A Hybrid Model for Network Security Systems: Integrating Intrusion Detection System with Survivability

Tarun Bhaskar[1], Narasimha Kamath B[2], and Soumyo D Moitra[3]
(Corresponding author: Tarun Bhaskar)
Computing & Decision Sciences, GE Global Research, Whitefiled Road, Bangalore-560066, India[1]
i2 Technologies, Bangalore, India[2]
Operations Management Group, Indian Institute of Management Calcutta, Kolkata, India[3]
(Email: tarun.bhaskar@ge.com)
(Received Oct. 6, 2006; revised and accepted May 22, 2007)

Abstract

Computer networks are now necessities of modern organisations and network security has become a major concern for them. In this paper we have proposed a holistic approach to network security with a hybrid model that includes an Intrusion Detection System (IDS) to detect network attacks and a survivability model to assess the impacts of undetected attacks. A neural network-based IDS has been proposed, where the learning mechanism for the neural network is evolved using genetic algorithm. Then the case where an attack evades the IDS and takes the system into a compromised state is discussed. We propose a stochastic model which enables us to do a cost/benefit analysis for systems security. This integrated approach allows systems managers to make more informed decisions regarding both intrusion detection and system protection.

Keywords: Genetic algorithms, intrusion detection systems, neural networks, simulation, survivability

1 Introduction

The dramatic growth of the Internet and other computer networks has been accompanied by a significant increase in network intrusions and attacks on computer systems. Given the enormous dependence of both individuals and organizations on information networks, including the Internet, it is important to develop cost-effective measures to mitigate this threat. Articles abound in the literature that address this issue: [1, 10, 12, 21, 27, 30]. According to a survey done by ICSA Labs (a subsidiary of security firm TruSecure), the number of attacks on organisations has increased tremendously in 2003 and the average cost of cleaning has also gone up to $100,000. The number of incidents reported to Carnegie Mellon's Computer Emergency Response Team/Coordination Center (CERT/CC) has increased from the range of 2000-3000 in early and mid 1990s to 52,658 in 2001, 82,094 in 2002 and 137,529 in 2003. In February 2000, several web-sites including Yahoo, Amazon, E-bay etc. were shut down due to denial of service attack on their servers. As per the data published by The US General Accounting Office (GAO), 250,000 attacks were made on the Federal Computer Systems and only 1-4% of those were detected. Such examples clearly demonstrate the need for good network security for computer networks. Among the key concerns regarding the security of computers are (1) the detection of intrusions and (2) the survivability of networked systems under attack. Not only do we need to quickly and efficiently detect network intrusions and attacks, but we should also have the most appropriate defenses if and when a computer system is attacked, since experience shows that there will inevitably be some attacks that either escape detection or cause damage in spite of being detected.

The issue is the *extent* to which we deploy defense mechanisms against these attacks. Stronger defenses will imply higher costs. We have to consider trade-offs between security and costs, where compromising on costs could include possible functional limitations to the system ([12, 17]). That is, while we need to enhance security, we also need to decide *by how much* should the network security be enhanced so as to be cost effective. The most appropriate level of security would be based on the organization's needs, its financial abilities, and the potential threats it faces. In view of this, a cost/benefit analysis of network systems security is important.

In this paper we consider both the detection issue as well as the level of security that would be appropriate. The objectives of this paper are the following:

- To integrate intrusion detection with the survivability (or impact) analysis to provide a complete view

of network security.

- To develop a new model for IDSs based on neural networks, that uses genetic algorithms (GA) for developing the learning rule.
- To develop a model for probabilistically predicting the state of the system under attack.
- To perform a cost/benefit analysis for an organisation implementing network security.

The model and approach presented in this paper allows us to analyse the performance of an IDS. It also helps us to analyse the state of the system if an attack *goes through*. In cases where an attack goes through, we estimate the impact on the system, that is, the degree to which its functionality has survived. Thus we have developed a hybrid[1] model that tracks the complete sequence of events associated with a network intrusion or attack. The synergistic advantage of our approach is that improvements in the performance of the IDS can be directly incorporated into survivability estimation. In general, a systems manager would like to manage both the security as well as the investment for that security. The hybrid model we propose has the potential to lead to a Decision Support System (DSS) that could help systems managers make more informed decisions about the IDSs for their sites and about the kind of protection their systems should have.

The structure of rest of the paper is as follows. In Section 2, we discuss the existing literature and some unresolved problems. In Section 3, we discuss the integrated model for network security. In Section 4, we develop the IDS which uses GA to evolve the learning rule for Artificial Neural Networks (ANN). Section 5 considers the case where the attack penetrates the system, develops the stochastic model and discusses the cost/benefit analysis. Section 6 concludes the paper by summarising the contribution of the paper and identifying future research directions.

2 Research Issues

To arrive at the proposed hybrid model, this paper draws on two streams of research: intrusion detection (in particular, GA for intrusion detection) and analysis of the impact of network attacks on systems (specifically, survivability). We briefly sketch the outline of the relevant issues within each stream.

2.1 IDS Related Literature

There are two broad techniques for network security: protection and detection ([23]). The protection technique tries to protect the system from attack. The most commonly used protection device is the firewall which allows only valid data to pass through it. Another approach is using an IDS, which collects information from a variety of systems and network sources, and analyses the data stream for signs of intrusion or misuse. The modelling of an IDS has always been an important problem for researchers in this area. [9] proposed an IDS model based on historical data and [23] provides a detailed survey of the work done on this topic. The success of an IDS is measured by the false positives and the true positives.

Researchers have shown that the efficiency of an IDS can be improved by using data mining techniques. [20] presented a data-mining based model for an IDS. Neural networks are among the most effective data mining techniques. [31] compared different data mining techniques and found that neural networks were better at identifying malicious connections. [3] applied neural networks for modelling an IDS. [21] also stated that the efficiency of an IDS improves by using neural networks. The major problem with neural networks is that the training consumes a lot of time and processing power because of the gradient-based learning algorithm.

GA have also been used for modelling an IDS. The concept of genetic algorithm was given by [15] and was successfully used as an optimisation method by [14]. [2] used genetic algorithms for learning the behavior of the computer user. [8] used genetic programming as suggested by [19] for an IDS and achieved very good results.

