms produced.

2.2. Background: anomaly detection in host-based modeling

The problem of intrusion detection, and more specifically that of anomaly detection, was historically first formulated

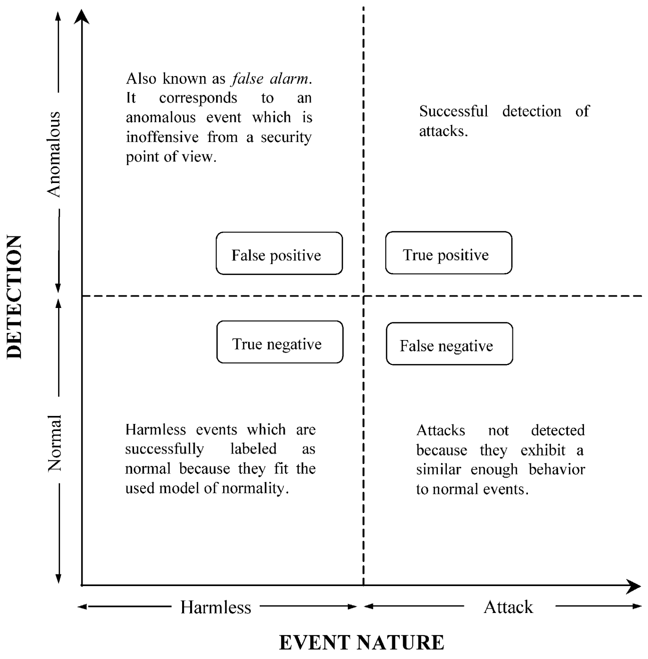
DETECTION
Anomalous
Normal
Also known as *false alarm*. It corresponds to an anomalous event which is inoffensive from a security point of view.
Successful detection of attacks.
False positive
True positive
True negative
False negative
Harmless events which are successfully labeled as normal because they fit the used model of normality.
Attacks not detected because they exhibit a similar enough behavior to normal events.
Harmless
Attack
EVENT NATURE

Fig. 3. Possible cases when anomaly detection based on the Suspicion Hypothesis is used for attack identification.

and studied within the context of host-based analysis. This circumstance can be easily understood if it is considered the initial framework in which the necessity for such systems was articulated: the attempt to establish an analogy between the security-oriented monitoring tasks performed by human administrators and the requirements of an automatic detection system.

In nearly all cases, administrators carry out their analysis by inspecting log files, which reflect those interesting events recently happened in the system. These events are basically related to activities in which users and applications make use of system resources, like the time of the day when a user logged in, the sequence of commands a user types, and so on. The first and natural approach attempted to build anomaly detectors that compared previously computed user profiles to the current user activity. The earlier works of Anderson [10] and Denning [11] followed this direction. Likewise, subsequent systems like NADIR [12], IDES [14], NIDES [15], or Emerald [13] (built upon the two previous) presented several interesting and novel techniques, and improved the detection performance of user-based modeling.

Although the analysis of user activity is a natural approach to detect intrusions, experience has showed that it is far from being accurate. The reason for this can be found in the lack of strict patterns in the user behavior. It is possible to identify several causes for these changes: new programs are installed and users start to use them, the user suddenly changes his/her working hours, the user constantly improves his/her abilities learning new commands, etc. As a consequence of such variability, user profiles are very inaccurate and detection systems raise a large amount of false alarms.

The results obtained under the approach of estimating user dynamics allowed a more in deep reflection on the features that defines the host behavior. Despite users constitute essential pieces in the system, all the actions carried out by them, like resource access (memory, files, CPU), are finally done by using programs. Furthermore, from certain perspective it is possible to divide the application code into two kinds of instructions (at high level, not assembly instructions): (1) those that compose the program logic and structure, like assignation of values to variables, condition evaluation, flow-control structures, and so on; and (2) those that constitute the effective access to system resources, such as process execution, file open/read/write, etc. In current commercial operating systems (OS) the final access to resources is limited to kernel and other internal OS components. Programs obtain the required services by executing the specific *system call* that provides the needed function. The set of system calls can be thus viewed as a set of available services offered by the OS to the 'client' applications to access system resources.

Therefore, and since the code of a given application should not change, the sequence of system calls executed by a program should be regular and predictive. On the other hand, several attacks involve program misuses (even code changes, like during a buffer overflows) which could be certainly detected by observing deviations in relation to normal executions. The analysis of system call sequences was first proposed by Forrest [16]. The idea, briefly commented before, is that of tracing the system calls that processes issue during their execution. After learning this behavior, it is possible to perform prediction of the next call given the past sequence. When a considerable amount of predictions fails to be correct, the detection system raises an alarm.

Several approaches based on the modeling of system calls executed by a program have been proposed to build host-based intrusion detection systems. For instance, a comparison of the use of several learning models for this purpose is presented in Ref. [17].

3. Criteria to classify anomaly detection methods in networks

In every field of science and technology, categorizing a phenomenon allows to systematically study the most relevant aspects involved in its investigation. In a remarkable work related to taxonomies of computer security flaws, Landwehr et al. [31] made a shrewd reflection which deserves to be cited: "A taxonomy is not simply a neutral structure for categorizing specimens. It implicitly embodies a theory of the universe from which those specimens are drawn. It defines what data are to be recorded and how like and unlike specimens are to be distinguished."

Current anomaly detection methods can be classified in accordance with the organization provided by the scheme in Fig. 4. These criteria allow not only to classify techniques, but also to understand the main aspects of the problem addressed by a given method. In what follows, each one of the proposed criteria is described in detail.

3.1. Network feature analyzed

As many other systems, a network is a complex assembly of components with non-trivial relationships that can be studied from several points of view. To consider the notion of 'normal behavior' of the network as a whole could conduce to a clear failure. When the problem of anomaly detection is posed within this context, it is necessary to formulate explicitly which network features, or facets, the model of normality refers to. Thus, detectors can be classified according to the feature modeled.

Traditionally, traffic has been the unique network-related aspect addressed by detection systems. Methods that obtain models of normality by analyzing network traffic can be divided into two groups, according to the way that the traffic is inspected. Techniques termed 'Flows Analysis' (see Fig. 4) are characterized by the study of the temporal evolution of several measures related to traffic flows.

