Intrusion Response Systems: Survey and Taxonomy

Alireza Shameli-Sendi, Naser Ezzati-jivan, Masoume Jabbarifar, and Michel Dagenais

Department of Computer and Software Engineering
École Poytechnique de Montréal, Montreal, Canada

Summary
This paper presents a taxonomy of intrusion response systems (IRS), classifying a number of research papers published during the past decade that provide us with many valuable insights into the field of Internet security. In recent years, we have seen impressive changes in how attackers gain access to systems and infect computers. We discuss the key features of IRS that are crucial for defending a system from attack. Choosing the right security measures and responses is an important and challenging part of designing an IRS. If we fail to do so, our automated response systems will reduce network performance and wrongly disconnect users from a network. We address this challenge here, and introduce the concept of "response cost", in an attempt to meet users needs in terms of quality of service (QoS) and the interdependency of critical processes. This taxonomy will open up interesting areas for future research in the growing field of intrusion response systems.

Key words:
Intrusion, Response system, Security, Taxonomy, Risk assessment, Prediction, Response cost

1. Introduction

Our use of software systems, information systems, distributed applications, etc. is continuously growing in size and complexity [6]. Today, cyber attacks and malicious activities are common problems in distributed systems, and they are rapidly becoming a major threat to the security of organizations. It is therefore crucial to have appropriate Intrusion Detection Systems (IDS) in place to monitor, trace, and analyze system execution. Only then can we hope to identify performance bottlenecks, malicious activities, programming functional, and other performance problems [4]. Intrusion Response Systems (IRS), by contrast, continuously monitor system health based on IDS alerts, so that malicious or unauthorized activities can be handled effectively by applying appropriate countermeasures to prevent problems from worsening and return the system to a healthy mode. Unfortunately, IRS receives considerably less attention than IDS [15].

Usually, the attacker exploits security goals: the confidentiality and integrity of data, and the availability of service (referred to as *CIA*), by targeting vulnerabilities in the form of flaws or weak points in the security procedures, design, or implementation of the system [2], [42]. The main issue in choosing a security measure is to correctly identify the security problem. For example, we do not want to isolate a whole server from a network on which many services are installed, nor do we want to kill processes that are using a considerable amount of CPU resources unless we are sure they are being attacked. Thus, implementing an appropriate algorithm in IDS and IRS, and choosing the right set of responses, must take into account whether or not the network is being attacked with a very high positive value. It is essential that we counter attacks with new features, a complete list of responses, accurate evaluation of those responses in a network model, and an understanding of the cost of each response in every network element. If we fail to do so, our automated IRS will needlessly reduce network/host performance, wrongly disconnect users from the network/host, and eventually result in a DoS attack on our network. We must, therefore, establish a tradeoff between slowing down system performance and maintaining maximum security [22].

In this paper, we propose a taxonomy of IRS and present a review of existing IRS. Our aim in the paper is to identify the weaknesses of IRS and propose guidelines for improve them.

The rest of this paper is organized as follows: in Section 2, we propose our taxonomy of IRS and describe their main elements. A review of recent existing IRS is presented in Section 3. Section 4, we discuss the current state of the intrusion response field, and suggestions for future research which can improve the current weaknesses of IRS. Finally, in Section 5, we present our conclusions.

2. A taxonomy of intrusion response systems

Depending on their level or degree of automation, IRS can be categorized as:

- ***Notification systems:*** These systems mainly generate alerts when an attack is detected. An alert can contain information about the attack, such as attack description, time of attack, source IP, user account, etc. The alerts are then used by the administrator to select the reactive measures to apply, if any. This approach is not designed to prevent attacks or return system to a safe mode. The major challenge in this approach is the delay between the intrusion and the human response.

Manuscript received January 5, 2012
Manuscript revised January 20, 2012

- ***Manual response systems:*** In these systems, there are some preconfigured sets of responses based on the type of attack. A preconfigured set of actions is applied by the administrator when a problem arises. This approach is more highly automated than the notification system approach.
- ***Automated response systems:*** These systems are designed to be fully automated, so that no human intervention is required, unlike the two methods described above, where there is a delay between intrusion detection and response. One of the major problems with this approach is the possibility that an inappropriate response will be executed when a problem arises. Another challenge with executing an automated response is to ensure that the response is adequate to neutralize the attack. The characteristics of this approach are depicted in Figure 1, and are the following:

2.1. IRS input

IDS are tools that monitor systems for signs of malicious activities. They are closely related to automated fault identification tools. We use network-based IDS (NIDS) to monitor the network and host-based IDS (HIDS) to monitor the health of a system locally [21], [28], [30], [36], [20].

IDS are divided into two categories: *anomaly-based*, and *signature-based*. In anomaly-based techniques, a two step process is employed. In the first step, called the training phase, a classifier is extracted using a popular algorithm, such as a Decision Tree, a Bayesian Network, a Neural Network, etc. [19], [35], [37]. The second step, the testing phase, concentrates on classifier accuracy. If the accuracy meets our threshold, it can be used to detect anomalies. Anomaly-based detection is able to detect unknown attack patterns and does not need predefined signatures. However, it is difficult to define normal behavior, and the malicious activity may look like a normal usage pattern. In signature-based techniques (also known as misuse detection) [41], we compare captured data with well-defined attack patterns. Pattern matching makes this technique deterministic, which means that it can be customized for every system we want to protect, although it is difficult to find the right balance between precision, which could lead to false negatives, and genericity, which could lead to false positives [33], [53]. Moreover, signature-based techniques are stateless. Once an attack matches a signature, an alert is issued and the detection component does not record it as a state change. One solution to the limitation of detection based only on stateless signatures is to use a finite state machine (FSM) to track the evolution of an attack [4]. That way, while an attack is in progress, the state changes and we can trigger appropriate responses based on a confidence level threshold, which would result in a lower false positive rate. The detection component has all the detailed information about the malicious activity, such as severity, confidence level, and the type of resource targeted. The output of the detection component is based on the Intrusion Detection Message Exchange Format (IDMEF) [44]. This is a standard that can be used to report alerts about attacks or malicious behaviors. Briefly, each alert embodies the following:

- *Analyzer Identification:* the analyzer that originated the alert.
- *Create Time:* the time at which the alert was created.
- *Detect Time:* the time at which the event(s) leading up to the alert occurred.
- *Analyzer Time:* the current time on the analyzer.
- *Source:* the source of the event leading up to the alert, including Node, User, Process, and Service.
- *Target:* the intended victim of the event leading up to the alert, including Node, User, Process, Service, and File.
- *Classification:* name and description of the alert.
- *Assessment:* consisting of three fields (Impact, Action, and Confidence):

    - *Impact:* This field shows the analyzers assessment of the events impact on the target. The Impact field has three attributes: *Severity*, *Completion*, and *Type*. The severity attribute value can be *high*, *medium*, or *low*, and is very important information for the prediction component, as explained in the prediction section. The completion attribute indicates whether or not the attack was successful, and so its value can be failed or successful. If we want to detect the progress of the attack early on, an FSM can send an alert for each state reached. Thus, the completion attribute of all the alerts generated while the attack is in progress will be recorded as failed. Only the final alert of each FSM execution will earn the successful completion value. The type attribute indicates the nature of the attempt related to the alarm.