[12] consider the cost factors associated with IDSs and present an approach for assessing intrusion detection models to optimize benefits and to minimize costs under a given set of conditions. Although, they develop a cost model of an IDS with a number of component costs, they only consider cost metrics in terms of levels on a scale of 0 to 100, rather than actual costs in monetary values. They have considered only four types of intrusions (probing, DoS, illegal local access and illegal root access). [4] proposed a cost-effective model for electronic data processing systems. [6] provides a long list of various attack methods that includes not only network attacks but also physical attacks and accidents that could damage a computer. Among the network attacks, he identifies Trojan horses, information changes in transit, viruses, input overflow, network service and protocol attacks and interprocess communication attacks.

Next we turn to the issues related to the impact of attacks on systems and survivability.

2.2 Survivability Related Literature

Survivability is the degree to which a network computing system continues to provide essential services in the presence of attacks and failures, and recover full services in a timely manner. Survivability is mainly dependent on the type of network, the nature of the attacks and the type of defense mechanism deployed by the organisation. [16] has undertaken an extensive survey of the nature of attacks on computer systems and has reported the analysis of data on computer security violations. [10] have

[1] In this paper we use the term hybrid in the particular sense of combining the two issues of intrusion detection and survivability analysis

discussed the survivability requirements and strategies to achieve it. A case analysis ([11]) has also been carried out on how the survivability can be defined and analyzed. [7] has applied the deception techniques like honey pots, address aliasing and multiple address translation as cost efficient techniques to increase the effectiveness of the system against attacks. [24] have used the spiral model to discuss the survivability life cycle and have come up with a system architecture for different types of attacks.

[13] have developed high performance solutions to achieve survivable systems in an unbounded environment by applying emergent algorithms. The defense mechanism plays an important role if the intruder targets the intrusion detection system for launching the attack. [28] have identified the vulnerabilities in general IDSs and have given possible solution techniques. The issue of survivability analysis has been addressed from the design perspective in [18] where they assess the effect of faults through scenario graphs. The importance of improving cyber security using 1) an objective function that incorporates survivability as well as costs and 2) a monitoring system for early detection is emphasized in [30]. In this paper we have attempted such an integration of intrusion detection and survivability analysis.

2.3 Unresolved Issues

To date, literature addressing the challenge of considering intrusion detection together with an analysis of the impact of the attacks that *'go through'* is sparse. To the best of our knowledge only [17] have discussed the cost/benefit analysis of IDS implementation. The costs of security compromises are evaluated in terms of "Annual Loss Expectancy (ALE)" and "Return on Security Investment (ROSI)". However, they did not address the impact of attacks, which we attempt to do using the proposed hybrid model. In addition to a new and faster method for intrusion detection based on genetic algorithms, we present a new method of modelling intrusions or attacks and also estimate the degree to which a system has survived an attack.

Another contribution of this hybrid model is that it can be the basis of a DSS for network security. In any organization that uses the Internet or any information network, there will always be a need to assess the network security situation and perhaps the need to install or upgrade an IDS. Also, given the continually changing nature of network intrusions and attacks, there will be a need to regularly assess the network security for the organization's systems. Thus a DSS that could review both the performance of the IDS and the system survivability can help in arriving at better decisions regarding alternative security measures. The approach developed in this paper is a preliminary but necessary step towards developing such a DSS.

3 A Network Security Model

It is futile to expect absolute security of a network system. As the sophistication of attackers increases, any computer connected to an open system (such as the Internet) may be attacked and compromised to some degree. The most common defensive step that such an organisation takes is to deploy an IDS. Since new types of attacks are constantly evolving there is a need for an IDS with a faster learning method and this is the objective of the proposed IDS model.

Figure 1 depicts the response of an IDS vis-a-vis an attack/no attack scenario, highlighting the false positives and true positives. We are interested in analysing the system which is under attack. There can be two cases in which the system gets compromised due to the attack:

- Detected by the IDS, but not stopped; (for example, the distributed denial of services (DDoS) attack is almost impossible to stop even if we know about it).
- Undetected by the IDS.

We assume that the IDS has been deployed along with a suitable preventive mechanism. This may not be foolproof in the sense that some detected attacks penetrate the system. Then the impact of an attack, if it goes through, has to be evaluated.

The different phases of a networked system with respect to attacks and security are shown in Figure 2. There can be several levels of protection and we need to decide on the most *suitable* level for a given organisation, given its financial and operational constraints.

If an organization wants to have a better control over its system, it should have an effective detection and prevention system in place. But it needs to invest more to get a better system. The organization needs to analyze the impact of any attack, which goes into the system and based on the impact and its financial and operational constraints, it needs to decide on the investment on the security of the system.

In this paper we try to build a framework for the holistic approach for network security strategy of an organization. We first develop an IDS, which tries to detect any malicious connection in an efficient way. Then based on the performance of the IDS, we try to analyze the impact of any attack, which gets into the system. Since it is very difficult to get any real life data on the impact of any attack on a network system, we use simulation to do the impact analysis. This analysis can be used to make decisions about investment on network security of an organization. In the next section we discuss the design of an efficient IDS and we discuss the simulation study for impact analysis in the subsequent sections.

Prediction by the IDS	Actuality: Attack	Actuality: No Attack
Attack	Detected	False Alarm
No Attack	Not Detected	Correct Prediction

Figure 1: Output of the IDS

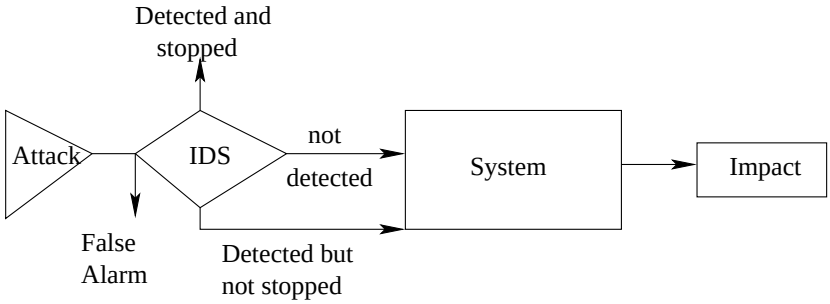

Figure 2: States of security systems

4 GA Based Neural Network for IDS

4.1 Genetic Evolution and Learning

The aim here is to develop an IDS which adapts to the environment. Evolution and learning are the two most fundamental processes of adaptation. Since learning through neural network is a complex, time consuming process, the connection between learning and evolution can be used to decrease the complexity of the problem and hence speed up the adaptation. A framework to establish a relationship between evolution and learning has been given by [5].

