An Application of Clonal Selection Process of an Artificial Immune System to Implementing Intruder Detection System

JungWon Kim[a], Jong Uk Choi[b], and Hwa Soo Kim[c]

[a]University College of London
Tel: +44- 171-380-7329, Fax: +44-171-387-1397, Email: J.Kim@cs.ucl.ac.uk

[b]School of Information & Telecommunication , Sangmyung University
7 Hongji-dong, Chongro-gu, Seoul, Korea 110-743
Tel: +82-2-2262-5222, Fax: +82-2-2262-5333, E-mail: juchoi@markany.com, juchoi@pine.sangmyung.ac.kr

[c]National Defense University Seoul Korea
Tel:+82-2-300-2141, E-mail: hskim@kndu.ac.kr

ABSTRACT

This research aims to unravel the significant features of the human immune system, which would be successfully employed for a novel network intrusion detection model. Several salient features of the human immune system, which detects intruding pathogens, are carefully studied and the possibility and the advantages of adopting these features for network intrusion detection are reviewed and assessed.

Keyword: IDS, Immune system

Introduction

An intrusion detection system (IDS) is an automated system for the detection of computer system intrusions using audit trails provided by operating systems or network monitoring tools. The main goal of an IDS is to detect unauthorised use, misuse and abuse of computer systems by both system insiders and external intruders. The research on IDSs started from Anderson's work (Anderson, 1980) which aimed to improve the auditing facilities and the surveillance abilities of computing systems. In his work, he defined and classified threats as follows (Lunt, 1988), (Lunt, 1993):

- *External penetrators* (who are not authorized to use the computer)
- *Internal penetrators* (who are authorized to use the computer but not the data, program, or resource accessed), including
- *Masqueraders* (who operate under another users id and password)
- *Clandestine users* (who evade auditing and access controls)
- *Misfeasors* (who authorized to use the computer and resources accessed but misuse their privileges)

Anderson proposed a new idea to detect each of these intruder types using audit trails. Firstly, he claimed that abnormal frequencies of failed login attempts might imply the presence of external penetrators. Secondly, abnormal frequencies of failed access attempts to files, programs, and other resources would indicate the presence of internal penetrators. Thirdly, if currently observed users behaviour is significantly deviated from the users' profiled normal behaviour, it could be interpreted as the penetration of masqueraders. In addition, clandestine users can breach the auditing process and the access control of an operating system. Clandestine users typically use a system privilege or operate at a level below which auditing is being performed. When clandestine users use a system privilege, it would be detected by investigating all use of functions which cripple auditing, change the user identity numbers of audited users, or change other auditing parameters. Alternatively, clandestine users operating at a lower level than an auditing level could be detected by auditing service or kernel calls. Finally, the detection of misfeasors requires monitoring particularly the abnormal accesses and usage patterns of legitimate users who have access to vulnerable resources.

Denning's Generic Intrusion Detection Model

Dorothy Denning is the pioneer of intrusion detection research and she propounded the first generic intrusion

detection model in 1987(Denning, 1987), which is independent of any particular system, application environment, and system vulnerability or intrusion type. Since her proposal of generic intrusion detection model, various types of intrusion detection systems(IDS) has been developed. Most of these systems follow the basic notion of her generic IDS, which is a real-time expert system whose knowledge is derived from statistical inference based on the audit trails of users or system resources.. This model was employed for building a number of intrusion detection systems (Ilgun et al., 95), (Jackson et al., 94), (Lunt, 88), (Lipins and Vaccaro, 89), (Lunt et al., 92), (Mykerjee et al., 94). Denning inspired many researchers to build automated intrusion detection systems by providing the first general framework based on Anderson's rudimentary notion (Anderson, 1980). These systems proved the validity of the generic model.

The fundamental structure of this model stems from a rule-based expert system. The main units of a rule-based expert system are a knowledge base, which is a collection of knowledge represented in the form of rules, a working memory, which is a global database of facts used by the rules and an inference engine, which makes inferences by selecting rules that match the available facts. In Denning intrusion detection model, a working memory stores profile facts describing the normal behaviour of subjects with respect to objects and provide the signatures of abnormal behaviours. A statistical metric and model are used to present profiles. A subject can be an individual system user, a group of system users or a system itself, while objects can be files, programs, messages, records, terminals etc. When a subject acts upon a specific object, it generates an event, which alters the statistical metric state of both subject and object. A knowledge base contains activity rules to be fired for updating profiles, detecting abnormal behaviour, and producing reports. An inference engine makes its inference by triggering rules matching profile facts.

Genetic Algorithm Approach

The validity of Genetic Algorithms were tested by Me (Me, a), (Me, b) as a complement to a model-based approach. The genetic algorithm in his experiment was used as a tool to search efficiently for the evidences supporting the intrusion models of a model-based misuse detection system. Genetic algorithm aims to find events, which have the greatest risk to a system amongst the subsets of all the possible intrusion models. The selected subset, which is considered as the greatest risk to a system, can be interpreted as ones, which are the most likely to occur. The system needs not to be committed to match all the observed events to the corresponding events in all the intrusion models of a knowledge base. Rather the system is expected only to pinpoint the events related with the selected models by a genetic algorithm. This allows a system to save a large amount of computation to collect evidences for a model-based misuse detection system.

The genetic algorithm starts its evolution by setting up a population, which is initialized by a set of randomly selected possible solutions. Each member is evolved through crossover and mutation and evaluated its fitness to an optimal solution by a fitness function. Me sets up the population of initial solutions as a N-dimensional hypothesis vector, H, which maximizes the product W * H. Here, N is a number of possible intrusion types. The element of H, Hi, is turned on by 1 if the intrusion i is presented and turned off by 0 otherwise. W is an N-dimensional weight vector, which is proportional to the risk with each intrusion. In order to make the hypothesis realistic, a constraint is added; the number of events for a particular intrusion which is observed from a audit trail should be greater than or equal to the number of events which are required to cause that the specific intrusion. The finally gained solution through the evolution of a genetic algorithm represents the one set which collects different types of intrusion models and these models are deemed as they can subvert the system as much as the greatest risk.

