J Comput Virol (2007) 3:39–49
DOI 10.1007/s11416-007-0036-2

EXTENDED VERSION OF WTCV'06

Intrusion detection and virology: an analysis of differences, similarities and complementariness

Benjamin Morin · Ludovic Mé

Received: 8 October 2006 / Accepted: 12 January 2007 / Published online: 7 February 2007

Abstract In this paper, we analyze the differences, similarities and complementariness which exist between two major domains of nowadays information security: intrusion detection on one hand, virology and anti-viruses technologies on the other hand. This analysis is built from two points of view. First, we compare, through the definitions that have been proposed by researchers of the two communities, the goals that are actually pursued in each domain. Then, we compare the techniques that have been developed to reach these goals. In the conclusion, we summarize our analysis and suggest that alert correlation is one way to make the two fields cooperate.

1 Introduction

Research in virology and research in intrusion detection both emerged in the 1980s, but evolved separately until today. Viruses and intrusion detection are nowadays two major building blocks for information security. An anti-virus software is embedded in almost each computer, not to say in each computer hosting the Windows operating system. Intrusion detection is more restricted to the professional area. Nevertheless, intrusion detection systems (IDSes) are largely used by network administrators in small or large companies, or in research or academic environments.

Using an anti-virus software or an IDS is actually an answer to the same kind of concerns: even if preventive security tools (e.g., authentication, access control, cryptography) are deployed, it is often or even always possible, for a malicious individual, to bypass the protections. Thus, a second level of defense is mandatory, which is constituted by tools such as anti-virus softwares and IDSes. The idea is that if it is not possible to prevent attacks against our computers, at least it may be possible to detect these attacks. Then, once detected, it may also be possible, in some cases, to stop the attack.

The goals of the two domains thus appear to be pretty similar. In this case, why are the two domains separate? Is there an intrinsic difference between the two forms of attacks justifying such a separation? Presumably not. Some researchers work in both domains [18,6,19], and IDSes have already been used as anti-virus softwares: in [20], the ASAX [13] IDS is used as an expert system to detect viruses in the VIDES platform.

This paper is an attempt to position each of these domains with respect to the other. We do not claim here any new major scientific contribution. Rather, our objective is to identify similarities and differences, and see how the tools from the two domains may be integrated in a unified framework that would benefit to the final users by increasing the hosts and networks security levels.

The reader should notice that this article is written by authors from the intrusion detection field, not by computer virologists. Therefore, this paper is open to discussion, and even written to give rise to discussion. It only reflects the (potentially biased) opinion of intrusion detection specialists, although we make an effort to be impartial.

The remaining of this article is organized as follow. In Sect. 2, we compare, through the definitions that have been proposed by researchers of the two communities, the goals that are actually pursued in each domain. Then,

B. Morin (✉) · L. Mé
Supélec, Rennes, France
e-mail: benjamin.morin@supelec.fr

in Sect. 3, we compare the techniques that have been developed to reach these goals. Finally, in the conclusion, we summarize our analysis and suggest that alert correlation is one way to make the two fields cooperate.

2 Comparing domains

In this section, we compare the definitions that have been proposed by researcher of the two domains. First, we present the generally accepted definition of intrusion detection, then we come to the Cohen's and Adleman's definitions of virology, to finally be able to compare these definitions in order to position each domain with respect to the other. It is here a first attempt in comparison, the next section being devoted to a comparison based on the technics used by the tools presently used to detect intrusions or viruses.

2.1 A definition of intrusion detection

To enforce security properties within an information system, one has first to define a *security policy*, i.e., a set of rules and principles that ensure one or several of the properties of the "security triad":

1. Confidentiality: preserving authorized restrictions on information access and disclosure. A violation of the confidentiality property is the unauthorized disclosure of information.
2. Integrity: guarding against improper information modification or destruction. This includes ensuring information non-repudiation and authenticity. A violation of the integrity property is an unauthorized modification or destruction of information.
3. Availability: ensuring timely and reliable access to and use of information. The violation of the availability property is the disruption of access to or use of information or an information system.

Classically, preventive security tools are deployed to implement the security policy. For instance, access control and cryptology are both fields of computer security whose objective is to *prevent* security policy violations. Therefore, the question of *detecting* such violation might look trivial, if not pointless.

The intrusion detection field has emerged from the insufficiencies and failures of classical security mechanisms to protect computers from attacks. Intrusion detection is nothing but a second line of defense whitch actually finds its origin in log audit activities. The report written by James P. Anderson in 1980 [2], considered as the seminal work on intrusion detection, proposed changes in computer audit trails exploitation, in order to provide information ensuring that the activities that have occurred on a computer system were in compliance with the security policy. Auditing was indeed necessary because of the numerous flaws (i.e., vulnerabilities) introduced in a system during its design, implementation or configuration. These vulnerabilities can be exploited to bypass security mechanisms used to enforce a security policy.

From the scientific perspectives, the domain still lacks theoretical ground, and there is no formal definition of intrusion detection. Nevertheless, there is a commonly accepted one, coming from Anderson's view. This definition is relative to the security policy and can be stated as follows:

Definition 1 Intrusion detection is the process of monitoring computer networks and systems for violations of the enforced security policy.

Automating the analysis of large audit trails had become necessary as computer networks grew in size. This automation is the goal of intrusion detection systems (IDS), the foundations of which are attributed to the Denning's intrusion detection model [9].