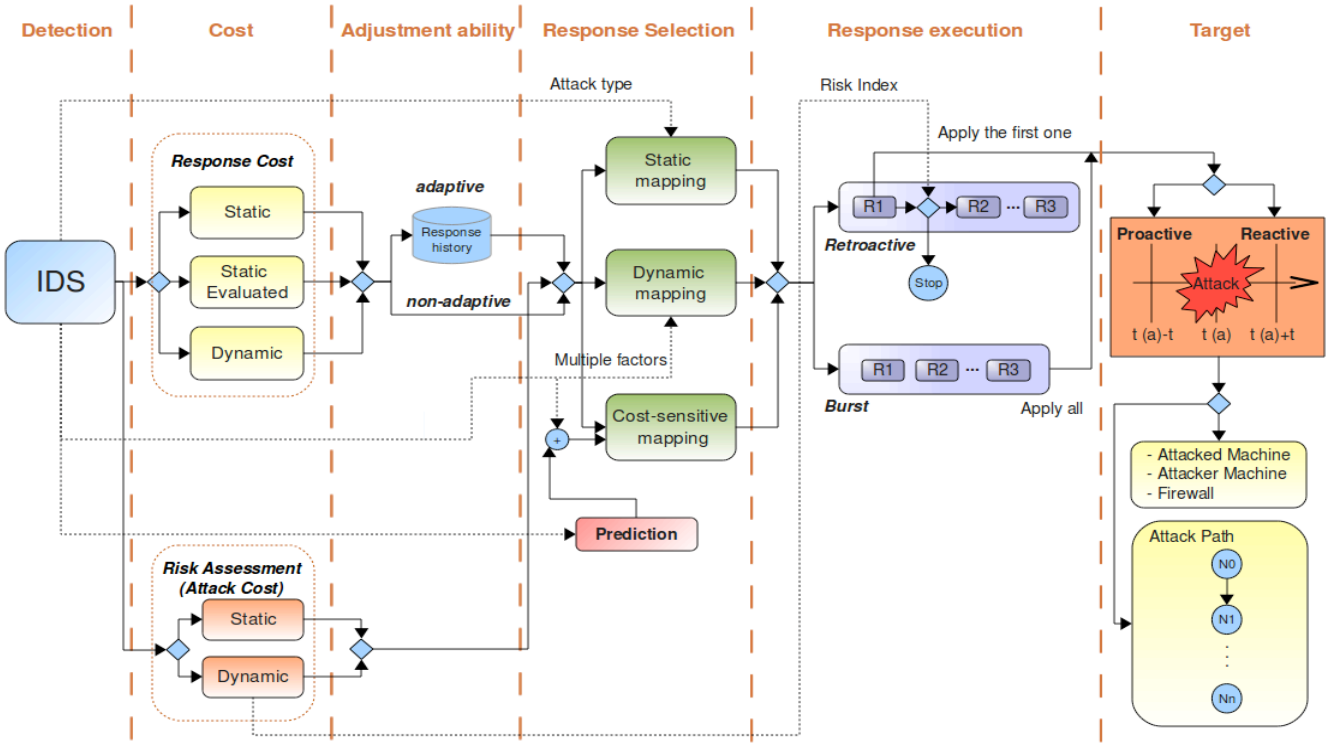


Fig. 1 Taxonomy of Intrusion Response Systems.

- *Action:* This field is filled in if the IDS detects an attack and reacts to it. Otherwise, it will be left blank.
- *Confidence:* This field reflects the validity of the analyzer estimation. Its value can be *low*, *medium*, or *high*. However, different values can be assigned to it. For example, in the FSM mechanism, a weight can be associated with each state, the sum of all the weights being 100. Confidence in this case means confidence level. The confidence level related to each alert is equal to the sum of the weights of all the states previously seen.

2.2. Response cost model

Response cost evaluation is a major part of the IRS. Although many automated IRS have been proposed, most of them use statically evaluated responses, avoiding the need for dynamic evaluation. However, the static model has its own drawbacks, which can be alleviated by designing a dynamic evaluation model for the responses. Dynamic evaluation will also more effectively protect a system from attack, as threats will be more predictable. Verifying the effect of a response in both dynamic mode and static mode is a challenge, as accurate parameters are required to evaluate that response. If, for example, we have an Apache process under the control of an attacker, this process is now a gateway for the attacker to access our network. The accepted countermeasure would be to kill this hijacked process that has become potentially dangerous. When we apply this response, we will increase our data confidentiality and integrity (*C* and *I* of *CIA*) if the process was doing some damage on our system. But, the negative impact is that we lose Apache availability (*A* of *CIA*), since our Web server is now dead and our website is down. Let us imagine another scenario, where we have a process on a server consuming a considerable amount of CPU resources that is doing nothing but slowing down our machine (a kind of CPU DoS). This time, killing the process will improve service availability (system performance), but will not change anything in terms of data confidentiality and integrity. We now have two very different results for the same response. Also, some of the responses effects depend on the network infrastructure. For example, applying a response inside the external DMZ is probably very different from doing so inside the LAN or "secure zone" in terms of *CIA*. Responses cannot be evaluated without considering the attacks themselves, which are generally divided into the following four categories [13], [23]:

1. ***Denial of service (DoS):*** The attacker tries to make resources unavailable to their intended users, or consume resources such as bandwidth, disk space, or processor time. The attacker is not

looking to obtain root access, and so there is not much permanent damage.

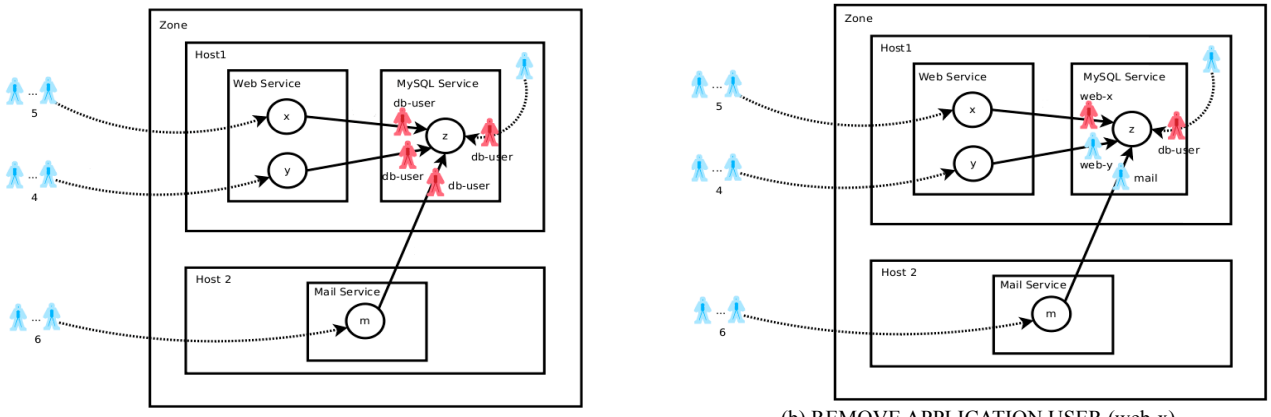
2. ***User to root (U2R):*** An individual user tries to obtain root privileges illegally by exploiting system vulnerabilities. The attacker first gains local access on the target machine, and then exploits system vulnerabilities to perform the transition from user to root level. After acquiring root privileges, the attacker can install backdoor entries for future exploitation and change system files to collect information [64].
3. ***Remote to local (R2L):*** The attacker tries to gain unauthorized access to a computer from a remote machine by exploiting system vulnerabilities.
4. ***Probe:*** The attacker scans a network to gather information and detect possible vulnerabilities. This type of attack is very useful, in that it can provide information for the first step of a multi-step attack. Examples are using automated tools such as ipsweep, nmap, portsweep, etc.