It was believed that a genetic algorithm would be an advantageous component for an intrusion detection system through the above experiment. Regardless of the satisfactory test results, several questions are raised. First one is the definition of a set showing the greatest risk. This set is defined simply by considering the predefined extent to risk and the absence and the presence of each intrusion model. This definition totally disregards the correlation of different intrusion models. A more sophisticated fitness function, which should consider a number of intrusion models, collectively is ignored. Second question is the definition of constraints. The order of events of an intrusion model is disregarded when the constraint is defined. The intrusion is not taken place merely by the collection of required events. It is caused as a result of the sequential feedback from the required events.

Genetic Programming Approach

Another attempt to use an evolutionary algorithm has been presented by (Crosbie and Spafford, 95b). They proposed autonomous multi-agent systems for network intrusion detection (). In this proposal, each agent is required to be adaptive to dynamically changing network environments. As one approach to implementing adaptive agents, Crosbie and Spafford used a genetic programming (GP). In this work, a number of GP agents evolve by learning the normal and abnormal building blocks from prepared training scenarios, which are mixture of both intrusive and non-intrusive activities. This GP agent evolves as batch style learning and the evolved best agent is selected for a stand-alone IDS.

The agent implemented using GP consists of a set of operators (which are logical, arithmetic, and conditional) and a set of primitives. The primitives are extracted event from monitored network traffic such as total number of socket connections, average time between connections, minimum time between connections, maximum time between socket connection, destination port to which the connection is intended and source port from which the connection originated. Combining these primitives and operators in any way as a parse tree form generates a single agent. The fitness value of generated agent is evaluated at each evolution step based on its performance. To evaluate the performance, training data were prepared by simulating four different types of intrusions on network traffic data and assigning the probability of each intrusion. This predefined probability indicates the degree of an intrusion. The absolute differences between the agents reported suspicious degree and the training scenarios assigned probability determines agents fitness value.

This work emphasised the autonomous adaptability of GP and it showed good adaptability. The evolved prediction rules have only selected events from presented initial events. It showed the possibility of using GP to perform feature selection process. However, this work has a critical problem, which requires extensive and deep training scenarios. This means it has to simulate all intrusions to collect right training data. From practical point of view, it is very difficult to simulate all intrusions. Moreover, it requires assigning the probability of each intrusion and it is not easy to evaluate the degree of risk.

Computer Immunology Approach

While most of the above intrusion detection systems have developed based on the rigorous research in the computer security community, a novel approach was proposed by the researchers in the artificial intelligence community. They have investigated how a natural immune system works well for protecting humans from dangerous foreign pathogens such as bacteria, viruses, and parasites and modelled a computational immune system to protect a computer system from computer viruses and intrusions (Somayaji et al., 97), (Kephart, 94), (Forrest et al., 97), (Forrest et al., 94). This idea started from that several researchers view computer viruses as a form of artificial life (Spafford, 94).

One of them, Forrest (Forrest et al., 97) has attempted to build a computational immune model for the intrusion of a computer system. She regarded the intrusion detection problem as distinguishing "self" from "other" which is the cornerstone of a natural immune system. This view can be said to be very similar to the view of anomaly detection. She suggested two approaches to implement this basic notion. In the first one (Forrest et al., 96), she profiles the normal behaviours for privileged system processes and compares them to currently observed behaviours. This idea is not much different from a typical anomaly detection idea. The only difference of her experiment is that she profiled the system calls of privileged system processes rather than user commands or system usage. She argued the profiles of system calls resulted in a massive reduction of audit data. In the second approach (Forrest et al., 97)(Forrest et al., 94), she follows the protecting mechanism of a natural immune system more closely. Like the first idea, a normal behaviour of a user or a system, self is profiled. A system generates a set of detectors, which fail to match some normal behaviour. While a system is executing, if a detector is activated, it is implied there is a change on behaviour and this behaviour is deemed as an abnormal behaviour. In (Forrest et al., 94), this idea was tested for detecting computer viruses.

In (Forrest et al., 96), the validity of the first idea of computer immunology was assessed using several different tests. They selected sendmail as the privileged process to be profiled. In order to profile the system calls of sendmail's normal executions, they retrieved the system calls which belonged to a k+1 size window and recorded chronologically the system calls within a window. By sliding a window along the trace of system calls, the various combinations of system calls could be profiled. When the patterns of sendmail's normal executions were profiled, special efforts were made to maintain the generality of the profile. Therefore they artificially created 112 different types of messages which would be sent by sendmail. For various operating systems they reprofiled the system calls of sendmail process for each of 3 different window sizes. The results obtained were combined into a single built profile which was then subjected to two tests in order to confirm its validity. In the first test, each of the normal system calls of sendmail were compared to the normal system calls of other processes and approximately showed 5-32% mismatching. Forrest, Hofmeyr and Longstaff interpreted this result as the only information of the order of system calls can distinguish the behaviour of different processes. In the second test, Forrest, Hofmeyr and Longstaff simulated three different types of system process behaviour. First, successful intrusions using sendmail vulnerability, second, failed intrusion attempts using sendmail vulnerability and third, error conditions. Although discrepancies among the mismatching rates with sendmail normal system calls depended on specific type of abnormal behaviour as expected, the test results did not show this expectation. The mismatching rates of the first and the second types of process behaviour showed at average over 2%. However, this rate is similar to the mismatching rates of the third type of process behaviour, 2.3% and 2.7%. Thus, normal system call profiles did not show any clue to distinguish error conditions from intrusions.

This first computer immunology approach does not show any distinguishing feature of human immune systems such as distributed anomaly detection. Instead, it tried to show the concept, self and non-self on a single host and this is not much different from the conventional anomaly detection mechanism. The interesting point of this work is that it showed the system call sequences of privileged process could be a good indicator to describe self of a monitored host. Even though it is not the first attempt, it proves again this audit level will be quite promising in terms of building a light-weight IDS. However, it needs to be tested on more extensive intrusions in order to prove whether this audit level is good enough to replace other heavy-weight audit level such as auditing all application programs and auditing all user commands typed.

