2004

Behavior-based System for Generation of Security Solutions

Shobha Potluri

Pranathi Venkatayogi

Bharat Bhargava
Purdue University, bb@cs.purdue.edu

Report Number:
04-011

BEHAVIOR-BASED SYSTEM FOR GENERATION OF SECURITY SOLUTIONS

Shobha Potluri
Pranathi Venkatayogi
Bharat Bhargava

Department of Computer Sciences
Purdue University
West Lafayette, IN 47907

CSD TR #04-011
March 2004

Behavior-based System for Generation of Security Solutions

Shobha Potluri, Pranathi Venkatayogi, and Bharat Bhargava
Department of Computer Sciences and
the Center for Education and Research in Information Assurance and Security (CERIAS)
Purdue University

Abstract

We propose building an integrated agent-based simulation environment to find the effective security profile within the budget considerations. The key ideas involved are: dynamic firm profile; behavior based dynamic attacker profile generator; learning-based IDS, and an evolutionary search via genetic algorithms for the effective security profile. The significant contribution of this work is that it integrates the management and computer science concepts to build a seamless interactive simulation environment in order to address an important problem in the field of information security.

Contents

1. Introduction

Information warfare has been a major threat to business enterprises and it assumes special significance in the post 9/11 world. In today's highly competitive environment the success of a firm is dependent on how well the firm manages its information resources. Business managers have to make complex decisions regarding the security of these resources while operating under budget constraints. Thus, information security has become a critical issue to the success and stability of any business enterprise. Investing in correct security technologies is crucial to countering the threats from perpetrators, especially in a dynamic environment where the attackers continuously learn and adapt to the security technologies and keep devising innovative techniques to breach the firm's security. Current security research concentrates on providing technical solutions to the security threats from attackers [2, 17]. This means coming up with a combination of different security solutions such as firewalls, IDS, log systems, encryption etc. These studies address specific security threats and assume certain behaviors of the attackers. The effectiveness of these solutions reduces once the attackers change their strategies. Hence, providing security solutions catering to the dynamic nature of the attackers is critical.

We propose to design a system that provides security solutions for a firm in light of the dynamic nature of the attacker. Simulations are carried out in which the attackers attack the firm. Based upon the vulnerability of the firm, the attackers are successful in breaching the security and hence in inflicting damages. The vulnerability of the firm depends on the security solutions that it has chosen to guard itself from the expected attacks. Successfully tracking the attacks for a given security configuration of the firm will help in learning more about the firm's preparedness. Analyzing different firm security solutions in the light of attacks will be a good search tool for coming up with effective security measures that a firm needs to adopt. This could be a valuable aid in business decision-making especially given the importance of the problem of information security and the cost involved.

In order for this tool to be effective, the real world situation needs to be simulated. Simulation needs to mimic the actual scenarios as much as possible. Unlike the present studies [2, 17], we propose to enhance the simulations by including the dynamic behavior of the attackers. In the real world the attacker behavior changes based upon the results from the previous attacks. This learning aspect on the part of the attackers makes them more lethal and from the perspective of the firm leads to an unpredictable environment. Unless the firm becomes more effective in predicting the attacker moves, the losses would mount.

We propose an integrated simulation framework that can simulate the attacks from a set of dynamic attackers. The firm is modeled as an entity with a specific set of security solutions. Attacks are carried out by attackers who learn from the effectiveness of their past attacks. The system that we are planning incorporates the dynamic behavior of the attacker by including learning mechanisms ([3], [6], [13], [21]). This simulates the real world scenario. An intrusion detection system (IDS) screens the attacks so that only the most lethal ones affect the firm. A population based stochastic guided search algorithm based on evolutionary principles is used to come up with effective security solutions for the firm. Multiple security solutions are screened at every time step for their effectiveness in warding off the attacks. Attackers enhance their strategies based upon the effectiveness of the attacks at the previous time step. When the simulation ends based upon a user defined termination criterion, we get multiple solutions that are effective in guarding the firm from the attackers.

The user can define the "effectiveness" of the security solutions. This could be the ability of the security solution to protect against the vulnerabilities, or simply the cost of the security components that the firm chooses. This modular object oriented framework can be easily extended to incorporate any definition for the effectiveness of the firm's security solutions as long as it can be numerically evaluated.

Inclusion of the IDS in the proposed integrated framework leads to the choice of security components that help the firm withstand the toughest attacks. An IDS is a default and relatively common security solution adopted by most firms in the real world. The reasons [25] for this being:

- An IDS prevents problems by increasing the perceived risk of discovery and punishment of attackers
- It can detect attacks and other security violations that are not prevented by other security measures
- It can document the existing threat to an organization
- It acts as quality control for security design and administration, especially of large and complex enterprises
- It provides useful information about intrusions that do take place, allowing improved diagnosis, recovery, and correction of causative factors.

The current architecture can also be extended to the defense environment, where the behavior profiles of the enemy can be analyzed to develop effective defense mechanisms. Thus in the following pages we present a framework that includes behavior based model for of dynamic attackers, an environment that closely mimics the real world scenario for the simulation of attacks and an easily extensible search procedure that comes up with effective security solutions for a firm. This could be a valuable business tool for the managers for making crucial decisions about the choice of security solutions or any enterprise.

2. Problem Statement

The research problem is to develop a system that will aid business decision makers in generating effective security solutions for any business enterprise in light of the dynamic nature of the attacker. This system comprises a simulation environment that mimics the real world as much as possible. Simulations are carried out to evaluate the effectiveness of the different security profiles in light of attackers who change their strategies dynamically. The presence of dynamic attackers who learn from the effectiveness of their previous attacks and the IDS makes this integrated framework realistic. Security solutions are evaluated based on user defined criteria such as their cost, budget constraints of the firm, effectiveness of the solutions in withstanding various attacks and any other user defined criteria.

3. Research Direction

The main thrust of our research is in two directions:

1. Design of a framework that helps generate effective security profiles for the firms in the presence of dynamic attackers. This entails research in the following areas:

- Representation of a firm along with its security components
- Modeling the attacker who can change attack strategies dynamically along the course of the simulation
- Inclusion of an Intrusion Detection System that is representative of a security component of the organization
- Developing a simulation that provides for the efficient interaction of the various components

2. Design and conduct experiments to simulate and validate the proposed architecture and illustrate its efficacy. The interaction and dynamic adaptability of the various components will be studied. Experiments will be designed for analyzing the following:

- The efficacy of the representation of the organization in mimicking the real world
- Improvement in the effectiveness of the security solutions over time in light of changing attacks
- Whether the components chosen satisfy the user defined optimality criteria
- Effect of attacker learning mechanism on the security solutions generated
- Efficiency of the learning process of the Intrusion Detection System
- Most effective choice of security components to overcome vulnerabilities of the organizations

4. Related Work

Some of the related issues addressed in past and present information security research have focused on

- Generating security policy assuming static perpetrator behavior
- Stochastic simulation of attacks
- Psychometrics – modeling psychological aspects of a attacker
- Learning-based intrusion detection systems

Rees et al [17] have developed a Policy Framework for Interpreting Risk in eCommerce Security (PFIRES). This work proposes a life-cycle based approach to dynamically develop security policies. The key objective of their research is to provide information security professionals and top management a framework through which usable security strategy and policy for e-commerce applications can be created and maintained in line with the standard information technology lifecycle.

Soumyo and Konda [15] have developed a model to evaluate trade-offs between cost of defense mechanisms for networked systems and the resulting expected survivability after a network attack. The costs will be those of deploying and maintaining various defense mechanisms to protect a system or site against attacks, including the costs of any constraints on the system imposed by the defense mechanism. The benefits will be those of increased survivability, the ability of systems to recover from attacks, of the system or site. A primary objective of their research is to develop and apply a reasonably realistic simulation model that can help systems managers and chief information officers to understand survivability issues better and evaluate the tradeoffs involved in decisions about network systems design, including their defense mechanisms.

The above research studies assume static attacker behavior and ignore learning on the part of both attackers and victims. Presumably there is a continuous cycle of increasing sophistication on both sides, and this is what our approach is aiming at.

Attacker behavior has been studied through analysis of security incidents on the Internet. Attackers have been classified into various groups such as attackers, spies, terrorists, corporate raiders, professional criminals and vandals depending on what their intention is for breaking into a computer or a computer network [11].