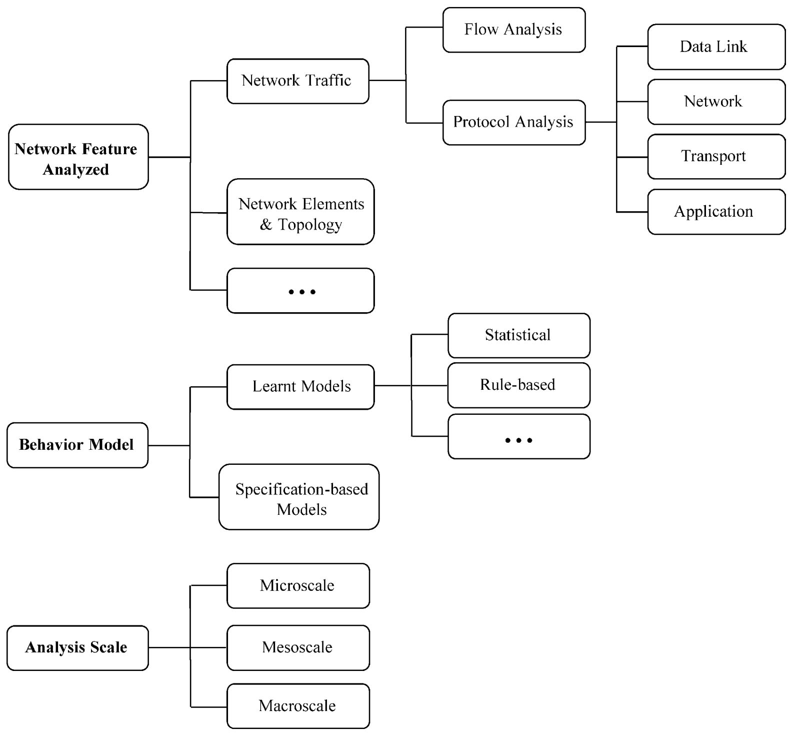

Fig. 4. Three proposed criteria to classify anomaly detection methods in networks.

Such measures are often event counts over time. Typical examples of event counts in a TCP/IP environment are (these and other counts can be also computed for a network segment instead of focusing on a given host):

- ☐ The number of bytes sent/received during a fixed time interval by a given final system.
- ☐ The number of IP/TCP/UDP/ICMP packets sent/ received by a given final system during a fixed time interval.
- ☐ The number of TCP/UDP connections initiated during a fixed time interval.
- ☐ The number of requests received by an HTTP/DNS/ FTP/SSH/etc. server during a fixed time interval.
- ☐ Etc.

As was briefly commented in the first example provided in Section 2.1, these event counts present a regular waveform when they are plotted as temporal series. Several methods based on their analysis have been proposed and will be discussed subsequently.

On the other hand, protocols are an essential piece of networking and, of course, they have been widely used in attack technology. In order to detect protocol misuses, several approaches have focused on the modeling of protocol usages. From this point of view, it is possible to classify systems according to the network layer modeled:

- ☐ Data link (Ethernet, Token Ring, etc.)
- ☐ Network (IP in most of cases)
- ☐ Transport/Control (TCP, UDP, RTP, ICMP, etc.)
- ☐ Application (HTTP, DNS, Telnet, FTP, SSH, POP, SMTP, etc.)

In contrast to the previous discussion, more facets besides traffic can be considered in order to characterize a network behavior. For instance, network topology can be an interesting point to study. In a ‘classical’, wired environment (say, a local area network), the topology is a relatively invariant feature and, therefore, irrelevant from certain perspective. Nevertheless, this situation drastically changes in ad hoc networks, in which patterns of connectivity continuously evolve and can become a key point to study the presence of hostile activities.

In spite of the two provided main criteria (network traffic and topology), other network features can be easily added to the proposed scheme.

3.2. Behavior model

As stated before, one of the essential components of any anomaly detector is the model of normal behavior of the system which serves as pattern of correctness In general terms, there are two main approaches to the construction of such a model. The first of them is based on the application of machine learning techniques, in order to automatically obtain a representation of normal behavior from the analysis of system activity. On the other hand, the alternative is to manually provide specifications of correct behavior. Anyway, this criterion concerns the model construction itself and, therefore, it is independent of whether the feature that is modeled is a network protocol, traffic flows, or any other relevant characteristic.

3.2.1. Learnt models

The natural and historically first formulated approach towards the construction of models of normal behavior of a system consists in observing it while it is working under normal conditions, and applying machine learning techniques in order to obtain a model that: (a) explains, by means of a reduced representation, the amount of data observed; and (b) be able of extrapolate to other, non-observed situations. Within this context, there could be so many different types of models as existing learning techniques. Classic examples are rule-based systems, statistical algorithms (from simple estimators to Markov Chains, or powerful Hidden Markov Models), and Artificial Neural Networks.

3.2.2. Specification-based models

In specification-based detectors, the model is not estimated by analyzing training data, but it is constructed by an expert human. In this case, the model is usually composed of a set of manually developed specifications that capture *legitimate* system behavior. Note that, if specifications are complete enough, the system would be capable to detect illegitimate patterns of behavior. In the best case, the false alarm rate is minimized because this kind of models avoids the problem of harmless activities that has not been previously seen. This is an unavoidable problem in learning-based approaches, in which the notion of normality is obtained exclusively by analyzing training data.

Specifications are provided by using any kind of formal tool. For example, FSMs (Finite State Machines) seem to be appropriate to model network protocols, while mere descriptions in a specific language could be useful to establish other types of legitimate behavior.

The main drawback of this approach is that development of high-quality and useful specifications is often difficult and time-consuming [32]. Nevertheless, the amount of efforts devoted to develop specifications should be conceived as a task comparable to others existing in the security field, e.g. configuration of filters in a firewall, or construction of attack signatures in misuse-based ID systems. Even if the development of specifications could be a more arduous task, its main advantage is that, theoretically, it has to be done only once.

3.3. The notion of analysis scale

Although it is not always well identified, the notion of scale of analysis is implicit in every anomaly detection method proposed until the date. In order to achieve our objective of obtaining accurate models of normality of a network, a deep comprehension of the phenomena involved in its dynamics is required. Nevertheless, there are several points of view, or *dimensions*, from which to carry out such a study.

From a mere functional perspective, the elements involved in a network can be taken into consideration. Within a simple scheme, a *service* could be identified as an atomic entity. A *host* is composed by several services, and a *network* by several hosts. Although very simplistic, this classification allows us to distinguish between those methods that try to obtain models of normality for a given service, from those that perform this task by modeling a complete host, or even the whole network.

On the other hand, if the problem is focused from the point of view of network traffic analysis, a similar notion concerning the time dimension arises. The behavior of a traffic variable over time (for instance, the number of TCP/UDP connections initiated, or the number of IP packets sent/received) can be studied in several time scales. It is a well-known fact that these traffic-related measures exhibit, under normal conditions of operation, periodic patterns that can be easily visualized. For example, if we count the number of packets received by a service during time intervals of 1 h and we plot the obtained series, a characteristic pattern of activity will be displayed. It is possible to identify diurnal and nocturne regimes in such a plot, or even more fine-grain divisions as well as weekly and seasonal profiles. In any case, the traffic model used to measure normality is related to this behavior at one or several time scales.

Finally, the notion of scale can be also identified when protocol analysis is used for anomaly detection. The inspection of individual packets can be viewed as a low-scale analysis, in contrast to the study of packet streams (medium scale), or even the simultaneous evaluation of different connections within the whole network (high scale).