The Clonal Selection of an Artificial Immune System

Even though new antibodies surviving negative selection are assured to be self-tolerant, their efficacy to detect antigens is unknown when they are released from the bone marrow and the thymus. This is because new antibodies are randomly generated and they are verified only not to be self. They might hold non-self patterns but not antigen patterns. In order to exclude these ineffectual detectors, the human immune system adopts the evolution of antibodies towards the existing antigen patterns (Tizard, 1995). During this evolution process, the human immune system uses its own unique niching strategy to maintain generality and diversity of antibodies as one part of clonal selection process (Forrest et al, 1993).

[Clonal Selection of the Human Immune System]

B-cells and T-cells, which are antibody secreting cells, employ various gene shuffling mechanisms and somatic mutation in their development stage as their strategy for maintaining the diversity of antibodies. Besides these mechanisms, human immune systems adopt different strategies. As one species evolves via natural selection, human immune systems also evolve through a process called clonal selection. The genes, which determine the specificity of antibodies, continuously evolve toward having the capacity to detect more prevalent pathogens at any moment and reproduce new lymphocytes with high affinity for those specific pathogens (Playfair, 1996), (Roitt and Brostoff, 1998), (Tizard, 1995) .

When B-cells are developed in bone marrow, only a limited energy source is given to them. They are active immediately after release from bone marrow, but they are rapidly exhausted and die. However, they do not simply disappear if they are activated by binding to specific antigens. When B-cells are directly or indirectly activated by antigens, they are divided and differentiated into a number of clones of antibody secreting cells, plasma cells, before they are exhausted. Plasma cells have the same antigen-binding properties as the receptors of parent B-cells or they can have mutated antigen-binding properties. As B-cells bind more to specific antigens, they have more chances to be selected for cloning. Similarly, as they bind less to specific antigens, they have fewer chances to be selected to clone and eventually tend to decrease. According to the different population of existing specific antigens, only the fittest antibodies survive.

Furthermore, when B-cells are activated by antigens, they produce memory cells for the reoccurrence of same antigens. The life of B-cell and its clone, the plasma cell is relatively short. However, some of the B-cell clones survive as memory cells. Some of the memory cells are exposed antigens and differentiated into plasma cells without undergoing somatic mutation and other memory cells undergo somatic mutation to be differentiated into plasma cells. Particularly the plasma cells generated without somatic mutation allow the secondary response of immune systems. When new pathogens are detected by B-cells, they generally take some time. We call this primary response. Compared to the primary response, the secondary responses by memory cells are very fast and efficient. Moreover, the memory cell provides an associate memory property (Dhaeseler, 1997). The new antigens have a structure that is not the same but is similar to the structure of previously detected antigens, and can be detected by memory cells. This is because the binding of antibody and antigen is approximate. For example, when a body is infected by cowpox, the immune system takes time to detect and eliminate it. But if somebody is infected by smallpox after being infected by cowpox, he/she is rapidly cured by the secondary response of immune system. This is because cowpox and smallpox are similar enough to induce secondary response by memory cells.

The Clonal Selection Algorithm

As discussed in the previous section, the clonal selection of a human immune system shows several significant mechanisms to maintain diversity and generality of antibodies. Among those, for the artificial immune system, we focus only on its natural selection mechanism in order to supplement for the efficiency drawback of negative selection algorithm. By using the evolution process of clonal selection, the computational time of negative selection due to random generation can be reduced. In addition, the problem of tuning the appropriate number of detectors may be solved by multimodal convergence feature of a niching strategy.

Forrest et al (1993) presented the niching strategy of their artificial immune system which follows the analogy of the human immune systems. They explored whether it is able to i) detect common patterns of randomly presented antigens and ii) to discern and maintain the diverse antigen population. In their model, they created

one population of antibodies and one population of antigens randomly. They used the GA to evolve the antibody population under a constant antigen population. This algorithm was devised by observing the following unique features of the human immune systems:

1. From a specific antibody's point of view, antigens are usually encountered sequentially.
2. From an specific antigen's point of view, it responds with only a subset of antibody secreting cells.
3. There is competition among antibody secreting cells which bind given antigens.
4. The antibody secreting cells are evolved by somatic mutation.

Conforming to the niching strategy of the human immune system which is brought by the clonal selection, for each generation, their modified GA selects an arbitrary size of random sample from the antibody population and a single random antigen from the antigen population. After each antibody in the sample is matched against a selected antigen, the fitness score of only one antibody showing the highest match score is increased while the fitness scores of the others remain the same. In summary, the observed properties of the human immune systems are turned into the following clonal selection algorithm:

1. A single antigen is randomly chosen.
2. A fixed size of the antibody population sample is chosen at random.
3. Each antibody in the sample is compared with the randomly selected antigen.
4. The antibody in the sample which showed the highest match score has added the match score to its fitness value. The fitness values of other antibodies remain the same.
5. This process is repeated for many antigens.

Using this algorithm, Forrest et al (1993) showed antibodies evolved to be generalists that match to most antigens to some extent. Their analysis of this result showed that antibodies evolved towards finding common schema that is shared among many antigens. Through the various experiments, they observed that this algorithm could sustain multiple inconsistent antibody patterns, which appear as the multiple peaks at a search space, and the similarity among antigens does not affect this capability. Moreover, they compared this niching strategy of the artificial immune system with the fitness sharing algorithm (Smith, Forrest, and Perelson, 1993). From this comparison, they reported that as the result of antibody sampling mechanism, the niching strategy of the artificial immune system controls its generality via the antibody sample size. To be more precise, when the sample size decreases, the selective pressures are moved towards generating a population of more general antibodies.

Even though the negative selection algorithm provides several strengths for network intrusion detection, it is necessary to resolve the excessive computational time caused from the random generation approach. Dhaeseleer (1997) introduced more efficient detector generation algorithms: a linear-time algorithm and a greedy algorithm. The basic idea is to provide an efficient method to enumerate all candidate detectors and thus allowing the negative selection algorithm to select valid detectors from this complete candidate detector set. However, this algorithm can be used only for a binary immune system using a simple r-continuous-bits matching rule. This is because they enumerate all possible valid detectors by counting the recurrence of all the potential r-continuous-bit binary strings unmatching self strings. Dhaeseleer also suggested the use of a non-binary alphabet immune system as an important future investigation because it is more natural in many cases. Furthermore, as analysed in the previous chapter, even when this formula can be applied (the case which uses a binary encoding and r-continuous matching function), it turned out infeasible when it is applied on real network traffic data due to its excessive quantity.