The Physical Conditions, Emotional State, Cognitive Capabilities and Social Status (PECS) architecture proposed by Schmidt [18] is intended to support the design process of agent-based simulation models. Individual human behavior and decision making, interaction between individuals, as well as interactions of individuals with their environment form the crux of this approach. This reference model provides a concept for the construction of agents, a communication infrastructure, an environment component and domain independent model architecture. It proposes a general, methodologically founded construction scheme which can be applied to various application areas and must be filled with specific attributes and dynamic behavior.

The learning mechanisms for enhancing the ability of the attackers based upon the efficiency of the previous attacks can be implemented using any of the machine learning techniques such as the genetic algorithms [6], classifier systems [13], genetic programming [3] and neural networks [21].

An exhaustive study about the various categories in which the IDS falls is available in the literature [25]. This was done to enable us to choose the category best suited for the current work.

The concept behind Intrusion Detection Systems [25] is to inspect all network activity (both inbound and outbound) and identify suspicious patterns that could be evidence of a network or system attack. The two approaches to develop IDS are knowledge based and behavior based approach. These approaches have been analyzed and their relative merits and demerits are compared **(Table 1)** to choose the one appropriate to our requirements. The location of where the IDS is deployed affects the nature of the attacks it can detect. IDS can be installed on the network, on a specific host, or an application within a host as represented in **Table2.**

	Definition	Advantages	Disadvantages
Knowledge Based Approach	-contains information about specific attacks and vulnerabilities and looks for attempts to exploit these vulnerabilities. When such an attempt is detected, an alarm is triggered. Accuracy depends on the regular update of knowledge about attacks.	-the potential for very low false alarm rates -contextual analysis proposed by the intrusion detection system is detailed, making it easier to take preventive or corrective action.	-Maintenance of the knowledge base of the intrusion detection system and maintaining it up to date -Knowledge about attacks is much focused causing it to be closely tied to an environment. -Detection of insider attacks is difficult
Behavior Based Approach	Normal or expected behavior extracted from reference information is compared with the current activity, any deviation observed, is detected as an intrusion.	-can detect attempts to exploit new and unforeseen vulnerabilities and contribute to the automatic discovery of new attacks. -do not face the generalization issue. -They help detect abuse of privileges types of attacks that do not actually involve exploiting any technological vulnerability.	-high false alarm rate - periodic online retraining of the behavior profile is required which results in either in unavailability of the intrusion detection system or in additional false alarms.

Table 1: Categorization based on Approach [25]

Type of IDS	Definition	Advantages	Disadvantages
Network Based IDS	-located on the segment being monitored promiscuously examining every packet on the network.	-straightforward to implement and deploy.	- a single sniffer cannot be relied upon to monitor an entire subnet. -easy target for attacks such as DoS.
Host Based IDS	-run as an application on a network-connected host.	-ability to detect violation of security policy by an insider. -ideal for high availability servers.	-closely tied to the operating system. -No protection against network-layer DoS attacks (SYN flooding, ping)
Application Based IDS	-operate at the Application layer and are tailored to monitor specific applications for suspicious user patterns and application log messages.	-defend against sophisticated attacks that go undetected by both the network and host based intrusion detection systems.	-hard to manage and deploy. -one is required for each type of critical network application in the enterprise.

Table2: Categorization Based on Location of IDS [25]

5. Proposed Architecture

The proposed Architecture consists of the following components:

Firm
Intrusion Detection System
Attackers

Firms and attackers interact with each other, with firms trying to protect their resources, and malicious agents or attackers trying to breach the security. Firms try to face the attacks by choosing various security solutions. The overall view of the interrelationships between the firm, the attackers and the security solutions is shown in **Figure 1**.

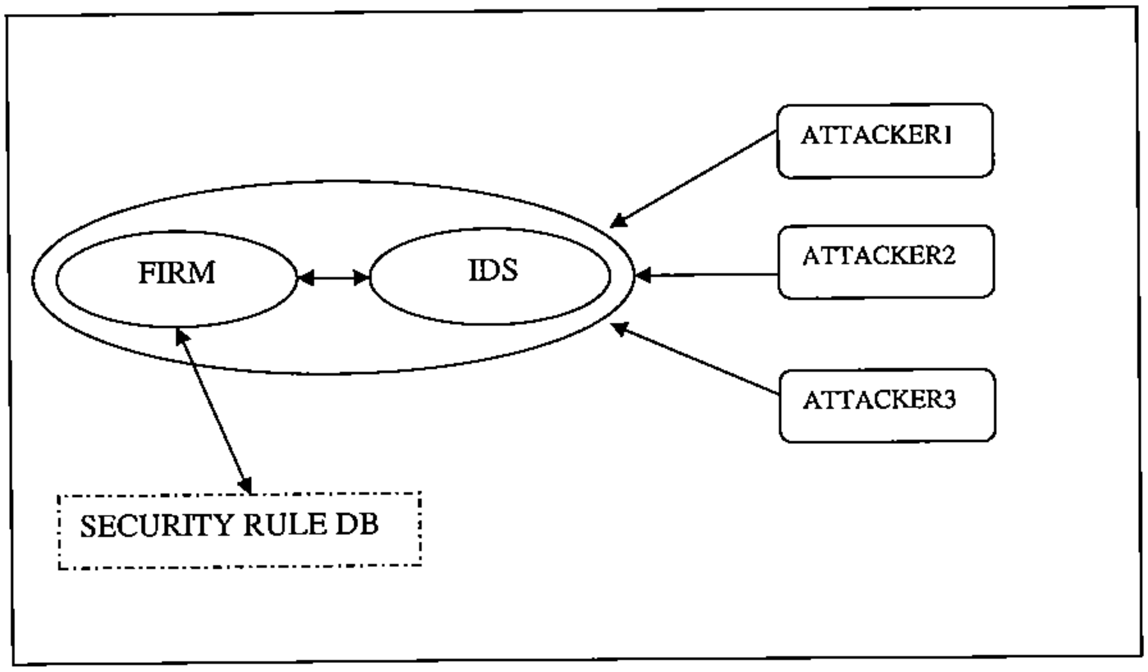

Figure 1: Integrated Agent-based Architecture for Effective Security Profile Generation

Firms are defined as the entities that own the information systems and reap the benefits of these resources through more effective and efficient management and operation of their primary business activities [2]. Firms seek to harvest these benefits while trying to protect their resources from external malicious agents. The information system resources of firms are classified into four categories, namely: network resources, operating systems, databases, and applications [22]. Each of these resources has vulnerabilities associated with it.
Vulnerability as defined by Microsoft Security Research Centre is "a security exposure that results from a product flaw, and which the maker of the product should fix". Exploiting of these vulnerabilities causes loss to the firm. Firms attempt to minimize their losses from damage or loss to their information resources by investing in security solutions while operating under budget constraints.

To exploit the vulnerabilities certain resources are required. These resources may be classified into two broad categories, financial resources and technical resources. Financial resources include monetary resources like assets, cash etc. Technical resources include skill sets like programming languages known, software available, etc.

For efficient representation of the various components involved in the architecture, we term the following notation.
Notation1: δ (p,n,S) denotes an n-bit substring S starting from p
Notation2: Max (a,b,...z) denotes the maximum value among a,b,...z

Information required for the architecture is categorized and stored in the database using the relational data model. The various relations are:

1. R_v *(ISR, ISRType, Vul, VulRep, Loss, RR [])* – This relation shows the associations between the resources of the firm, their corresponding vulnerabilities, losses that can be incurred and resources to be possessed to exploit the corresponding vulnerabilities.
   ISR denotes the particular information system resource owned by various firms.
   ISRType denotes the type of particular information system resource,
   Vul is the string denoting the vulnerability associated with the particular ISR and ISRtype.
   VulRep is a bit string representation of Vul. All the vulnerabilities (Vul) associated with a particular ISR and ISRType are enlisted and a unique bit string VulRep is assigned to them.
   Loss represents the loss incurred by the firm when the vulnerability (Vul) is exploited (in $).
   RR[] is the list of resources that need to be possessed to exploit vulnerability (Vul).
   An example tuple of the above relation could be (Network, LAN, "DDOS", 001, 10000$, [Networking Software,....])

2. R_a *(RR, RC)* - This relation shows the association between the resource and the cost to acquire the resource.
   RC is the cost of acquiring resource RR.

3. R_c*(Comp,VulRep[],Cost)* – This relation shows the association between the security components, the vulnerabilities they defend against and the cost of the component.
   Comp denotes the security component such as firewall, IDS, authentication mechanisms
   VulRep[] is a list of bit strings representative of vulnerabilities(Vul).
   Cost is the cost of the security component.