In the first category, where the attacker is slowing down our system, we are looking for a response that can increase service availability (or performance). In the second and third categories, since our system is under the control of an attacker, we are looking for a response that can increase data confidentiality and integrity. In the fourth category, attackers are attempting to gather information from the network and about possible vulnerabilities. Thus, responses that improve data confidentiality and service availability are called for in this case. A dynamic response model offers the best response based on the current situation of the network, and so the positive effects and negative impacts of the responses must be evaluated online at the time of the attack. Evaluating the cost of the response in online mode can be based on resource interdependencies, the number of online users, the users privilege level, etc. There are three types of response cost model:

1. ***Static cost model:*** The static response cost is obtained by assigning a static value based on expert opinion. So, in this approach, a static value is considered for each response (RC_s = *CONSTANT*).
2. ***Static evaluated cost model:*** In this approach, a statically evaluated cost, obtained by an evaluation mechanism, is associated with each response (RC_{se} = *f(x)*). The response cost in the majority of existing models is statically evaluated. A common solution is to evaluate the positive effects of the responses based on their consequences for the confidentiality, integrity, availability, and performance metrics. To evaluate the negative impacts, we can consider the consequences for the other resources, in terms of availability and performance [11], [12]. For example, after running a response that blocks a specific subnet, a Web server under attack is no longer at risk, but the availability of the service has decreased. After evaluating the positive effect and negative impact of each response, we then calculate the response cost. One solution is as follows [17], obviously the higher *RC*, the better the response in ordering list:

$$RC_{se} = Positive_{effect} / Negative_{impact} \quad (1)$$

3. ***Dynamic evaluated cost model:*** The dynamic evaluated cost is based on the network situation (RC_{de}). We can evaluate the response cost online based on the dependencies between resources and online users. For example, the consequences of terminating a dangerous process varies with the number of interdependencies of other resources on the dangerous process and with the number of online users. If the cost of terminating the process is high, maybe another response would be better. Evaluating the response cost respect to the resource dependencies, the number of online users, and the user privilege level leads us to have an accurate cost-sensitive response system. The following example will explain why the response effect has to be calculated based on resource dependencies. Let us imagine two scenarios: 1) all services (web and mail) are using the MySQL shared user application (db-user) Figure 2a; and 2) all services (web and mail) are using a separate user application (web-user and mail-user) Figure 2b. If the web services in scenario 1 are attacked and we remove db-user when the attack is detected, it is obvious that web and mail processes cannot continue to run . In contrast, if the web services in scenario 2 are attacked and we remove web-user, the mail process and other web service processes will be unaffected. Thus, in the first scenario, where all the services are using the same MySQL user, selecting other locations (based on the attack path such as a firewall point or web server point) or other responses, are the better options. Thus, resource dependency model improves IRS in terms of their ability to apply appropriate responses, while meeting users needs in terms of QoS and the interdependencies of critical processes. The majority of the proposed IRS uses Static Cost or Static Evaluated Cost models, as Table 1 in Section 3 illustrates.



(a) REMOVE APPLICATION USER (db-user)

(b) REMOVE APPLICATION USER (web-x)

Fig. 2 Two scenarios of in which the application user is removed

2.3. Adjustment ability

There are two types of adjustment model: 1) *non-adaptive*; and 2) *adaptive*. In the non-adaptive model, the order of the responses remains the same during the life of the IRS software. In fact, there is no mechanism for tracing the behaviors of the deployed responses. In the adaptive model, the system has the ability to automatically and appropriately adjusts the order of the responses based on response history [15]. We can define a *Goodness* (*G*) metric for each response. Goodness is a dynamic parameter that represents the history of success (*S*) and failure (*F*) of each response for a specific type of host [14]. This parameter guarantees that our model will be adaptive and helps the IRS to prepare the best set of responses over time. The following procedure can be used to convert a non-adaptive model to an adaptive one [14]:

$$G(t) = S - F$$
$$R_{effectivness}(t_0) = (RC_s \mid RC_{se} \mid RC_{de}) * G(t)$$
$$R_{effectivness}(t) = R_{effectivness}(t-1) * G \qquad (2)$$

One way to measure the success or failure of a response, or a series of responses, is to use the result of the online risk assessment component. We discuss this in the "Response execution" section. Now, *G* can be calculated as proposed in [14]: if the selected response succeeds in neutralizing the attack, its success factor is increased by one, and if it fails, that factor is decreased by one. The important point to bear in mind is that the most recent results must be considered more valuable than earlier ones. Let us imagine an example where the results of *G* and *F* for a response are *10* and *3* respectively, the most recent result being *F=3*. Unfortunately, although *G=7* indicates that this response is a good one, and it was appropriate for mitigating the attack, over time and with the occurrence of new attacks, this response is not sufficiently strong to stage a counter attack.

2.4. Response selection

There are three response selection models:

1. ***Static mapping:*** An alert is mapped to a predefined response. This model is easy to build, but its major weakness is that the response measures are predictable.
2. ***Dynamic mapping:*** The responses of this model are based on multiple factors, such as system state, attack metrics (frequency, severity, confidence, etc.), and network policy [5]. In other words, responses to an attack may differ, depending on the targeted host, for instance. One drawback of this model is that it does not learn anything from attack to attack, so the intelligence level remains the same until the next upgrade [10], [46].
3. ***Cost-sensitive mapping:*** This is an interesting technique that attempts to attune intrusion damage and response cost [16], [17]. Some cost-sensitive approaches have been proposed that use an offline risk assessment component, which is calculated by evaluating all the resources in advance. The value of each resource is static. In contrast, online risk assessment component can help us to accurately measure intrusion damage. The major challenge with the cost-sensitive model is the online risk assessment and the need to update the cost factor (risk index) over time.

2.5. Response execution

There are two types of response execution:

1. ***Burst:*** In this mode, there is no mechanism to measure the risk index of the host/network once the response has been applied. Its principal weakness is the performance cost, as all the responses are applied when a subset may be enough to neutralize the attack. The majority of the proposed IRS use burst mode to execute responses.
2. **Retroactive:** there is a feedback mechanism which can measure the response effect based on the result of the most recently applied response, the idea being to make a decision before applying the next in a series of responses. There are some challenges that must be addressed if this mode is to be used in the adaptive approaches; for example, how to measure the success of the most recently applied response, and how to handle multiple occurrences of malicious activities [18]. As shown in Figure 1, we have to measure the risk index after running each response. The risk assessment component can help us do this, but the difficulty is that the risk assessment must be conducted online. Retroactive approach is firstly proposed in [17]. We have named it retroactive. As mentioned, the idea is to have a decision-making before applying the next response in a set of responses. There are a number of ways to implement the retroactive approach, among them the following: 1) *Use a response selection window:* the first idea that firstly proposed in [17] is using response selection window. Every response has a static risk threshold associated with it. The permission to run each response corresponds to the current risk index of the network. When the risk index is higher than the static threshold of the response, the next response is allowed to run. With a response selection window, the most effective responses are selected to repel intrusions. 2) *Run responses independently:* This is a simple idea, which involves measuring the risk index of one response, to make a decision about the next one. 3) *Group responses:* This is a good idea if measuring the risk index of a single response does not provide enough information to make the decision about running the next response and cannot be applied in a production environment. It involves defining a round-based response mechanism. Figure 3 illustrates six responses to a specific malicious activity which are ready in the pending queue before the start of the first round. Whether or not to run the next round of responses is based on the risk index of the network. Once a round of responses has been run, a new risk index is measured by the Online Risk Assessment component after a specific delay. As shown in Figure 3, every response has a Response Effectiveness, which defines how the selected response is ordered in the pending queue. Figure 4 shows two possible scenarios for consideration after the first round of responses has been launched. In the first scenario, the risk index of the network decreases, so the next round is not required. With this knowledge, the network can be prevented from being overly impacted. In contrast, in the second scenario, the risk index shows that malicious activity is continuing, in spite of the application of the first round of responses. In this case, the second round of responses has to be applied. There are some challenges to be overcome here. The first is to determine how many responses in a round is considered enough to neutralize an attack. Is the number sufficient to avoid having to run the next round and overly impact the network? Is the number sufficient to accurately measure the risk index? Clearly, it would be helpful to define some attributes for the responses, in order to analyze them better and order them more effectively. The responses with fewer characteristics could be placed in a group and applied as a group. Unfortunately, there is no strong attack dataset available for testing the ideas of IRS researchers [40]. This problem is common to all security researchers. Such a dataset would enable us to determine whether or not one round of responses is enough or if the number of responses in a round is sufficient to neutralize an attack. This was also a challenge in [17], as the authors could not establish the strength of their proposed model.