As the result, instead of using one of these algorithms, the negative selection algorithm for network intrusion detection introduced in this work should be replace by the niching strategy of Smith, Forrest, and Perelson's (1993) artificial immune system to build a valid detector set.

The Clonal Selection Algorithm for Network Intrusion Detection

In the first phase, the clonal selection algorithm build self/non-self profiles. In this research, the raw network traffic packets were gathered and these packets were parsed and built into self/non-self profiles according to its connection label.

These profiles are equipped with previously identified fields, which can distinguish normal and abnormal network activities. Then, the profiles are encoded in an appropriate data representation. In the second phase, when all the self/non-self profiles are encoded, the clonal selection algorithm starts generating detectors. We slightly modify Forrest et al's clonal selection algorithm() by adding the negative selection as one operator to it. The second phase of this algorithm for generating a detector set can be summarized as follows:
For each connection profile and its corresponding detector set:

1. D detector patterns are generated at random and their fitness values are initialised with zeroes.
2. A sample of N detector patterns is randomly selected from the generated D detector patterns.

3. A single intrusion pattern is randomly selected from the non-self profile.
4. Each detector in the sample is compared to all the self patterns from the self profile and the degree of similarity is measured. If any detector matches any self pattern completely, this detector is replaced by a new detector with ranom genes.
5. Each detector in the sample is compared to the selected intrusion pattern and the degree of similarity is measured.
6. The fitness value of the single detector in the sample that shows the largest similarity is increased. The fitness values of other detectors remain the same.
7. The processes 2-5 are repeated (for typically three times the number of antibodies (Smith, Forrest, and Perelson, 1993)).
8. P_b % detector patterns are selected as parents and genetic operators such as crossover, mutation are applied to generate new detectors.
9. P_w % detector patterns are deleted to make space for children.
10. A new detector population is created by including the selected parent detectors and the offspring detectors generated in 8
11. Processes 2 - 9 are repeated until the fitness values cease to change.

After finishing the second phase by performing above, the clonal selection algorithm builds new self profiles by parsing newly captured network packets. In the third phase, the detector patterns in each detector set are compared to the patterns in each corresponding new self profile. If any detector pattern matches the new self pattern, the algorithm generates an alarm signal.

As seen in section 4.2, this niching strategy controls the generality of each detector according to a detector sample size. For practical reasons, we expect this algorithm to create more general detectors so that each detector can match more than one intrusion. This means that even though each detector cannot bind to one intrusion exactly, it can match a number of intrusions to some degree. This approach is more likely to be suitable for network intrusion detection. This is because, as we can clearly see in the next section, the length of each self chromosome used in this work and the search space which these self chromosomes form is much larger and complex than the search spaces handled in most of work using a simple negative selection algorithm (Dhaeseleer, 1997). Furthermore, we expect the computation time of the clonal selection to be less due to using evolution rather than random search. Finally, the appropriate number of detectors will also be naturally determined based on the multimodal convergence of evolution process.

Network Traffic Data for Clonal Selection

When the negative selection algorithm was developed, the first data set which was described in section 3.3. Even though this data allows us to test the first prototype of the system, the amount of data was not enough to describe the connection-oriented network behaviour. This is because it was collected only for a very short period (which is approximately 15 minutes.) This led us to find a better quantity of data set. The second data set was provided by the DARPA Intrusion Detection Evaluation Program (http://www.ll.mit.edu/IST/ideval /index.html). This program was set up and managed by MIT Lincoln lab to survey and evaluate the state of art in intrusion detection research. It gathered about 4 gigabytes of compressed network packets of 7 weeks and simulated a wide range of various network intrusions. These intrusions are categorised into four groups: Denial-of-service intrusions, unauthorised access from a remote machine, unauthorised access to local superuser privileges by a local unprivileged user and surveillance and probing attacks. In other words, for each intrusion class, a number of different intrusions but using a similar attack scenario are simulated. Compared to the first data set, this data set provides more extensive types of intrusions.

The data from these two data sets originally had the fields of network packet capturing tools format such as time stamp, source IP address, source port, destination IP address, destination port and etc. However, the primitive fields of captured network packets are not enough to build a meaningful profile. Consequently, it is essential to build a data-profiling program to extract more meaningful fields, which can distinguish normal and abnormal. Many researchers have identified the security holes of TCP protocols (Porras and Valdes, 1998) and so the fields used by our profiles are selected based on the extensive study of this research. They are usually defined to describe the activities of each single connection.

The automated profile program was developed to extract the connection level information from TCP raw packets and it was used to elicit the meaningful fields of the first data set. For the second data set, Lee(1999) provided the pre-processed data set which extract the meaningful fields from DARPAs original data.

For each TCP connection, the following fields are extracted for both two data sets:

- Connection identifier: each connection is defined by four fields, initiator address, initiator port, receiver address and receiver port. Thus, these four fields are included in the profile first in order to identify each connection.
- Known port vulnerabilities: many network intrusions attack using various types of port

vulnerabilities. There are fields to indicate whether an initiator port or a receiver port potentially holds these known vulnerabilities.

- 3-way handshaking: TCP protocol uses 3-way handshaking for a reliable communication. When some network intrusions attack, they often violate the 3-way handshaking rule. Thus, there are fields to check the occurrences of 3-way handshaking errors.
- Traffic intensity: network activities can be observed by measuring the intensity over one connection. For example, number of packets and number of kilobytes for one specific connection can describe the normal network activity of that connection.
- Connection Content: these fields describe the activities of network connection user. One example is the number of failed logins or whether a su command is attempted and succeeded etc.

Thus, in total, network profile fields have 35 different ones for the first data set and 41 different fields for the second data set. While the first set includes only the first four types of fields, the second set adds the connection content type of fields. The details of these fields are introduced in Appendix.

Implementation

This section describes the detailed implementation of the clonal selection algorithm that is proposed in this work. This stage is developed in order to build an initial gene library. It introduces the genotype and phenotype representation, the genetic operators and finally the fitness functions which are based on the similarity between a detector pattern and a self pattern.