2.2 Definitions of virology

Compared to intrusion detection, virology benefits from stronger theoretical foundations. There are two major formalization efforts in studying viruses, namely the work by Cohen [7], then the one by Adleman [1].

In order to position intrusion detection with respect to virology, we give a summary of the existing definitions of viruses. Further details can be found in Filiol's book [11].

Cohen's work is considered as the seminal work on computer virology. According to Cohen, a virus can be formalized as a word on a Turing machine tape that duplicates or mutates on the tape when it is activated. An informal definition of a virus can thus be stated as follows:

Definition 2 A computer virus is a program that can infect other programs by modifying them to include a possibly evolved copy of itself.

The notion of replication in central to Cohen's definition of viruses. More recent studies comply with this definition. For example, Bonfante et al. [3,4] proposed a new virus model, where self reproduction is considered a distinctive feature of viruses. Filiol [11] also states that any sequence that reproduces itself is a virus.

Deciding whether a computer program is a virus thus consists in deciding whether this program replicates itself. Cohen's main theorem states that deciding

whether a program is a virus is undecidable. In other words, there does not exist any program capable of deciding that a program is a virus either by enumerating all viral sequences or all non-viral sequences. Therefore, classical anti-virus techniques used nowadays are deemed to fail.

In [1], Adleman proposed a more general formalization of computer viruses. Notably, Adleman's model relaxes the virus replication characteristic. This model allows to capture a broader class of attacks, named *computer infections*, for which a classification is proposed. This classification includes viruses, as well as other forms of computer attacks, like trojan horses and logic bombs. The preferred term used to denote Adleman's computer infections is now *malwares*, whose origin is unclear. Initial references to it can be found in [18]. Brunnstein also used this term in 1991 [5], where a transition from anti-viruses to anti-malware programs is proposed. Filiol proposes in his book [11] the following definition of malware:

Definition 3 A malware is a simple or self-reproducing offensive program that is installed in an information system without users' knowledge, in order to violate confidentiality, integrity or availability of that system, or susceptible of falsely incriminating the owner or user of the program in a computer offense.

In his paper, Adleman also refines the complexity class of the infection detection problem, but the problem still remains undecidable.

2.3 Discussion based on definitions

2.3.1 Following Cohen's definition: an IDS should not directly detect viruses

In the light of Cohen's formal definition, computer viruses do not seem to fit into the scope of intrusion detection. Indeed, this definition of viruses makes no explicit reference to any security policy violation, which is a key concept in intrusion detection. In other words, a program that corresponds to Cohen's definition does not necessarily violate a security policy. Thus, this program would not be considered as an attack too by an IDS. Moreover, to our best knowledge, the detection of intrusions that can be characterized by the self-replication property of a program is not addressed in the intrusion detection literature. IDSes cope with potentially isolated attacks, lead by individuals whose objective is to take the control of some hosts; the objective of such individuals is not to have their attack necessarily replicated.

Computer worms are one exception to this. A computer worm can be defined as an autonomous self-replication program. Autonomy means that worms do not need user intervention to duplicate themselves, which makes their spreading much faster. Autonomy is made possible in ever increasing networked environments, where computer softwares exhibit remotely exploitable vulnerabilities. This difference between classical viruses and worms is referred to as *locality* in the malware taxonomy proposed by Swimmer [20]. Because of their self-replication characteristic, worms definitely fit into Cohen's definition. They also fit into the scope of intrusion detection, because they exploit a vulnerability which may lead to a security policy violation. Moreover, the fast propagation of worms has side effects on the network which can lead to a denial of service. For this reason, intrusion detection techniques based on the profiling of network traffic have been proposed to detect worms propagation. In these approaches, self-duplication is used as a mean to detect worms. Worms are thus an example of malware that has been studied in depth by both communities.

One may notice that the self-duplication property of Cohen's definition implies that a virus provokes the modification of other programs. This is an integrity violation, which should be detected by an IDS as a security policy violation.

2.3.2 Following Adleman's definition: an IDS should detect viruses

A common characteristic of Cohen's and Adleman's models is that they consider infected programs, i.e., programs that have been modified by a virus. In other words, virology assumes the presence of a virus within the code or data of a program. Whether the decision is made based on a static analysis of the program or by executing it in a protected environment, the problem remains the same: virus detection consists in deciding if a program contains a viral sequence or not.

The objective of intrusion detection is not to detect infected computer programs, but to analyze the *activities* of some monitored active entities (e.g., users, processes), in order to identify the subset of these activities that violates the security policy enforced within the monitored system.

Adleman's definition of malwares, which includes viruses, is similar to the definition of intrusions: in both cases, there is an explicit reference to a violation of the security properties. In the light of Adleman's model, worms are thus not the only common object under study in intrusion detection and viruses. Also are backdoors and trojans, whose detection can typically be achieved using an IDS.

For example, a brute force attack intended to guess a password should be detected by an IDS. Nevertheless, unless if realized by a virus, a classical anti-virus system should not detect such an attack. On the other hand, if it is a virus that realizes the attack, an IDS should detect it exactly as if it was realized by any user or process of the monitored system, i.e., by analyzing activities.