The importance of the scale of analysis in anomaly detection methods lies in the fact that certain anomalies/attacks are only observable at certain scales. For instance, the old attack known as ‘land’ is characterized simply by an IP packet in which both source and destination addresses are equal [42]. This attack can be easily detected by inspecting each packet separately. On the other hand, detection of certain forms of DDoS attacks usually requires some kind of correlation or aggregation mechanism among different sources and connections, since the inspection of individual packets does not reveal any sign of anomaly.

Based on this discussion, several analysis levels, or *scales*, can be proposed to classify anomaly detection methods and, hence, the kind of anomalies/attacks that they are capable to detect. A simple scheme, in which three scales are considered, is the following:

☐ *Microscale (μ-models).* Methods based on the analysis of low-level features. According to the feature modeled, examples of low scales of analysis are:
  - Analysis of individual packets, in the case of protocol analysis.

- Traffic analysis during short periods of time (e.g. <1 s).
- Traffic analysis destined to a specific service within a host.

☐ *Mesoscale (m-models)*. Methods based on the analysis of medium-level features. According to the feature modeled, examples of medium scales of analysis are:
- Analysis of connections or packet streams.
- Traffic analysis from several seconds to minutes.
- Analysis of traffic destined to a specific host within the network.

☐ *Macroscale (M-models)*. Methods based on the analysis of high-level features. According to the feature modeled, examples of high scales of analysis are:
- Simultaneous analysis of several connections and event correlation within the whole network.
- Traffic analysis during hours, days, months, and so on.
- Traffic analysis across all the hosts within the whole network

4. Methods based on the analysis of traffic flows

Within the context of network management, recognition and identification of anomalous behavior in a network under surveillance is often based on the experience acquired from years of work in such issues. Currently, there exists a broad spectrum of tools to help in this process, albeit the human factor is not completely removed in none of them. As defined in Ref. [38], an IP flow level data is a unidirectional series of IP packets of a given protocol traveling between a source and a destination IP/port pair within a certain period of time. Observed properties in traffic flows have enabled the chance of developing methods to detect anomalies present in them. The detection is carried out by identifying statistical deviations from regular patterns of activity. Despite anomaly detection for management purposes is not the focal point of this work, several works related to failure detection have been successfully applied to detect certain attacks whose result is, in particular, to produce a network failure (DoS is a typical example).

Since flows are essentially temporal series that exhibit regular patterns, methods based on stochastic processes and signal analysis have been applied in order to characterize them and, therefore, detect anomalous behavior. Until now, these methods have demonstrated efficacy to identify those attacks that generate anomalies in the patterns of normal flows, for example, due to the generation of an unusual amount of packets. The examples discussed below illustrate these techniques more deeply.

4.1. Case study I: CRM model (Cabrera–Ravichandran–Mehra)

The work of Cabrera et al. in Ref. [8] intends, according to the authors, not to construct a classifier of network traffic but to evaluate the discriminating capabilities provided by certain statistical measures to classify traffic flows. The motivation for this approach comes from the following observation: certain attacks, like DoS and port scans, cause an increase in the total number of connections initiated during a given time interval.

Let us suppose that T is a time scale factor (for example $T=1$ h, or $T=1$ min), and let us consider the following definitions:

☐ $X_N^T(k)$: Total number of normal connections which were initiated in the interval $[kT, (k+1)T]$.

☐ $X_A^T(k)$: Total number of attack connections which were initiated in the interval $[kT, (k+1)T]$.

Several works, like those of Paxson and Floyd [23–25], have revealed the presence of regular patterns in the behavior of services like *telnet* or *ftp*. For example, connection arrivals can be well-modeled by Poisson processes. On the other hand, and as was pointed out in Section 1.1, there exists a large amount of network attacks that produce anomalies in certain network *invariants*. Based on these and other studies, authors analyze the evolution of the two previously exposed measures applied to the dataset used during the DARPA Intrusion Detection Evaluation Program [18,19]. The main conclusions can be summarized in two observations:

(1) The normal connections ($X_N^T(k)$) follow three operating regimes: a day regime, an evening regime, and a night regime. Moreover, the day and night regimes seems to adjust to Poisson processes.
(2) Attack connections ($X_A^T(k)$) present heterogeneous behavior. The DoS and scan attacks appear in bursts, while other forms of attack appear on a single connection.

Based on these experimental results, the authors empirically explore the discriminating power of the total number of initiated connections with the aim of detecting attacks. Several models are studied by estimating the probability density functions for each regime and setting thresholds as basic detection mechanism. The well-known Kolmogorov–Smirnov test is used to achieve this purpose. The results obtained are between 60 and 85% of correct detections for day regime with no false alarms, and inferior in the case of night regime. The design of detection techniques for the evening regime is not addressed in that work.

These measures, called *network models* by the authors, are complemented with *application models*, in which several variables for each connection are studied with the aim of detecting attacks different in behavior from those exposed. These variables include the number of bytes interchanged during the connection (number of bytes from the originator and number of bytes from the responder), the connection duration, the status flag, etc. for the case of the *telnet* service. After several empirical studies and statistical comparisons, the authors conclude that there is a strong evidence that attack and normal connections corresponds to different underlying processes. The detection rates obtained show that certain measures provide about the same discrimination capabilities as the best results in the DARPA Evaluation.

4.2. Case study II: signal analysis of network traffic

In the work of Barford et al. [20], results concerning signal analysis of several classes of network anomalies are reported. The approach, based on the use of wavelet filters, is applied to datasets consisting of six months of IP flows and SNMP measurements collected on a large university network. In an earlier work, authors identified several similarities and differences within a number of anomalies [21]. Their objective was not merely to cluster anomalies but to try to characterize common properties exhibited by each group. Identification of such invariants attributes is an important milestone in order to subsequently detect anomaly behavior by using automated methods.

After a visual analysis, three broad categories of anomalies were pointed out by authors:

- ☐ *Network Operation Anomalies*, which includes devices outages, modifications in network behavior caused by configuration changes, etc.
- ☐ *Flash Crowd Anomalies*: rapid rise in traffic flows of a particular type or to a well known destination. They are caused by a sudden interest in a specific service (for instance, a new software patch in a repository server or a highly interesting content in a Web site).
- ☐ *Network Abuse Anomalies*, mainly DoS flood attacks and port scans. In some forms, they are different from the former two in that they are not always detectable by inspecting bit or packet flows. Particularly, port scans usually generates many distinct address/port pairs and, therefore, many flows.

Even though the two first classes of anomalies have an undoubted interest, from the point of view of this work only the last will be commented. The nature of DoS flooding attacks has been widely studied and, besides the work of Barford et al., other like that of Moore et al. [22] has shown that flow data can be effective for identifying them.