Genotypes and Phenotypes

The clonal selection algorithm employs the evolution mechanism of the classification rules, which classify non-self from self. One of natural ways of expressing classification rules is as a set of disjunctive normal form (DNF) rules. The if-part of each rule is a conjunction of one or more conditions to be tested and the then side of the rule describes the class label assigned to the rule. In the context of this research, the generated single detector will have a conjunctive rule as its phenotype. Therefore, the non-self patterns that can be detected by generated detectors will be a disjunction of these conjunctive rules.

There is some recent work which employed especially GP for the evolution of classification rules including our previous work financial fraud detection (Bentley, 2000a), (Bentley, 2000b). It takes advantage of the flexibility of GP that can adopt various functions sets, such as negation, larger-than, etc. This flexibility allows its phenotypes, classification rules, to express more complex conditions. However, this research trades the strong expressive power of GP for the simplicity of standard GA that can use bit string representation as its genotype. This is because one significant aspect of this research is the analysis of the pitfalls that the previously proposed AIS algorithms might have. In order to do so, we follow the closest genotype representations to theirs, but practically powerful and economic enough to express given phenotypes. The genotype and the phenotype representations which are used in this work is not claimed as the best way to do so. However, it is suitable for the purpose of this research.

Genotype Encoding and Phenotype Interpretation

Genotypes consist of 41 genes where each gene represents each feature of a detector. As described in section 4.3, the profile built for this work has 41 features and this number determines the total number of corresponding genes in the detectors. Each gene comprises existing feature values. For instance, in the case of Protocol_type feature, its valid feature values are tcp, udp and imcp. Thus, this feature has only three possible values and leads the corresponding gene to have only three nucleotides. As seen in figure 1, each nucleotide is a binary bit whose value one represents to include the corresponding feature value in the condition part of a classification rule and whose value zero indicates vice versa. This kind of genotype representation allows a single feature of each detector rule to have more than one value, which can be combined by OR operator by assigning a separate bit to each existing feature value. In addition, the valid genes of each detector rule are combined by AND operator. Thus, the genotype in figure 1 can be interpreted as "IF (field1 belongs to ([6..10] or [41..50]) and field2 belongs to ([1..2] or [5..9]) and .... and field5 belongs to [53..66])THEN it is intrusion". It should be noted that in the case that all the existing nucleotides of a single gene have identical values, which would be all ones or all zeroes. It will result in the exclusion of this feature from the condition part of a given classification rule. This is because both cases (the former means any value of this gene and the latter implies none of valid gene values) can be interpreted that any value of this gene is irrelevant to the class decision. Furthermore, for the case that all the

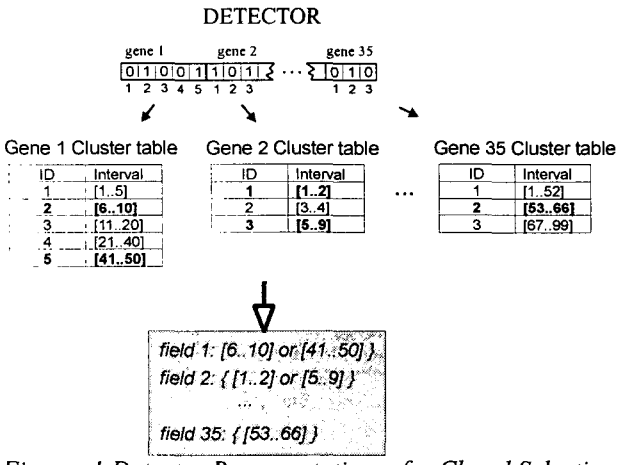

Figure -1 Detector Representation for Clonal Selection

This kind of genotype representation is not novel. This encoding scheme has been very popular and its descriptive power has been proved to be sufficient enough to handle practical rule evolution problem. It was proposed by De Jong(De Jong et al., 1993) for this work to use GA for concept learning, which also aims the evolution of attribute-based classification rules. This approach can be compared to another classic genotype encoding schemes for the same domain As being mentioned GP as an alternative approach before, there are some other approached to express classification rules for their evolution. For instance, real-value encoding, fuzzy rule encoding, etc. can be more powerful and flexible encoding scheme. However, for the same reason described earlier, the direct comparison of genotype encoding scheme, which is employed in this research, to these method is not conducted. We narrow our attention to the binary encoding methods that are used in Michigan approach for the classification rule evolution.

This different method is that assigning an appropriate number of bits which sufficiently cover the valid range of gene values (Goldberg, 1989). For instance, if the feature 1 has three values and it requires two bits, which is large enough to indicate three different cases, 01, 10, 11. For this method, the required number of bits to encode each gene is 2** when ** is a number of existing gene values. The advantage of De Jongs genotype encoding method over this method is that it allows a disjunction of feature values for each feature in a single detector rule while the other method represents only one valid value for each feature per detector rule. By doing so, one single detector rule has more expressive power and it becomes more general, which means that it can match more intrusion patterns. As a consequence, the number of detector rules, which is required to detect a given number of intrusion patterns, will become smaller. As being discussed in chapter 3, the lightweight detectors are required for the network intrusion detection context and this strength of De Jongs method certainly advocates the lightweight feature of generated detector.

This genotype representation is a popular approach to the evolution of classification rules, but the somewhat different method that was used by other AIS algorithms for concept pattern recognition. Forrest et al (1994) and Potter (1997) developed the AIS for concept pattern learning and they used the different genotype encoding method which is introduced above. Even though they used a rather restricted genotype description, they managed to endow the lightweight feature to generated detectors by using a match threshold concept, which looks closer to the approximated binding procedure of human immune systems. The brief description of this method is that each antibody and antigen are represented by a fixed length of binary strings and when a predefined number of bits are complementary, these two string are considered to match each other. A given single antibody string can match more than one antigen string as long as the number of their complementary bits is above a specified threshold. In this case, a matching threshold value affects the degree of antibody generality, but it is difficult to determine its appropriate value. To overcome this problem, Potter lets this value to evolve together when an antibody evolves by his clonal selection algorithm.

