Journal of Zhejiang University SCIENCE
ISSN 1009-3095
http://www.zju.edu.cn/jzus
E-mail: jzus@zju.edu.cn

An immunity-based technique to detect network intrusions*

PAN Feng (潘 峰)†[1], DING Yun-fei (丁云飞)[2], WANG Wei-nong (汪为农)[1]

([1]*Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200030, China*)

([2]*Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield S10 2TN, UK*)

†E-mail: fpan@sjtu.edu.cn; pan_feng_hao@hotmail.com

Received Dec. 5, 2003; revision accepted Apr. 21, 2004

Abstract: This paper briefly reviews other people's works on negative selection algorithm and their shortcomings. With a view to the real problem to be solved, authors bring forward two assumptions, based on which a new immune algorithm, multi-level negative selection algorithm, is developed. In essence, compared with Forrest's negative selection algorithm, it enhances detector generation efficiency. This algorithm integrates clonal selection process into negative selection process for the first time. After careful analyses, this algorithm was applied to network intrusion detection and achieved good results.

Key Words: Artificial immune system, Network intrusion detection, Negative selection, Clonal selection
doi:10.1631/jzus.2005.A0371 **Document code:** A **CLC number:** TP393.08

INTRODUCTION

Research interest in immune system has increased over the past few years, and because of its special information processing capabilities, it has been applied to solve many problems (de Castro and Von Zuben, 1999; Hunt and Cooke, 1996). Among these research areas, network security is one of the hot spots and has been considered as analogous to immunity in natural systems. Researchers at the University of New Mexico did a lot of research in this area (Forrest *et al*., 1997), with negative selection algorithm being the core of their research. They used this algorithm to detect virus and network intrusions and achieved good results.

Network intrusion detection is very complex because it is hard to characterize the normal and abnormal behavior of a system in network environment. Many people have done lots of research in this area. The general assumption is that the normal behavior of a system can often be characterized by a series of observations over time. Also, normal system behaviors generally exhibit stable patterns when observed over a period of time. There are many approaches to implement such anomaly detection (Denning, 1987). In this paper, we use Dipankar's method to characterize intrusions in computer network (Dasgupta and Gonzalez, 2002). He used traffic statistics to represent normal behaviors.

OVERVIEW OF HUMAN IMMUNE SYSTEM

First let us review the main mechanism of the human immune system (de Castro and Von Zuben, 1999). The overall human immune system is implemented through interactions among a large number of different types of cells. Among these cells, B and T cells (both being leukocytes) are the most important. Their main function is to distinguish self-cells, or cells of the human body, from non-self cells which are dangerous foreign cells. Each lymphocyte is specialized in reacting to a limited number of structurally related harmful foreign cells known as antigens (Ag).

B-cells are antibody (Ab) secreting cells and T-cells kill antigens or help or suppress the development of B-cells. When antibody recognizes a spe-

* Project (No. 60073034) supported by the National Natural Science Foundation of China

cific antigen, antibody binds to antigen. Both B-cells and T-cells have their own unique genetic structures. They are developed in the bone marrow and the thymus respectively. At the bone marrow and the thymus, several gene libraries uniquely corresponding to the domain of B-cells and T-cells contain the candidate genes to express B-cells and T-cells. An antibody is generated by joining randomly selected gene segments from the gene libraries.

Before leaving the bone marrow and the thymus, maturing B-cells and T-cells have to pass the last test, negative selection. Because in the development process of B-cells and T-cells, brand-new cells can be generated via various genetic operators. Therefore, it is possible for a randomly generated antibody to bind to self-cells. To prevent this from happening, maturing B-cells and T-cells binding to self-cells are killed when they are still in the bone marrow and in the thymus respectively instead of being released into the body.

Mature B-cells and T-cells that passed the negative selection are released from the bone marrow and thymus, continuously circulate in the blood system inside the body, and will activate and evolve when they encounter corresponding antigens. When the antibody of B-cells recognizes harmful antigens, they bind to them and activate directly or indirectly with the help of T-cells. When the binding affinity between antibody and antigen exceeds a certain threshold, B-cells will be activated and immediately followed by clonal selection process. The activated B-cells are divided into a number of clones that have the same antigen-binding properties as the parent B-cells or muted antigen-binding properties. On the other hand, if any antigen cannot activate B-cells within a limited time, they will die off soon. So, based on the existing antigens, only the fittest B-cells antibodies survive. Furthermore, when antigens activate B-cells, they produce memory cells for reoccurrence of the same antigens in the future. Because of these memory cells, antigens that have been identified previously are detected much quicker (known as secondary response).

SHAPE-SPACE MODEL AND NEGATIVE SELECTION ALGORITHM

The whole set of cells available for antigen recognition, called immune repertoire, has to be complete in order to properly protect our body from harmful invaders. In order to quantitatively describe the interactions between immune cells and antigens, Perelson and Weisbuch (1997) introduced the concept of shape-space model. The shape-space idea is that the degree of binding between antibody and antigens can be described with a set of features. What these features are is not important. This set of feature is called generalized shape of an antibody combining site by L parameters. Then a point in the L-dimensional space specifies the generalized shape of an antigen's binding region with regard to its antigen binding properties.