As was mentioned above, the proposed method is based on the application of wavelet analysis to the measurement string, treating it as a signal and ignoring any semantics (such as packet headers). At this point, we forgo explaining the theoretical foundations of wavelet analysis, and interested readers can seek advices directly from the original work in Ref. [20]. The key to the detection of anomalies by using this mathematical tool lies in its inherent time–frequency analysis. This property allows dividing the signal into different components at several frequencies. In the case of traffic flow, low frequency-part corresponds to patterns of very long duration, like several days. Mid frequency-part captures daily variations in the data, and high frequency-part consists of short-term variations. In order to obtain these three components, wavelet coefficients are grouped into three intervals and signals are subsequently synthesizing from them.

Once obtained the long-term, medium-term and short-term patterns of behavior, an algorithm termed *deviation score* is proposed to automatically identify regularities in data. By the very nature of short-term anomalies, such as several forms of DoS attacks and port scans, they are detected within mid-band and high-band components. By separating these parts from long-term behavior, the decomposition facilitates to easily visualize these anomalies as isolated peaks in the upper bands. Subsequently, detection by setting thresholds can be performed.

Without the intention to unfavorably compare this work, its application to anomaly detection methods in network security is limited to only some types of attacks. Despite the attractive approach proposed and the high-quality results obtained, this and other related methods seem more accurate for fault detection than attack identification. DoS flooding and certain forms of port scans are detected due to the inherent anomalous alterations generated in patterns of activity. Nevertheless, low-frequency scans and other forms of DoS attacks do not generate such patterns, albeit their behavior is obviously anomalous. In these cases, other methods should be employed to detect them.

5. Anomaly detection based on protocol analysis

Network protocols are one of the fundamental pieces of networking since they are involved in every information interchange throughout the network. Each protocol is carefully designed to support a specific facet of the communication process, so that devices and applications use it according to an established set of formal rules. Nevertheless, protocol specifications usually suffer ambiguities that enable its use in several ways that go beyond those for which it was thought up. Some of these uses include communications that carry out malicious activities, from powerful intelligence gathering techniques to the most destroyer denial of service attacks. In addition to this, several implementations are not fully conformant with the recommendations and might introduce malformed packets in the network (for example, a list of common implementation problems in TCP is provided in RFC 2525 [37]).

Misuse in protocol utilization is certainly one of the most significant characteristics of several forms of hostile traffic, and its detection is crucial in any network-based anomaly detector. Among the broad spectrum of methods proposed to achieve this purpose, in the following sections several representative examples are commented. Because of their special interest, application-layer protocols are treated separately in Section 5.1.

5.1. Case study III: SPADE/SPICE

SPADE (Statistical Packet Anomaly Detection Engine) was initially developed as a module included in SPICE (Stealthy Portscan and Intrusion Correlation Engine) [1] detection system. Since 2001, it is also available for Snort™ system [35,36]. It was the first approach that proposed the idea of assigning to each single packet an *anomaly score*, in contrast to previous approaches based on analysis of traffic windows or complete sessions.

The anomaly score of a packet is a number which measures its degree of strangeness based on the recent history of the network. The scheme is conceptually simple and consists in a frequency-based mechanism: the fewer times a particular packet has been observed, the higher its anomaly score will be. In order to do this, the core of the model is composed by a probability table that maintains the occurrences of different kinds of packets along history. Classification of packets into different kinds is carried out by using their joint occurrence of certain packet header field values. In addition to this, the network history is weighted with the aim of providing more importance to more recent events.

Within this context, functions of SPADE are basically three:

- ☐ To upgrade periodically the model (probability of occurrence table) as new traffic circulate across the network.
- ☐ To compute the anomaly score for each packet.
- ☐ Packets that receive an anomaly score higher than a fixed defined threshold are forwarded to SPICE. In its origin, SPICE was designed to detect port scans. This task is achieved by a correlation module which combines packets assumed to belong to the same scan.

5.2. Case study IV: PHAD, ALAD, LERAD, and NETAD

The work of Mahoney et al. in Refs. [2,3,5,40,41] presents several methods that address the problem of detecting anomalies in the usage of network protocols by inspecting packet headers. The common denominator of them all is the systematic application of learning techniques to automatically obtain profiles of normal behavior for protocols at different layers. These works constitute a remarkable effort in the study of anomaly-based systems.

PHAD (Packet Header Anomaly Detector, [2]) is a system that learns the normal ranges of values for packets header field at the data link (Ethernet), network (IP), and transport/control layers (TCP, UDP, and ICMP). During the training period, if a packet field is observed n times with r distinct values, then there must have been r ‘anomalies’. Thus, if this behavior continues, the probability that the next observation will be anomalous is estimated by using r/n. In addition to this, to take into consideration the dynamic nature of real-time traffic, PHAD weights the probability of occurrence of an event by the last time it occurred. Put simply, if an event last occurred t time units ago, then the probability that it will occur in the next time unit is approximated by $1/t$. Thus, each packet header field containing an anomalous value (i.e. a value not previously seen during the training stage) receives a score given by tn/r. Likewise, the scores of individual fields are added up to obtain the total packet score.

As stated, only anomalous fields receive a score. The notion of normal value of a field is obtained during the training stage by constructing lists of observed values for each of the 33 fields examined by PHAD. Since storing all the different observed values is prohibitive due to memory costs, an approach based on storing ranges instead of isolated values is proposed. A maximum limit, C, for lists of ranges is fixed, in such a way that if C is exceeded, then the method finds the two closest ranges and merges them. On the 1999 DARPA ID Evaluation, authors obtain a detection ratio of 72 over 201 attack instances with a rate of 10 false alarms per day. Additionally to this, several variants of the proposed method are in-depth studied and commented by authors in the same work.

ALAD (Application Layer Anomaly Detection, [3]) is a system designed to assign an anomaly score to TCP incoming connections to well known server ports. After a testing of a number of attributes and combinations of them, authors select five as those which provide the best performance:

- ☐ *P(source IP address/destination IP address)*. Note that there is a separate model for each local IP address. Only models for TCP ports <1024 are considered.
- ☐ *P(source IP address/destination IP address, destination TCP port)*. This is similar to the previous, but it distinguishes among different servers within the same host.
- ☐ *P(destination IP address, destination TCP port)*. It models the set of local servers which usually receive requests.
- ☐ *P(TCP flags/destination TCP port)*. It models the set of normal TCP flags for the first, next to last, and last packet of a connection. Note that it is a separate model for each service.
- ☐ *P(keyword/destination TCP port)*. This is an attempt to include application-layer information into the model. Thus, ‘keyword’ corresponds to the first word in the payload (i.e. the text between a linefeed and the following space).

These probabilities are estimated during the training stage and used in detection mode to obtain the anomaly score of packets and connections. After a joint evaluation of PHAD and ALAD, authors reach the conclusion that the detection coverage of both systems has practically no overlap. This fact means that anomalies (i.e. attacks) are present at different network layers and, therefore, its detection is successfully performed if the corresponding layer is modeled. The idea that classifying network attacks according to the network layer they exploit could be a more useful taxonomy than that used in the 1999 DARPA/MIT Lincoln Labs ID Evaluation was previously pointed out in Ref. [6], and these results confirm it.

