Evaluating Attack Resiliency for Host Intrusion Detection Systems

Jesus Molina
Department of Electrical
and Computer Engineering
University of Maryland
chus@glue.umd.edu

Michel Cukier
Center for Risk and Reliability
Department of Mechanical Engineering
University of Maryland
mcukier@umd.edu

Abstract- Host intrusion detection systems (HIDSs) are important tools used to provide security to computer systems. Many HIDSs exist and security practitioners need a way to determine the optimal security solution for their environment. Current evaluations of HIDSs focus on detection accuracy and typically do not account for the possibility that an adversary may subvert the HIDS and modify the outcome. As some elements from the HIDS need to reside within the system under supervision, evaluating the strength against HIDS subversion is critical.

This paper defines HIDS subversion and presents HIDS resiliency as a metric of HIDS strength in the event of an attack against the system being supervised. To estimate HIDS resiliency, we evaluated the independency between the system being supervised and the HIDS. Then we integrated resiliency into current frameworks to evaluate detection accuracy.

Keywords: host intrusion detection systems, benchmarking, metrics, attack resiliency, independency.

1 Introduction

The goal of a security mechanism is to prevent the system from reaching an insecure state. However, security mechanisms are not infallible. While catastrophic failures should be immediately evident (e.g., an attacker who deletes all data from the system), silent intrusions, where the attacker is not noticed, present a major concern. To prevent insecure states from going unflagged, we implemented a set of tools called intrusion detection systems (IDSs). The goal of an IDS is to accurately detect insecure states or transitions to insecure states (possibly raising an alert) by evaluating an input vector of heterogeneous data from different sources (e.g., file systems, network packets, and memory). Historically, IDSs were divided into two main types: network IDS (NIDS) and host IDS (HIDS). NIDSs consume network packets to investigate the state of the system, while HIDSs analyze logs within the host being monitored.

As HIDSs evolve, new tools are used to collect data. Currently, HIDSs are IDSs that collect internal data from the supervised system, and NIDSs are IDSs that collect data external to the host, usually by monitoring network interfaces. Hybrid IDSs collect both internal and external data. All NIDS elements typically reside outside of the host, and the IDS security is evaluated independently from the system being supervised. That is not the case for a HIDS, as at least some part of the HIDS resides inside the supervised system, and hence HIDS security is tightly bound to its deployment characteristics.

A IDS that operates correctly reports alerts in the event of an attack and does not mislabel any activity as malicious. As many different IDSs exist, security professionals need to evaluate this options to make educated selections for the best IDS for a specific environment. The most common method employed to evaluate an IDS is to measure the number of missed attacks (false negatives) in a predetermined universe of attacks and to measure the number of alerts which are not attacks (false positives) from a pool of non-malicious activity. HIDSs and NIDSs are currently been evaluated using this approach.

However, in the case of HIDSs, deployment characteristics are important for performing an accurate evaluation. In this study, we present one such deployment characteristic, *HIDS resiliency*, as the probability that the HIDS will not be subverted in the event of an attack against the system under supervision Evaluating resiliency presents several challenges. First, an attacker may corrupt any element in the path, from data collection to alert reporting, to compromise the result. However, often the security assessment of a HIDS remains confined to the detection engine's own security. While the HIDS and the detection engine are commongly thought to be the same, they are not. In fact, a typical method to defeat anti-virus or other integrity checkers is to install readily available rootkits which subvert the kernel, because the kernel acts as an element of the IDS. Second, the security evaluation depends on the deployment environment, so parameters associated with the environment need to be provided. Third, as often is the case in security evaluations, a certain degree of expert knowledge is necessary to estimate HIDSs resiliency.

In the next section we describe the related work. Section 3 contains definitions and terminology used throughout the paper. We describe HIDS independency in Section 4, and we introduce an independency cost. In Section 5, we describe how to use the independency measurements to estimate HIDS resiliency depending on environmental variables. Section 6 contains an analysis for current IDS frameworks and integrates HIDS resiliency into a HIDS cost framework. Section 7 contains the conclusions.

2 Related Work

The first known reports of subversion techniques against HIDSs appeared in 1993. The underground e-zine *Phrack* published a brief article on a simple technique that could be used to bypass the Tripwire [1] integrity checker by modifying its database. Articles continued to appear on the subject,

Received April 1, 2008

proposing, for example, to corrupt the kernel to subvert the HIDS [2].

IDS resiliency to attacks against the system being supervised has been cited as a possible IDS study characteristic in [3, 4]. In both studies, a brief description of the characteristic was given and the underlying importance of the characteristic was identified. However, few attempts exist to evaluate HIDS resiliency. In [5], a scorecard metric approach was used to judge the security of the IDS, which can be low, average or high. However, there was no formalization of how the measurement should be performed. Finally, the work in [6] proposed the concept of robust IDS evaluations, which is to analyze the behavior of the IDS in a hostile environment for evasion attacks against the detection engine.