The following sections explain each of the components of the proposed architecture. Section 5.1 explains the design of the firm component, section 5.2 explains the design of the IDS, section 5.3 explains design of an attacker component. Then section 5.4 explains the simulation environment and section 5.5 discusses the interaction among the various components.

5.1. Design of Firm Component

This section presents the representation of a firm in the proposed artificial environment. Firm is one of the major components of the architecture and each firm is represented as an agent with a set of profiles denoting the characteristics of the firm. As we move across generations of a firm, the firm must be able to update the solutions it chooses and be able to defend against the vulnerabilities it has.
The research questions involved in the firm component are as follows:

1. How to obtain security solutions which provide defense against the vulnerabilities of various information resources of a firm?

2. How the firm learns to update its choice of security components to protect against the unpredictable attacks it faces?

We propose to solve the above research questions with the following design. The firm is represented by a set of profiles: IT profile, vulnerability profile, security profile and exposed risk profile (Figure 2).

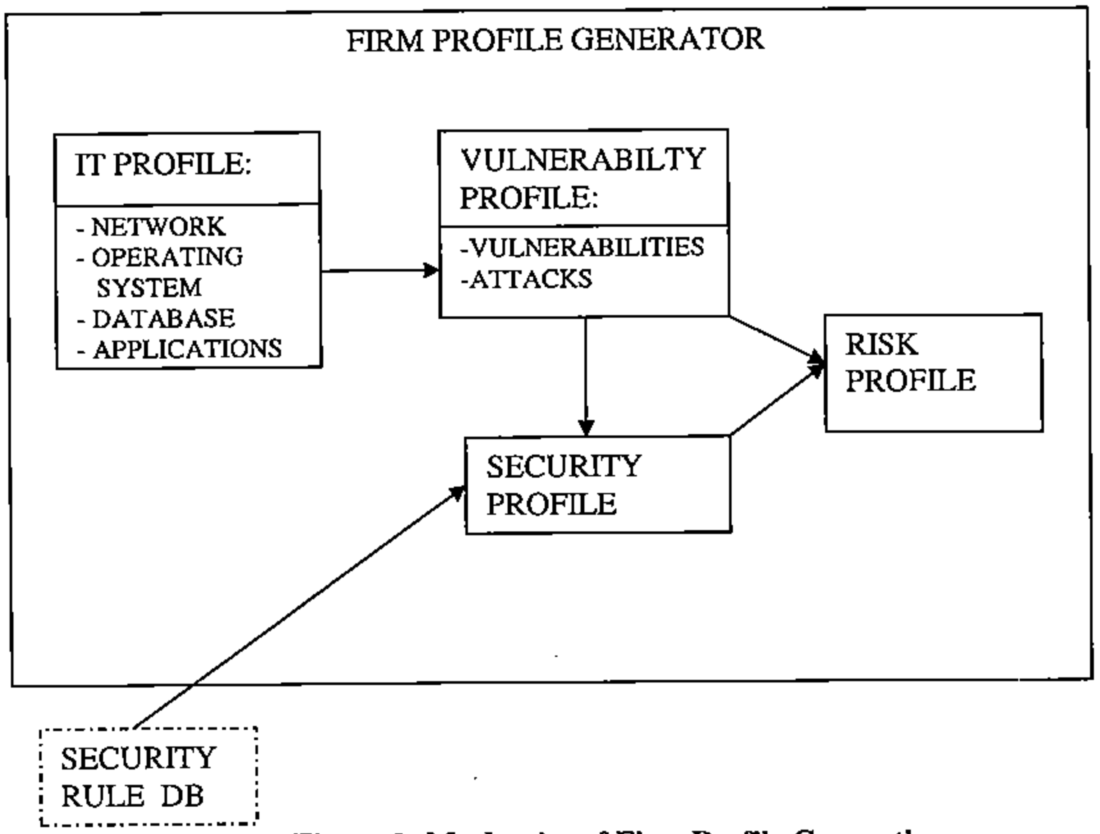

Figure 2: Mechanics of Firm Profile Generation

IT profile: IT profile is defined as the representation of information system resources owned by the firm. Firm's IT profile is represented by *4b* bit binary string *(IP)* giving 2^n possible combinations of IT resources where

$\delta(0,b,IP)$ represents the network resource such as LAN, WAN or MAN
$\delta(b, b, IP)$ represents the operating system resource such as WINDOWS, UNIX
$\delta(2b, b, IP)$ represents the database resource such as ORACLE, DB2
$\delta(3b, b, IP)$ represents the applications such as APPLETS, ACTIVEX

Vulnerability Profile: Each of the information system resources of the IT profile has the associated set of vulnerabilities represented as a vulnerability profile. Vulnerability profile is represented by a *4p-bit* binary string *(VP)* with each of the p-bit string representing the vulnerabilities of the corresponding IT resource.

δ (0, p, VP) represents the vulnerabilities of the chosen network resource
δ (p, p, VP) represents the vulnerabilities of the chosen operating system
δ (2p, p, VP) represents the vulnerabilities of the chosen database resource
δ (3p, p, VP) represents the vulnerabilities of the chosen application

Let V_n, Vo, Vd, Va denote the number of vulnerabilities of the type of network, operating system, database, and application chosen, respectively. The value of *p* is chosen in such a way that substrings representing vulnerabilities of various information system resources are of same size. Each vulnerability is represented by a μ-bit binary string where

$$Max\ (V_n, Vo, Vd, Va) \leq 2^{\mu}$$
$$p = Max\ (V_n, Vo, Vd, Va) * \mu$$

Security Profile: Security components are considered external to the firms and are represented by security rules. Each security rule is represented by *4p* bit strings similar to the format of the vulnerability profile where each μ-bit string can be used to find (from R_c) the security component (z) which protects against the respective vulnerability. Each component has cost (C_z) (from R_c) associated with it, such as cost of installation, purchase cost.
Each of the security rule SR_j has cost $cost_j$ calculated as the sum of individual components as shown below

$\forall$ *component z represented in the security rule*
$$cost_j = \sum C_{jz}$$

Each bit of the security rule can take three values representing high (1), moderate (#), and low level (0) of security respectively [12].Firms will select a security rule that matches their vulnerability profile. It is assumed that the firms tend to provide at least moderate level of security to their known vulnerabilities.
The security rule is matched to a vulnerability profile under the following condition

$\forall$ *bit k in the firm vulnerability profile VP and security rule* SR_j

$$\left.\begin{array}{l} VP_k = SR_{jk} \\ or \\ SR_{jk} = '\#' \end{array}\right\} \qquad (1)$$

Every security rule that satisfies the above condition is eligible to be chosen by the firm, if the rule satisfies the budget constraint of the firm.

$$\cos t_j \leq budget \qquad (2)$$

The final security rule adopted by the firm is a function of the strengths of the eligible security rules. Let, Ψ denote the set of all the eligible rules for the firm and s_j represent the strength of security rule *SR* $_j$. Then the probability of adopting security rule *SR* $_j$ is given by

$$p_{SR_j} = \frac{s_j}{\sum_{\Psi} s} \tag{3}$$

Where,

$SR_j \subset \Psi$

The adopted security rule denoted as *security profile (SP)* determines whether firm will or will not be able to provide security for each of its known vulnerabilities.

Risk Profile: The unprotected vulnerabilities expose the information resources of the firm to the threats from the malicious agents. This risk is represented through a *risk profile (RP)* of the firm and is modeled as *4k/μ* string. Each character in the string represents three levels of risk- no risk (1), unknown risk (0) and high risk (-1). Unknown risk represents the risk created from the unknown vulnerabilities that the firm did not seek to protect. The exposed risk *RP* at any bit k of firm is given by

$$RP_k = \begin{cases} 1 & [\forall j, j \in \{\mu(k-1)...\mu\}, (VP_j = SP_j) and (SP_j \neq \#)] \\ 0 \quad \text{if} & [\forall j, j \in \{\mu(k-1)...\mu\}, (VP_j = 0 or 1) and (SP_j = \#)] \\ -1 & [\forall j, j \in \{\mu(k-1)...\mu\}, (VP_j \neq SP_j) and (SP_j \neq \#)] \end{cases}$$

Where, 1, 0, and -1 represent no, unknown and high levels of risk, respectively.