Even though the method which maintains the lightweight property of detectors employed by Forrest et al(1994) and Potter(1997) looks closer to human immune systems, the expressive power of each detector rule is weaker than De Jongs encoding power. For example, a single detector encoded by De Jong's method in the figure 1 can be represented by (number) detectors encoded by Forrest et al and Potters encoding schemes. Thus, even though they provide the generality of generated detector via the threshold match method, this work advocates the De Jongs genotype encoding scheme to generate an even lighter detector. In summary, this method trades the lightweight feature of a generated detector for a larger search space This conclusion does not imply that a threshold matching function is generally poorer at showing its generality when it is compared to logical representation. Notably GPs unique tree structure employing various functions provides a strong expressive power and it used a matching threshold concept such as a sphere threshold or a linear threshold. However, we still prefer De Jongs approach since its phenotype, which is logical representation, more natural to read and thus it provides strong intelligibility of generated detector rules. The intelligibility is one of significant IDS features for a security officer to operate IDSs.. However, when the overall architecture of AIS for network intrusion detection developed in this thesis is recalled, our main concern to make a system light originates from a possible burden of network traffic by transferring a large number of detectors from a primary IDS to secondary IDSs. Therefore, the lightweight feature of each detector is considered as more significant

than longer computation time to generate detectors due to larger search space. This problem can be resolved by employing the parallel arrays of primary IDSs and this suggestion was already made in chapter 2.

Discretisation

As seen in the previous section, each network activity profile has 41 fields. From these 41 fields, the values of 32 fields are continuous and the values of the other 9 fields are discrete. Specifically, the continuous values of 32 fields show a wide range of values. In order to encode this various and broad range of values in the genotypes that are described above, a discretisation algorithm is needed. For the negative selection, we simply defined each discretizsed cluster having contains the same number of records. Even though the simple discretisation algorithm provides the clusters, it is not certain whether each cluster is formed without serious information loss. As the result, the clonal selection in this work used an entropy-based discretisation algorithm (Witten and Frank, 2000). The basic idea of this algorithm is that splitting a numerical attribute value recursively until the new cluster intervals does not minimize its entropy. In other words, this algorithm seeks for the splitting points where makes the information gain largest. Here, the information gain is defined as the difference between the information value without the split and that with the split.

However, this kind of naive entropy-based discretisation algorithm is also hindered by the overfitting problem. Since the entropy measures the purity of data points, this value has the minimum when the data points of one split belong to the same class. However, this again leads the generated clusters reflecting only a training data set. Fayyad and Irani (1993) proposed the advanced entropy-based discretisation algorithm to alleviate the overfitting problem. Instead of searching for the largest information gain, it adopts the minimum description length(MDL) principle to stop recursive splitting. This principle suggests that the best solution will lie on the point which minimise the overall cost of the learned system. The cost of the system is defined as the degree of system complexity plus the amount of information required to specify the errors when the current system is applied. In other words, when the system complexity increases during a training period, it reflects a training data set more closely and thus the prediction error on the training data decreases. By stopping the system evolution earlier, the complexity of the system decreases, but it should pay more for informing incorrectly predicted cases. Thus, MDL priniciple proposes to seek for the balanced point between the high cost caused by bulding up the system reflecting a training set perfectly and the high cost brought from sending extra information to correct wrong predictions when the system is not complicated enough.

Fayyad and Irani's new stop criteria of recursive splitting is

$$Gain(A,T,S) < \frac{\log_2(N-1)}{N} + \frac{\Delta(A,T,S)}{N} \quad \text{---- (1),}$$

where N is the number of examples in the set S, a feature A and partition boundary T and

$$Gain(A,T,S) = Ent(S) - E(A,T,S)$$

the class information entropy of the partition made by T is

$$E(A,T,S) = \frac{|S_1|}{|S|} Ent(S_1) + \frac{|S_2|}{|S|} Ent(S_2),$$

and $Ent(S)$ is the entropy of S.

$$\Delta(A,T,S) = \log_2(3^k - 2)$$
$$-[k \cdot Ent(S) - k_1 \cdot Ent(S_1) - k_2 \cdot Ent(S_2)]$$

and k_1 is the number of class labels represented in the set . Here, the first component of (1) is the information needed to specify the splitting point and the second is a correction cost required to transmit which classes corresponding to the upper and the lower subintervals. Since this recursive partition of each numerical attribute value is completed by evaluating each splitting point independently using above criteria, some numerical attributes are clustered very finely whereas others will be partitioned coarsely.

Fitness Functions

While the generation of detectors by applying genetic operators is performed at the genotype level, the fitness value measurement among competing detector candidates is operated at the phenotype level. For effective network intrusion detection, two important objectives of AIS are regarded: accuracy, and generality. The accuracy of generated detector rules can be measured by considering false negative errors and false positive errors.. As discussed in the previous chapter, the human immune system uses the unique method to reduce false positive errors via negative selection algorithm. The clonal selection algorithm developed in this work also embed this negative selection in its fitness evaluation stage. (See section 4.3). The generality of is also achieved by its unique fitness evaluation method using antibody sampling. This also is discssed in section 4.2. The intelligibility which was considered as one of significant golas to be achieved by the financial fraud detection is excluded from the objective function of clonal selection. Even though this feature will be the important feature, our system currently concentrates on equipping its accuracy and generality first. Thus, the intelligibility of this system still remains as the problem to be tackled and we can apply the similar multiobjective fitness functions which were used in the financial fraud detector. As a result, the clonal selection of the AIS developed in this work has a single fitness function which indicates the accuracy of generated detector rules.

Many other evolutionary rule learning algorithms usually employ fitness functions measuring the accuracy of generated detector rules by counting the number of examples correctly classified. and its modality is maintained by a nested search. However, the clonal selection algorithm used in this work maintains its generality via the emergent fitness sharing completed by the antigen sampling at the fitness evaluation stage. This emergent fitness sharing was achieved because it lets only the selected antibodies in a sample to compete with each other per generation. In this case, the antibodies within a hypersphere of a specific radius around a given antigen form a local competing group and the fittest one in this group becomes the generalist one to describe this local searcg area. Thus, the key point of emergent fitness sharing of clonal selection is the natural formation of competing groups based on the hypersphere with the given radius around the selected antigen. In other words, the multi modals in an antigen search space are searched by grouping candidate antibodies, which are similar enough to evolve towards a same target antigen, into one competing group. Therefore, the clonal selection algorithm used in this work adopts the similarity measurement in order to boost its accracy feature instead of counting correctly classified items. Furthermore, the distance measurement has the advantage which accelearte the evolution speed at an early stage. This is because the detectors are initially generated with random genes and many of them are still far from any of antigens. Thus, if the system uses the correctly detected antigens as its fitness function values, it can have zero values or very tiny values. This can result in making the evolution proceed very slowly at the early stage. However, since the similarity measurement informs to the system how far each detector resides from the current antigens, the evolution can progress towards the current antigens a lot faster even before any of existing detectors detect any antigen.