According to Chalmers the usual learning process, as in ANN, is a connectionist approach in which the nodes in a layer are connected to those in a different layer. The kind of emergence found in genetically-based systems differs from that found in connectionist systems. Connectionist systems support synchronic emergence, that is, emergence over levels, whereas genetic-based systems support diachronic emergence, that is, emergence over time. So he proposed a method to achieve synchronic emergence through evolutionary methods, which involve making an indirect connection between a genotype and a phenotype. The genotype is the collection of genetic information passed on between generations (in GA it is a string of bits). The phenotype is the behavioral expression of the genotype, an entity that interacts with the environment and is subject to selection by differential fitness.

The motivation behind indirect mappings from genotype to phenotype is to allow for an open-ended space search. A feature of current genetic search is that a genotypic space is precisely specified in advance, and the search cannot go outside this space. Say, for example, we specify a genotype of 5 bits. By doing this we restrict the search space to vary from 0 (00000) to 32 (11111). When this is coupled with a direct genotype to phenotype mapping, it translates directly into strong-delineated phenotypic space whose properties are well understood in advance. Synchronic emergence guarantees that high-level phenotypic characteristics are not limited in advance.

The problem discussed above gives rise to what is called genetic connection. Since it is difficult to know in advance precisely which low-level computations are appropriate for a specific high-level behavior, it makes sense to use genetic methods to search for an appropriate low-level computational form. The rest of this section deals with the application of genetic connection in modelling GA based neural network algorithm for IDS.

4.2 Data Source and Processing

We have used the Knowledge Discovery in Database (KDD) Cup data provided by the Lincoln Labs of Massachusetts Institute of Technology. The data set was generated via a simulated U.S Air Force LAN[2]. Originally the data consisted of raw TCP dump data from the network. From a sample of this raw data, connection records

[2]The data were downloaded from the Website: http://kdd.ics.uci.edu/databases/ kddcup99/kddcup99.html

were established based upon the sequence of TCP packets. The data which represent a connection consisted of 41 attributes and a final output indicating whether the connection is a normal or a malicious one. Since we are distinguishing only between normal and malicious connection, the output is binary.

The ratio of normal to abnormal sequences in the training and testing dataset is an important variable and affects the performance in multiple ways. After trying different proportions, [29] found that a balanced proportion (i.e. equal proportion of normal and abnormal data) gives the best results. This finding was ratified by [31] who also observed similar results. Therefore, for our experiment, we also take a balanced dataset.

4.3 The Learning Task and the Topology of the Neural Network

There are two standard categories of learning: supervised learning (learning with feedback regarding the desired action) and unsupervised learning (learning without any feedback). The problem and the data clearly indicate that our problem is of the supervised learning type. The next step is the selection of the neural network architecture which should be simple yet effective. We choose the simplest non-trivial topology of single-layer feed-forward network with 41 input and 1 output nodes. So the experiment was performed on a single-layer feed-forward network with supervised learning.

4.4 The Learning Mechanism

Our aim is to come up with a non-gradient based learning algorithm. For this we need to code complex forms of weight-space dynamics into a simple linear genome. We cannot express all possible kinds of weight-space dynamics under a single encoding. So for this experiment we assume that changes in the weight of a given connection are functions of only the information that is local to that connection, and that the same function will be employed for every connection. For a given connection, from input unit i to output unit j, local information includes four items:

a_i the activation of the input unit i.

o_j the activation of the output unit j.

t_j the training signal on output unit j.

w_{ij} the current value of the connection strength from input i to output j.

The genome must encode a function F, where $\Delta w_{ij} = F(a_i, o_j, t_i, w_{ij})$.

F is taken as a linear function of the four dependent variables and their six pairwise products. Thus, F is determined by specifying ten coefficients.

The genome specifies these ten coefficients directly, with the help of an eleventh scale parameter. We let

$$\Delta w_{ij} = k_0(k_1 w_{ij} + k_2 a_i + k_3 o_j + k_4 t_i + k_5 w_{ij} a_i + k_6 w_{ij} o_j + k_7 w_{ij} t_i + k_8 a_i o_j + k_9 a_i t_i + k_{10} o_j t_i).$$

The genome consists of 35 bits in all. The first five bits code the scale parameter k_0, which can take the values $0, \pm 1/256, \pm 1/128, ..., \pm 32, \pm 64$, via exponential encoding. The first bit encodes the sign of k_0 (0=negative, 1=positive), and the next four bits encode the magnitude. If these four bits are interpreted as an integer j between 0 and 15, we have

$$|k_0| = \begin{cases} 0 & \text{if } j = 0 \\ 2^{j-9} & \text{if } j = 1, 2 \ldots 15. \end{cases}$$

The other 30 bits encode the other ten coefficients in groups of three. The first bit of each group expresses the sign, and the other two bits express a magnitude of 0, 1, 2 or 4 via a similar exponential encoding. If we interpret these two bits as an integer j between 0 and 3, then

$$|k_i| = \begin{cases} 0 & \text{if } j = 0 \\ 2^{j-1} & \text{if } j = 1, 2, 3. \end{cases}$$

4.5 The Experiment

The steps of the experiment are explained in this section. The selected data were divided into 30 datasets. Each dataset was called a task. Of these datasets, 20 were used for training and 10 were held back for testing. The selection of those 20 datasets was random. This was done to make sure that the training and the testing set were changed after every epoch to avoid any biased training of the neural network. The overall procedure in one epoch was as follows:

- Each chromosome, representing one learning rule, was evaluated. To evaluate a chromosome, an appropriately sized network was configured for each of the 20 tasks. The following procedure was conducted for each task.
  - For each epoch, the network was shown all the training patterns, and the weights were updated according to the encoded learning rule. The absolute values of connection strengths were capped at 20 to prevent runaway learning rules.
  - The network was presented with each pattern once more, and its outputs were recorded. If the desired and actual outputs were on opposite sides of 0.5, the response was counted as an error.
  - Fitness was calculated as $100 * (1 - \frac{\text{number of errors}}{\text{number of patterns}})$, yielding a percentage value between 0 and 100. This function was used for its simplicity and ease of interpretation.
- The fitness of a chromosome was taken as its average fitness over all twenty tasks, and chromosomes were probabilistically selected for inclusion in the next

generation based on their cumulative fitness over generations. The selection mechanism was roulette selection with elitism (that is, the most-fit chromosome was always included in the next generation).