5.2. Design of Intrusion Detection System (IDS)

The IDS is one of the security solutions chosen by us for the firm. The dynamic behavior property of the attacker is to be accounted for in each of the components of the proposed architecture. Thus behavior based IDS in comparison to knowledge based IDS, which is based on tracking user behavior appears to be a natural choice. Hence we have decided to use behavior based approach for the IDS component. As the functionality of IDS component in our architecture is to detect and prevent as many attacks as possible, the rate of false alarms cited as the main drawback of behavior-based techniques will not be a big concern when compared to the accuracy with which behavior based IDS detects attacks.

In order for the security profile to be effective all the attacks be it on the network, host or at the application layer are to be guarded against. Hence simulation of IDS incorporating the features of the network, host and application based IDS is proposed to be developed. The issues involved in the design of IDS are as follows:

1. How to design the IDS such that it allows minimum attacks go undetected?
2. How to design the IDS to accommodate the attacks perpetrated by adaptive attackers?

We propose to solve the above research questions with a learning-based system for detecting intrusion presented in **Figure 3**. The IDS contains a set of post-attack profiles (Profile DB). The IDS initially is void of any profiles. IDS learns about the attack profiles based on its interaction with the firms and attackers. It compares current profile with attack profiles present in Profile DB to say if an attack has actually occurred.

The attacks from the attackers, the results of attacks from firms and the damage caused are given as input to the detection engine. The detection engine is responsible for comparing the incoming attacks with the profiles in the profile DB to identify attacks. If it can detect an attack correctly, the attack is considered as void, otherwise the respective firm encounters the damage pertaining to the attack. In the latter case, the information regarding the result from the attack (attack profile) is used by the IDS by adding attack profile to Profile DB to learn more about the respective attacker. Thus, the learning engine helps the IDS learn from the dynamic nature of the attacker behavior. The commercially available IDS' do not take into account the attacker behavior. The presence of an IDS that dynamically learns from the changing attacker and firm profiles makes this study a significantly better than the previous ones, aimed at simulating static firm/attacker behavior to study information security.

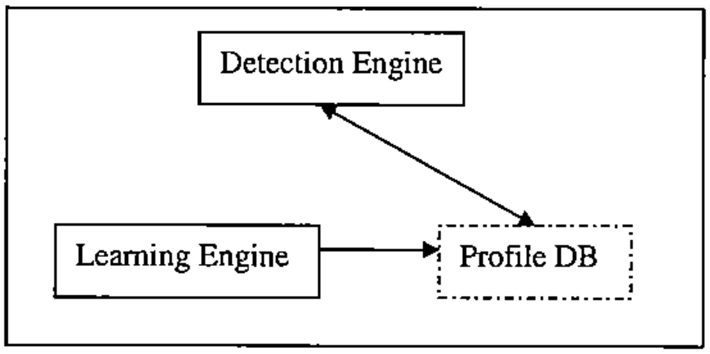

Figure 3: Learning-based System for Detecting Intrusions

The learning engine can be implemented using any of the machine learning techniques such as the genetic algorithms [6], classifier systems [13], genetic programming [3] and neural networks [21]. Thus the proposed architecture can integrate the various learning mechanisms already studied by the other researchers in order to realistically simulate the firm/attacker behavior. For scalability and configurability we plan to implement a distributed intrusion detection system based on the cooperation of autonomous agents [1].

5.3. Design of Attacker

An attacker is malicious and tries to learn as much information related to the firm as possible and tries to attack the firms in innovative ways. His interests in the firm may be monetary or sentimental. The continuous learning behavior of the attackers is to be incorporated in our architecture. The various attackers represented as agents can communicate with each other and exchange information about the firms' resources increasing the danger caused to the firm. All of these issues are to be taken care of. The research issues can be summarized as follows:

Research Questions:

1. How to incorporate the adaptive nature of a real world attacker in the simulated attacker?
2. How to implement communication mechanism among the various attackers?
3. How do various attackers coordinate to attack a particular firm?

We propose to solve the above research questions with the following design. In the initial setup, the attackers choose a firm to attack randomly. The perpetrators are represented as artificial agents and learning mechanisms are implemented to make the agents intelligent enough to represent a real world attacker. The agents learn by gaining intelligence from the knowledge bases that contain data from the real world environment. As shown in **Figure 4**, the agent-based architecture of an attacker consists of

- Intelligent module
- Interface module

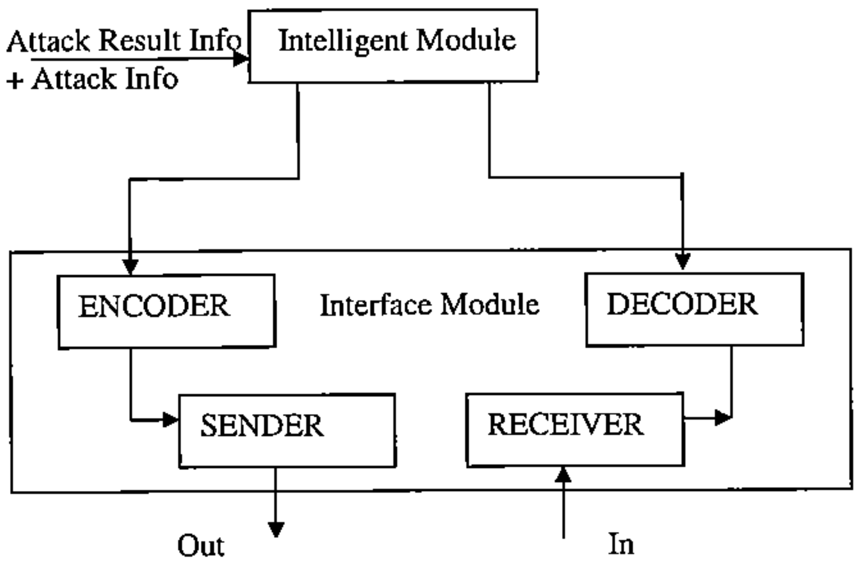

Figure 4: Agent Based Architecture of Attacker

Interface module

This module provides the agent interface with the external world with one of the main functionalities being transforming messages into the format agreed upon as an interface between agents and transferring to the other agents. The other functionality is to make the inverse transformation of the messages received from the external world and handing over them to intelligent module. This module is further subdivided into Encoder, Sender, Receiver and Decoder.

Encoder: Encodes the message to be sent to external agent, as defined by the communication protocol.

Sender: It sends the message to target agent.
Receiver: It receives the message from an external agent and hands over to the decoder to process it.
Decoder: It decodes the external message received into the format expected by the intelligent module.

Intelligent Module

The Intelligent Module models the intelligence of adaptive attackers. Intelligence helps attackers gain utility from every successful attack and exploitation of specific vulnerabilities in the firms. It also helps attackers choose strategies for attack based on the success or failure of previous attacks. Attacker will dynamically adapt to the security the firms are providing with the help of his intelligence.