LERAD (LEarning Rules for Anomaly Detection, [3]) constitutes an improvement of the two previous approaches by using a rule-learning algorithm. Two key ideas present in PHAD and ALAD are maintained: inclusion of a large number of attributes related to packet header, and the use of a non-stationary model, in which the probability of a field having some value depends on the time of its most recent occurrence, and not on its average frequency.

The idea behind the introduction of rules is to take into consideration the joint probability of occurrence of values within a packet header; that is, general rules that assign a probability to a set of attributes given that another set has certain values. LERAD monitors the same type of TCP connections than ALAD. Nevertheless, LERAD uses a learning algorithm to obtain rules, and also monitors an extended set of attributes.

Finally, NETAD (NEtwork Traffic Anomaly Detector, [40]) is another approach based on the consideration of each byte within the packet as an attribute with 256 possible values. The first 48 bytes of the packet starting with the IP header are modeled. To be precise, this system identifies nine common packet types and a different model is estimated for each of them. Examples of these packet types are: TCP SYN packets, TCP ACK packets to port 21 (FTP), TCP ACK packets to port 80 (HTTP), all IP packets, etc. Of course, a packet may belong to more than one type. In addition to this, several improvements are introduced concerning the calculation of the anomaly score for each packet. Although it can seem a very rudimentary method, the evaluation carried out by authors point out that NETAD performs quite well.

Interested readers can find in Ref. [5] an evaluation of the four methods commented before. Again, the 1999 DARPA/MIT Lincoln Labs ID Evaluation data sets are used as benchmarking framework and, among others conclusions, authors suggests the presence of artifacts, as was previously stated by McHugh in the critique presented in Ref. [6].

5.3. Case study V: specification-based protocol anomaly detection

Contrary to the previous approaches commented until now, anomaly detection is neither strictly linked with statistical notions of normality nor models learnt by using machine learning techniques or any other kind of estimation algorithms. An alternative approach is that known as 'specification-based models', and whose basic foundations were briefly described in Section 3.2. According to the criterion 'behavior model' provided in the taxonomy used along this work, in specification-based detectors the model is not estimated by analyzing training data, but it is constructed by an expert human. In this case, the model is usually composed of a set of manually developed specifications that capture *legitimate* system behavior.

In Ref. [32], Sekar et al. present this approach as well as a prototype with excellent detection performance. The model proposed by authors consists of developing protocol specifications by using *Extended Finite State Automata* (EFSA). An EFSA is basically a finite-state automaton extended to support two added functions: (a) transitions on events that may have arguments; and (b) state variables in which values can be stored. The specification is done through a special language designed for that purpose.

The aim of each specification is to monitor protocol behavior, like IP or TCP. For each initial protocol instance received, a new state machine is created an added to a list of active monitors. When a machine reaches the final state, it is deleted from the list. The monitoring task itself is performed by associating each received packet to a transition between states and verifying that conditions for such a transition are fulfilled. Thus, every communication leaves a trace along the machine, characterized by the sequence of states (remember that each state includes as well the values of state variables). These traces are subsequently used to acquire statistical properties of the protocol stream in terms of the frequency that a given transition has and the distribution of values of each state variable. Based on these learnt statistical properties, several kinds of attacks supported by protocol misuses can be detected.

Specifications for IP and TCP are provided by authors and tested with datasets from 1999 DARPA Intrusion Detection Evaluation Program [18,19]. It is showed how the most relevant attacks are detected within this approach: three attacks in the case of IP machine, and eight with TCP. An important point here is that while one machine detects a subset of attacks, the other detects another, disjoint subset. This corroborates the well-known fact that misuses are frequently protocol-specific and, therefore, a complete supervision of each protocol used in the networking environment is required.

As can be easily viewed, the proposed approach combines both specifications of correct patterns of behavior together with learning statistical notions of regularity. This technique seems to provide accurate detection of certain attacks while the false alarm rate is contained at a low level (5.5 per day on the average). Readers who wish to go more deeply into specific details are encouraged to inspect the original work in Ref. [32].

6. Application-layer anomaly detection: payload inspection

The vast majority of the research carried out in anomaly-based network IDSs through protocol analysis has been focused on the inspection of packet header information at the data link (Ethernet, for example), network (IP), and transport/control layers (TCP, UCP, ICMP, etc.). The basic idea behind the notion of application-layer anomaly detection is to extend these models of normality from analyzing only packet header data at network and transport/control layer to include application-layer payload as well [7].

Justification for this approach comes from the inability of the previous approaches to detect attacks on servers, like DNS, HTTP, FTP, or SMTP. Although still immature, traffic modeling and protocol analysis technologies have proved their capability to detect several forms of anomalies when DoS attacks, port scans and other malicious network traffic is launched against the monitored system. Nevertheless, this situation dramatically varies when other kinds of attacks are considered, especially those known as Remote-to-Local (R2L). This term was coined during the DARPA Intrusion Detection Evaluation Program developed at MIT Lincoln Labs [18,19], and it is generally defined as intrusion attempts from remote users with the aim of getting unauthorized local access to the target host. During the 1999 evaluation, less than 10% of not previously seen R2L attacks were detected by ID systems.

In order to understand the reason for this inaccuracy, the nature of R2L attack must be examined. These intrusion attempts are usually achieved by exploiting vulnerabilities in services offered by the destination host (e.g. buffer overflows). The basic characteristic of such attacks, which make almost impossible that formerly exposed systems were able to detect them, is the fact that anomalies are located in the packet payload, i.e. information destined to the server. Low-level packet features are usually normal and correct from a security point of view, in such a way that an analysis of network or transport layer contents does not reveal any sign of anomaly in traffic.

In an earlier work, Krügel et al. [7] established the necessity of separately analyzing the payload of IP packets according to the application that has created or that will receive them. This *service specific* approach, which also implies a different notion of normal traffic for each different service, seems to be a basic assumption in application-layer anomaly detection due to the diverse nature of payload structure depending on the particular service the data is destined to. Due to this reason, specific knowledge of the monitored service is required for the construction of accurate models of behavior.

6.1. Case study VI: KTK model (Krügel–Toth–Kirda)

As the majority of proposed systems in anomaly detection, this model operates in two separate modes: training and detection. The analyzed data from which the normal behavior is obtained is, however, novel in relation to previous approaches. The technique proposed in Ref. [7] performs the detection of anomalies by learning the normal behavior of application-layer protocol instances, i.e. payloads. To be precise, HTTP and DNS services are studied in the mentioned work.