2.6. Prediction and risk assessment

As we know, an IDS or individual detection components usually generate a large number of alerts, and so the output of an IDS is stream data, which is temporally ordered, fast changing, potentially infinite, and massive. There is not enough time to store these data and rescan them all as static data [37], [38], [39]. Thus, if we connect the detection component to the intrusion response component. After a few hours, the impact on our network is huge, and results in a DoS. The goal of designing prediction and risk assessment components is to help response systems to be more intelligent in terms of preventing the problem from growing and in returning the system to a healthy mode. Since the output of an IDS is stream data, prediction and risk assessment components must cope with these data, and we have to find appropriate algorithms to deal with them. These algorithms are used in IRSs, and their components are the following:

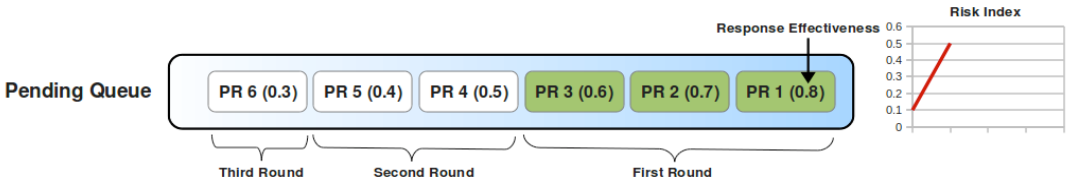


Fig. 3 Ordered pending responses before the start of the first round.

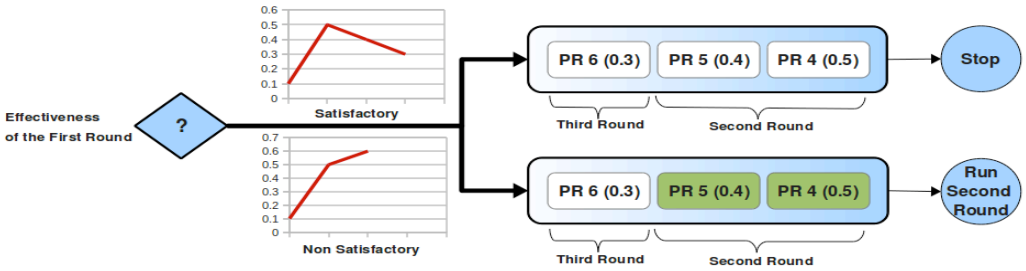


Fig. 4 Two possible outcomes for decision-making after the first round of responses has been run.

2.6.1. Prediction

In the prediction view, we have two types of IRS: 1) *Reactive*; and 2) *Proactive* [15], [51]. In the reactive approach, all responses are delayed until the intrusion is detected. The majority of IRS use this approach, although this type of IRS is not useful for high security. For example, suppose the attacker has been successful in accessing a database and has illegally read critical information. Then, the IDS sends an alarm about a malicious activity. In this case, a reactive response is not useful, because the critical information has already been disclosed. In general, the disadvantages of a reactive response are the following [51]:

- It is applied when an incident is detected, so the system remains in the unhealthy state it was in before the detection of the malicious activity until the reactive response is applied.
- It is sometimes difficult to return the system to the healthy state.
- The attacker has the benefit of time between the start of the malicious activity and the application of the reactive response.
- It takes more energy to return the system to the healthy state than to maintain it in that state.
- Since it is applied after an incident is detected, the system is exposed to greater risk of damage.

In contrast, the proactive approach attempts to control and prevent a malicious activity before it happens, and plays a major role in defending hosts and networks. A number of different schemes that predict multi-step attacks have been proposed. Some researchers have inserted the prediction step in the detection component. For example, the authors of [34] believed that, since existing solutions are only able to detect intrusions when they occur, either partially or fully, it is difficult to block attacks in real time. So they proposed a prediction function based on Dynamic Bayesian Networks, with a view to predicting the goals of intruders. Other researchers have worked on prediction algorithms based on detection output. In this method, detection components are distributed across a network and alerts are sent to the prediction component. Of course, there may be aggregation and correlation components between the detection and prediction components to reduce the number of false positives. Yu and Frincke [29] and Shameli-Sendi et al. [3] proposed the *Hidden Colored Petri-Net (HCPN)* and *Alert Severity Modulating* respectively to predict the intruders next goal. While most researchers use alert correlation to differentiate true alerts from alerts generated by detection components, called the *Alert Filtering* approach, the authors of [29] and [3] have taken a different approach. They maintain that, while multi-step attack actions are unknown, they may be partially detected and reported as alerts. They also maintain that all alerts can be useful in prediction, as the task of alert correlation is not only to find good alerts or to remove alerts.

2.6.2. Risk assessment

Again, most IDS generate a huge number of alerts over time. A large number of these alerts are duplicates and false positives [20], [61]. Many schemes have been proposed to overcome these weaknesses, some of which use an alert aggregation mechanism to reduce the number of alerts [20]. Others use an alert correlation mechanism to extract attack scenarios [26], [27], while a third group is attempting to assess the threat of intrusion [18], [23], [54], [55]. Also, alert information has only the severity field

(IDMEF format), which does not allow for a comprehensive description of the risk assessment or the level of threat. Risk assessment is the process of identifying and characterizing risk. In other words, risk assessment helps the IRS component determine the probability that a detected anomaly is a true problem and can potentially successfully compromise its target [18].

Thus, there are two types of risk assessment: 1) *static:* many researchers use offline risk assessment in IRS, assigning a static value to every resource in the network. Offline risk assessment has been reviewed in the Information Security Management System (ISMS) standards that specify guidelines and a general framework for risk assessment. It is described in many existing standards, such as NIST and ISO 27001 [43], [56]. Although they cannot satisfy the requirements of the online risk assessment environment, these standards are nevertheless fundamental and useful [2]. 2) dynamic*:* online risk assessment is a real time process of evaluation and provides a risk index related to the host or network [31]. Online risk assessment is very important in terms of minimizing the performance cost incurred. It does this by applying a subset of all the available sets of responses when that may be enough to neutralize the attack. In the second model, we can dynamically evaluate attack cost by propagating the impact of confidentiality, integrity and availability through service dependencies model or attack graph [1], [49], [66] or by general model based on attack metrics [18], [23], [3]. The type of IDS that works based on tracers [4] is capable of improving its analysis results by adding a "system state" feature [25]. A system state database provides a view of the state of each host, including CPU usage, memory usage, disk space, and a resource graph showing the number of running processes, the number of running threads, memory maps, file descriptors, etc. In fact, without knowledge of the state of the system, a real and accurate online risk assessment is impossible. So, an online response system that supports the system state would be a very novel model.

2.7. Response deactivation

The need to deactivate a response action is not recognized in the majority of existing automated IRS. The importance of this need was first suggested in [6]. These authors believe that most responses are temporary actions which have an intrinsic cost or induce side effects on the monitored system, or both. The question is how and when to deactivate the response. The deactivation of policy-based responses is not a trivial task. An efficient solution proposed in [6] is to specify, two associated event-based contexts for each response context: *Start (response context)*, and *End (response context)*. The risk assessment component can also help us decide when a countermeasure has to be deactivated. In [6], countermeasures are classified into one of two categories, in terms of their lifetime: 1) *One-shot countermeasures*, which have an effective lifetime that is negligible. When a response in this category is launched, it is automatically deactivated; and 2) *Sustainable countermeasures*, which remain active to deal with future threats after a response in this category has been applied.