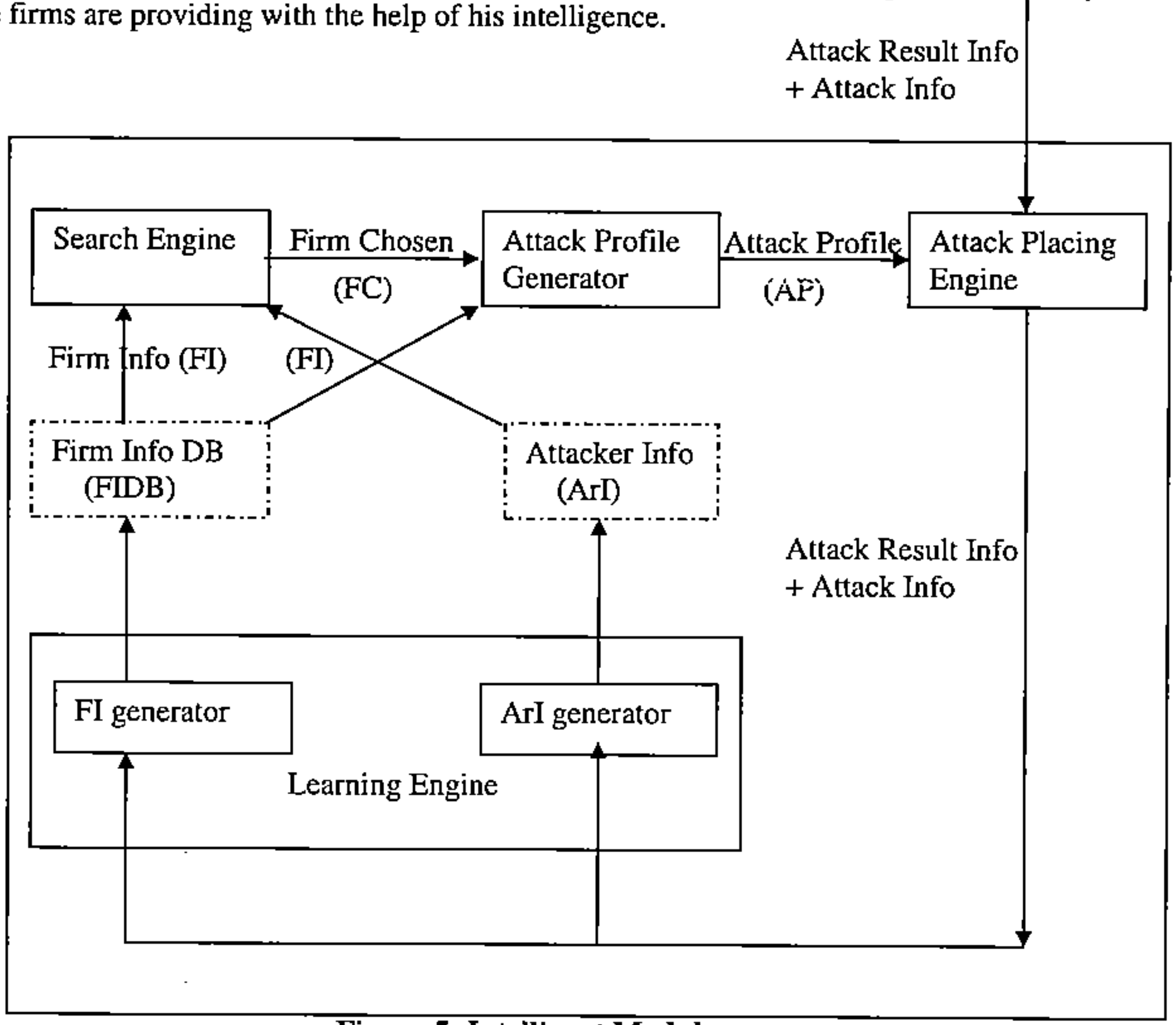

Figure 5: Intelligent Module

Intelligent Module (Figure 5) comprises of the following four components

- Search Engine
- Attack Profile Generator
- Learning engine

- Attack Placing Engine

Search Engine

The basic functionality of Search Engine is to choose a firm to attack. It can be modeled as an optimization function, which maximizes benefits incurred by the attacker, has maximum utilization of the attacker resources, or minimizes risks involved. The Firm Info from Firm Info Database(FIDB) and attacker info (ArI) are inputs to Search Engine, and its output is firm chosen (FC).

Attack Profile Generator

From FC given as input from Search Engine to this module, attack profile generator uses the knowledge of the firm obtained from FIDB along with attacker info (ArI) to generate attack profile.

Learning Engine

It can be subdivided into two major sub components

- FI generator
- ARI generator

Learning Engine receives attack info and attack result info obtained as feedback from the attack placing engine after the attack has been placed and updates AI and knowledge about the target firm. It also receives the attack related information of the attacks performed by other agents on the target firm and enhances its knowledge about that firm.

FI generator updates FI in FIDB, while ARI generator updates attacker resource info based on the feedback of the attack.

Attack Placing Engine

This modules places an attack on the FC, determines the result of the attack, losses to the firm due to attack, benefits to the attacker, new knowledge of vulnerabilities of the firm. All the above is given as feedback to learning engine.

The definitions of the information flow parameters in the intelligent module are as follows:

Firm Info: Firm Info *(FI)* is representative of knowledge attacker has about the firm. This includes the knowledge about the IT profile and the vulnerability profile and is represented as a duple *<AIP, AVP>*.AIP (AVP) is a 4b (4k)-bit string represented in a format similar to the IT profile (vulnerability profile) of the firm.

Attack Profile: Attack Profile *(AP)* is a $4k/\mu$ bit binary string where the bit values represent the capability of the attacker to exploit particular vulnerabilities. It has same format as the risk profile (*RP*) of the firm mentioned earlier. The probability of success of attack perpetrated by any attacker l with profile AP_l on a firm with risk profile RP is given by,

$$p_{lik} = \begin{cases} zero & AP_{lk} = 0 \\ very\ low & (AP_{lk} = 1) and (RP_k = 1) \\ low & \text{if } (AP_{lk} = 1) and (RP_k = 0) \\ high & (AP_{lk} = 1) and (RP_k = -1) \end{cases} \quad (6)$$

where, k represents the bit at which the vulnerability of the firm is exploited

If the attack against a firm is successful, the firm incurs losses associated with the vulnerabilities that were exploited. The security rule adopted by this firm also has an associated decrease in strength due to failure in defending firm's resources. However, if the attack against a chosen firm is unsuccessful, the adopted security rule gains strength.

Attacker Info (ArI): Attacker Info (*ArI*) is a list of resources the attacker possesses (RR) to exploit the vulnerabilities of the firm.

Attack Info: Attack Info (AI) is the information of the firm obtained by the attacker after the attack has taken place. It is represented as a duple <firm-id, losses-incurred-by-firm>
firm-id: Each firm in the environment is assigned a unique id, firm-id.
losses-incurred-by-firm: This parameter represents loss incurred by the firm due to the attack.

Attack Result Info: Attack Result Info (*ARI*) indicates result of the attack success (1) or failure (0).

5.4. Implementation of Attacker Simulator

An attacker has knowledge about the various techniques of placing attacks and tools to use. He or she is intelligent, and thus capable of inferring from this knowledge improved attacks. This intelligence and knowledge can be represented as an expert system. Also as shown above, each of the stages of an attack can be represented as a sequence of steps which can be mapped into rules of the expert system. Hence, we plan to use the concept of expert systems to aid in the simulation of an attacker. The following are the research tasks involved:

1. Obtain the information about various attacks, vulnerabilities of various components of a system, and various hacking tools available.
2. Provide generic simulation model such that various types of hackers from plain player to terrorist fall into the same model.
3. Simulation of a hacker to use various hacking tools, which are used by real world hacker.
4. Find representation for various attacks.
5. Simulation of hacker's intelligence like coming up with new attacks.
6. Can hacker get feedback from the environment e.g. whether an attack was successful, and if so to what extent?

To address the research goal of simulation of an attacker, the pattern of an attacker is to be identified. The following is the description of a classic pattern of an attacker

Classic Pattern of an attacker

The following flowchart [27] in Figure 6 gives various stages involved in placing an attack.

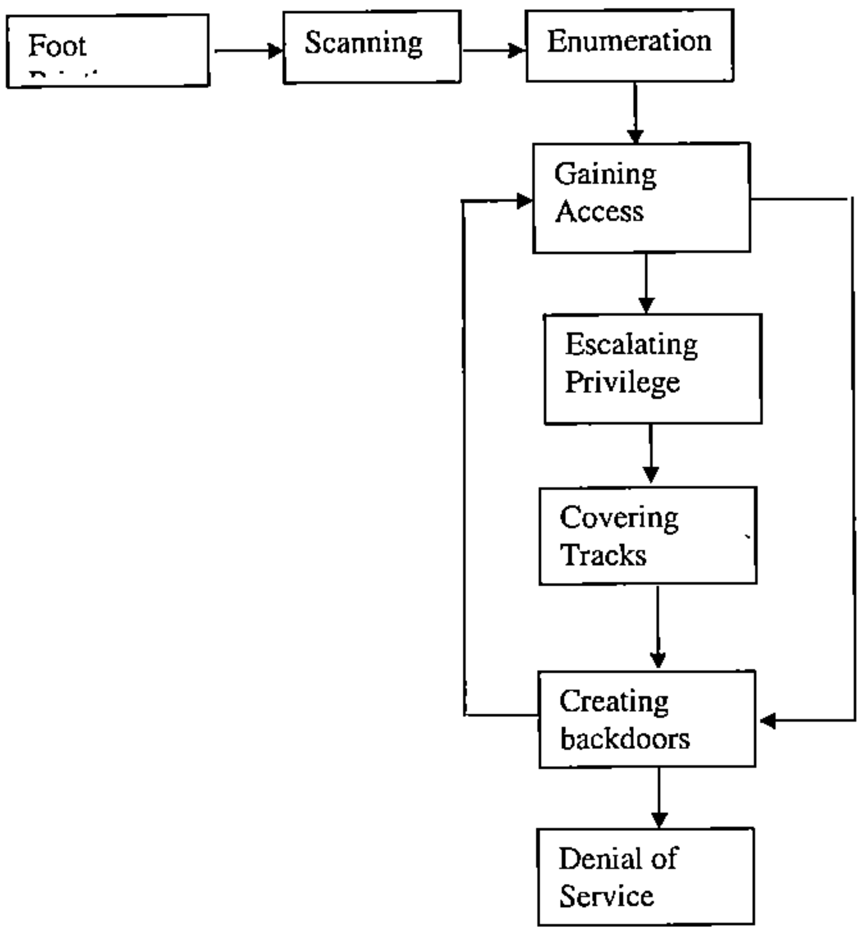

Figure 6: Classic Pattern of an attacker

1. Footprinting:

Footprinting is referred to as the art of gathering target information. The systematic footprinting helps an attacker to create a complete profile of an organization's security posture. Using various techniques such as whois, DNS, etc an attacker can identify critical information.