Due to the nature of lower layer protocols in our networking context (mainly IP and TCP/UDP), data destined to a server can be fragmented into a series of pieces that travel separately across the network. An attacker can distribute a malicious payload over several datagrams, in such a way that a system that operates by examining isolated packets could not be able of detect it. Because of this reason, and since the purpose is to analyze complete requests to a service, authors include a module termed PPU (Packet Processing Unit). Its main function is to read the network traffic and reassemble packets containing the same request to a service offered by a server. At this point, authors distinguish among different types of available services in a server. For example, in the case of HTTP it is possible to mention GET, POST or HEAD as frequently used type of services.

The second module of the proposed system is the SPU (Statistical Processing Unit). It processes groups of normal requests with similar statistical properties and obtains a model of normal behavior for each group. In the case of HTTP, GET requests are very similar and could be grouped intro a unique cluster. Nevertheless, POST requests may present significant differences due to the nature of the HTML form and the data typed by the user. Currently, those groups are done manually by an expert.

The model of normal behavior for a given group is constructed by analyzing three features of the requests contained in it: the likelihood of the type of request, the length of the request and the payload distribution. The choice of these features is justified by an analysis of the properties of several R2L attacks. Thus, the type of request is included because several exploits based on buffer overflows and other input validation errors use a network service that is rarely used. In the case of the length of the request, most of the payloads carrying inputs to overflow a buffer in the target service present a considerable longer request (it has to include the shell-code itself and additional padding). On the other hand, the lengths of normal requests are often quite similar. Finally, experiments carried out by the authors prove that normal requests have a very similar PDF (Probability Density Function). This fact is easily explained if two reasons are considered: (a) HTTP and DNS requests mainly contain characters and human readable strings, i.e. a small subset of all 256 possibilities in the ASCII code; and (b) characters are not uniformly distributed.

In the scheme proposed, a model of normality is obtained for each of the above mentioned features. During the detection stage, an *anomaly score* is computed for the evaluated request. This number measures the amount of variability found between the analyzed request and the normal behavior. For this purpose, several statistical techniques are used. The interested reader can consult them in the original work [7].

Regardless of the specific, formal background used to measure normality and anomaly, the most important contribution of this work is surely the problem statement (i.e. service-specific anomaly detection) together with the accurate identification of request features that remain statistically invariant except in the presence of certain forms of hostile traffic.

7. Some notes on the state of the art

Despite the wide list of methods carefully developed and proposed during the last twenty years, the fact is that anomaly detection is still an immature technology The lack of consolidated commercial tools based on it is surely the most convincing evidence for realizing that more research efforts are required. The purpose of this closing section is not to provide an exhaustive list of current open problems, but only to point out several research lines that deserve attention because of their relevance.

Some issues concerning performance limitations are not addressed here, although they have an undoubted importance. High-speed networks introduce non-trivial problems for monitoring tasks, and efficiency is a key point of every method that hopes to work in a real environment.

7.1. Evaluation methodologies

Along this paper, the 1999 DARPA Intrusion Detection evaluation performed at Lincoln Labs (MIT) has been cited several times In Ref. [6], McHugh carried out a critique of that evaluation with the aim of identifying its major shortcomings. For example, the methods employed to generate the data used for the evaluation seem to be inadequate, since synthetic data are not appropriate to simulate real environments.

In 2000, the Lincoln Adaptable Real-Time Information Assurance Test-bed (LARIAT) program was initiated with the aim of providing a benchmarking environment [34]. Unfortunately, this platform is not available for use to the general public. In any case, the common methodology used by researchers to test and evaluate ID methods in general, and anomaly detection techniques in particular, is to create a simulated network with background traffic interlaced with malicious activity. Nevertheless, and as was pointed out in Ref. [33], this ad-hoc methodology suffers from several limitations. For instance, questions as the network architecture, its complexity or capabilities are totally unspecified, although it is nevertheless true that no recommendations appear to exist.

In a profound revision of the field of testing and benchmarking methodologies for ID, Athanasiades et al. [33] concludes that it is clear that present approaches are inadequate. In spite of the great contributions of efforts like the above mentioned, new common frameworks and methodologies are required to improve the existing practices employed in the field. More capable tools are needed, which be able to generate adequate network background traffic as well as launch appropriate attacks.

7.2. The cryptographic challenge

One of the most worrisome points in anomaly detection is the clash of interests between two *seemly* opposite mainstays of network security: the tend to add more and more ciphering to network traffic in order to provide privacy to communications, and the increasing requirement to capture and access traffic content with the aim of analyzing it in search of security violations In fact, this challenge is neither new nor specific of the anomaly detection approach. It underlies in all mechanisms which require plain access to certain packet fields that are carried throughout the network, secured under the protection provided by cryptographic codes. For example, all signature-based ID systems are also subject to this constraint.

The naive solution to this strong drawback is to move the analysis location into the protocol stack, specifically above the layer responsible for deciphering the content. This approach does not allow performing the analysis of traffic before it has reached the destination host, for example, at a border gateway or router, being thus a very poor solution. On the other hand, the use of local, intermediary elements to manage keys arises as an alternative approach. Thus, this component (let us say, a kind of key server) will carry out all the tasks concerning key management for applications inside the monitored network. By using authentication mechanisms, traffic inspectors could retrieve appropriate keys from it and use them to decipher each packet and analyze its contents. This approach presents several drawbacks that limit its application. For instance, it disables the end-to-end nature of nearly all existing privacy solutions, like IPSec. Furthermore, and regardless of performance considerations, such an approach will require that every application that uses any type of cryptographic protection shares the involved keys with the server. This last is far from being a trivial problem.

7.3. Inherent limitations of the anomaly detection approach

Despite the discussion presented along this paper, it is important to note two main subjects regarding the relationship between anomalies and attacks First, the fact that *most* hostile traffic could be labeled as anomalous events does not imply that *all* attacks belong to that category. The existence of normal traffic which can carry out malicious actions from a security point of view limits the application of this kind of methods.

On the other hand, the very definition of 'anomaly' implies several consequences when its role in network management and supervision is considered. Addition of new services, or in more general terms, changes in the network configuration will produce that the anomaly detector raise alarms unless the model of normality is adapted and new events are recognized as normal. Although this limitation is inherent to the very nature of the anomaly detection problem, administrators have to deal with it, which can become a serious constraint in constant evolving environments.

7.4. On the nature of attacks and anomalies in networks

Anomaly detection has been typically related to statistical notions of normality. Put simply, 'normal' has been conceived as 'frequent'. Even though more powerful statistical methods could not have been explored yet, formulation of alternative notions of normality could provide great benefits. The question here can be easily formulated as: *should 'normal' behavior be always conceived as 'frequent' behavior?* From a security point of view, it is not difficult to imagine scenarios in which both concepts are not strictly linked; that is, correct operations that are rarely observed (simply because users/systems do not carry out them), and also frequent security violations, for example, due to the administrator's permissiveness or the lack of security policies.