The initial approach to measure the similarity between a selected antigen and antibodies is applied on their genotype level. Thus, the antigen phenotype value should be converted into an appropriate cluster number which was used to define detector genotypes. Then, if the corresponding nucleotide of detector genotype gene is turned on(the bit value is 1), this pair matches. Whenever a given antigen gene and detector gene match, that detector gene is assigned a match distance 0. However, if the corresponding nucleotide of detector genotype gene is not turned on(the bit value is 0), it searches for the closest nucleotide turned on. Then, the number of bits between the closest detector nucleotide, which is turned on, corresponding to the antigen cluster number is calculated as the match distance of these two genes. The second step is counting the matched fields and measuring the distance between unmatched fields. This finally defines the final match score, M_{gb}, between an antigen, A_g, and a detector, A_d. That is

$$M_{gb}(A_g, A_d) = \sum_g \left(\frac{d_{mg}}{N_{cg}} \cdot \frac{1}{N_g} \right)$$

,

where d_{mg} is the match distance between the g th antigen gene and the g th detector gene, N_{cg} is the number of total clusters comprising the g th gene and N_g is the number of total genes. Here, the gene match distance for each gene pair is divided by the total number nucleotides of the given gene in order to gain the gene match distance relative to the defined gene length. In addition, the gene match distance is scaled in order to make its maximum score 1 by dividin it by the total number of genes.

This match score represents the similarity between a given antigen and a detector. When this score is zero, they match completely and have the highest similarity. In the cotrast, when this score shows 1, their distance is the largest and so their similarity becomes the least. As we have seen in the section 4.2, the match score of the detector is kept until the detector/antigen sampling and evaluation procedure finishes after the certain number of repeated work. Then, the match score stored in the detectors are compared and only the detector which has the highest similarity (= the least match score) is selected for updating its fitness values. If the selected detectors are more than one, the clonal selection chooses only one detector to break the ties. The chosen detector update its fitness score by adding (1-match score) and represents one with the highest fitness score as the more desirable one.

Genetic Operators

Child rules are generated using two genetic operators: crossover and mutation.

Crossover

The crossover used in this work is the classic single-point crossover of genetic algorithm. It finds two random points in the parent genotypes and splicing the genotypes at that point. These spcliced genotypes are crossed over, and that the binary numbers within the genes are also crossed. This operator is used to generate all new offspring detectors, i.e., crossover is applied with a probability of 100%.

Mutation

The clonal selection algorithm uses a various types of mutation operators. As the first one, it uses the conventional mutation operator to modify randomly the

binary numbers in each gene by a single bit. The second mutation operator is the generalisation mutation. This mutation is designed to generate a more generalised offspring. To the new detector to be more general, this operator truns on one bit and results in having one more condition in a phenotype detector. For example, after applying the generalisation operator, a detector

(A1 = small) and (A2 = good) might create a new offspring
(A1 = small or medium) and (A2 = good).

To make the effect of this mutation the gradual change on the offspring, the clonal selection algorithm selects one nucleotied randomly and it is turned on only any of its adjacent bits is turned on. For the case when all nucleotides are turned off, any nucleotide can be truned on.

The next mutation operator applied is the specialisation mutation. On the contrast to the generalisation mtation, this mutation simply drops one bit and generates an offspring with one less condition. For instance, a detector *(A1 = small or medium) and (A2 = good)* can generate a new offspring
(A1 = small) and (A2 = good) after being applied by the sepecialisation mutation.
This mutation is applied in a similar way to the generalisation mutation's. That is that any ranomly selected nucleotide is turned off only if any of its adjacent bits is turned off. In the case when all nucleotides are turned on, any nucleotide can be truned off.

The forth mutation operator is the shift mutation. It shifts all the bits of a chosen gene to the right direction or the left direction. The shift direction is determined at random. Thus, a detector
(A1 = small) and (A2 = good) might have its offspring
(A1 = medium) and (A2 = good) because of the application of the shift mutation.

Finaly, the delete mutation is also introduced in this system. It aims to delete an entire gene from a given detector. For example, a detector
(A1 = small) and (A2 = good) can have its offspring
(A1 = small) after being applied by the delete mutation.
This mutation works by turning on all the nucleotides and results in excluding the gene when a detetor is mapped into phenotypes. This mutation is designed in order to operate as a feature selection operator. Since the candidate features included in a training set often include irrelevant ones and they can disrupt the search method immensly, the clonal selection algorithm uses the deletion mutation to filter out those genes.

Others
The clonal selection employs population overlapping, where the worst n% of the population are replaced by the new offspring generated from the best m%. Typically values of n = 80 and m = 40 seem to provide good results. The population size was normally between 100 and 200 individuals.

Conclusion

This research has investigated network-based IDSs and provided a set of general requirements for them by a careful examination of the literature. Based on these requirements, three principal design goals were identified. After sketching the simplified human immune system, their salient features that can contribute to build a competent network-based intrusion detection system were analysed. This analysis show that the human immune system is equipped with a number of sophisticated mechanisms, which satisfy the three identified design goals. Consequently, the design of a novel network-based IDS based on the human immune system is promising for future network-based IDSs.

This research also investigated the existing network-based IDSs. They were categorised into three different approaches: monolithic, hierarchical and co-operative and problems were identified for each approach. In order to resolve these problems, a novel artificial immune model was presented. This model combines the three evolutionary stages: gene library evolution, negative selection and clonal selection into a single methodology. These three processes are co-ordinated across a network to satisfy the three goals for designing effective IDS's: being distributed, self-organising and lightweight. Analysis of the characteristics of this unified evolutionary approach show that, unlike existing approaches, the proposed artificial immune model does satisfy the requirements of network-based IDSs. Consequently, algorithms based on this model show considerable promise for future IDSs.