More generally speaking, any execution of a virus payload that engenders malicious activities should be detected by an IDS. As such, virus detection should fall within the scope of IDSes. Moreover, notice that, as it monitors activities, IDSes are a priori less impacted by the undecidability result related to virus detection.

2.4 Summary

At this point, we have shown that, *from a theoretical point of view*, intrusion detection encompass anti-virus detection. Actually, this is not surprising, as the definition of intrusion detection refers to any violation of the security policy.

Nevertheless, intrusion detection and virology are not redundant. *From the practical point of view*, they tackle computer security from different points of view, which are complementary. Current IDSes are generally not designed to detect viruses. Likewise, as noted by [20], anti-virus vendors have been reluctant to detect other forms of malicious software than pure virus.

However, this situation is slowly changing. Indeed, anti-virus softwares now detect and stop worm propagation. This could be interpreted as a natural evolution of anti-virus that acquire HIDS functionalities. In the same time, currently existing IDSes now acquire viruses detection capabilities.

3 Comparing techniques

In Sect. 2, we have compared, through the definitions that have been proposed by researchers of the two communities, the goals pursued in intrusion detection and anti-virus fields. Another way to position intrusion detection with respect to virology is to compare the techniques developed in these two domains to reach those goals. This is the aim of this section.

In the case of intrusion detection, we will see that there is a large gap between practical achievements and the theoretical objectives.

This section is organized as follows: we first present the characteristics of IDSes, then we give four examples of implementations which are the most representative according to us. The positioning with respect to anti-virus softwares will be addressed throughout these two sections, where it is the most relevant.

3.1 Characteristics of IDSes

In this section, we present the different characteristics of present IDSes, as proposed by Debar et al. [8], and compare these characteristics to the ones of anti-virus softwares.

3.1.1 Architecture of an IDS

According to the Intrusion Detection Working Group at the IETF, an IDS is composed of two distinct components, the architecture of which is illustrated in Fig. 1:

1. The *sensor* is responsible for the capture, feature extraction and normalization of the *activity* of the monitored entity, as seen from a *data source*. The sensor provides *events* to the *analyzer*.
2. The *analyzer* is responsible for the detection of misuse or anomalous activities, and sends alerts to a *manager* where alert correlation and management processes take place.

The sensor and analyzer functions correspond to the two most discriminating characteristics of IDS taxonomy proposed by Debar et al., namely the *data source* employed to detect attacks and the *detection method*

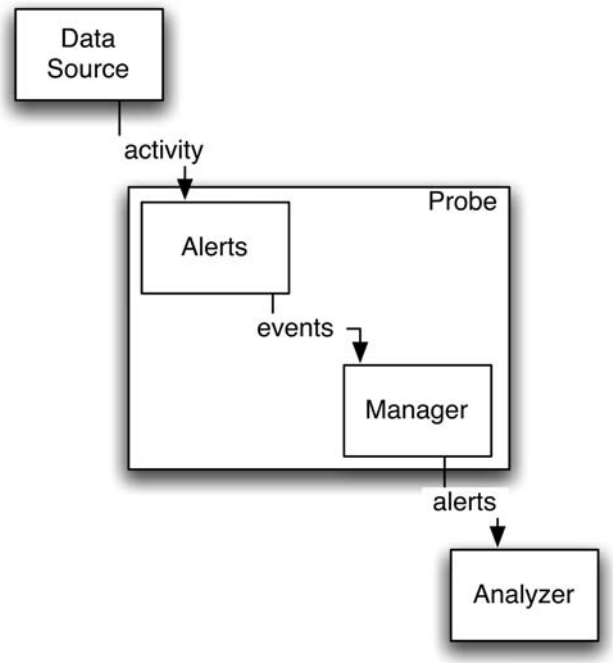


Fig. 1 Architecture of an IDS

that allows to distinguish attacks from normal activities. These two characteristics are further explained in this section. This taxonomy includes three other classification criteria, namely the behavior on detection, the detection paradigm, and the usage frequency. These criteria are not as discriminative as the first two to distinguish two IDSes pairwise, but they provide interesting elements for the anti-virus comparison. We will briefly discuss them in the remainder.

We may notice that in [20], Swimmer shows that a classical anti-virus software fits into the IDS Debar et al's taxonomy.

3.1.2 Data sources

Exploitable data sources to monitor the activity of an information system are generally split into three categories: the network traffic, the operating system's audit trails and the applicative logging information. IDSes that take advantage of the former are called Network-based IDS (NIDS), while IDS that take advantage of the two latest are called Host-based IDS (HIDS) because their analysis is achieved on the target host. Nevertheless, the term HIDS is generally used for an IDS analyzing operating system level data, while IDS exploiting application data are rather named application-based IDS.

Network-based intrusion detection: NIDSes analyze network packets in order to detect signs of intrusions. NIDSes either listen to traffic passively using a dedicated interface in promiscuous mode, or are put inline, in which case they are called *intrusion prevention systems* (IPS) because of their traffic blocking capabilities.

Network-based IDSes received a wide acceptance by the industrial community. This is mainly due to their easy deployment, to the absence of impact on the performances of the monitored network, and to the fact that one single sensor is capable of monitoring the activity of several hosts provided it is located at a strategic point in the network.

The scope of NIDSes is not only large from the topological point of view, but also from the point of view of the attack types that can be detected. Any attack vectored to the target through the network can potentially be detected by a NIDS, without any modification of the monitored system.