The ideal solution would be the discovery of what can be termed *attack invariants*. An attack invariant can be defined as a property of some network facet (e.g. traffic volume) which is always verified, except in the presence of an attack. Evidently, a better understanding of the nature of anomalies, and especially those related with network attacks, is undoubtedly required in order to improve current detection mechanisms in such direction. This requirement is especially relevant since the most important challenge in anomaly detection is the choosing of features to be modeled. Such features must accurately characterize the service, system or network utilization patterns in order to obtain a precise model of the normal behavior. But at the same time, they must suffer a strong enough change when the observed activity is completely different from those behaviors labeled as normal. Even if it would be possible to identify every parameter involved in their behavior, complexity of current networks hinders a brute-force analysis. Furthermore, the detection of anomalies which could have security implications is increasingly complex since anomalous behavior seems to exist in networks by default.

7.5. Recommendations toward improved methods

Based on the methods surveyed along this work, this analysis concludes by identifying a list of desirable properties which could considerably improve the detection quality of current methods.

- ☐ *Analysis at several scales*. Concerning the scale of analysis, several methods based on protocol analysis (e.g. some of the previously commented) perform the detection by analyzing the fields within each protocol instance. Instead, the approach here proposed is focused mainly on properties of packet sequences. Since there exist attacks that exploit both features, it would desirable to develop methods capable to simultaneously work in both scales. Otherwise, the spectrum of successfully detected attacks will be biased.
- ☐ *Hybrid models: inclusion of specifications and specific knowledge*. From specification-based approaches to application-layer detectors, more and more knowledge is included in each new method proposed with the aim of reducing the false alarm rate. 'Blind', isolated methods (e.g. pure statistical analysis or data mining techniques) do not seem to work well because probably there is not enough information in data to infer what normal is. Additional knowledge, like that explicitly provided by specifications or implicitly included within the detection method, is required to capture the essential nature of normal, correct behavior.
- ☐ *Anomaly-specific detectors*. It seems clear that it is possible to classify a large amount of the existing attacks according to the network traffic feature which exploits, or on which is based. Thus, several attacks are well detected with flows analysis because they cause a visible change in patterns of normal behavior. Methods based on protocol analysis have proven their accuracy dealing with attacks based on malformed packets and related misuses. Surely, a tool based on the existence of several, heterogeneous detectors will obtain better results than each of them working separately.

8. Conclusions

Networks are becoming increasingly complex at the same time that security concerns do not cease to grow and require more and more attention and resources. With the aim of ensuring that a network system works appropriately, constant monitoring is essential. But performing a suitable, manual inspection of activities has turned into an unachievable task since several years ago. Among the broad spectrum of methods and tools available to support this necessity, anomaly detection has proven to be a very valuable approach for network management as well as network security.

In this work, an analysis of the state of the art in the field of anomaly detection for network intrusion detection has been presented. Far from being a mere, commented list of developed systems, the exposition has been articulated upon a proposed taxonomy in which the most relevant features of current solutions are included. Thus, the network feature analyzed, the type of behavior model, and the scale of analysis have been proposed as basic criteria to classify current methods as well as key notions to the problem itself. Subsequently, representative prototypes of each paradigm have been briefly discussed. Albeit many distinguished systems have not been mentioned here due to space reasons, the case studies carried out provide a sound picture of the state of the art.

To conclude the survey, several recommendations with the aim of improving current anomaly detection methods have been pointed out. In any case, anomaly detection is still a hard problem in several domains. Regardless of the current needs, a deeper knowledge is required until this technology achieves a solid maturity for its implantation in environments increasingly changeable at the same time that strongly threatened.

Acknowledgements

This work has been partially supported by Spanish Government (MECD) under National Program PNFPU with reference AP2001-3805, and Spanish MCYT under project TIC2002-02798 (FEDER funds 70%).

References

[1] S. Staniford, J.A. Hoagland, J.M. McAlerney, Practical automated detection of stealthy portScans, IDS Workshop of the Seventh Computer and Communications Security Conference, Athens, Greece 2000.

[2] M.V. Mahoney, P.K. Chan, PHAD: packet header anomaly detection for identifying hostile network traffic, Florida Institute of Technology Technical Report CS-2001-4 2004.

[3] M.V. Mahoney, P.K. Chan, Learning nonstationary models of normal network traffic for detecting novel attacks, Proceedings of the Eighth International Conference on Knowledge Discovery and Data Mining 2002; 376–385.

[4] R.A. Kemmerer, G. Vigna, Intrusion detection: a brief history and overview, IEEE Computer 35 (4) (2002) 27–30.

[5] M.V. Mahoney, P.K. Chan, An analysis of the 1999 DARPA/Lincoln Laboratory evaluation data for network anomaly detection, Florida Institute of Technology Technical Report CS-2003-02 2003.

[6] J. McHugh, Testing intrusion detection systems: a critique to the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory, ACM Transactions on Information and System Security 3 (4) (1999) 262–294.

[7] C. Krügel, T. Toth, E. Kirda, Service specific anomaly detection for network intrusion detection, Proceedings of the 17th ACM Symposium on Applied Computing (SAC), Madrid, Spain 2002; 201–208.

[8] J.B.D. Cabrera, B. Ravichandran, R.K. Mehra, Statistical traffic modeling for network intrusion detection, Proceedings of the Eighth IEEE International Symposium on Modeling, Analysis and Simulation of Computer Telecommunication Systems 2000; 466–473.

[9] J. Allen, A. Christie, W. Fithen, J. McHugh, J. Pickel, E. Stoner, State of the practice of intrusion detection technologies, Technical Report CMU/SEI-99-TR-028, Software Engineering Institute, Carnegie Mellon University, 2000.

[10] J.P. Anderson, Computer security threat monitoring and surveillance, Technical Report, James P. Anderson Co., Fort Washington, 1980.

[11] D. Denning, An intrusion detection model, IEEE Transactions of Software Engineering SE-13 (2) (1987) 222–232.

[12] J. Hochberg, K. Jackson, C. Stallings, J.F. McClary, D. DuBois, J. Ford, NADIR: an automated system for detecting network intrusion and misuse, Computers and Security 12 (3) (1993) 235–248.

[13] P.A. Porras, P.G. Neumann, EMERALD: event monitoring enabling responses to anomalous live disturbances, 20th NIS Security Conference, October 1997.

[14] T.F. Lunt, A. Tamaru, F. Gilham, R. Jagannathan, P.G. Neumann, C. Jalali, IDES: a progress report, Sixth Annual Computer Security Applications Conference, Tucson, AZ, USA, December 1990.

[15] D. Anderson, T. Frivold, A. Tamaru, A. Valdes, Next Generation Intrusion Detection Expert System (NIDES), SRI International, 1994.

[16] S. Forrest, S.A. Hofmeyr, A. Somayaj, T.A. Longstaff, A sense of self for unix processes, IEEE Symposium on Research in Security and Privacy 1996; 120–128.