International standards, such as the Common Criteria for Information Technology Security Evaluation (CC) [7] can be used as a possible tool [8] to measure the overall technological achievement of the HIDS. The CC provides a numerical score called the Evaluation Assurance Level (EAL), ranging from one to seven, to measure the technological achievement of a system against a certain security target, called a Protection Profile (PP). While the CC is a useful and accepted evaluation standard, there are serious drawbacks for its use in evaluating HIDSs. First, not all HIDSs come with an EAL from the manufacturer, as EALs are resource intensive to acquire both with respect to time (often months) and cost (often thousands of dollars). Second, as the EAL is tailored to a security target, EALs may or may not include features like attack resiliency. For example, the PP for an IDS security scanner [9] assumed a non-hostile environment and that no attack would be attempted to subvert it. The security evaluation for the Dragon NIDS and HIDS [10] assumed that the threat level of the environment is unsophisticated and that the hardware would not be tampered with. These two examples show that comparing products by their EAL is only valid in the case that they share exactly the same PP target.

Some HIDS implementations attempted to specifically increase HIDS resiliency by means of embedded hardware [11], virtualization [12] and trusted computing [13]. All of these techniques claim to increase the independency between the IDS and the system under supervision. These claims are well founded, but they suggest the need to describe independency and to formalize the relationship with IDS subversion to evaluate their technological achievement.

3 Definitions

HIDSs may suffer integrity, confidentiality or availability attacks. If the attacker's objective is to perform an attack against the system being supervised, availability and confidentiality attacks will be of limited use. Indeed, most availability attacks against the HIDS are considered to be attacks against the system and are labeled as alarms. Confidentiality attacks provide limited information on the system, which is inferred from configuration files and output data.

Integrity attacks are more important with respect to attacking the system being supervised. Indeed, a clever change of the HIDS allows the attacker to circumvent it. If the modification leads the HIDS to miss an otherwise being supervised detectable attack, we declare the HIDS to be subverted. While the attacker may compromise the integrity of any element in the HIDS, we can reduce the target for the attacker to modify the HIDS's output in the event of an attack. This definition implies that the attackers objective is not to gain control of the HIDS, but to change the HIDS output, which may be a simpler problem.

Formally:

Definition 1. *An intrusion detection system, $\mathcal{H}$, with output, $\mathcal{O}$, resulting from a set of data, $\mathcal{D}$, is* **subverted** *by an attacker if for the same set of data, $\mathcal{D}$, the attacker can change the output to $\mathcal{O}' \neq \mathcal{O}$.*

For the simplest type of HIDS, implemented as a function, h, with inputs $\mathbf{X} = \{x[1], x[2] \ldots x[N]\}$ that are classified as normal data or as an attack, $h : X \rightarrow \{0, 1\}$, the attacker subverts the system if the HIDS response is the inverse given $\mathbf{X}$, i. e., "0" instead of "1". Depending on the HIDS deployment characteristics, HIDS subversion varies in complexity. The following definition describes a metric of this complexity:

Definition 2. *We define HIDS resiliency to subversion as the probability that the HIDS will not be subverted in the event of an attack to the supervised system.*

In the remainder of this paper, we refer to HIDS resiliency as the HIDS resiliency to subversion attacks. To evaluate HIDS resiliency we did not make any assumptions about the privilege level gained on the supervised system, or the initial privilege of the attacker on the supervised system. We assumed that the attacker had no special implicit privileges on the HIDS. If the attacker decides to subvert the HIDS, we need to evaluate the probability of the attacker succeeding. There are two possible routes to perform a subversion attack on a HIDS: through the host, or out of band. For the case of HIDSs, usually the easiest approach for an attacker is to subvert the HIDS using the system being supervised as the attack vector. We will assume that out of band attacks are more costly than attacks through the system being supervised. If shared elements exist between the HIDS and the supervised system, an attacker may corrupt or tamper with elements from the supervised system to subvert the HIDS. Hence, a primary goal for the HIDS is to be independent of the supervised system, so that an attack launched against the system under supervision will not impact the HIDS resiliency. To estimate resiliency, we first evaluate *HIDS independency* as the level of isolation between the HIDS and the supervised system. The more independent a system is, the more resilient will be to attacks through the host supervised.

We define this property as follows:

Definition 3. *We define HIDS independency as the level of isolation between the HIDS and the supervised system for a certain privilege level on the supervised system.*

The study of HIDS independency varies depending on the privilege level achieved by the attacker on the supervised system as part of an attack. Each privilege level may expose certain elements to the attacker and this exposure may exist

Received April 1, 2008

exclusively for this privilege level.

The following sections describe how to study HIDS resiliency and HIDS independency and the relationship between them.

4 Evaluating Independency

If the HIDS is isolated from the supervised system, studying HIDS resiliency does not depend on that system and hence the problem is similar to that of studying the security of other computer systems. However, achieving complete isolation is not possible on HIDSs. At a minimum, the data collection agent must reside inside the host and in many cases, other HIDS components are shared with the supervised system.