2. Scanning

With footprinting a list of network IP address ranges, DNS servers and mail servers can be obtained. To determine whether these are alive and reachable from the internet, scanning is done with the help of various tools like ping sweeps and port scans.

3. Enumeration:
This is another information gathering technique, with higher level of intrusiveness. Enumeration involves direct connections and active queries. Various enumeration techniques are present based on the operating system being used, which can be identified from footprinting or scanning.

4. Gaining Access:
Enough data has been gathered by now (based upon the above mentioned strategies), so the attacker can use various exploits (such as buffer overflows) to access the information about the target.

5. Escalating privilege:
The attacker tries to escalate his privileges to system level, if only user level access was obtained before.

6. Covering Tracks:
Once total ownership of the system is gained, the attacker tries to hide this fact from the system administrators to prevent detection. The attacker clears logs and hides the tools he uses.

7. Creating Backdoors:
Once the attacker succeeded in obtaining privileged access, he creates trap doors, so that he can now gain the privileged access whenever he wants to. One way he can do this is by infecting the start up process.

8. Denial of Service:
If the attacker has been unsuccessful in gaining access to the system by the various techniques, he can use his exploits by launching a denial of service attack, and thus disabling the target as a whole.

An attacker tries to gather information initially through the techniques of foot printing, scanning and enumeration. Once the attacker has got enough information, he tries to gain access if possible. If this is not possible, he resorts to the attack of a defeated attacker-denial of service. If possible, the attacker escalates his privilege level, covers his tracks and creates backdoors. Once the backdoors are created, he can now gain access frequently.

Each of the stages in the classic pattern of an attacker can be represented as a sequence of steps to be followed to achieve the desired effect. As an illustration, the representation of the stage of foot printing as sequence of steps is given below.

Step 1: Registrar information, associated whois servers information and a listing of potential domains that match the target is obtained by placing a query to the whois.crsnic.net server. The query can be placed by using the command whois.

```
$ whois "Johnson." @whois.crsnic.net
```

Step 2: Once the registrar is identified, a query to search the specific registrar for all instances of the entity name is placed. The different domains associated with the name specified in the query will be listed.

Step 3: Domain names are selected based on the information the hacker has about the organization's type of business (e.g. if organization is Acme Networks then acme.net is selected as the domain name). A domain query is placed to get the following information:

a. Domain Name;
b. Administrative Contact;
c. When the record was created and updated;
d. Primary and secondary DNS servers.

The administrative contact may give information about the person responsible for the internet connection or firewall.
The record creation and modification dates indicate the accuracy of the information.

Step 4: Once the associated domains are identified, a query to the DNS is placed which could cause the disclosure of internal host names and IP addresses to the attacker. This query can be placed using tools like *nslookup*. The knowledge of internal IP addresses is equivalent to having a complete profile of the organizations internal network.

Step 5: Once the potential networks have been identified, tools like *traceroute* can be used to identify the network topology, access control devices (such as firewall), and potential access paths into the networks.

Approach for Solving Research Issues

We plan to answer the above research issues involved in implementation of attacker simulator as below:

1. Building knowledge base

Various books and online resources give information about currently used hacking tools, various well known attacks, and vulnerabilities of various components of a system (network, OS, database, or application).

Different types of attacks are:

1. Denial of Service
   An attack that targets resources within the network, with the intention of reserving resource and keeping legitimate users from gaining access.
2. Virus/Trojan Horses/Worms

A virus is malicious code that can plant itself into operating systems and programs and modify them. A Trojan horse is a virus that has been hidden inside of legitimate software.

3. Distributed Denial of Service
   It is coordinated attack where armies of zombie [37] machines are employed and controlled by a single master to overwhelm the resources of victims with floods of packets.
4. IP Spoofing
   IP spoofing is accomplished when an outside hacker uses a discovered IP address to gain access to the trusted environments.
5. Replay Attacks
   Replay attacks occur when a hacker intercepts a communication between two parties and replays the message.
6. DNS attacks
   DNS attacks targets resources within the network with the intention of reserving resource and keeping legitimate users from gaining access.
7. Web Defacement
   Such attacks entail on changing the content on web page subtly, thus disseminating false information.

The various hacking tools available as listed in [30, 31], are:

1. Smurf
   Smurf tool is used to cause Denial of Service attacks using ICMP directed broadcast messages.
2. fping
   fping is a ping(1) like program which uses the Internet Control Message Protocol (ICMP) echo request to determine if a host is up.
3. udpscan
   This tool is used to identify open UDP ports by sending a bogus UDP packet then waiting for an ICMP message of "PORT UNREACHABLE".
4. nslookup
   Given an ip address, this tool gives the name of the machine, by sending DNS q.
5. whois
   Given a domain name, this tool provides links to all the sites which have the given domain name as the substring in the web link.

6. *Generic Simulation model*

Hacker's effectiveness depends, among others, on the pool of resources at his/her disposal [35].We propose to develop a model which takes the resource level as a parameter, thus providing generic simulation model.

7. *Usage of hacking tools*

The various hacking tools will be represented as a set of IF - THEN rules in the knowledge base. The rules also specify the resultant output obtained after using the tool.

8. *Representation of attacks*

Every attack has a well-defined sequence of steps to be followed. Each of the steps is represented as IF-THEN rules in the knowledge base.

9. *Modeling of hacker's intelligence*
The inference engine finds out new combinations of attacks and adds new sets of rules to its knowledge base based on the feedback (success or failure) of an attack.

Tools to be Used

There are various freeware tools [38] that can be used to develop expert systems such as:

1. **BABYLON**: This is a modular, configurable, hybrid environment for developing expert systems. It provides the following knowledge representation formalisms: frames, rules, logic (Prolog) and constraints. It requires Common Lisp.

2. **ES**: The ES Expert system development tool supports backward/forward chaining, and fuzzy set relations.

3. **GEST (Generic Expert System Tool):** This shell can be used in a variety of problem domains and supports backward and forward chaining. Its knowledge representation schemes include frames, rules and procedures. Support is also present for fuzzy logic and certainty factor maintenance. It includes blackboard architecture. The user interface utilizes the Symbolics windowing system and is menu and mouse driven.

4. **CLIPS (C Language Integrated Production System):** A forward-chaining rule-based tool written in C by NASA. It can be easily embedded in other applications and includes an object-oriented language called COOL.

5. **FuzzyCLIPS**: This version of CLIPS provides handling of fuzzy concepts and reasoning, in addition to the other CLIPS features.

6. **RT-Expert for DOS, Personal Edition**: A rule-based system with allows for integration of the expert system with C or C++ code.

Among the various tools Clips [29] is the tool chosen. The reasons for choosing CLIPS are summarized as follows:

1. Knowledge Representation: CLIPS provides a cohesive tool for handling a wide variety of knowledge with support for three different programming paradigms: rule-based, object-oriented and procedural. *Rule-based programming* allows knowledge to be represented as heuristics, or "rules of thumb," which specify a set of actions to be performed for a given situation. *Object-oriented programming* allows complex systems to be modeled as modular components (which can be easily reused to model other systems or to create new components). The *procedural programming* capabilities provided by CLIPS are similar to capabilities found in languages such as C, Java, Ada, and LISP.