[17] C. Warrender, S. Forrest, B. Pearlmutter, Detecting intrusions using system calls: alternative data models, Proceedings of IEEE Symposium on Security and Privacy 1999; 133–145.

[18] R. Lippmann, J.W. Haines, D.J. Fried, J. Corba, K. Das, The DARPA off-line intrusion detection evaluation, Computer Networks 34 (4) (1999) 579–595.

[19] MIT Lincoln Labs, DARPA Intrusion Detection Evaluation 1998, http://www.ll.mit.edu/IST/ideval.

[20] P. Barford, J. Kline, D. Plonka, A. Ron, A signal analysis of network traffic anomalies, Proceedings of ACM SIGCOMM Internet Measurement Workshop 2002.

[21] P. Barford, D. Plonka, Characteristics of network traffic flow anomalies, Proceedings of ACM SIGCOMM Internet Measurement Workshop, San Francisco, CA, November 2001.

[22] D. Moore, G. Voelker, S. Savage, Inferring internet denial-of-service activity, Proceedings of USENIX Security Symposium, Washington, DC, USA, August 2001.

[23] V. Paxson, Why Understanding Anything About the Internet is Painful Hard, AT&T Center for Internet Research at International Computer Science Institute, Berkeley, CA, 1999.

[24] V. Paxson, Empirically-derived analytic models of wide-area TCP connections, IEEE/ACM Transactions on Networking 2 (4) (1994) 316–336.

[25] V. Paxson, S. Floyd, Wide-area traffic: the failure of Poisson modeling, IEEE/ACM Transactions on Networking 3 (3) (1995) 226–244.

[26] S.M. Bellovin, Packets Found on an Internet, Computer Communication Reviews 23 (3) (1993) 26–31.

[27] M. Bykova, S. Ostermann, B. Tjaden, Detecting network intrusions via a statistical analysis of network packet characteristics, Proceedings of the 33rd Southeastern Symposium on System Theory, March 2001; 309–314.

[28] P. Criscuolo, Distributed denial of service, Trin00, tribe flood network, tribe flood network and stacheldraht, Computer Incident Advisory Capability (CIAC) CIAC-2319 (2000).

[29] CERT® Coordination Center, CERT® Advisory CA-1998-01 Smurf IP Denial-of-Service Attacks, Available at: http://www.cert.org/advisories/CA-1998-01.html

[30] CERT® Coordination Center, CERT® Coordination Center Statistics 1988–2003, Available at: http://www.cert.org/stats/cert_stats.html

[31] C.E. Landwehr, A.R. Bull, J.P. McDermott, W.S. Choi, A taxonomy of computer program security flaws, ACM Computing Surveys 26 (3) (1994) 211–254.

[32] R. Sekar, A. Gupta, J. Frullo, T. Shanbhag, A. Tiwari, H. Yang, S. Zhou, Specification-based anomaly detection: a new approach for detecting network intrusions, Proceedings of the Ninth ACM Conference on Computer and Communications Security, Washington, DC, USA, November 18–22, 2002; 265–274.

[33] N. Athanasiades, R. Abler, J. Levine, H. Owen, G. Riley, Intrusion detection testing and benchmarking methodologies, Proceedings of the First IEEE International Workshop on Information Assurance (IWIA), Darmstadt, Germany, March 2003; 63–72.

[34] L.M. Rossey, J.C. Rabek, R.K. Cunningham, D.J. Fried, R.P. Lippmann, M.A. Zissmann, LARIAT: Lincoln adaptive real-time information assurance test-bed, Presentation in International Symposium on Recent Advances in Intrusion Detection (RAID 2001), October 10–12, 2001, Davis, CA 2001;.

[35] M. Roesch, Snort: lightweight intrusion detection for networks, Proceedings of the USENIX LISA Conference, November 1999.

[36] Snort™, The Open Source Network Intrusion Detection System, Available at: http://www.snort.org

[37] V. Paxson, M. Allman, S. Dawson, W. Fenner, J. Griner, J. Heavens, K. Lahey, J. Semke, B. Volz, Known TCP Implementation Problems, RFC 2525 1999 http://www.ietf.org/rfc/rfc2525.txt.

[38] K. Claffy, G. Polyzos, H.W. Braun, Internet traffic flow profiling, Technical Report TR-CS93-328, University of California at San Diego, 1989.

[39] Z. Zhang, C. Manikopoulos, J. Jorgenson, Architecture of generalized network service anomaly and fault thresholds, Proceedings of MMNS, Lecture Notes in Computer Science 2216 2001; 241–255.

[40] M.V. Mahoney, Network traffic anomaly detection based on packet bytes, Proceedings of the 18th ACM Symposium on Applied Computing (SAC), Melbourne, FL, USA 2003; 346–350.

[41] M.V. Mahoney, P.K. Chan, Detecting novel attacks by identifying anomalous network packet headers, Technical Report CS-2001-2, Florida Institute of Technology, 2001.

[42] CERT® Coordination Center, CERT® Advisory CA-1997-28 IP Denial-of-Service Attacks, Available at: http://www.cert.org/advisories/CA-1997-28.html

Juan M. Estévez-Tapiador is a PhD candidate in the Department of Electronics and Computer Technology of the University of Granada, Spain. He received an MS in Computer Sciences and Engineering from the University of Granada, where he obtained the 'Best Student' Academic Award. Currently he is holding a PhD grant from the Spanish Ministry of Education. Since 2000, he is a member of the 'Research Group on Signals, Telematics and Communications' of the University of Granada, where he has been involved in several public and private research projects. His research is focused on computer and network security, especially in intrusion detection and response systems, new security paradigms and group communications security.

Pedro García-Teodoro received his BSc in Physics (Electronics speciality) from the University of Granada, Spain, in 1989. This same year he was granted by 'Fujitsu España', and during 1990 by 'IBM España'. Since 1989 he is Associate Professor in the Department of Electronics and Computer Technology of the University of Granada, and member of the 'Research Group on Signal, Telematics and Communications' of this University. His initial research interest was concerned with speech technologies, especially automatic recognition, field in which he developed his PhD Thesis in 1996. From then, his profile has derived to that of computer networks, and although he has done some works in telematics applications and e-learning systems, his main current research line is centred in computer and network security.

Jesús E. Díaz-Verdejo is Associate Professor in the Department of Electronics and Computer Technology of the University of Granada (Spain).He received his BSc in Physics (Electronics specialty) from the University of Granada in 1989 and has held a PhD grant from Spanish Government. Since 1990 is member of the 'Research Group on Signal, Telematics and Communications' of the University of Granada. This year he became Assistant Professor at this University. In 1995 he obtained a PhD degree in Physics. His initial research interest was related with speech technologies, especially automatic speech recognition. Currently he is working in computer networks, mainly in computer and network security, although he has developed some work in telematics applications and e-learning systems.