2.8. Attack path

The majority of existing automated IRS apply responses on the attacked machine, or the intruder machine if it is accessible. By extracting the *"attack path"*, we can identify appropriate locations, those with the lowest penalty cost, for applying them. Moreover, responses can be assigned to calculate the dynamic cost associated with the location type, as discussed in the "Response cost model" Section. The numerous locations and the variety of responses at each location will constitute a more effective framework for defending a system from attack, as its behavior will be less predictable. An attack path consists of four points: 1) the *start point*, which is the intruder machine; 2) the *firewall point*, which includes firewalls and routers; 3) the *midpoint*, which includes all the intermediary machines that the intruder exploits (through vulnerabilities) to compromise the target host; and 4) the *end point*, which is the intruders target machine. Although, research on the attack path has been carried out and some ideas as to its usefulness have been formulated [58], [63], [65], it has rarely been implemented in an IDS or IRS.

3. Classification of existing models

In this section we discuss recent IRS and provide a summary of all the proposed IRS of interest in Table 1, which presents their detailed characteristics as is given in [15]. Curtis et al. [7], [8], [5] propose a complex dynamic mapping based on an agent architecture (AAIRS). In AAIRS, multiple IDS monitor a host and generate alarms. The alarms are first processed by the Master Analysis agent. This agent indicates the confidence level of the attack and passes it on to an Analysis agent, which then generates a response plan based on degree of suspicion, attack time, attacker type, attack type, attack implications, response goal, and policy constraints. Lee et al. [13] propose a cost-sensitive model based on three factors: 1) operational cost, which refers to the cost of processing the stream of events by IDS; 2) damage cost, which refers to the amount of damage to a resource caused by an attacker when the IDS is ineffective; and 3) response cost, which is the cost of applying a response when an attack is detected. The authors focus on the DARPA 1998 dataset, which is based on network connections. The resources that are being attacked in this dataset are network services and applications on some hosts. Damage and response costs have been statically defined based on four categories (ROOT, R2L, DoS, and PROBE).

Table 1: Classification of existing IRSs based on proposed taxonomy.

IRS	Year	Response Selection	Risk Assessment	Risk Assessment Criteria	Prediction ability	Adjustment ability	Response evaluation model	Response execution	Response lifetime
DC&A [45]	1996	Dynamic mapping			Reactive	Non-adaptive	Static Cost	Burst	Sustainable
CSM [10]	1996	Dynamic mapping			Reactive	Non-adaptive	Static Cost	Burst	Sustainable
EMERALD [46]	1997	Dynamic mapping			Reactive	Non-adaptive	Static Cost	Burst	Sustainable
BMSL-based response [32]	2000	Static Mapping			Reactive	Non-adaptive	Static Cost	Burst	Sustainable
SoSMART [60]	2000	Static Mapping			Reactive	Non-adaptive	Static Cost	Burst	Sustainable
PH [52]	2000	Static Mapping			Reactive	Non-adaptive	Static Cost	Burst	Sustainable
Lee's IRS [13]	2000	Cost-sensitive	Static		Reactive	Non-adaptive	Static Cost	Burst	Sustainable
AAIRS [5], [7], [8], [9]	2000	Dynamic mapping			Reactive	Adaptive	Static Evaluated Cost	Burst	Sustainable
SARA [59]	2001	Dynamic mapping			Reactive	Non-adaptive	Static Cost	Burst	Sustainable
CITRA [62]	2001	Dynamic mapping			Reactive	Non-adaptive	Static Cost	Burst	Sustainable
TBAIR [57]	2001	Dynamic mapping			Reactive	Non-adaptive	Static Cost	Burst	Sustainable
Network IRS [16]	2002	Cost-sensitive	Static		Reactive	Non-adaptive	Dynamic Evaluated Cost	Burst	Sustainable
Tanachaiwiwat 's IRS [50]	2002	Cost-sensitive	Static		Reactive	Non-adaptive	Static Cost	Burst	Sustainable
Specification-based IRS [49]	2003	Cost-sensitive	Dynamic	Resource Dependencies	Reactive	Non-adaptive	Dynamic Evaluated Cost	Burst	Sustainable
ADEPTS [48]	2005	Cost-sensitive	Static		Proactive	Adaptive	Static Cost	Burst	Sustainable
FAIR [47]	2006	Cost-sensitive	Static		Reactive	Non-adaptive	Static Evaluated Cost	Burst	Sustainable
Stakhanova's IRS [14]	2007	Cost-sensitive	Static		Proactive	Adaptive	Static Evaluated Cost	Burst	Sustainable
DIPS [23]	2007	Cost-sensitive	Dynamic	Attack metrics	Proactive	Non-adaptive	Static Cost	Burst	Sustainable
Jahnke [66]	2007	Cost-sensitive	Dynamic	Attack Graph	Reactive	Non-adaptive	Dynamic Evaluated Cost	Burst	Sustainable
Strasburg's IRS [12]	2008	Cost-sensitive	Static		Reactive	Adaptive	Static Evaluated Cost	Burst	Sustainable
IRDM-HTN [17]	2010	Cost-sensitive	Dynamic	Attack metrics	Reactive	Non-adaptive	Static Evaluated Cost	Retroactive	Sustainable
OrBAC [6]	2010	Cost-sensitive	Dynamic	Resource Dependencies	Proactive	Adaptive	Static Evaluated Cost	Burst	Deactiveable
Kheir's IRS [1]	2010	Cost-sensitive	Dynamic	Resource Dependencies	Proactive	Non-adaptive	Dynamic Evaluated Cost	Burst	Sustainable

Toth and Kruegel [16] present a network model that takes into account relationships between users and resources, since users perform their activities by utilizing the available resources. The goal of a response model is to keep the system in as high a state of usability as possible. Each response alternative (which node to isolate) is inserted temporarily into the network model and a calculation is performed to determine which response has the lowest negative impact on services. In this model, every service has a static cost, and there is only the "block IP" response to evaluate as a way to repel an attack. When the IDS detects an incoming attack, an algorithm attempts to find the firewall/gateway that can effectively minimize the penalty cost of the response action.

Tanachaiwiwat et al. [50] propose a cost-sensitive method. Although they claim that their method is adaptive, they have, in fact, implemented a non adaptive mechanism. They point out that verifying the effectiveness of a response is quite expensive. They check, IDS efficiency, alarm frequency (per week), and damage cost, in order to select the best strategy. The alarm frequency reveals the number of alarms triggered per attack, and damage cost assesses the amount of damage that could be caused by the attacker. An appropriate list of response is available in the proposed model.

Balepin et al. [49] propose two different ways to arrange resources: in a resource type hierarchy, or on a system map. They have adopted a dynamic way to add new nodes for every type of alert that is raised by the IDS that did not already exist on the map. Actually, every node is representative of a system object, such as a file, a running process, a socket, etc. Also, each node has a list of response actions that depend on the type of node, and there is a mechanism to assign a cost to each node.

Foo et al. [48] present a graph-based approach, called ADEPTS. The responses for the affected nodes are based on parameters such as confidence level of attack, previous measurements of responses in similar cases, etc. Thus, ADEPTS uses a feedback mechanism to estimate the success or failure of an applied response. This model is non adaptive, because it does not observe or analyze the behaviors of the deployed responses.

Papadaki and Furnell [47] proposed a cost-sensitive response system that assesses the static and dynamic contexts of the attack. A database for analyzing the static context is needed to manage important characteristics of an attack, such as targets, applications, vulnerabilities, and so on. In terms of evaluating the dynamic context of an attack, there are some interesting ideas embodied in the proposed model. The two main features of this model are: 1) the ability to easily propose different orders of responses for different attack scenarios; and 2) the ability to adapt decisions in response to changes in the environment. To evaluate the characteristics of each response action, they have proposed the following parameters: counter-effects, stopping power, transparency, efficiency, and confidence level.