- After the 500th generation, the best chromosome was selected, and the learning rule was encoded using it. The fitness of the learning rule was tested on the 10 datasets that were held back for the testing task.

The same process was repeated for ten epochs and the results were analyzed. The above process was repeated with different parameters of genetic algorithm. A two-point cross-over and elitist selection were used. The cross-over rate was varied from 50% to 80% with an increment of 5% in every step. The mutation rate was varied from 1% to 5% with an increment of 0.5%. The algorithm was coded in Java and run on a Linux based machine with CPU speed of 1.4 GHz and 512 MB SDRAM.

Using a GA based method not only helps us in avoiding the cumbersome job of gradient based learning and devise a learning algorithm from the data itself, it also gives us some flexibility over the time required for training the algorithm. It would be useful if we discuss a few steps to control the time consumed in the training process of the proposed method. The time taken for training of the model depends on several factors. The first and the most important factor is the data and division of the data into training and test sets. In this experiment we are dividing the 30 datasets into 20 training and 10 test sets. The time taken by the algorithm can be brought down significantly by reducing the testing task. But reducing the size of the testing data can also effect the performance of the IDS. So one need to decide on a suitable size of test data after based on the trade-off. The second important factor is the number of generations. We have used 500 generations as the terminating point for each run of the GA. The time consumed can be reduced by reducing the number of generations. The selection, cross-over and mutation operators also effect the time taken for training the network.

4.6 Results

Let us now analyze the results obtained from the experiments conducted as explained above. As discussed, we tried the proposed algorithm with different sets of parameter values. The best fitness was 92.4% which was achieved with 55% cross-over rate and 1% mutation rate.

For illustration, we provide the two best results we obtained during the experiment. The best fitness value was 92.4% and the respective values of k's were 2, 0, 0, 1, -1, 0, 0, 0, -2, 2, 0. The corresponding learning functions derived from this set of values are:

$$\begin{aligned}\Delta w_{ij} &= 2(o_i - t_i - 2a_j o_i + 2a_j t_i) \\ &= 2(o_i - t_i)(1 - 2a_j) \\ &= 4(a_j - 0.5)(t_i - o_i).\end{aligned}$$

The second best fitness was 90.2% with k values: -1, 0, 0, 1, 1, 0, 0, 0, -2, 4, 0 respectively. So the corresponding learning rule is:

$$\begin{aligned}\Delta w_{ij} &= -1(o_i + t_i - 2a_j o_i + 4a_j t_i) \\ &= [2a_j(o_i - t_i) - (o_i + t_i)].\end{aligned}$$

The performance of the IDS is evaluated based on the average efficiency. The average fitness over ten epochs was 80.4%. This means that on an average the IDS can detect a malicious connection in about 80% of the cases. We have investigated the effect of the best case and average case IDS on the survivability of the system in Section 5.3.

Since these rules have been evolved using data and a genetic algorithm, the theoretical authenticity of these rules need to be evaluated. The similarity of these rules with the popular delta rule is evident. This resemblance shows that the rules evolved do not deviate much from our current theoretical understanding of the learning rules. As discussed in [31], training a neural network is a time consuming task due to the calculation of gradient. Since we have avoided the gradient method, the complexity in the training of the network has been reduced.

Having developed a methodology for an IDS, we now consider the survivability of a networked computer system when it experiences an attack that was either not detected or was detected but not thwarted.

5 Survivability Model

On an average the IDS can detect about 80% of the attacks. The most sophisticated IDS detects around 98% of attacks. The undetected attacks may compromise the system state depending on the type of attack and defense mechanism installed in the system. Even some detected attacks may not be preventable. The survivability model determines the compromised state of the system and assesses the degree to which it has survived (or survivability). The model consists of three parts: a stochastic process for the generation of attacks on the system, a model for the state transition process of the attacked system given a level of defense, and a method of estimating the expected survivability of the system. This will provide a managerial perspective on the trade-off between costs and system survivability to determine the most appropriate level of defense for a system, given that no IDS is 100% effective.

In this paper the model proposed by [25] has been generalized by adding more types of attacks, defense mechanisms and incorporating attack classes. The model also takes into account the possibility that the system can be in a compromised state when the next attack occurs. An empirical study of network attacks has been done by [26].

Based on publicly available documents and data we have formulated a simulation model to determine the expected survivability of systems and the average damage done to them under different conditions. The outcome

of an attack is determined by the attack type, the system configuration, the initial system state and the defense mechanism. The stages through which a system passes are shown in Figure 3. These are used to compute the system state transition matrix. The process is simulated as the system moves from an initial state to the final state, and this final state is used to compute the survivability. The cost is assumed to increase with the level of the defense mechanism and we determine the benefits in terms of the survivability of the system.

We now introduce a few notations that have been used to model survivability of the system.

Notations

- {J} Attack types; we consider 6 levels of attack types.
- i, j Index of attack type, i, j in {J}; i denotes the prior incident and j denotes the subsequent (or current) one.
- $P(j)$ Probability that an incident is of type j.
- $\tau(i, j)$ Inter-incident time between incidents i and j.
- a Arrival rate of incidents.
- {S} System states; we say that the system can be in any of six specified states, 1 being fully functional and 6 being non-functional.
- {B} Attack class; attacks can fall into two categories, 1 being an attack detected but not stopped and 2 being an undetected attack.
- {D} System design; we have considered only one design.
- {M} Defense mechanism; six layers of defense mechanism with the cost and strength of defense increasing with each layer.
- r, s Index of system state, r, s in {S}
- d Index of system design, d in {D}
- m Index of defense mechanism, m in {M}
- T Transition probability matrix with elements $p(r, s)$, where $p(r, s)$ is a function of i, j, d, m.
- b Index of attack class, b in {B}
- α Proportion of attacks that were detected but not stopped to attacks which went undetected.