A network-based IDS utilising the artificial immune model is being implemented in order to prove the validity of this approach. Current work is focusing on building initial self profiles and detectors from normal and abnormal TCP/IP packets, which were collected from a real network environment. As the first attempt of this effort, the negative selection stage was implemented and experiments showed its infeasibility for its application to the essential profiling fields of real network data. This result directs this research to re-define the role of negative selection algorithm within the overall artificial immune system framework. Finally, the intrusion detection mechanism of clonal selection stage were investigated and the clear understanding of task of clonal selection stage helped us to comprehend the distinct job of negative selection stage.

The contributions of this work will provide an applicable

methodology for designing an artificial immune system to be able to perform network intrusion in a truly distributed, self-organising and lightweight way.

Reference

[1] (Balasubramaniyan et al., 1997) Balasubramaniyan, J. S. et al., "Software Agents for Intrusion Detection", Department of Computer Sciences, Purdue University, 1997.available at http://www.cs.purdue..edu/coast/coast-library.html

[2] (Bentley, 2000a) Bentley, P. J., "Evolutionary, my dear Watson-Investigating Committe-based Evolution of Fuzzy Rules for the Detection of Suspicious Insurance Claims", in Proceeding of the second Genetic and Evolutionary Computation Conference (GECCO 2000), July 8-12, Las Vegas, Nevada, USA.

[3] (Bentley, 2000b) Bentley, P. J., "Evolving Fuzzy Detectives: An Investigation into the Evolution of Fuzzy Rules". Chapter in Suzuki, Roy, Ovasks, Furuhashi and Dote (Eds), Soft Computing in Industrial Applications. Springer Verlag London Ltd. ISBN 1-85233-239-X.

[4] (De Jong et al., 1993) De Jong, K. A., Spears, W. M., and Gordon, D. F., "Using Genetic Algorithms for Concept Learning", Machine Learning, Vol.13, No.2/3, pp.161-188, 1993

[5] (Dhaeseleer et al., 1997) Dhaeseleer, P. et al, "A Distributed Approach to Anomaly Detection", ACM Transactions on Information System Security., 1997. Available at http://www.cs.unm.edu/~patrik

[6] (Fayyad and Irani, 1993) Fayyad, U. M., and Irani, K. B., Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning, Proceeding of The Thirteenth International Joint Conference on Artificial Intelligence, pp.1022-1027, 1993.

[7] (Flockhar, 1995) Flockhar, I. W., "GA-MINER: Parallel DataMining with Hierarchical Genetic Algorithms" Final Report, EPCC-AIKMS-GA-MINER-REPORT 1.0, Edinburgh Parallel Computing Center, 1995.

[8] (Forrest et al., 1997) Forrest, S.; Hofmeyr, S; Somayaji, A, "Computer Immunology", Communications of the ACM, Vol.40, No.10, pp.88-96, 1997.

[9] (Forrest et al., 1996) Forrest, S. et al., "A Sense of Self for Unix processes", Proceedings of 1996 IEEE Symposium on Computer Security and Privacy, Los Alamos, CA, pp.120-128, 1996.

[10] (Forrest et al., 1994) Forrest, S. et al, "Self-Nonself Discrimination in a Computer", Proceeding of 1994 IEEE Symposium on Research in Security and Privacy, Los Alamos, CA: IEEE Computer Society Press, 1994.

[11] (Forrest et al, 1993) Forrest, S. et al, "Using Genetic Algorithms to Explore Pattern Recognition in the Immune System", Evolutionary Computation, 1(3), 191-211, 1993.

[12] (Goldberg, 1989) Goldberg, D. E., Genetic Algorithms in Search, Optimization & Machine Learning, Addison-Wesley, 1989.

[13] (Kephart, 1994) Kephart, J. O., "A Biologically Inspired Immune System for Computers", Artificial Life IV, Proceeding of the Fourth International Workshop on the Synthesis and Simulation of Living Systems, pp.130-139, 1994.

[14] (Lee, 1999) Lee, W., A Data Mining Framework for Constructing Feautres and Models for Intrusion Detection Systems, Phd Thesis, Dept of Computer Science, Columbia University, 1999.

[15] (Mykerjee et al, 1994) Mykerjee, B.; Heberlein, L. T.; Levitt, K. N., "Network Intrusion Detection", IEEE Network, Vol.8, No.3, pp.26-41, 1994.

[16] (Noel 1982), Noel, C., "Credit Scoring Systems: A Critical Analysis", Journal of Marketing, Vol.48, Spring, pp.82-91, 1982.

[17] (Playfair, 1996) Playfair, J. H. L., Immunology at a Glance, 6th Ed, Blackwell Science, 1996

[18] (Porras and Valdes, 1998) Porras, P. A.; Valdes, A., "Live Traffic Analysis of TCP/IP Gateways", Proceeding of ISOC Symposium of Network and Distributed System security, 1998. Available at http://www2.csl.sri.com/emerald/downloads.html

[19] (Potter, 1997) Potter, M. A., The design and analysis of a computational model of cooperative co-evolution, PhD Thesis, George Mason University, Fairfax, VI., 1997.

[20] (Roitt and Brostoff, 1998) Roitt, I., Brostoff, J., and Male, D., Immunology, Fifth Ed., Mosby International Ltd, 1998.

[21] (Smith, Forrest and Perelson, 1993) Smith, R. E., Forrest, S., and Perelson, A. S., "Searching for Diverse, Cooperative Populations with Genetic Algorithm", Evolutionary Computation, 1(2), 127-149, 1993.

[22] (Somayaji et al., 1997) Somayaji, A.; Hofmeyr, S.; Forrest, S., 1997, "Principles of a Computer Immune System", Proceeding of New Security Paradigms Workshop, Langdale, Cumbria, pp.75-82, 1997.

(Tizard, 1995) Tizard, I. R., Immunology: Introduction, 4th Ed, Saunders College Publishing, 1995.

(Witten and Frank, 2000) Witten, I.H. and Frank, E. Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, Morgan Kaufmann Publishers, 2000.