In [14], Stakhanova et al. proposed a cost-sensitive preemptive IRS. This model focuses on detecting anomalous behavior in software systems. It monitors system behaviors in terms of system calls, and has two levels of classification mechanism to detect intrusion. In the first detection step, when both normal and abnormal patterns are available, the model attempts to determine what kind of pattern is triggered when sequences of system calls are monitored. If the sequences do not match the normal or abnormal patterns, the system relies on machine learning techniques to establish whether the system is normal or anomalous. These authors have presented a response system that is automated, cost-sensitive, preemptive, and adaptive. The response is triggered before the attack completes. There is a mapping between system resources, response actions, and intrusion patterns which has to be defined in advance. Whenever a sequence of system calls matches a prefix in an abnormal graph, the response algorithm decides whether to repel the attack or not, based on a confidence level threshold. Multiple candidate responses may be available, and the one with the least negative effect is selected based on utility theory. The effectiveness of each applied response is measured for future response selection. If the selected response succeeds in neutralizing the attack, its success factor is increased by one; otherwise it is decreased by one.

Haslum et al. [23] have proposed a real time intrusion prevention model. This model is cost-sensitive, and the prediction module has been implemented, as well as a dynamic risk assessment module based on a fuzzy model. Fuzzy logic is used here to capture and automate the risk estimation process that human experts carry out using their experience and judgment based on a number of dependent variables. In a fuzzy automatic inference system, the knowledge of security and risk experts is embedded into the rules for creating the fuzzy model. They have also designed a prediction model based on the hidden Markov model (HMM) to model the interaction between the intruder and the network [24]. That model can detect the U2R, R2L, and PROBE categories of attacks, but not the DoS category.

Jahnke et al. [66] present a graph-based approach for modeling the effects of attacks against resources and the effects of the response measures taken in reaction to those attacks. The proposed approach extends the idea put forward in [16] by using general, directed graphs with different kinds of dependencies between resources and by deriving quantitative differences between system states from these graphs. If we assume that *G1* and *G2* are the graphs we obtain before and after the reaction respectively, then calculation of the response's positive effect is the difference between the availability plotted in the two graphs: *A(G2)-A(G1)*. Like [16, 49], these authors focus on the availability impacts. Strasburg et al. [12] proposed a structured methodology for evaluating the cost of a response based on three parameters: operational cost (*OC*), impact of the response on the system (*RSI*), and response goodness (*RG*). The response cost model is: $RC = OC + RSI - RG$. *OC* refers to the cost of setting up and developing responses. The *RSI* quantifies the negative effect of the response on the system resources. *RG* is defined based on two concepts: 1) the number of possible intrusions that the response can potentially address; 2) the amount of resources that can be protected by applying the response.

Mu and Li [17] presented a hierarchical task network planning model to repel intrusions, in which every response has an associated static risk threshold that can be calculated by its ratio of positive to negative effects. The permission to run each response is based on the current risk index of the network. When the risk index is greater than the response static threshold, the next response is allowed to run. They propose a response selection window, where the most effective responses are selected to repel intrusions. There is no evaluation of responses in this work, however, and it is unclear how the positive and negative effects of responses have been calculated. In that framework, the communication component is responsible for receiving alerts from multiple IDS. An alert filter, and verification and correlation components have all been considered. Intrusion response planning is in place to find a sequence of actions that achieve a response goal. These goals are the same as those in [5]: analyze the attack, capture the attack, mask the attack, maximize confidentiality, maximize integrity, recovery gracefully, and sustain service. Each goal has its own sequence of responses. For example, if the goal is to analyze an attack, the earlier responses in the sequence have to be weak, but later responses have to be strong. In [18], the authors propose a D-S evidence theory to assess risk.

Kanoun et al. [6] were the first to propose a risk-aware framework to activate and deactivate response policies, which consists of an online model and its

architecture. The likelihood of success of an ongoing threat or an actual attack, as well as the cumulative impacts of the threat and the response, are all considered before activating/deactivating a strategic response. The main contribution of the proposed model is to determine when a strategic response should be deactivated and how. These authors believe that the deactivation phase is as important as the activation phase.

Kheir et al. [1] propose a dependency graph to evaluate the confidentiality and integrity impacts, as well as the availability impacts. The confidentiality and integrity criteria were not considered in [16, 49, 66]. In [1], the impact propagation process proposed by Jahnke et al. is extended by adding these impacts. Now, each resource in the dependency graph is described with a 3D CIA vector, the values of which are subsequently updated, either by active monitoring estimation or by extrapolation using the dependency graph. In the proposed model, dependencies are classified as structural (inter-layer) dependencies, or as functional (inter-layer) dependencies.

4. Discussion

In this paper, we have reviewed all the recently proposed IRS models. Although an appropriate taxonomy was proposed in 2007 in [15], we are looking for new features to improve their design as well as network security, in an effort to neutralize attacks. The new IRS features we have introduced here are capable of repelling attack perfectly.

In the last five years or so, we have seen impressive changes in the ways in which attackers gain access to systems and infect computers. The main problem with choosing a security measure is identifying the security problem. It is important, for example, that we not isolate a whole server from a network and disrupt the many services we have installed there, nor do we want to kill processes that are using considerable amounts of CPU resources if we are not sure they have been attacked. Consequently, the appropriate algorithms must be implemented in an IRS, and the right set of responses with a very high positive value must be selected whether or not an attack is in progress. To design an appropriate algorithm to trigger responses, the attack level (user access, root access, application access) and, most importantly, the context of the attack (resource exhaustion, a malware dropper, a rootkit, malware itself, port opening, remote access, etc.) has to be considered. Countering attacks requires preparation of a complete list of responses, an accurate evaluation of those responses in a network model, and understanding the impact of each response in every element of the network. Otherwise, our automated IRS will:

- reduces network/host performance,
- wrongly disconnect users from the network/host,
- result in high costs for administrators re-establishing services, and
- become a DoS attack for our network, which will eventually have to be disabled.

Today, many services are available and used by large numbers of users. It is extremely important to maintain the users QoS, the response time of applications, and critical services in high demand. We have discussed some techniques to control the number of responses applied when a problem arises. As we have pointed out, running responses in burst mode decreases not only network performance, but also that of the attacked machine. Also, we have explained how to use the retroactive approach to determine the number of effective responses for repelling an attack. Another important issue related to IRS support that we have mentioned is identifying the attack path, since extracting it can enable us to specify the appropriate locations for applying responses. With respect to the location type, appropriate responses can be assigned to calculate dynamic cost. In this way, an attack path-based IRS finds the best locations for applying responses at the lowest penalty cost. We also discussed two important components that provide IRS intelligence: 1) the risk assessment component; and 2) the prediction component, and the challenges associated with using these algorithms in streaming mode. Below are some suggestions for future research on the development of IRS:

- Design a rapid and accurate ***online response cost evaluation framework*** for IRS to meet network demands.
- Propose a framework for ***adaptive*** IRS taking into account response history in order to tackle weaknesses in the response goodness calculation over time mentioned in this paper.
- Design a ***retroactive*** approach based on the concept of grouping, in order to prevent overly impacting the network and to accurately measure the risk index.
- Prepare a ***strong, real dataset*** of single and multi-step attacks. Such a dataset is needed by all security researchers and will be useful for testing the retroactive approach incorporating the grouping concept.
- Design an Online Risk Assessment component with a ***"system state"*** feature, in order to accurately measure the risk index.
- Design an IRS that supports an ***attack path***, in order to meet users needs in terms of QoS and

to enable the application of responses where the penalty cost is the lowest.