5.1 Model Description

Conducting real world experiments to model network security incidents may be difficult, costly and in some cases, unethical. So we propose a simulation model that has three components and is run with data that were publicly available. Where no data were available, we made suitable assumptions and conducted sensitivity analysis. In actual applications, managers can use values of the parameters based on the data they have for their organizations to analyze the survivability of their system.

5.1.1 Modelling of the Attacks and Their Impacts

In order to forecast the attacks, we model the process as a marked, stochastic point process, where the attacks occur at random points in time. For each attack we have considered the attack type and attack class. Therefore the mark space will be two dimensional ({B x J}), characterized by severity of attack and attack class.

Most of the research work for testing IDS uses the DARPA/Lincoln Laboratory off-line evaluation data set. These data are from extensive experiments that were performed in 1998 and 1999 at DARPA ([22]). But this data set does not address attack simulation. We have used the data sets available in [26] and on the CERT/CC Website, to obtain the probability of occurrence and the inter attack distribution. These are presented in Table 1.

Table 1: Probabilistic data sets for network attacks

#	Attack type (j)	Prob(j)	distribution
1	Root break-in	0.51	Exp(111)
2	Account break-in	0.23	Exp(94)
3	Denial of services	0.02	Gamma(0.5,144)
4	Information Corruption	0.02	Gamma(0.5,152)
5	Access attempt	0.15	Exp(78)
6	Information Disclosure	0.07	Weibull(0.5,57)

The six attack types described in Table 1 are indicative of the type of attacks. First, a random number is generated and the attack type is determined from the cumulative probability distribution. The inter attack time is then generated from the appropriate distribution for that attack type. The technical detail to classify a particular attack is beyond the scope of this paper. We would like to take this up in our future research, as to how an attack can directly be classified. This is a bit difficult as response to any undetected attacks is reactive rather than proactive. This is an indicative study that uses simulation for the analysis, starting from the distribution computations of the available dataset.

In order to simulate the attack we first determine the type of attack and then determine whether it was detected or undetected at the IDS. For the purpose of analysis we consider the probability density function of the inter-attack times. When the process is Poisson, the probability density function of the inter-incident time (τ's) is given by

$f(t) = Pr(t \leq \tau \leq t + dt) = a * e^{-at}$,

where a is the rate of occurrence of attacks. The distribution function is given by $F(t) = 1 - e^{-at}$.

Some past evidence ([26]) suggest that the exponential model is a reasonable approximation for the arrivals of attacks. However, the possibility of other distributions cannot be ruled out. For example, a mixture of exponentials may provide a good fit. Further, epidemiological models may be more meaningful in cases where one has more information about the attack process, as in the case

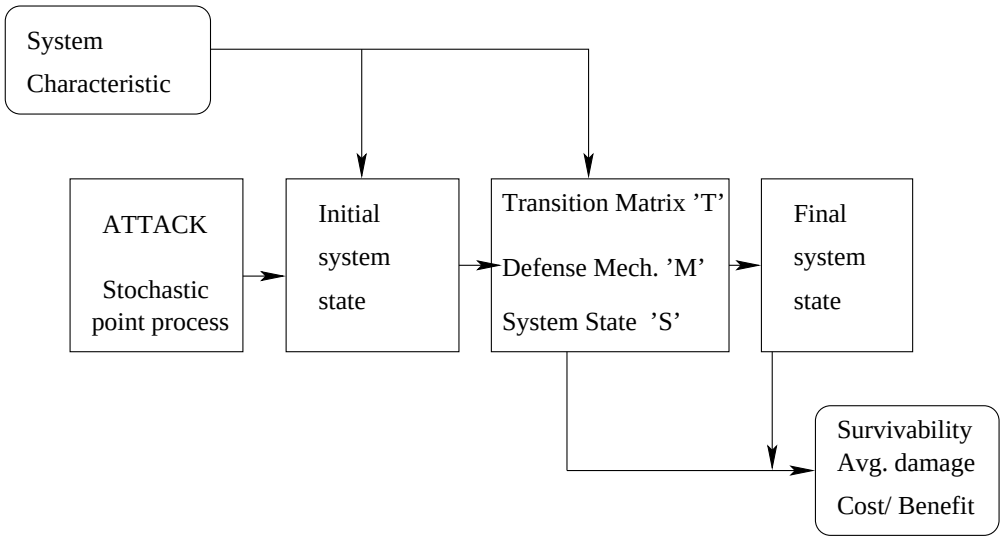

Figure 3: Simulation model for survivability

of viruses. When network attack data are available, a distribution that fits the data may be estimated and applied in the analysis.

5.1.2 The State Transition Process

The configuration of the system is a mix of the design D and the defense mechanism M. We have assumed six hypothetical levels of defense mechanisms with cost increasing with effectiveness. Since the different design data were unavailable we have assumed D as constant although it can be varied if data are available. The response model assumes state transitions to depend on the attack type and attack class, i.e $p(r,s) = p(r,s|j,d,m,b)$. The states of the transition matrix are ordered based on degree of compromise, that is, from $s = 1 \equiv$ normal (fully functional) to $s = \mathrm{S} \equiv$ non-functional. Given an attack, the system can never go to a better state and thus T is an upper triangular matrix.

$$T = \begin{pmatrix} p_{11} & p_{12} & p_{13} & p_{14} & p_{15} & p_{16} \\ 0 & p_{22} & p_{23} & p_{24} & p_{25} & p_{26} \\ 0 & 0 & p_{33} & p_{34} & p_{35} & p_{36} \\ 0 & 0 & 0 & p_{44} & p_{45} & p_{46} \\ 0 & 0 & 0 & 0 & p_{55} & p_{56} \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

The columns represent the subsequent system state denoted by s and the rows denote initial state denoted by r. Constraints are imposed on the elements of T, $\{p(r,s)\}$, in terms of the dependence on s, j, m and b. Also the system must end up in some state or the other. That is,

$$\sum_s p(r,s) = 1 \quad \forall r.$$

We do not know of any source which provide data on the state of the attacked system or the loss incurred by the system. Our model assumes a set of hypothetical states which can be mapped onto actual states as and when the data is available.