The existence of common elements between the HIDS and the supervised system allows a path for the “reusability” of attack stages, i.e. attacking the supervised system will provide a vector to attack the HIDS.

Definition 4. *We define a HIDS as perfectly independent if no shared mechanism exists between the IDS and the supervised system for all privilege levels on the supervised system.*

A basic property of a perfectly independent HIDS is that its resiliency does not depend on the supervised system. The definition does not imply that a perfectly independent HIDS is resilient to all attacks, as other vectors to subvert the HIDS may exist without using the supervised system as the attack vector. For example, a PCI card may act as the monitor of a platform and display a very high independency. But if the PCI card features a network connection (independent from the host), it may be attacked through this connection. However, this vector of attack may be studied separately from the host.

4.1 Studying the HIDS Data Path

A study of HIDS independency should begin with clearly understanding the elements employed by the HIDS, from data collection to alert reporting. To subvert the HIDS, the attacker may attempt to launch a range of attacks against any element of the HIDS. The situation is further complicated as most HIDSs require common system elements to transfer or modify the data (e.g., network card, hard disk controller or kernel driver). Moreover, for complex systems, several redundant elements may be used for collecting and reporting, each element contains different properties. An attacker may corrupt any element in the HIDS data path from data collection to alert reporting with the objective of subverting the HIDS. A simple HIDS model [14] consists of three parts: the *agent*, which collects the data; the *director*, which corresponds to the detection engine; and the *notifier*, which reports, if necessary, the results from the director. We assume that all HIDSs feature a single director, but can consist of many agents and notifiers. As seen in Figure 1, both the agent and the notifier may be composed of further active intermediaries. We denote such active intermediaries as proxies, and the communication paths between proxies as communication channels. In Figure 1, we represent each communication channel as S_i, and each proxy as P_i. We describe each HIDS element depending on the type of resources employed by each proxy and communication channel. For example, a hard disk controller uses firmware and hardware, and a detection engine resides in memory while storing configuration files on a filesystem. An attacker may exploit a specific element by tampering with any of the shared resources used by that element. Normally, subversion attacks, which involve restarting the HIDS, will be notified. Hence we will only evaluate shared resources while the HIDS is running in a normal mode.

4.2 Defining the Privilege Levels

Once the elements are identified, the complexity needed to exploit them to subvert the HIDS should be evaluated for the different privilege levels. We assumed that no privilege level on the supervised system immediately led to any privilege level on the HIDS.

While taxonomies of access levels may differ, for this work we used the taxonomy proposed in [15]. This taxonomy refers to the privilege level on the supervised system. For example, the superuser privilege entails access to the software of the supervised system, including firmware and possibly the BIOS, and the physical access privilege level grants access to the system’s hardware. Our goal is to evaluate the effort, time and resources needed by the attacker to realize the threat, i.e., subvert the IDS in the event of an attack to the supervised system. We assumed that the more complex the modification of a specific HIDS element is, the closer the element is to being isolated from the supervised system.

4.3 Evaluating HIDS Independency

To calculate the independency score for each node, the most straightforward metric is relative cost. The use of relative cost is a common practice to evaluate IDSs [16, 17, 6]. The evaluator sets a specific and known cost baseline. If the HIDS evaluation is performed after the HIDS deployment, the simplest technique is to set the baseline to the cost for the attacker to achieve the privilege level for which we are evaluating independency in the supervised system. Once the baseline is set, we need to evaluate the relative effort for the attacker to subvert the HIDS for each element. If the element does not contain any shared resources for that privilege level, we will assign infinity (∞) as the relative cost for that element. If the HIDS evaluation was performed using another baseline in another system, the practitioner can easily modify the cost by simply specifying the ratio between both baselines. In HIDSs where attacks are detected in real or near real-time, the cost to subvert the monitor after the intrusion might be high, as the attacker only has a small window of opportunity to compromise HIDS elements before detection. The cost of subverting the HIDS after the attack is bounded in time by the monitor’s reaction time. This will increase the cost of certain attacks, e.g., brute force attacks on the HIDS administrator key. Another issue is self-monitoring: some HIDSs monitor their own elements and

Received April 1, 2008

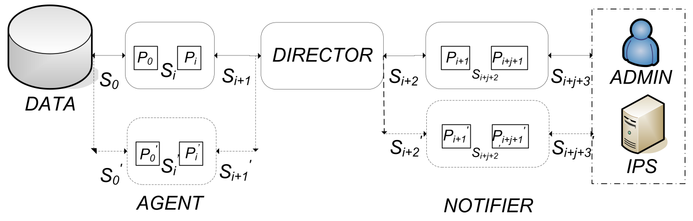

Figure 1: HIDS Data Path

consequently trigger an alarm if attacked. While performing the cost based analysis, these factors should be taken into account to increase the cost for the attacker.