5. Conclusion

In the past decade, various very effective Intrusion Response Systems have been developed. At the same time, we have seen impressive changes in the way attackers infect computers. As a result, it is impossible to create a perfect IRS that repels the majority of attacks. As mentioned in this paper, existing automated IRS suffer from weaknesses that prevent them from neutralizing attacks. Significant research will be required to address all those weaknesses and design a framework with a high level of capability. We have proposed a taxonomy of IRS and discussed future research that could improve the current systems substantially, which would in turn improve the intrusion response mechanism to enable it to accommodate more intelligence for the decision making process.

Acknowledgment

The support of the Natural Sciences and Engineering Research Council of Canada (NSERC), Ericsson Software Research, and Defence Research and Development Canada (DRDC) is gratefully acknowledged.

References

[1] N. Kheir, N. Cuppens-Boulahia, F. Cuppens and H. Debar, "A service dependency model for cost sensitive intrusion response," Proceedings of the 15th European Conference on Research in Computer Security, pp. 626-642, 2010.

[2] A. Shameli-Sendi, M. Jabbarifar, M. Shajari and M. Dagenais, "FEMRA: Fuzzy expert model for risk assessment," Proceedings of the Fifth International Conference on Internet Monitoring and Protection, pp. 48-53, Barcelona, Spain, 2010.

[3] A. Shameli-Sendi and M. Dagenais, "Real Time Intrusion Prediction based on improving the priority of alerts with Hidden Markov Model," Journal of Networks, 2012.

[4] G. N. Matni and M. Dagenais, "Operating system level trace analysis for automated problem identification," The Open Cybernetics and Systemics Journal, vol. 5, 2011, pp. 45-52.

[5] A. Curtis and J. Carver, "Adaptive agent-based intrusion response," Ph.D. thesis, Texas A&M University, USA, 2001.

[6] W. Kanoun, N. Cuppens-Boulahia, F. Cuppens and S. Dubus, "Risk-Aware Framework for Activating and Deactivating Policy-Based Response," Fourth International Conference on Network and System Security, pp. 207-215, 2010.

[7] C. Carver and U. Pooch, "An intrusion response taxonomy and its role in automatic intrusion response," IEEE Workshop on Information Assurance and Security, 2000.

[8] C. Carver, J. M. Hill and J. R. Surdu, "A methodology for using intelligent agents to provide automated intrusion response," IEEE Systems, Man, and Cybernetics Information Assurance and Security Workshop, pp. 110-116, 2000.

[9] D. Ragsdale, C. Carver, J. Humphries and U. Pooch, "Adaptation techniques for intrusion detection and intrusion response system," IEEE International Conference on Systems, Man, and Cybernetics, pp. 2344-2349, 2000.

[10] G. White, E. Fisch and U. Pooch "Cooperating security managers: a peer-based intrusion detection system," IEEE Network, vol. 10, 1996, pp. 20-23.

[11] C. Strasburg, N. Stakhanova, S. Basu and J. S. Wong, "The Methodology for Evaluating Response Cost for Intrusion Response Systems," Technical Report 08-12, Iowa State University.

[12] C. Strasburg, N. Stakhanova, S. Basu and J. S. Wong, "A Framework for Cost Sensitive Assessment of Intrusion Response Selection," Proceedings of IEEE Computer Software and Applications Conference, 2009.

[13] W. Lee, W. Fan and M. Miller, "Toward Cost-Sensitive Modeling for Intrusion Detection and Response," Journal of Computer Security, vol. 10, no. 1, 2002, pp. 5-22.

[14] N. Stakhanova, S. Basu and J. Wong, "A cost-sensitive model for preemptive intrusion response systems," Proceedings of the 21st International Conference on Advanced Networking and Applications, IEEE Computer Society, Washington, DC, USA, pp. 428-435, 2007.

[15] N. Stakhanova, S. Basu and J. Wong, "Taxonomy of Intrusion Response Systems," Journal of Information and Computer Security, vol. 1, no. 2, 2007, pp. 169-184.

[16] T. Toth and C. Kregel, "Evaluating the impact of automated intrusion response mechanisms," In proceeding of the 18th Annual Computer Security Applications Conference, Los Alamitos, USA, 2002.

[17] C. P. Mu and Y. Li, "An intrusion response decision-making model based on hierarchical task network planning," Expert systems with applications, vol. 37, no. 3, 2010, pp. 2465-2472.

[18] C. P. Mu, X. J. Li, H. K. Huang and S. F. Tian, "Online risk assessment of intrusion scenarios using D-S evidence theory," 13th European Symposium on Research in Computer Security, pp. 35-48, Malaga, Spain, 2008.

[19] P. Berkhin, "Survey of clustering data mining techniques," 2001.

[20] F. Xiao, S. Jin and X. Li, "A Novel Data Mining-Based Method for Alert Reduction and Analysis," Journal of Network, vol. 5, no. 1, 2010, pp. 88-97.

[21] K. Scarfone and P. Mell, "Guide to Intrusion Detection and
Prevention Systems," Technical report, NIST: National Instutute of Standards and Technology, U.S. Department of Commerce, 2007.

[22] D. B. Payne and H. G. Gunhold, "Policy-based security configuration management application to intrusion detection and prevention," IEEE International Conference on Communications, 2009, Dresden, Germany.

[23] K. Haslum, A. Abraham and S. Knapskog, "DIPS: A framework for distributed intrusion prediction and prevention using hidden markov models and online fuzzy risk assessment," In 3rd International Symposium on Information Assurance and Security, pp. 183-188, Manchester, United Kingdom, 2007.

[24] K. Haslum, M. E. G. Moe and S. J. Knapskog, "Real-time intrusion prevention and security analysis of networks using HMMs," 33rd IEEE Conference on Local Computer Networks, 2008, Montreal, Canada.

[25] A. Montplaisir, "Stockage sur disque pour accès rapide d'attributs avec intervalles de temps," M.Sc.A. thesis, École Polytechnique de Montréal, 2011.

[26] B. Zhu and A. A. Ghorbani, "Alert correlation for extracting attack strategies," International Journal of Network Security, vol. 3, no. 3, 2006, pp. 244-258.

[27] C. Kruegel, F. Valeur and G. Vigna, "Alert Correlation," In Intrusion Detection and Correlation, 1st ed., vol. 14., New York: Springer, 2005, pp. 29-35.

[28] G. Stein, C. Bing, A. S. Wu and K. A. Hua, "Decision Tree Classifier For Network Intrusion Detection With GA-based Feature Selection," In Proceedings of the 43rd annual Southeast regional conference, Georgia, ISBN:1-59593-059-0, pp. 136-141, 2005.

[29] D. Yu and D. Frincke, "Improving the quality of alerts and predicting intruder's next goal with Hidden Colored Petri-Net," Computer Networks, pp. 632-654, 2007.

[30] N. B. Anuar, H. Sallehudin, A. Gani and O. Zakaria, "Identifying False Alarm for Network Intrusion Detection System Using Hybrid Data Mining and Decision Tree," Malaysian Journal of Computer Science, ISSN 0127-9084, 2008, pp. 110-115.

[31] A. Arnes, K. Sallhammar, K. Haslum, T. Brekne, M. Moe and S. Knapskog, "Real-time risk assessment with network sensors and intrusion detection systems," In Computational Intelligence and Security, vol. 3802 of Lecture Notes in Computer Science, pp. 388-397, 2005.

[32] T. Bowen, D. Chee, M. Segal, R. Sekar, T. Shanbhag and P. Uppuluri, "Building survivable systems: an integrated approach based on intrusion detection and damage containment," In DARPA Information Survivability Conference and Exposition, pp. 84-99, 2000.

[33] Difference between Signature Based and Anomaly Based Detection in IDS, URL http://www.secguru.com/forum/difference between signature based and anomaly based detection in ids.