Portability: CLIPS is written in C for portability and speed and has been installed on many different operating systems without code changes. Operating systems on which CLIPS has been tested include Windows 95/98/NT, MacOS X, and Unix. CLIPS can be ported to any system, which has an ANSI compliant C or C++ compiler.

Integration/Extensibility: CLIPS can be embedded within procedural code, called as a subroutine, and integrated with languages such as C, Java, FORTRAN and ADA. CLIPS can be easily extended by a user through the use of several well-defined protocols.

Verification/Validation: CLIPS includes a number of features to support the verification and validation of expert systems including support for modular design and partitioning of a knowledge base, static and dynamic constraint checking of slot values and function arguments, and semantic analysis of rule patterns to determine if inconsistencies could prevent a rule from firing or generate an error.

Fully Documented: CLIPS comes with extensive documentation including a Reference Manual and a User's Guide.

Low Cost: CLIPS is maintained as public domain software.

The disadvantage of CLIPS is that it is less effective for heuristic classification problem solving than for synthesis because they lack support for backward chaining. Since our implementation does not involve heuristics, this is of no major concern.

5.5. Design of environment for interaction among agents

The current approach involves an integrated intelligent agent-based learning system [2] that is used to simulate attacks and evaluate the effectiveness of the security solutions in preventing these attacks.

This environment is created using the framework of *Synthetic Environment for Analysis and Simulation [26] (SEAS)* developed at Krannert Graduate School of Management, Purdue University. SEAS emulates the US Department of Defense's "War Gaming" paradigm in business and economic settings. It is the application of computer generated modeling techniques, hero-to-fore use to create virtual realities to set up virtual economies. Specifically, SEAS allows for the creation of situation-specific economies through mathematical rule-sets derived from theoretical and empirical work. The goal is to permit scale controlled experiments where human and synthetic players can play together.

5.6. Interaction of the Components

The interaction among the various components of the integrated architecture is as shown in Figure 7.Attacker module generates the attack profile, based on the attacker's knowledge about the vulnerabilities of the firm. IDS compares the incoming attack profile with profiles in its profile DB. If IDS matches the profile with any of its profiles, it recognizes a possible attack, declares the attack result as failure. In the simulation, once the attack is detected, IDS sends the attack result and attack info as feedback to the attacker to help him learn about firm.

If IDS does not match current attack profile with any one of its profiles in profile DB, it passes the attack profile to attack engine. Attack engine compares current attack profile with the risk profile input from the target firm. It compares profile parameters of both the profiles to see if attacker really has knowledge about the vulnerabilities of the firm. If attack engine identifies that attacker does not have above mentioned knowledge, it sends the attack result information as failure along with attacker profile as a feedback to the attacker. Otherwise it calculates and sends the losses incurred along with attack result to the firm. Firm updates its security components from the information obtained from attack engine and sends attack profile to IDS, to store the profile in its DB, for further comparison. The feedback loop as mentioned above continues to improve the attacker's knowledge of firm and IDS knowledge about the attacker.

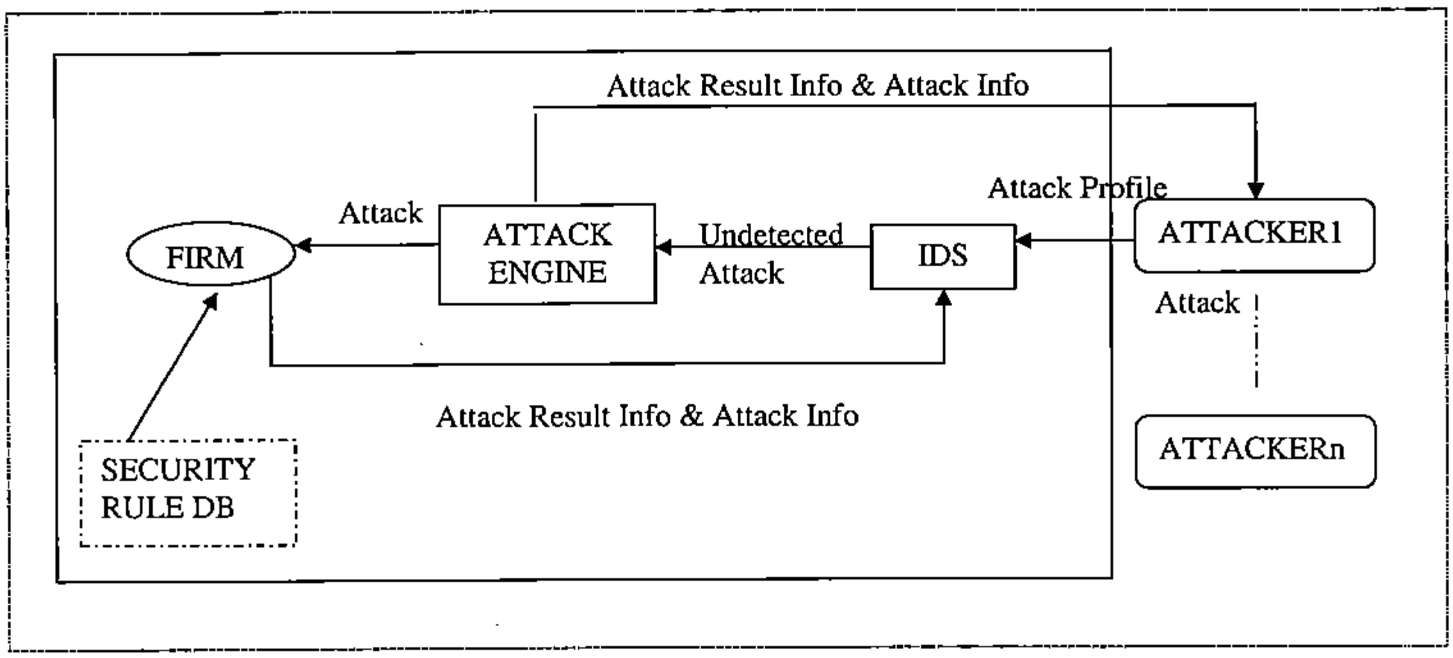

Figure 7: Interaction of components

5.6.1. Simulation of Attacks

The attacks have been simulated by starting with the simple case where we assume that the attacker is static and does not change his behavior based on his learning. In the proposed architecture we would enhance this simulation to incorporate the learning capability which makes attacker and attack simulation more practical.

The simulated attacks by attackers pass through the IDS.IDS based on its knowledge from the user profile database either recognizes the attack, in which case the attack is considered void, or lets the attack pass through it. If the attack is allowed to pass through

IDS, based on the knowledge the attacker has about firm's vulnerabilities, various resources he has, and the effectiveness of the current firm profile the damage caused to the firm is calculated. Depending on the extent of the damage undergone and its budget constraints the firm's security profile is enhanced.

5.6.2 Evolutionary Principles for Optimizing Security Profile

The security profile of the firm is evaluated in light of the attacks. The fitness of each of the firm's security profile is computed based on its effectiveness to withstand an attack. Genetic algorithms (GA), a stochastic population based search is used to come up with the effective security profile within the budget considerations.

6. Experiments

The proposed architecture has been partially implemented to include the firm profile generator, attack simulator and the GA engine for estimating effective security profile. The following are the proposed experiments:

6.1. Experiment A: Testing the Firm Profile Generation module

Purpose: The purpose of the experiment is to test incorporated learning behavior on the firm side using genetic algorithms and see if the firm is able to get to an effective selection of security components.

Method: Firms represented as human agents and attackers represented as artificial agents are placed in the SEAS environment. Only the static nature of the attackers is simulated. Several generations are run until an effective firm profile is obtained.

***Input Parameters*:**
1. Information about various properties of organization as
    a. Budget of firm
    b. Wealth of firm
    c. Type of Network
    d. Type of OS
    e. Type of Database
    f. Types of Applications
2. Capabilities of various attackers indicating the types of resources attacker is capable of attacking.

Output Parameters:
Selection of security components that represents an effective profile for the organization.

Analysis and Conclusion to be drawn:
The selection of security components is to be done from the real life, without taking the simulation environment into

consideration. This is compared with the output obtained from the experiment and based on this the accuracy of the method
used is estimated.

6.2. Experiment B: Testing the Attacker Simulation Module

Purpose: The purpose of the experiment is to test learning behavior incorporated on the attacker side. Only the static behavior of firm is simulated.

Method: Attackers represented as artificial agents are placed in the SEAS environment. The experiment is run for several generations in the sense of genetic algorithms.