After assigning a relative cost to each HIDS element for a privilege level, we defined the lowest independency relative cost assigned to an element for this privilege level as the overall HIDS independency for that privilege level, denoted as $Cost^{priv}$. If the element is redundant, i.e., another data path exists from collection to reporting, only the highest cost between redundant elements is used to assess the overall independency, as an attacker needs to subvert at least one element on each data path to subvert the HIDS. Attacks achieving a certain privilege level will exploit the shared element with the lowest independency relative cost to subvert the HIDS. Low $Cost^{priv}$ values will hence imply low resiliency values.

4.4 Case Scenario: Studying Samhaim Independency

HIDS independency is a deployment characteristic. Consequently, even for the same HIDS, the results may vary depending on how it is deployed. Our aim is not to describe the independency of the specific HIDS. Rather, this section illustrates the proposed method to use for a particular case: evaluating a host-based file integrity verifier. We estimated the independency relative costs of Samhain [**?**], an integrity verifier. Samhain was deployed on a server, with an installation of Gentoo Linux. The version of Samhain used was 2.3.1. We deployed Samhain using the default installation options.

To illustrate our method, we restricted the independency evaluation to the supervisor privilege level ($Cost^R$). The selected baseline cost is the cost to achieve root access for an outsider. In this case, the baseline models the effort of finding an unknown vulnerability in the software that runs on the supervised system, which is up to date. We will estimate the relative cost of the other attack vectors to subvert the HIDS compared to the baseline. In other words, we will estimate how much more (or less) effort is needed for the attacker to subvert the HIDS for various attack vectors compared to finding an unknown vulnerability in the software running on the supervised system. Note that the exact values of these relative costs are less important than their order of magnitude so that the most easily launched attack vector is correctly identified.

In our implementation, Samhain read the files specified by our policy, verified the integrity against a database and then logged the result on a CD, while sending a message with the logs to an email address. By default, logs, messages and configuration files were unencrypted. The HIDS elements and data path are represented in Figure 2. HD is the hard disk containing the files subject to inspection. P_0 represents the hard disk Integrated Drive Electronics (IDE) controller, P_1 the operating system kernel (IDE driver, filesystem) and both elements are part of the agent. The Samhain process acts as the detection engine. Samhain also uses a filesystem to store the configuration files. Two different notifiers are used, consequently we studied both paths. P_2 represents a smtp process, the mail server. P_3 represents the kernel network stack, network card driver and P_4 the network card. For the redundant path, P_6 is the syslogd login daemon, P_7 stands as the CD IDE controller and P_8 the filesystem on the CD. For the communication channels, S_0 represents the IDE internal registers, S_1 is the PCI Bus, S_2 is the kernel to user space communication, S_3 inter process communication, S_4 the user space to kernel communication and finally S_5 represents the internal network connection. The rest of the path is outside the host and hence independent until the notification reaches the administrator.

After we defined the data path of the HIDS, we evaluated the relative cost of each shared element for the supervisor privilege. We compared the different costs of subverting the HIDS for each element from the supervisor level. The results, shown in Table 1, are based on expert knowledge. We estimated that modifying the internal operation of hardware proxies or attacking restricted communication paths (e.g., the PCI bus) requires either insider help or an intimate knowledge of the system, so we assigned $Cost^R = 10$ (i.e., an effort 10 times higher than that of finding an unknown vulnerability in the software running on the supervised system). Many current controllers provide a mechanism to upgrade the internal firmware, hence providing an easier attack vector (e.g., replacing the firmware with a corrupted version). While the cost may fluctuate depending on the specific hardware, we set $Cost^R = 5$ for firmware based hardware.

Software is simpler to subvert: specifically there are tools that can automatically corrupt the detection engine itself, so we assigned $Cost^R = 0.3$. However, the attacker may also corrupt other proxy software, like smtp. As this requires some expert knowledge to correctly modify the communication, we estimated $Cost^R = 0.7$ for both the network OS stack and smtp.

The lowest relative cost is associated with S_2. Many rootkits exist on the Internet that are designed to modify the operating system. Therefore, user space applications will provide deceptive outputs. As rootkits are far more com-

Received April 1, 2008

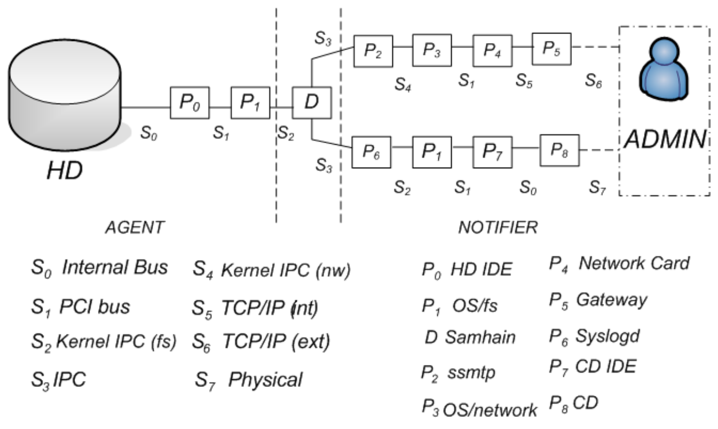

Figure 2: Samhain Data Path

mon than patches that target specific binaries, we estimated the relative cost to be very low. Hence, if no mitigation techniques are in place, we set $Cost^R = 0.1$. The existence of two redundant paths for notification does not affect the overall cost, as the lowest independency cost is associated with one of the proxies of the agent.