However, network-based IDSes suffer from several drawbacks. First, they are not able to analyze ciphered traffic, which is increasingly used in network protocols. Second, the ever increasing traffic bandwidth makes the exhaustive capture of packets a challenging task, potentially leading attacks not to be detected. Last but not least, diagnosis of NIDSes is generally not reliable enough because the information contained in network traffic is generally not sufficient to rule on the success or failure of attacks.

Host-based intrusion detection: Host-based intrusion detection systems (HIDS) take advantage of audit trails to monitor the activity of processes or users at the host level. Audit trails can be provided by the operating system (e.g., Sun BSM) or by individual applications (e.g. web server log files), in which case sensors are sometimes referred to as application-based.

System calls invoked by processes constitute a data source of particular interest in host-based intrusion detection, because they provide fine-grained information about what is actually executed by a host. Therefore, system calls allow for a more reliable diagnosis about the impact of attacks.

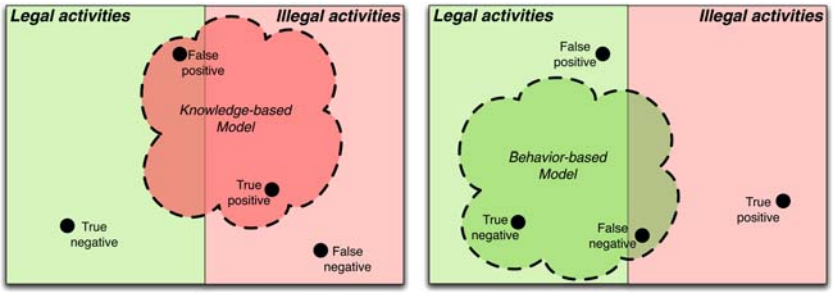
Host-based intrusion detection has two major drawbacks. First, the mere capture of the activity can turn out to be time and space consuming, thereby degrading the overall performance of the monitored host. Moreover, time spent to analyze events adds to their capture. Second, the scope of HIDSes is limited to a single host or application. Therefore, several sensors need to be deployed in large networks for the monitoring to be effective, which makes alert management and correlation at the manager level crucial. For these reasons, the development of HIDSes has not been as successful as for NIDSes in the industrial community.

Anti-virus softwares vs. HIDS/NIDS: From the analysis localization point of view, classical anti-virus softwares are similar to HIDS. Yet, almost any computer is now equipped with an anti-virus, so we may wonder why HIDS did not achieve the same deployment. We believe that there are two reasons for this. The first one is that HIDS have a continuous impact on the overall performance of the monitored host, and their installation is intrusive because they require deep interactions with the operating system. The second reason is that IDSes still suffer from a significant amount of false alarms. From an IDS vendor point of view, these two reasons are all the more intolerable as their softwares are to be used by privates.

3.1.3 Detection method

In this section, we focus on the detection paradigm, that is to say the approach used to distinguish malicious activities from normal ones. There are two opposite categories of attack detection paradigms: knowledge-based

Fig. 2 Detection methods



and anomaly-based intrusion detection. The former is best known as *misuse* intrusion detection and the latter as *anomaly* intrusion detection. We prefer the *knowledge* and *behavior* terminology, due to Debar et al. [8], because we believe it is more explicit.

Figure 2 illustrates these two conceptually opposite detection methods. Behavior detection consists in modeling normal activities, while the knowledge-based intrusion detection consists in modeling malicious activities. We will give further details in the remainder. In both cases, the challenge is to build models that are simultaneously *complete* (i.e., the model allows to detect *all* malicious activities) and *accurate* (i.e., the model allows to detect *only* malicious activities). Incompleteness leads to *false negatives* (i.e. attacks that are not detected), while inaccuracy leads to *false positives* (i.e. false alerts), which are the two infamous problems that current IDSes face.

Knowledge-based detection: Conceptually, knowledge-based intrusion detection consists in building a model of known malicious (i.e., forbidden) activities, and comparing the current activity with this model. The elements of the model that encode the characteristics of malicious activities are generally called *signatures*. Therefore, knowledge-based intrusion detection is a recognition problem, where the objective is to decide whether the currently monitored activity matches one of the signatures.

The obvious major drawback of knowledge-based intrusion detection resides in their inherent incompleteness: attacks for which no signature exists are not detected. Therefore, the signature database must be up-to-date in order not to miss attacks. The so-called "0-days" (i.e., attacks exploiting unknown vulnerabilities) are of course a major concern for that type of detection.

Knowledge-based IDSes are presumably more accurate than behavior-based ones. However, in order not to miss attacks, signature providers tend to create generic signatures, in order to capture most attacks. Generic signatures are also susceptible to capture legitimate activities, leading thus to false positives.

The major advantage of knowledge-based intrusion detection is that it recognizes the exploited vulnerabilities. Therefore, the diagnosis provided by the IDS allows security operators to take fast and appropriate countermeasures.

Anti-virus softwares vs. knowledge-based detection: Knowledge-network-based IDSes are the most common kind of IDSes. Most of them contain signatures intended to detect malwares based on their payload, including viruses and worm propagation. However, the signatures used by IDSes to detect viruses are generally both incomplete and inaccurate compared to those found in anti-virus systems.