***Input Parameters*:**
Values of parameters as represented in the attacker profile are obtained from the real world and taken as initial data set for the attacker.

***Output Parameters*:**
The attacks which the module generates when run through different generations in the sense of genetic algorithms are captured.

Analysis and Conclusion to be drawn:
Strength of attacks is compared across generations. We expect to see increasing trend in the strength.

6.3. Experiment C: Intrusion Detection Module

Purpose: The purpose of the experiment is to test the functionality and learning behavior of the Intrusion detection component.

Method: Normal users and attackers are represented as artificial agents and are placed in the SEAS environment. The learning behavior of the attackers is also taken into consideration. The Intrusion Detection component simulated is tested against the artificial agents.

***Input Parameters*:**
Values of parameters, as represented in the attacker profile, are obtained from the real world for normal users as well as attackers, and taken as initial data set for the artificial agents.

***Output Parameters*:**
The behavior of the IDS as it is run through various generations is captured. The number of false alarms generated is also captured.

Analysis and Conclusion to be drawn:

We expect to see decreasing trend in number of undetected attacks and the number of false alarms generated across generations.

6.4. Experiment D: Integrated Framework

Purpose: The purpose of the experiment is to integrate all the components and test the system as a whole.

Method: Firms represented as human agents, users represented as artificial agents and the IDS are placed in the SEAS environment. The dynamic nature of the attackers is also simulated. Several generations are run until an effective firm security profile is obtained.

Input Parameters:
Values of parameters as represented in the firm and attacker profiles are obtained from the real world and taken as initial data set for the respective profiles.

Output Parameters:
Selection of security components that represents an effective profile for the organization.

Analysis and Conclusion to be drawn:
The selection of security components is also done from the real life, without taking the simulation environment into consideration. This is compared with the output obtained from the experiment and based on this the accuracy of the method used is estimated.

7. Conclusions

7.1. Summary of Research Problems and Research Plan

We propose building an Integrated Agent-Based simulation environment to find the effective security profile within the budget considerations. The key ideas involved are: dynamic firm profile; behavior based dynamic attacker profile generator; learning-based IDS, and an evolutionary search via genetic algorithms for the effective security profile. The significant contribution of this work is that it integrates the management and computer science concepts to build a seamless interactive simulation environment in order to address an important problem in the field of information security.

7.2. Significance of the Problem

An easy to use and realistic managerial decision tool to choose the right security profile within the budget constraints is the need of the hour. We have proposed an outline for the design of an agent-based architecture with behavioral and information security concepts as the most natural way of implementing real world ideas in an integrated manner.

References

1. J.S. Balasubramaniyam, J.O. Garcia-Fernandez, D. Isacoff, E. Spafford, and D Zamboni, "An Architecture for Intrusion Detection using Autonomous Agents," COAST Technical Report, May1998.

2. A.R. Chaturvedi, M. Gupta, S. Mehta and L. Valeri, "Fighting the Wily Attacker: Modeling Information Security Issues for On-Line Financial Institutions using the SEAS Environment", in proceedings of *INET JAPAN 2000 Conference*, July 2000.

3. M. Crosbie, "Applying Genetic Programming to Intrusion Detection," in *Proceedings of 1995 AAAI Fall Symposium on Genetic Programming*, November, 1995.

4. D. E. Denning, "An intrusion-detection model," *IEEE Transactions on Software Engineering*, 13, 2, 222-232, February 1987.

5. S. Forrest, S.A. Hofmeyr and A. Somayaji, "Computer Immunology," *Communications of the ACM*, 40, 10, 88-96, October 1997.

6. D.E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley: Reading, MA, 1995

7. S. Gordon, "Generic Virus Writer," 6th International Virus Bulletin Conference, Brighton, UK, September 1996

8. M.Gupta, "An Analysis of Information Security Issues in an Artificial Environment," Working Paper, Krannert School of Management, December 2001

9. S.A. Hofmeyr, "An Immunological Model of Distributed Detection and its Application to Computer Security," Ph.D. Thesis, University of New Mexico, May 1999

10. J. H. Holland, "Adaptation in Natural and Artificial Systems," University of Michigan Press, Ann Arbor, MI, 1975

11. J.D. Howard, "An Analysis of Security Incidents on the Internet," Ph.D. Thesis, Carnegie Mellon University, 1995, Chapter 6. (http://www.cert.org/research/JHThesis/Chapter6.html)

12. Irvine, C., and Levin, T. "A Note on Mapping User-Oriented Security Policies to Complex Mechanism and Services", Technical Report, Department of Computer Science, Naval Postgraduate School, Monterey, CA

13. S. Kumar and E. Spafford, "A pattern matching model for misuse intrusion detection," in *Proceedings of the 17th National Computer Security Conference*, 11-21, October 1994

14. T. Lane and C.E. Brodley, "Temporal Sequence Learning and Data Reduction for Anomaly Detection," in *Proceedings of the Fifth ACM Conference on Computer and Communications Security*, pages 150-158, 1998

15. S.D. Moitra and S.L. Konda, "A Simulation Model for Managing Survivability of Networked Information Systems", Technical Report, CMU/SEI, December 2000

16. S. Potluri, "Application of SEAS to Security Profile Generation," Working Paper, Purdue University, October 2001

17. J. Rees, S. Bandyopadhyay and E. Spafford, "Policy Framework for Interpreting Risk in e-Commerce Security," Center for Education, Research and Information Assurance (CERIAS), Technical Report #TR-2000-01.

18. B. Schmidt, "The Modelling of Human Behavior," SCS Publications, 2000

19. E. Shaw, K. Rubin and J. Post, "The Insider Threat to Information Systems," *Security Awareness Bulletin,* n. 2, June 1998, 27-46; Steven Harrington, "Computer Crime and Abuse by IS Employee," *Journal of System Management,* vol. 5, n. 2, autumn 1995, pages 6-10

20. S. Stolfo, A.L. Prodromidis, S. Tselepis, W. Lee, D. W. Fan and P. K. Chan, "JAM: Java Agents for Meta-learning over Distributed Databases", Technical Report, Department of Computer Science, Columbia University, 1997

21. K.M.C. Tan "The Application of Neural Networks to UNIX Computer Security," In *Proceedings of the IEEE International Conference on Neural Networks, Vol.1* pp. 476-481, 1995.

22. Tudor, J. K., "Information Security Architecture: An Integrated Approach to Security in Organization," CRC Press, September 2000.

23. "Global Information Security Survey", Information Week, July 2000

24. "Information Security Industry Survey", Information Security Magazine, July 2001.(http://www.infosecurity mag.com/)

25. R. Bace and P. Mell, "Intrusion Detection Systems," NIST Special report on IDS, August 2001. Available at: http://csrc.nist.gov/publications/nistpubs/index.html.

26. R. Chaturvedi and S. R. Mehta, "Simulations in Economics and Management: Using the SEAS Simulation Environment", *Communications of the ACM*, March 1999.

27. Stuart McClure, Joel Scambray, George Kurtz, *Hacking Exposed: Network Security Secrets and Solutions*, third edition, McGraw-Hill, 2001.

28. Robert S. Engelmore, Edward Feigenbaum, Expert Systems and Artificial Intelligence, Available at : http://itri.loyola.edu/kb/c1_s1.htm

29. Gary Riley, A tool for building expert systems, Available at: http://www.ghg.net/clips/CLIPS.html.

30. hdc archives, Available at: www.hackers.com/html/archives.html.

31. Defensive and hacking tools, Available at: www.hackingexposed.com/tools/tools.html.

32. Vulnerabilities in operating systems, Available at: www.securityfocus.com.

33. The Twenty Most Critical Internet Security Vulnerabilities, Available at: www.sans.org/top20.htm

34. Fyodor, Exploit world! Master Index for All Exploits, Available at: http://www.insecure.org/sploits_all.html.

35. @stake, center of excellence: attack simùlation, Available at: http://63.251.138.38/atstake/acrobat/coe_attack_simulation.pdf.

36. eEye digital security, CHAM (Common Hacking Attack Methods) and its use in Retina the Network Security Scanner, Available at: http://www.eeye.com/html/Research/Papers/DS20010106.html.

37. Michael A. Vatis. Cyber Attacks During the War on terrorism: A Predictive analysis, 2001, Institute for Security Technology Studies at Dartmouth College.

38. Expert System tools, Available at: http://www.cs.cofc.edu/~manaris/ai-education-repository/expert-systems-tools.html.