Elem.	Description	Resource used	$C.^R$
$Base.$	root privilege	find unknown exploit	1
P_0	HD IDE	hardware/firmware	5
P_1	fs driver	OS	0.8
D	Samhain	memory/filesystem	0.3
P_2	smtp	memory/filesystem	0.7
P_3	network driver	OS	0.8
P_4	network card	hardware/firmware	5
P_6	syslogd	memory /filesystem	0.7
P_7	CD IDE	hardware/firmware	5
P_8	CD	filesystem	∞
S_0	IDE registers	internal bus	10
S_1	PCI bus	internal bus	10
S_2	kernel IPC	OS system call	0.1
S_3	shared libraries	IPC	0.2
S_4	network stack	OS	0.7
S_5	TCP/IP (int)	network connection	5

Table 1: Independency Study for Samhain

The amount of shared resources provides the attacker with many possible attack vectors to use to subvert the HIDS. While mitigation techniques may be used, the wide range of shared elements will probably make them either impractical or insufficient. For example, encrypting and signing configuration files and renaming the executable process will increase the independency relative cost for the detection engine. But the problem of sharing kernel elements persists[1].

[1]Samhain implemented certain techniques to avoid system call redirection, but not for the tested kernel version.

5 Evaluating Resiliency

Independency is used to evaluate the relationship between the HIDS and the supervised system by identifying shared elements and their possible use as attack vectors against the supervised system. In this section we describe how to estimate HIDS resiliency based on environment factors and independency.

The two main questions we need to answer are:

1. Does the attacker care to be detected?
2. Does the attacker have the necessary skill to perform an attack against the supervised system?

The motivation for the attacker to launch an attack against the supervised system could depend on the services provided by the supervised system and the type of organization. For example, if the supervised system is an informative web server, the attacker probably will not care about being detected in the event of web site defacement. In most cases, HIDSs stand as the last layer of defense in the system. Hence access controls, firewalls and NIDSs possibly have already “cleaned ” most of the less dangerous and automated attacks. We introduce the attacker motivation as a variable θ, $0 \leq \theta \leq 1$, where $\theta = 0$ is the extreme case that none of the attackers care about being detected (and hence the resiliency will be always 1) and $\theta = 1$ all attackers care and will attempt if possible to subvert the HIDS. To consider only the worst case scenario, the security practitioner should set θ to 1. Table 2 provides some values for θ in specific environments. These values are based on expert judgment. Therefore, the order of magnitude is more important than the absolute value itself.

In the previous section, we discussed how to estimate the relative cost of various shared elements compared to the baseline of finding an unknown vulnerability in the software running on the supervised system for different privilege levels. Thus, we are making the implicit assumption that the attacker has enough skill to reach that privilege level. We now link the relative cost introduced when evaluating HIDS independency, $Cost^{priv}$, with the probability of subverting the HIDS. If $Cost^{priv} << 1$, the probability that the attacker will succeed in the attempt is quite high, as the at-

Received April 1, 2008

Value of θ	Possible Scenario
0	No subversion attempts
0.1	Unprotected home machine
0.2	Unprotected university machine
0.3	Protected home machine
0.4	Protected university machine
0.5	Small business
0.6	Large business
0.7	Government institution
0.8	Military
1	All attackers attempt subversion

Table 2: Estimated θ for Specific Environments

tacker has shown enough skill to easily subvert the HIDS, $P^{priv}_{success} \approx 1$. However, for the case of $Cost^{priv} >> 1$, the probability that the attacker will succeed is quite low, $P^{priv}_{success} \approx 0$. In the extreme case that the HIDS is perfectly independent and $Cost^{priv} = \infty$, we have $P^{priv}_{success} = 0$, as we assumed that out of band attacks (i.e. not through the host) are more costly.

When discussing the probability values of subverting the HIDS, we did not include the attacker's motivation. This motivation is assumed to be independent of the probability of subversion. When calculating the resiliency of the HIDS, P^{priv}_{res}, we combine both factors in the following equation:

$$P^{priv}_{res} = 1 - \theta P^{priv}_{success} \quad (1)$$

The overall resiliency is computed for all possible privilege levels where the resiliency is first weighted by the frequencies of attacks (the frequencies are associated with the level of privilege reached through the attack):

$$P_{res} = \sum_{priv \in S} \alpha^{priv} P^{priv}_{res} \quad (2)$$

where S is the set of possible privilege levels on the supervised system and α^s are the frequencies of attacks reaching that privilege level.