If we know the transition probabilities in each case, we can input that data directly into the model. Otherwise, there is a need to develop a model to generate the elements $\{p(r,s)\}$ of the transition probability matrix T, or compute them by considering the intermediate states of the system. In most cases the attack takes the system to some compromised state because it takes time for the system administrator to perform corrective action. Estimating these transition matrices is critical but extremely complex, since S^2 x J x D x M x B probabilities must be estimated. We have assumed some simplifying rules so that we could generate the transition probabilities in the absence of suitable data, using the different parametric equations given below. We utilize the known properties of the transition matrix to obtain the different equations. A more detailed discussion can be found in [25]. Some of the properties are: $s \geq r$; probability of degradation is lower if the incident is less severe; probability of degradation is lower if the defense is stronger; probability of degradation is lower if the incident is detected by the IDS but could not be stopped.

$p(r,s) = p(r,s,j,b,cost(m) : \pi_0, \chi_0, \pi_1, \chi_1, \pi_2, \chi_2)$,
where π and χ are parameters that are estimated as described below. There are two cases, $s = 1$ and $s > 1$.

$p(r,s) = \pi_2 * (1 - exp(-\pi_1[cost(m) - \pi_0]))$ for $s = 1$.
$p(r,s) = \chi_2 * exp(-\chi_1[cost(m) - \chi_0])$ for $s > 1$.

These are simple but commonly used functional forms that are concave and convex respectively, and so reflect decreasing returns with cost. The equations hold for b_1 ={system attack detected but not stopped at the IDS} and for b_2 = {system attack undetected at the IDS}. In the first case there is a chance that the system administrator takes some early action to stop the system from degrading further. The values $\{p(r,s)\}$ for the transition probability matrix is generated accordingly in each

of these cases, by assuming that this shifts the location parameter π_0 by 1% and χ_0 by 2%.

$\pi_0 = r * 0.01, \quad \chi_0 = r * 0.02$ for b_1

$\pi_0 = \chi_0 = 0$ for b_2.

π_1 and χ_1 are the critical shape parameters that determine the relationship of the transition probabilities to the $cost(m)$ of the defense mechanism. These values increase as r increases from 1 to 6, and influence how the survivability varies with cost.

The boundary conditions and heuristic knowledge is used to obtain the relationships among the parameters. For example keeping the defense level same, the probability of degradation is lower if the attack is less severe. Also for the same severity level, the probability of degradation is lower if the defense is stronger and so on. The values $p(r, s)$ for T is generated using the parameter values computed from the equations presented below.

$\pi_1 = \pi_4 * (\frac{r}{r+1})$

$\pi_2 = \pi_3 * j$

$\chi_1 = \chi_4 * (\frac{r}{r+1})$

$\chi_2 = \chi_3 * ((7 - s) - (0.2 * j))$,

where π_4 and χ_4 are the critical shape parameters for the base case, that is when system is fully functional. π_3 and χ_3 are the scale parameters. The values of these parameters and constants were calibrated to give reasonable values of transition probabilities subject to the properties of the system. If the subsequent attack occurs before recovery from the earlier attack, then the system may be in a compromised state. We have considered this by simulating recovery times from compromised states.

Survivability as defined earlier is the degree to which a system has been able to withstand an attack or attacks, and is still able to function at a certain level in its new state.

SURV = performance level at new state/normal performance level.

Another possible way of measuring survivability is:

$SURV(s) = \sum_k w(k) * \phi(s, k)$,

where $\phi(s, k)$ is the degree to which the compromised function/service k has survived in state s and $w(k)$ is the importance level of the function/service.

This assumes that a complete set of states S of the system has been defined, and the system administrator can assess $\phi(s, k)$ for each s and k. In view of the data requirements, it may be necessary to aggregate the state space S, and the different functionalities and services K. The states in {S} may be classified here as normal (fully functional), under attack, compromised, recovered, almost dead and dead (non functional). The model assumes that the system will be in one and only one of these states. Then $\phi(s, k)$ could be the average level to which function or service k survives in each of those states s. This is a flexible approach, and can be applied in many situations. We have not come across any data sources in this respect and have used the normalized measures for $\phi(s, k)$. Then SURV(s) will be between 0 and 1, where 0 means total failure and 1 means completely normal.

5.2 Simulation of the Attacks and their Impacts

The simulation consists of simulating attacks and simulating the transition of the attacked system to its final state. We assume the attacks to be independent. We further assume a range of cost scaled between 0 and 100 for various defence mechanisms. The state transition probabilities were generated as explained earlier and are assumed to be constant over time. That is, no learning mechanism is modelled.

5.3 Simulation Results and Analysis

Through this simulation we determine the survivability and the average damage of the system for different costs (representing different defense mechanisms). The stronger the defense mechanism, the more likely it is to withstand an attack, that is, to stay in its normal state, and less likely to end up in a compromised state. The simulation was carried out for different relationships between the cost of the defense mechanism and the state transition probabilities of the system. The simulation was run for 100 attacks at a time and repeated with different seed values. Of the total intrusions reaching the system, it was assumed that only 10% were detected but not stopped by the IDS. The benefit to the organization is the increased survivability of the system. This provides the manager with the basis to perform a cost-benefit analysis.

The 'Specified' case corresponds to the attack types as defined in Table 1. The 'Uniform' case assumes that each attack type has an equal probability of occurrence. So the relative probabilities of the attack types play an important role in determining the survivability of the system. Since the attack types are not equally harmful, we see a clear increase in the survivability from the Specified Case to the case of Uniform Attacks. Figure 4 plots the survivability for different values of cost.

$a_0 = 1.0$, $a' = 0.001$, $\pi_3 = 0.3$, $\pi_4 = 0.35$, $\chi_3 = 0.05$, $\chi_4 = 0.02$.

The average damage to the system decreases with increase in IDS effectiveness. An average IDS (80% intrusion detection) performs reasonably well and there is no significant improvement (only around 1%) over the best case (94% intrusion detection). The simulation was run for 100 attacks. We find there is a significant advantage (a gain of around 30%) to incorporate an IDS. The system damage is graphically presented in Figure 5.