The principles of knowledge-based intrusion detection are the same as classical anti-virus systems. Therefore, Cohen's theorem about the undecidability of virus detection also applies to this category of intrusion detection approach. As a consequence, knowledge-based intrusion detection is sensitive to obfuscation techniques intended to hide attacks.

Behavior-based detection: Behavior-based (i.e., anomaly) detection consists in comparing the observed behavior of the monitored system with a reference model previously constructed, which describes the expected behavior of this system. A significant deviation between the current behavior and the model can be the sign of an intrusion. The model of safe behavior can be either established by learning techniques or explicitly specified by an expert.

Existing detection techniques falling in this category vary, depending on the type of system being monitored and the model construction approach. In [9], Denning proposed to build profiles of users based on metrics like the type of programs used, the rate of keystrokes, or the amount of CPU and memory consumed. However, the most productive line of research in the anomaly detection area consists in modeling the behavior of processes

by means of the system calls they invoke during execution. Several techniques have been proposed in the literature, most of which are based on statistical profiles. Examples are given in Sect. 3.2.

The main advantage of anomaly-based IDSes resides in their completeness: the detection does not require any knowledge of existing attacks. Therefore, this approach can potentially detect unknown or obfuscated attacks.

The major drawback of behavior-based intrusion detection lies in its inability to spot the exploited vulnerability explicitly. This leads to additional investigation efforts by security operators to understand the situation at issue.

It is also difficult to correctly capture the behavior of the system, i.e. capture all and only normal activities. Indeed, an incomplete model leads previously unseen normal activity to trigger false alarm; on the contrary, malicious activities that occur during the learning period leads attacks not to be detected.

Updating the normal behavior model is also challenging compared to the knowledge-based approach, where the malicious activities model update merely consists in adding new attack signatures to the existing database.

Anti-virus softwares vs. behavior-based detection: There is a fundamental difference between behavior-based approach in intrusion detection and behavior detection in virology (also known as detection by heuristics).

Behavior-based virology basically consists in using generic virus infection evidence in order to detect viruses without relying on a specific signature. Behavior intrusion detection consists in dynamically detecting deviations in the behavior of processes which are subject to computer infections. Their objective is not to recognize computer infections by analyzing the payload of a malware program.

In the context of behavior-based detection, mimicry attacks are the analog of obfuscation attacks in knowledge-based detection. Mimicry consists in masquerading a malicious activity so that it is not discernible from legitimate activity. We conjecture that Cohen's theorem probably applies when trying to decide whether a process behaves well with regard to the model of safe behavior, but this would require further investigations.

3.1.4 Other classification criteria

We now study the three other IDS classification criteria of Debar et al's taxonomy [8]. These criteria are not as discriminant as the data source and detection model previously discussed, but are interesting to study when comparing intrusion detection and virology.

Behavior on detection: The behavior on detection denotes the type of reaction that is undertaken by an IDS when an attack is detected. We may distinguish passive reaction, which only consist in raising an alert toward a human security operator or an alarm correlation mechanism, and active reaction which consists in trying to stop the attack by, e.g., blocking a network connection, killing a process or closing a user account. Most IDSes use passive reaction because of the non-negligible false positive rate. This is in contrast with anti-virus softwares, which do not only stop viruses, but also try to remove them. This aspect is not addressed in intrusion detection. One may notice however, that some knowledge-based NIDSes, called IPSes are able to block incoming malicious traffic because they are positioned inline, instead of passively sniffing traffic.

Usage frequency: Usage frequency applies to the way an IDS performs its analysis. There are two categories: continuous and periodic monitoring. Most IDSes perform continuous monitoring, by analyzing, for instance, the network traffic or the system calls invoked by processes during their execution. Anti-virus softwares also run continuously to scan each income, whatever the way it actually enters into the system. In addition, anti-virus systems are generally also used periodically, for example at boot time, to scan the files present on a host.

Detection paradigm: The taxonomy distinguishes two types of paradigms, transition-based and state-based IDSes. Transition-based detection consists in detecting intrusions by means of the transitions between insecure states. State-based detection consists in recognizing insecure states. Continuous-based IDSes (most of the IDSes) and anti-virus softwares are transition-based. Nevertheless, when used periodically, a anti-virus software is state-based.

3.2 Examples of IDS

In this section, we briefly present four existing types of IDSes, corresponding to the four possible combinations of the detection method and data source, {network-based, host-based} × {behavior-based, knowledge-based}.

3.2.1 Knowledge and network-based IDSes

Knowledge and network-based IDSes are the most widespread type of IDSes in operational contexts, because of their aforementioned advantages. Among

these types of IDSes, the open source IDS Snort[1] is probably the most widely used. Snort signatures consist of conjunct of constraints over the network packets header fields and payloads, as well as meta-information which describe the attack the signature is supposed to detect. The set of allowed constraints notably include fixed-string patterns and regular expressions that are matched against the payload of IP packets.