6 Integrating Resiliency into IDS Frameworks

In the previous section, we calculated the HIDS resiliency for specific environments and independency measurements. In this section we will provide practical uses of this metric by including it in a cost framework as a modifier of the probability of detection.

6.1 The Role of Resiliency in Detection Accuracy

An idealized HIDS system will correctly report the state of the system. In other words, no normal actions in the system are labeled as an alarm and any attack or intrusion is labeled as an alarm. The two main characteristics used to measure accuracy are the number of true positives in a certain universe of attacks, referred to as the probability of detection (P_D) and the number of false positives in a certain universe of valid actions collected by the HIDS, referred to as the probability of false alarms (P_{FA}). Some HIDSs provide several functional points, in the form of pairs of P_D and P_{FA}. The graphs created by pairing these functional points on a certain HIDS are called Receiver Operating Characteristics (ROC) curves.

Cost(Intrusion,Alert)	Description
C(0,0)	Normal HIDS operation
C(1,0)	Cost of not reacting to an intrusion
C(0,1)	Cost of a false alarm
C(1,1)	Cost of reacting to an intrusion

Table 3: Cost Related to HIDS Evaluation

Two conditions must exist to correctly detect an attack: the HIDS was not subverted as part of the attack and the detector correctly labeled the intrusion as an alarm. These two events are not necessarily independent. Hence, if we assume the worst case scenario where the attacker will not attempt a subversion attack in the event an attack is missed by the detector (see Figure 3):

$$Pr(Alert|Intrusion) = P_D P_{res} \quad (3)$$

and

$$\begin{aligned} Pr(Noalert|Intrusion) &= (1 - P_D) + \\ P_D(1 - P_{res}) &= 1 - P_D P_{res} \quad (4) \end{aligned}$$

6.2 Resiliency in Evaluation Frameworks

To provide the optimal functional point for an IDS and to compare different IDSs, several evaluation frameworks have been proposed [3, 17, 19]. We will integrate our proposed metric, resiliency, into the cost framework described in [16]. In [6] it was demonstrated that most of the previous frameworks can be transformed into a cost framework by setting predetermined costs.

In a cost framework, P_D and P_{FA}, as well as $1-P_D$ and $1 - P_{FA}$ are associated with relative costs. These costs define the following situations: reacting to an intrusion, missing an intrusion, raising a false alarm and not raising an alarm in normal operation, respectively. In Table 3, we summarize the meaning of the different costs. The likelihood of an intrusion, p, is introduced as an environmental variable. We can describe such a framework with a single equation as presented in [6]:

$$\begin{aligned} Cost_{IDS} &= p(C(1,0)(1 - P_D P_{res}) + C(1,1) P_D P_{res}) \\ &+ (1-p)(C(0,0)(1 - P_{FA}) + C(0,1) P_{FA}) \quad (5) \end{aligned}$$

In the cost framework, we include the resiliency as a multiplication factor of the probability of detection, P_D, as shown in equations 3 and 4. The probability of detection and the resiliency are not independent, because we assumed that the attacker will only attempt a subversion attack if s/he is aware that the detection engine will detect the activity.