The occurrence rate of attacks does not have any impact on expected survivability, because it is computed on a per attack basis. Sensitivity analysis shows that the survivability appears to be most sensitive to π_3 and χ_3. That is, the initial level of the transition probabilities is most important, rather than how they change with m. Detailed sensitivity analysis can be found in [26].

It can be seen that we now have a tool to undertake a cost/benefit analysis for determining the appropriate level of security for an organization. The cost/survivability

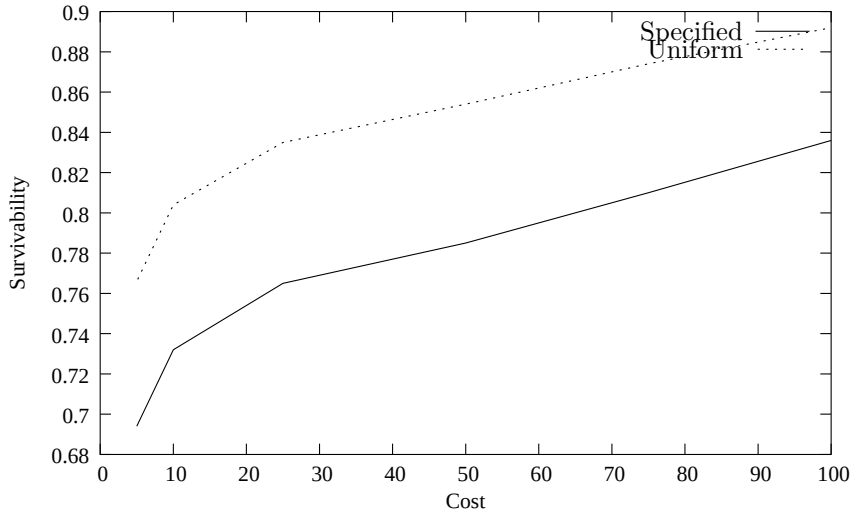

Figure 4: Survivability under different attack types vs cost

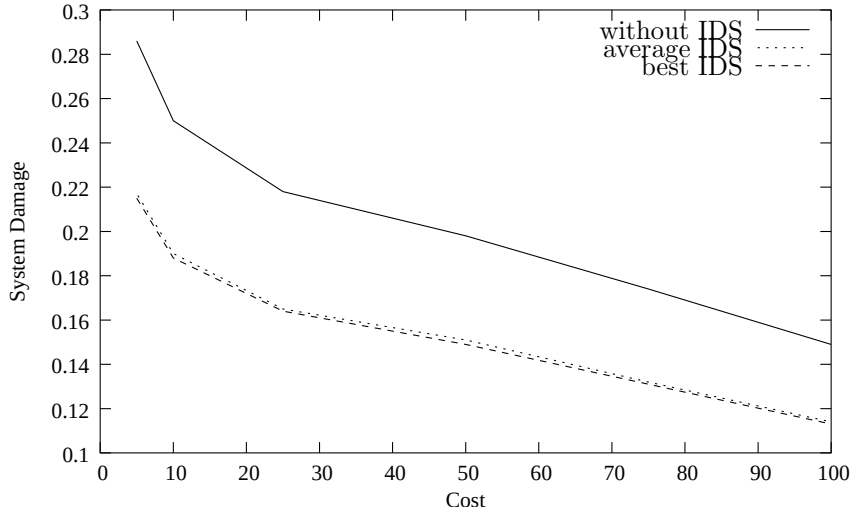

Figure 5: Average damage and effect of IDS

curve (or the cost/damage curve) indicates the trade-off involved between the cost of the defense system and the protection it provides. The right balance between the two will be determined by the preferences and priorities of an individual organization, and will necessarily vary from one organization to another. The advantage of this approach is that, given the specific organizational situation, management can arrive at a more informed decision about appropriate security measures.

6 Conclusions

In this paper we have discussed a network security system which consists of a detection mechanism and a protection mechanism. We have proposed a GA-based neural network model for the intrusion detection system. Neural networks have been quite effective in modelling an IDS. The main problem with a neural network based modelling system is that it uses a gradient-based training algorithm consuming a lot of processing time and power. In this paper we have shown that using genetic algorithm based learning for the neural network, we can get rid of the cumbursome gradient-based training method. An interesting feature of this method is that it does not use any predefined learning method but the learning is done through the data itself. So the learning process changes with the dataset, making the model more adaptive. The system response has been modelled probabilistically through a state transition matrix where the state transition probabilities are functions of the type of attack and the defense mechanism. We have outlined a set of reasonable constraints on the transition probabilities and developed a model to generate them in the absence of data. The model has been shown to be capable of supporting cost/benefit analysis and this should be of use to system managers and admin-

istrators in managing the security of their systems.

The major limitation of this study was the availability of cost figures and also the transition states reached by different types of attacks. The model we have developed is generic in nature and can be used as a template. One needs to populate the model with the private data that would be available within the concerned organisation. Also, testing and validation needs to be performed on the model and the inputs before it can actually be implemented.

Attack detection and classification are the two important decisions of intrusion detection systems. All the attack types cannot be detected and it is important to determine the survivability of the system in this uncertain environment. Our work demonstrates that it is possible to model attacks and determine the survivability as well as average damage of the network system. An extension of this work should include obtaining the relevant data and fine tuning the model for the different scenarios under consideration.

Acknowledgments

A significant portion of this work was done when the first two authors were pursuing the doctoral program at Indian Institute of Management Calcutta

References

[1] S. Axelsson, *Intrusion detection system: A survey and taxonomy*, Technical report, Chalmers Institute of Technology, Sweden, 2000.

[2] B. Balajinath and S. V. Raghavan, "Intrusion detection through learning behavior model," *Computer Communications*, vol. 24, no. 12, pp. 1202-1212, 2001.

[3] J. M. Bonifacio, A. M. Cansian, and A. C. Carvalho, "Neural networks applied to intrusion detection systems," *Proceedings of the International Conference on Computational Intelligence and Multimedia Application*, pp. 276-280, 1997.

[4] T. Bui and T. Sivasankaran, "Cost-effectiveness modelling for a decision support system in computer security," *Computers and Security*, vol. 6, no. 2, pp. 139–151, 1987.