The following example is an excerpt of the attack signature database:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS
    (msg:"WEB-IIS CodeRed v2 root.exe access";
     flow:to_server,established; uricontent:"/root.exe"; nocase;
     reference:url,www.cert.org/advisories/CA-2001-19.html;
     classtype:web-application-attack;)
```

This signature is supposed to allow the detection of the propagation of the Code Red worm. Actually, the mere presence of the `root.exe` in an HTTP request to a machine on TCP port 80 is sufficient for this signature to provoke an alert. The success or failure of an attack would not be evaluated in this case.

Snort signatures are also used to detect activities which might not necessarily denote an intrusive activity. For example, the following signature detects hosts that do not respond in a network:

```
alert icmp $EXTERNAL_NET any -> $HOME_NET any
    (msg:"ICMP Destination Host Unreachable";
     icode:1; itype:3; classtype:misc-activity;)
```

Some advocate that alerts provoked by this kind of signature are false positives because hosts not responding in a network are a classical phenomenon. We argue that these alerts are actually irrelevant in most contexts, but can be used to detect abnormal situations, the cause of which can be related to security. For example, an increase in the frequency of the above alert can be caused by a random scanning worm. Thus, these kinds of signature should not be deactivated. Rather, some additional component should be proposed to filter out from the alert stream the "normal" alerts, i.e., alerts that are generated without relation to a security problem. The Ref. [22] proposes such a filter.

This example illustrates that security operators who use IDSes are not only concerned with viruses, but also with *abnormal* activities in their network, the origin of which can be related with a security policy violation, but not necessarily. From this point of view, anti-viruses and IDSes are complementary in order to achieve a comprehensive protection of an information system.

[1] http://www.snort.org.

3.2.2 Behavior and network-based IDSes

In the PAYL IDS proposed by Wang and Stolfo [24], the model of normal network traffic is composed of the frequency distribution of every byte value in the payload of packets, for various network protocols. The rationale for this is that packets carrying shellcodes are likely to have a different byte frequency distribution than usual traffic, since shellcodes contain programs that are to be executed on the target server. The Mahalanobis distance is used to compare the byte frequency distribution of a given packet with the profile of normal traffic previously learned. A distance that exceeds a given threshold is indicative of an attack.

Actually, this approach proposes to study the frequency distribution of n-grams, i.e. byte sequences of length n, but the authors demonstrated that their approach is effective with $n = 1$ (which amounts to considering the frequency distribution each byte value). Indeed, the PAYL IDS achieves 100% detection rate with a false positive rate of only 0.1% on a widely used evaluation traffic corpus.

However, Kolesnikov and Lee [15] demonstrated that this approach was vulnerable to mimicry attacks that blend exploits with normal appearing byte padding, such as blended polymorphic attacks. Therefore, Wang et al. [23] proposed a revised version of their initial IDS, based on randomized higher-order n-grams (i.e. a mixture of n-grams, whose length is $n > 1$). This version is resilient against simple mimicry attacks because it is more difficult for smart worms to mimic normal traffic, as they are not aware of the normal traffic model used by the IDS.

This detection technique is interesting because it is radically different from classical anti-virus approaches, both by data source and the detection method employed. It achieves promising results.

3.2.3 Knowledge and host-based IDSes

The ASAX system developed by Habra et al. [13] is an expert system whose purpose is to analyze universal audit trails (e.g., Sun Basic Security Module). The attack signatures are specified in a rule-based language, RUSSEL, which allows to recognize sequences of (potentially non-adjacent) events within an audit trail.

Several languages in the same vein have been proposed since ASAX, like ADeLe [21] or Sutekh [16, 17], which allow for more complex logical connectors between event patterns, such as negation, disjunction and conjunction.

These recognition systems basically work following the same principle: at initialization, signature specifications are translated into finite state automata, whose transitions are event patterns. During detection, each event (e.g., system call) is fed into the automata. Automata whose current triggering event patterns match the event evolve to the next state. Reaching a final state is indicative of the achievement of an attack scenario.

In an attempt to show that viruses can be detected by host-based IDSes, Swimmer proposed in [6,20] a knowledge-based HIDS called virus intrusion detection expert system (VIDES). VIDES detects system-executable malwares by running the target programs inside a virtual operating system and analyze their side effects on the emulated system. The VMWare software is used for the emulation purpose and ASAX is the expert system used to diagnose the presence of malwares.

According to Swimmer, VIDES achieves promising results in detecting viruses known at the time, by using signatures based on generic interactions of the virus with the operating system.

These results are not surprising, because knowledge-based HIDS are very close to anti-virus softwares, as previously explained. More importantly, this approach illustrates what we already evoked in Sect. 2.3.2: one difference between IDSes and anti-viruses lies in the fact that the former exploits external interactions between the virus and the attacked system (i.e., symptoms) while the latter exploits intrinsic characteristics of the viral sequence. Actually, the behavior-based detection approach in virology, which is based on the analysis of actions of a virus on the system, corresponds to the knowledge-based intrusion detection approach.

Swimmer's work is interesting because, to our best knowledge, it is the first attempt to join the virology and intrusion detection domains, where they are the closest.