[34] L. Feng , W. Wang , L. Zhu and Y. Zhang, "Predicting intrusion goal using dynamic Bayesian network with transfer probability estimation," Journal of Networks and Computer Applications, vol. 32, no. 3, 2009, pp. 721-732.

[35] A. O. Adetunmbi, S. O. Falaki, O. S. Adewale and B. K. Alese, "Network Intrusion Detection based on Rough Set and k-Nearest Neighbour," International Journal of Computing and ICT Research, vol. 2, no. 1, 2008, pp. 60-66.

[36] A. Lazarevic, L. Ertz, V. Kumar, A. Ozgur and J. Srivastava, "A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection," Proceedings of the Third SIAM International Conference on Data Mining, 2003.

[37] J. Han and M. Kamber, "Data Mining: Concepts and Techniques," 2nd ed., San Francisco: Elsevier, 2006.

[38] M. Gaber, A. Zaslavsky and S. Krishnaswamy, "Mining Data Streams: A Review," ACM SIGMOD Record, vol. 34, 2005.

[39] C. Aggarwal, J. Han, J. Wang and P. Yu, "A Framework for Projected Clustering of High Dimensional Data Streams," Proceedings of the 30th VLDB Conference, Toronto, Canada, 2004.

[40] MIT Lincoln Laboratory, 2000 darpa intrusion detection scenario specific data sets, 2000.

[41] The Snort Project, Snort users manual 2.8.5, 2009.

[42] G. Antoniol. Keynote paper "Search based software testing for software security: Breaking code to make it safer," In ICSTW 09: Proceedings of the IEEE International Conference on SoftwareTesting, Verification, and Validation Workshops, IEEE Computer Society, 2009.

[43] G. Stoneburner, A. Goguen and A. Feringa, "Risk management guide for information technology systems," http://csrc.nist.gov/publications/nistpubs/800-30/sp800-30.pdf.

[44] H. Debar, D. Curry and B. Feinstein, "The Intrusion Detection Message Exchange Format (IDMEF)," http://www.ietf.org/rfc/rfc4765.txt.

[45] E. Fisch, "A Taxonomy and Implementation of Automated Responses to Intrusive Behaviour," Ph.D. thesis, Texas A&M University, 1996.

[46] P. Porras and P. Neumann, "EMERALD: event monitoring enabling responses to anomalous live disturbances," National Information Systems Security Conference, 1997.

[47] M. Papadaki and S. M. Furnell, "Achieving automated intrusion response: a prototype implementation," Information Management and Computer Security, vol. 14, no. 3, 2006, pp. 235-251.

[48] B. Foo, Y. S. Wu, Y. C. Mao, S. Bagchi and E. Spafford, "ADEPTS: adaptive intrusion response using attack graphs in an e-commerce environment," International Conference on Dependable Systems and Networks, pp. 508-517, 2005.

[49] I. Balepin, S. Maltsev, J. Rowe and K. Levitt "Using specification-based intrusion detection for automated response," In 6th International Symposium on Recent Advances in Intrusion Detection, pp. 136-154, 2003.

[50] S. Tanachaiwiwat, K. Hwang and Y. Chen, "Adaptive Intrusion Response to Minimize Risk over Multiple Network Attacks," ACM Trans on Information and System Security, 2002.

[51] N. B. Anuar, M. Papadaki, S. Furnell and N. Clarke, "An investigation and survey of response options for intrusion response systems," Information Security for South Africa, pp. 1-8, 2010.

[52] A. Somayaji and S. Forrest, "Automated response using system-call delay," In 9th USENIX Security Symposium, pp.185-198, 2000.

[53] M. F. Yusof, "Automated Signature Generation of Network Attacks," B.Sc. thesis, University Teknologi Malasia, 2009.

[54] P. Arnes, F. Valeur and R. Kemmerer, "Using hidden markov models to evaluate the risk of intrusions," Int.

Symp. Recent Advances in Intrusion Detection, Hamburg, Germany, 2006.
[55] W. Li and Z. Guo, "Hidden Markov Model Based Real Time Network Security Quantification Method," nswctc, International Conference on Networks Security, Wireless Communications and Trusted Computing, pp. 94-100, 2009.
[56] International Standard Organization, ISO/IEC 27005, Information Security Risk Management, 2008.
[57] X. Wang, D. S. Reeves and S. F. Wu, "Tracing based active intrusion response," Journal of Information Warefare, vol. 1, 2001, pp. 50-61.
[58] Y. Chen, B. Boehm and L. Sheppard, "Value Driven Security Threat Modeling Based on Attack Path Analysis," 40th Hawaii International Conference on System Sciences, Big Island, Hawaii, January 2007.
[59] S. M. Lewandowski, D. J. V. Hook, G. C. O'Leary, J. W. Haines and M. L. Rossey, "SARA: Survivable autonomic response architecture," In DARPA Information Survivability Conference and Exposition, pp. 77-88, 2001.
[60] S. Musman and P. Flesher, "System or security managers adaptive response tool," In DARPA Information Survivability Conference and Exposition, 2000.
[61] Z. Li, Z. Lei, L. Wang and D. Li, "Assessing attack threat by the probability of following attacks," in Proceedings of the International Conference on Networking, Architecture, and Storage, IEEE, pp. 91-100, 2007.
[62] D. Schnackenberg, H. Holliday, R. Smith, K. Djadhandari and D. Sterne, "Cooperative intrusion traceback and response architecture citra," in IEEE DARPA Information Survivability Conference and Exposition, pp. 56-68, 2001.
[63] Y. Zhang, X. Fan, Y. Wang and Z. Xue, "Attack grammar: A new approach to modeling and analyzing network attack sequences," In: Proc. of the Annual Computer Security Applications Conference (ACSAC 2008), pp. 215-224, 2008.
[64] M. Sabhnani and G. Serpen, "Formulation of a Heuristic Rule for Misuse and Anomaly Detection for U2R Attacks in Solaris Operating System Environment," In Proc. Security and Management, pp. 390-396, 2003.
[65] S. Savage, D. Wetherall, A. Karlin and T. Anderson "Practical network support for IP traceback," In ACM SIGCOMM, August 2000.
[66] M. Jahnke, C. Thul and P. Martini, "Graph-based Metrics for Intrusion Response Measures in Computer Networks," In: Proc. of the 3rd LCN Workshop on Network Security. Held in conjunction with the 32nd IEEE Conference on Local Computer Networks (LCN), Dublin, Ireland, 2007.

Alireza Shameli-Sendi received his B.Sc. and M.Sc. with honours from Amirkabir University of Technology (Tehran Polytechnic), and is currently finishing his Ph.D. at École Polytechnique de Montréal, Montreal, Canada. His primary research interests include information security, vulnerability analysis, intrusion detection, and proactive intrusion response systems.

Naser Ezzati-Jivan is currently studying in the department of Computer and Software Engineering at École Polytechnique de Montréal, Montreal, Canada. His research interests include the operating system instrumentation, system trace analysis, and Network Security. He received his B.Sc., M.Sc. in High Performance Firewalls from Sharif University of Technology and Isfahan University in 2000 and 2002 respectively.

Masoume Jabbarifar received her B.Sc. and M.Sc. from Amirkabir University of Technology (Tehran Polytechnic), and is currently finishing her Ph.D. At École Polytechnique de Montréal, Montreal, Canada. Her primary research interests include time synchronization, wireless sensor network, trace analysis, and security standards.

Michel Dagenais is Professor in the Department of Computer and Software Engineering at École Polytechnique de Montréal, Montreal, Canada. He received his Ph.D. in Electrical Engineering from McGill University, Montreal, Canada in 1987. His primary research area is distributed and multi-core systems, tracing and performance analysis. He also participates in projects related to security policies and security standards.