[5] D. Chalmers, "The evolution of learning: An experiment in genetic connection," *Proceedings of 1990 Connectionist model Summer School*, pp. 81–90, Morgan Kaufmann, 1990.

[6] F. Cohen, "Information system attacks: A preliminary classification scheme," *Computers and Security*, vol. 16, pp. 29–46, 1997.

[7] F. Cohen, *A Mathematical Structure of Simple Deceptive Network Deceptions*, Fred Cohen and Associates, 1999.

[8] M. Crosbie and G. Spafford, "Applying genetic programming to intrusion detection," *Proceedings of AAAI 1995 Fall Symposium*, 1995.

[9] D. E. Denning, "An intrusion detection model," *IEEE Transactions on Software Engineering*, vol. 13, no. 2, pp. 222-232, 1987.

[10] R. J. Ellison, D. A. Fisher, R. C. Linger, H. F. Lipson, T. Longstaff, and M. R. Mead, *Survivable Networked System*, Technical report, Carnegie Mellon University, 1997.

[11] R. J. Ellison, D. A. Fisher, R. C. Linger, T. Longstaff, and M. R. Mead, *A Case Study in Survivable Network System Analysis*, Technical Report CMU/SEI-98-TR-014, Carnegie Mellon University, 1998.

[12] W. Fan, W. Lee, S. J. Stolfo, and M. Miller, "A multiple model cost-sensitive approach for intrusion detection," *Proceedings of the Eleventh European Conference on Machine Learning (ECML 2000)*, Barcelona, Spain, May 2000.

[13] D. A. Fisher and H. F. Lipson, "Emergent algorithms: A new method for enhancing survivability in unbounded systems," *Proceedings of 32nd Hawaii Conference on System Sciences*, 1999.

[14] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, 1989.

[15] J. H. Holland, *Adaptation in Natural and Artificial Systems*, Ann-Arbor: University of Michigan Press, 1975.

[16] J. Howard, *Analysis Of Security Incidents On The Internet*, PhD thesis, Softwarre Engineering institute, Carnegie-Mellon University, 1995.

[17] C. Iheagwara, A. Blyth, T. Kevin, and D. Kinn, "Cost effective management frameworks: the impact of ids deployement technique on threat mitigation," *Information and Software Technology*, vol. 46, pp. 651–664, 2004.

[18] S. Jha and J. M. Wing, "Survivability analysis of networked systems," *Proceedings of the 23rd International Conference on Software Engineering*, July 2001.

[19] J. R. Koza, *Genetic Programming: On Programming of Computers by means of Natural Selection*, The MIT Press, 1992.

[20] W. Lee, S. J. Stolfo, and K. Mok, "A datamining framework for building intrusion detection model," *IEEE Symposium on Security and Privacy*, pp. 120–132, 1999.

[21] R. Lippmann and R. K. Cunningham, "Improving intrusion detection performance using keyword selection and neural networks," *Computer Networks*, vol. 34, pp. 597–603, 2000.

[22] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, "The 1999 darpa offline intrusion detection evaluation," *Computer Networks*, vol. 34, pp. 579–595, 2000.

[23] T. F. Lunt, "A survey of intrusion detection techniques," *Computer and Security*, vol. 12, no. 4, pp. 405-418, 1993.

[24] N. R. Mead, R. J. Ellison, H. F. Linger, T. Longstaff, and J. McHugh, *Survivable Network Analysis Method*, Technical Report CMU/SEI-2000-TR-013, Carnegie Mellon University, 2000.

[25] S. D. Moitra and S. L. Konda, *The Simulation Model for Managing Survivability of Networked Information Systems*, Technical report, Carnegie-Mellon University, 2000.

[26] S. D. Moitra and S. L. Konda, "An empirical investigation of network attacks on computer systems," *Computers and Security*, vol. 23, no. 1, pp. 43–51, 2004.

[27] T. Verwoerd and R. Hunt, "Intrusion detection techniques and approaches," *Computer Communications*, vol. 25, no. 15, pp. 1356-1365, 2002.

[28] C. Wang and J. C. Knight, *Towards survivable intrusion detection*, Technical report, Carnegie Mellon University, 2000. (http://www.cert.org/research/isw/isw2000/papers/38.pdf)

[29] R. L. Williams and R. Sharda, "Bankruptcy prediction using neural networks," *Decision Support Systems*, vol. 11, pp. 545-557, 1994.

[30] O. C. Yue, "Cyber security," *Technology in Society*, vol. 25, no. 4, pp. 565-569, 2003.

[31] D. Zhu, G. Premkumar, X. Zhang, and C.-H. Chu. "Data mining for intrusion detection: A comparison of alternative methods," *Decision Sciences*, vol. 32, no. 4, pp. 635-660, 2001.

Tarun Bhaskar is working with GE Global Research in Bangalore, India. He received his PhD from the Indian Institute of Management Calcutta. His thesis was in the area of Project Scheduling. He obtained his Bachelor Honours in Mechanical Engineering from Bihar University, India. His main research interests are Project Scheduling under Uncertainty, Robust Optimization, Soft Computing Techniques, Network Security and Combinatorial Optimization Problems.

Narasimha Kamath B earned his PhD from IIM Calcutta and his thesis was in the area of supply chain management. Currently he is working for i2 Technologies and is responsible for coming up with good planning solutions for its clients using the Supply Chain Planner product. He obtained his Bachelors in Mechanical from Regional Engineering College, Surathkal, India. He has worked for GE Medical Systems as lead system designer. Business process modelling, supply chain management and decision making are his current research interests.

Soumyo Darshan Moitra has a B.Sc. Honours from the University of Sussex, an M.A. from Cornell University, an M.S. from Syracuse University and a Ph.D. from Carnegie-Mellon University. He was an Assistant Professor at Baruch College, City University of New York and then a Member of Technical Staff at Bell Communications Research. Since 1995, he has been a Professor of Operations Management at the Indian Institute of Management, Calcutta. He has been a Humboldt Fellow at the Max Planck Institute, Freiburg, Germany, a Visiting Professor at NTT Laboratories, Japan, and a Visiting Scientist at the Software Engineering Institute, Carnegie-Mellon University.