3.2.4 Behavior and host-based IDSes

The first IDS falling in this category is the model proposed by Denning [9]. However, we choose to present here the *immunological* approach proposed by Forrest in [10,12] because it initiated a long series of works in the same vein (profiling processes based on system calls), and also because the biological allusion makes it (at least terminologically) relevant with the subject of this article.

This approach is inspired by the natural immune system, where cells are capable to distinguish self from non-self. In the context of computer security, the self of a given process is a list of short system call sequences considered safe, established through a learning process. During monitoring phase, the sequences of system calls invoked by a running process are matched against the process profile and a distance is computed on a statistical basis. Any significant deviation is indicative of a potential intrusion (through code injection for instance).

Improvements of the approach basically consist in making the length of sequences variable or taking into account the parameters of system calls.

Despite their common biological inspiration and detection location, anti-viruses and immunology-based IDSes are different from the detection method viewpoint. The former detects attacks based on their intrinsic characteristics (e.g., the fact that a program sequence self-duplicates), while the latter detect attacks by means of symptoms of the executed code on the target.

3.3 Summary

Most anti-virus techniques implement the static, signature-based, host-based approach to malicious program detection [20]. From the detection method viewpoint, anti-viruses which consist in looking for patterns of known viruses within a string of bits are very similar to knowledge-based network IDSes. From the analysis localization viewpoint, anti-viruses are like HIDs.

One notable difference between knowledge-based HIDS and classical anti-viruses lies in the fact that the former exploits external interactions between the virus and the attacked system (i.e., symptoms like invoked system calls) while the latter exploits intrinsic characteristics of the viral sequence.

The behavioral anti-virus approach should not be confused with behavior-based intrusion detection. Behavior-based anti-viruses are actually very similar to knowledge-based HIDSes, as they also exploit interactions between a code execution and the system. Behavior-based intrusion detection measures deviations in the behavior of the monitored system (e.g., a network, a host, a process) in order to detect attacks. This is fundamentally different from techniques which take advantage of knowledge about the characteristics of attacks. Behavior based detection is supposedly more resilient to obfuscation techniques which modify the shape of viruses.

Lastly, contrary to virology, intrusion detection does not address the problem of malwares removal from an infected system. Stopping attacks automatically is still not widely proposed, except in the knowledge-based

NIDS community (IPS), because of the IDSes' lack of accuracy.

4 Conclusion

Our objective in this paper was to identify similarities and differences between virology and intrusion detection in order to position each of these domains with respect to the other. We deliberately discussed these differences only through scientific and technical arguments. Nonetheless, other aspects might also be of interest. From the sociological viewpoint for instance, Swimmer [20] speculates that one reason why there is so little communication between virology and intrusion detection communities is that the former tends toward secrecy while the latter tends toward openness.

From the theoretical point of view, the objects under study in virology seem to be included in the intrusion detection field, which is not surprising owing to the generality of the definition of intrusion detection. However, the two domains differ by their methodology : intrusion detection considers objects by their extrinsic characteristics, whereas virology considers them by their intrinsic characteristics. These two visions are complementary.

Another difference between virology and intrusion detection is that the former has sound theoretical basis, but surprisingly few contributions [4], while the latter lacks a formal ground, but has been the subject of numerous research papers. We believe that intrusion detection would benefit from the theoretical basis of virology, while virology would benefit from the numerous detection techniques that have been investigated in intrusion detection.

There is a clear trend toward the convergence of intrusion detection and virology in scientific conferences, particularly by means of the malware concept. Indeed, the malware topic is increasingly evoked in conferences which were originally devoted to intrusion detection. The interested reader may refer for example to the RAID 2006[2] conference session titles or to the DIMVA 2007[3] call for paper.

From the practical point of view, it seems that most anti-virus techniques implement the same kind of technology, namely the signature, host-based approach to malicious program detection, whereas IDSes consider different data sources and detection methods. However, the fact is that signatures host-based IDSes have barely not been used to detect viruses (except the work by Swimmer [20]). Therefore, from the practical point of view, IDSes and anti-viruses are complementary. One may also notice that anti-viruses are progressively acquiring HIDS capabilities.

Another illustration of the complementarity of approaches is given by policy-based IDSes. This detection method consists in monitoring information flows within a system and detecting sequences of operations which lead to a security policy violation, each operation being legal if considered individually [14,25,26]. Such IDSes allow for the detection of privilege escalation attacks, but will not detect activities that do not violate the underlying security policy, contrary to anti-virus softwares. For example, a document of some user stolen by a malware executed by the same user (by means of an infected mail or macro virus for instance) would not lead a policy-based IDS to raise an alert, because a discretionary access control (DAC) policy allows a process run by a user to access any of the user's resource. On the contrary, for the same situation, an anti-virus software could detect the malware.