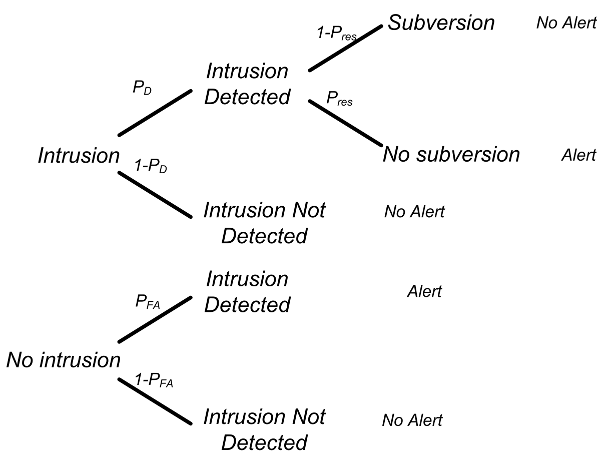

Figure 3: Detection Tree

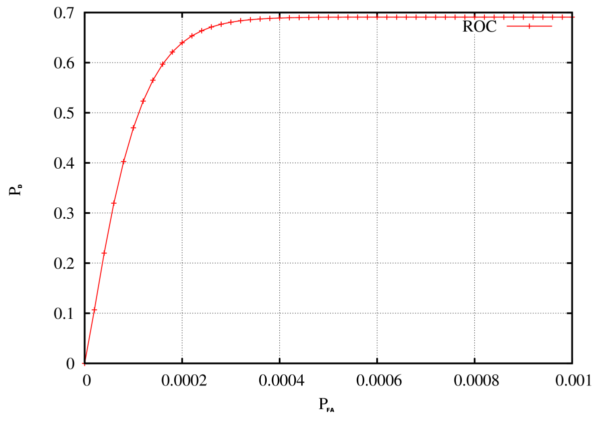

Figure 4: Sample ROC

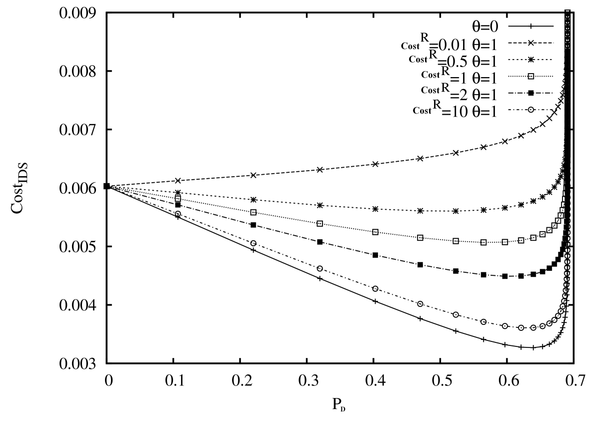

Figure 5: $Cost_{IDS}$ for different independency relative costs. The related costs are: $C(0,0) = 3 \times 10^{-5}$, $C(0,1) = 5$, $C(1,1) = 2$, $C(1,0) = 100$

6.3 Numerical Analysis: Estimating the Optimal Operational Point of a HIDS

A key problem in IDSs is, given a certain ROC data set, to find the optimal operational point given a certain environment, $Cost^*_{IDS} = \min_{(P_D, P_{FA}) \in ROC} Cost_{IDS}$. This section illustrates how different values of $Cost^{priv}$ will modify the value of $Cost_{IDS}$. For demonstration purposes, we will use ROC curves and attack probability values from the DARPA 1998 data set [20]. Figure 4, shows the ROC curve we use for the analysis.

Let us suppose the costs $C(Intrusion, Alert)$ are constant and provided. We set the value of the probability of intrusion, p, $p = 6.25 \times 10^{-5}$ as published in the DARPA evaluation. We also suppose that all attackers care about being detected and will attempt to subvert the IDS, $\theta = 1$. In the previous section, for a given privilege level, we described how the probability of a subversion attack succeeding, $P^{priv}_{success}$, was linked to the relative cost compared to the baseline of finding an unknown vulnerability in the software running on the supervised system, $Cost^{priv}$. We introduce the following simple function as an estimate of $P^{priv}_{success}$ that has the properties described in the previous section:

$$P^{priv}_{success} = \frac{1}{1 + Cost^{priv}} \tag{6}$$

The optimal operation point of this IDS, disregarding resiliency, is $P_D = 0.6534, P_{FA} = 0.00022$, with an expected cost of $Cost_{IDS} = 0.00335$ for the HIDS. If we now include P_{res} in the analysis, we find that for different values of $Cost^R$ (we suppose the frequency of attack in privilege levels other than root are negligible), the optimal operational point changes as presented in Figure 5.

Table 4 provides the optimal probability of detection and the associated cost for the different values of $Cost^R$.

$Cost^R$	P_D	$Cost_{IDS}$
0.01	0	0.0061
0.5	0.5234	0.0056
1	0.5649	0.0051
2	0.6213	0.0045
10	0.6397	0.0036

Table 4: P_D and $Cost_{IDS}$ for sample $Cost^R$

The HIDS is unusable for $Cost^R = 0.01$: the low in-

Received April 1, 2008

dependency does not provide a cost benefit compared to not having a HIDS, for any operation point of the provided ROC curve. This fact demonstrates the importance of the study of independency, indicating that a HIDS with a good detection engine will only benefit the organization if it is not easily subverted. For the other HIDSs, a different optimal operational point should be used to compensate for the reduction in the probability of detection and the increased comparative weight of the probability of false alarms. This solution is of course dependent on the assigned values for cost, but the importance of resiliency should be stressed as we have chosen a very low p. Increasing the value of p will provide even more extreme results, as increasing the number of attacks will provoke a spike in the number of subversion attempts. In this case we assumed the worst case scenario, with $\theta = 1$. However, we believe it makes sense for HIDSs to evaluate the operational point on the pessimistic side.

7 Conclusions

In this work we introduced HIDS subversion as a technique to circumvent HIDSs. Two metrics were proposed to evaluate the HIDS strength against subversion. HIDS independency is a qualitative, attack-independent metric which provides a measure of the isolation between the HIDS and the supervised system. HIDS resiliency is a quantitative, attack-dependent metric, which includes environmental attributes of the deployment scenario. We showed the feasibility of measuring these characteristics by performing an analysis on an example system.

This study of resiliency demonstrates that traditional methods of evaluating HIDS that only consider accuracy are incomplete. Furthermore, by introducing the independency property and corresponding metrics, we showed the feasibility and importance of quantifying HIDS characteristics other than accuracy.