All malware detection approaches have their own advantages and drawbacks, and a correct protection approach will ultimately combine several heterogeneous sensors (i.e. different IDS sensors types and anti-virus softwares) rather than a single sensor to catch all. Nonetheless, multiplying heterogeneous sensors reinforce the need for a centralized supervision of the security of an information system. Alarm fusion and correlation is one field of intrusion detection by means of which the two fields could cooperate.

As suggested by Swimmer [20], we believe that the separation between the intrusion detection and the virology fields should cease because they are complementary, both from the theoretical and practical point of view. As a matter of fact, the merger has already taken place, as illustrated by the shared topics that both communities study. A holistic approach to the problem of defending against malware would be on call.

[2] Recent advances in intrusion detection, http://www.raid-ssymposium.org/.

[3] http://www.dimva.org.

References

1. Adleman, L.: An abstract theory of computer viruses. In: Advances in Cryptology. Lecture Notes in Computer Science, vol. 403, pp 354–374. Springer, New York (1988)
2. Anderson, J.P.: Computer security threat monitoring and surveillance. Technical report, James P. Anderson Company, Fort Washington, Pennsylvania, April 1980
3. Bonfante, G., Kaczmarek, M., Marion, J.-Y.: Toward an abstract computer virology. In: International Colloquium on Theoretical Aspects of Computing. Lecture Notes in Computer Science, vol. 3722, pp 579–593. Springer, New York (2005)

4. Bonfante, G., Kaczmarek, M., Marion, J.-Y.: On abstract computer virology from a recursion theoretic perspective. J. Comput. Virol. **1**(3), 45–54 (2006)
5. Brunnstein, K.: From AntiVirus to AntiMalware Software and beyond: another approach to the protection of customers from dysfunctional system behaviour. In 22th National Information Systems Security Conference, pp 12–26 (1999)
6. Charlier, B.L., Mounji, A., Swimmer, M.: Dynamic detection and classification of computer viruses using general behaviour patterns. In: Proceedings of 5th International Virus Bulletin Conference (1995)
7. Cohen, F.: Computer viruses. PhD Thesis, University of Southern California (1985)
8. Debar, H., Dacier, M., Wespi, A.: A revised taxonomy for intrusion-detection systems. Ann. des Télécommun. **55**(7-8), 361–378 (2000)
9. Denning, D.E.: An intrusion-detection model. IEEE Trans. Softw. Eng. **13**(2), 222–232 (1987)
10. D'Haeseleer, P., Forrest, S., Helman, P.: An immunological approach to change detection: algorithms, analysis and implications. In: Proceedings of the 1996 IEEE Symposium on Research in Security and Privacy. IEEE Computer Society Press, Oakland, pp 110–119 (1996)
11. Filiol, E.: Computer viruses: from theory to applications. Springer, New York (2005)
12. Forrest, S., Hofmeyr, S.A., Somayaji, A., Longstaff, T.A.: A sense of self for unix processes. In: Proceedings of the 1996 IEEE Symposium on Research in Security and Privacy. IEEE Computer Society, IEEE Computer Society Press, pp 120–128, May 1996
13. Habra, N., Charlier, B.L., Mounji, A., Mathieu, I.: ASAX: software architecture and rule-based language for universal audit trail analysis. In: Proceedings of the 2nd European Symposium on Research in Computer Security (ESORICS'92). Lecture Notes in Computer Science, vol. 648, pp 435–450. Springer, New York (1992)
14. Ko, C., Redmond, T.: Noninterference and intrusion detection. In: Proceedings of the IEEE Symposium on Security and Privacy (2002)
15. Kolesnikov, O., Lee, W.: Advanced polymorphic worms: evading IDS by blending in with normal traffic. In: USENIX Security Symposium (2006)
16. Pouzol, J.-P., Ducassé, M.: From declarative signatures to misuse IDS. In W. Lee, L. Mé, A. Wespi (eds.) In: Proceedings of the 4th International Symposium on the Recent Advances in Intrusion Detection (RAID'2001). LNCS, vol. 2212, pp 1–21, October (2001)
17. Pouzol, J.-P., Ducassé, M.: Formal specification of intrusion signatures and detection rules. In: Proceedings of the 15th IEEE Computer Security Foudations Workshop (CSFW'02). IEEE Computer Society, pp 64–76, June (2002)
18. Spafford, E.H.: Computer viruses as artificial life. J. Artif. Life **1**(3), 249–265 (1994)
19. Swimmer,M.: Review and outlook of the detection of viruses using intrusion detection systems. In Debar H., Mé L., Wu S.F. (eds.) In: Proceedings of the 3rd International Workshop on the Recent Advances in Intrusion Detection (RAID'2000). LNCS, vol. 1907. Springer, New York, October 2000 (Extended abstract)
20. Swimmer, M.: Malware intrusion detection. PhD Thesis, Hamburg University (2005)
21. Totel, E., Vivinis, B., Mé, L.: A language driven intrusion detection system for event and alert correlation. In: Proceedings ot the 19th IFIP International Information Security Conference, pp 209–224, Toulouse. Kluwer, Dordrecht, August 2004.
22. Viinikka, J., Debar, H., Mé, L., Séguier, R.: Time series modeling for IDS alert management. In: Proceedings of the ACM Symposium on Information, Computer and Communications Security (ASIACCS'06), pp 102–113. ACM Press (2006)
23. Wang, K., Parekh, J.J., Stolfo, S.J.: Anagram: A content anomaly detector resistant to mimicry attack. In: Zamboni, D., Kruegel, C. (eds.) Recent Advances in Intrusion Detection. Lecture Notes in Computer Science, vol. 4219, pp 226–248. Springer, New York (2006)
24. Wang, K., Stolfo, S.J.: Anomalous payload-based network intrusion detection. In: Erland Jonsson, Alfonso Valdes, Magnus Almgren (eds.) Proceedings of the 7th International Symposium on Recent Advances in Intrusion Detection (RAID'2004). Lecture Notes in Computer Science, vol. 3224, pp 203–222. Springer, New York, September 15–17 (2004)
25. Zimmermann, J., Mé, L., Bidan, C.: Introducing reference flow control for detecting intrusion symptoms at the os level. In: Wespi, A., Vigna, G., Deri, L. (eds.) Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID'2002). Lecture Notes in Computer Science, vol. 2516, pp 292–306. Springer, New York (2002)
26. Zimmermann, J., Mé, L., Bidan, C.: Experimenting with a policy-based hids based on an information flow control model. In Proceedings of the Annual Computer Security Applications Conference (ACSAC), December (2003)