Along with providing metrics to compare independency between HIDSs, the results also provide a necessary insight on the weaknesses that can be exploited by a motivated attacker. This study, however, has limitations, which we will try to overcome in future research. We assumed that external entities were independent, and hence they feature perfect resiliency. As a result, if our techniques are to be applied to NIDSs, this will result in no modification to their probability of detection. This in our opinion is consistent with reality: One of the reasons NIDSs are superior to HIDSs is their independency of the system being supervised.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 0237493.

Bibliography

[1] Kim, G.H., Spafford, E.H.: The design and implementation of tripwire: A file system integrity checker. In: Proc. of the Conference on Computer and Communications Security (ACM '94). (1994) 18–29

[2] Halflife: Bypassing integrity checking systems. Phrack Magazine **7, Issue 51**(51) (1997)

[3] Axelsson, S.: The base-rate fallacy and the difficulty of intrusion detection. ACM-Transactions on Information and System Security -TISSEC **3**(3) (2000) 186–205

[4] Mell, P., Hu, V., Lippmann, R., Haines, J., Zissman, M.: An overview of issues in testing intrusion detection systems. Technical Report NIST IR 7007, National Institute of Standards and Technology (2003)

[5] Fink, G., O'Donoghue, K.F., Chappell, B.L., Turner, T.G.: A metrics-based approach to intrusion detection system evaluation for distributed real-time systems. In: Proc. of the International Parallel and Distributed Processing Symposium (IPDPS '02). (2002) 204–228

[6] Cardenas, A.A., Baras, J.S., Seamon, K.: A framework for the evaluation of intrusion detection systems. In: Proc. of the IEEE Symposium on Security and Privacy (IEEE security'06), Washington, DC, USA, IEEE Computer Society (2006) 63–77

[7] ISO/IEC Standard 15408: Common Criteria for Information Technology Security Evaluation. version 2.0 edn. (1998)

[8] Chen, Y.: Protection profile analysis and security target for an intrusion detection system. Master's thesis, Lulea University (2003)

[9] Corporation, S.A.I.: Intrusion detection system scanner protection profile. Technical report (2005)

[10] Enterasys: Enterasys dragon eal-intrusion defense system. Technical report (2004)

[11] Molina, J., Arbaugh, W.A.: Using independent auditors as intrusion detection systems. In: Proc. of the 4th International Conference on Information and Communications Security (ICICS'02), Singapore (2002) 291–302

[12] Litty, L.: Hypervisor-based intrusion detection. Master's thesis, University of Toronto (2005)

[13] Trusted Computing Group: Trusted computing platform alliance (TCPA) main specification, version 1.1b. Technical report (2003)

[14] Bishop, M.: Computer Security: Art and Science. Addison Wesley, New York (2002)

[15] Weber, D.J.: A taxonomy of computer intrusions. Master's thesis, Massachusetts Institute of Technology (1998)

[16] Gaffney, J.E., Ulvila, J.W.: Evaluation of intrusion detectors: A decision theory approach. In: Proc. of the Symposium on Security and Privacy (IEEE security'01). (2001) 50–71

[17] Lee, W., Fan, W., Miller, M., Stolfo, S.J., Zadok, E.: Toward cost-sensitive modeling for intrusion detection and response. Journal of Computer Security **10**(1/2) (2002) 5–22

[18] Wotring, B.: Host Integrity Monitoring Using Osiris and Samhain, Rockland, MA (2005)

[19] Gu, G., Fogla, P., Dagon, D., Lee, W., Skoric, B.: Measuring intrusion detection capability: an information-theoretic approach. In: Proc. of the ACM Symposium on Information, Computer and Communications Security (ASIACCS '06), New York, NY, USA, ACM Press (2006) 90–101

[20] Lippmann, R., Haines, J.W., Fried, D.J., Korba, J., Das, K.: The 1999 DARPA off-line intrusion detection evaluation. Computer Networks **1**(34) (2000) 579–595

Author Biographies

Jesus Molina is a researcher at Fujitsu Laboratories of America. He holds a PhD and a MS in Electrical Engineering from the University of Maryland. He has previously worked at the Maryland Information Systems Security Lab developing advanced hardware based intrusion detection systems. He currently chairs of the authentication working group at the Trusted Computing Group. His research interests include intrusion detection development and evaluation, trusted computing and authentication systems.

Michel Cukier is an Assistant Professor at the Center for Risk and Reliability in the Mechanical Engineering Department at the University of Maryland. He earned his doctorate from LAAS-CNRS, Toulouse, France in 1996 on coverage estimation of fault-tolerant systems. From 1996 to 2001, he was a member of the Perform research group in the Coordinated Science Laboratory at the University of Illinois, Urbana-Champaign, working on adaptive fault tolerance and intrusion tolerance. His current research interests include security evaluation, intrusion tolerance, distributed system validation and fault injection.

Received April 1, 2008