As shown in Fig.1, all points lie in some finite volume V of L-dimension space. It is assumed that each antibody specifically interacts with all antigens that are within a small surrounding region, characterized by the parameter ε, and called a recognition region, of volume V_ε. Because each antibody can recognize all antigens within the region, a finite number of antibodies can recognize an almost infinite number of points in the volume V_ε, where similar patterns occupy neighboring regions of the shape-space, and might be recognized by the same antibody shape, if adequate ε is provided.

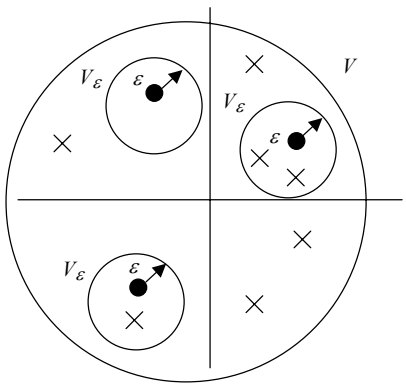

Fig.1 Ag-Ab representations and affinities

In this paper, using wavelet multi-resolution analysis, we initiate the clonal selection process by searching for optimum in local space. Here is how it works: An antibody activated into the clonal selection process will split into some new second-level antibodies with smaller recognition distance as shown in Fig.2. In this process, we must guarantee that the sum

of these new antibodies' recognition regions can cover the old antibody's (their parent antibody) recognition region. In essence, we think second-level antibody outperforms their parent antibody because it can recognize at higher precision.

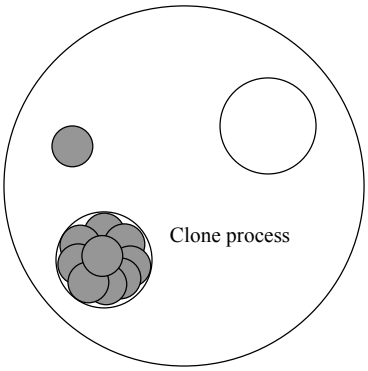

Fig.2 Clonal selection process

The idea of applying immune-logic principles in network security has progressed since 1994 (Forrest *et al.*, 1994). Forrest developed a negative selection algorithm based on the principles of self/non-self discrimination in the immune system. This algorithm is used to detect virus in computer system and intrusion in network environment.

In order to generate detectors (these detectors are analogous to the antibody in immune system) that can detect "non-self", but at the same time, will not detect "self", they must pass the negative selection process as shown in Fig.3 (Hofmeyr, 1999). It can be summarized as following:

1. Define self as a collection of strings of length l over a finite alphabet, a collection that must be monitored. For example, S may be a normal pattern of activity, which is segmented into equal-sized sub-strings;

2. Generate a set R of detectors, each of which fails to match any string in S;

3. Monitor S for changes by continually matching the detectors in R against S. If any detector ever matches, then a change is known to not match any of the original strings in S.

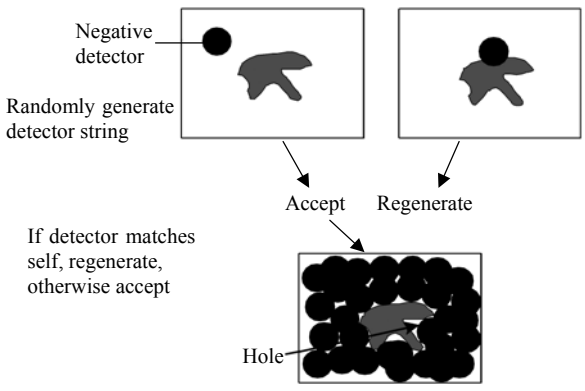

Fig.3 Negative selection algorithm

Forrest *et al*. gave a comprehensive analysis of this algorithm. Under certain matching rules (either r-continuous distance used by Hofmeyr or others), suppose the probability of two random strings a, b matching is defined as P_M. The size of the self set is defined as $|S_R|$. Then the number of retries required to generate a valid detector is a geometric random variable with parameter P_M, the expected number of trials, ρ until success is achieved (Hofmeyr, 1999):

$$E(\rho)=\frac{1}{(1-P_M)^{|S_R|}}$$

Hence the expected number of retries to generate a single valid detector is exponential in the size of the self set. The negative selection algorithm can be applied to any match rule. However, because of the exponential time complexity of negative selection, two other algorithms had been developed specifically for the contiguous bits rule. Both algorithms trade space complexity for time complexity and are not universal to other distance.

On the other hand, for a match rule with a constant probability of matching, there can exist non-self strings for which no valid detectors can be generated. Such strings are called holes because they are "holes" in the detection system's coverage of non-self set. D'haeseleer *et al*.(1996) suggested using different matching rules for different detectors could reduce the overall number of holes. That is to say, using multiple representation of detector.

MULTI-LEVEL NEGATIVE SELECTION ALGORITHM

From shape-space model's aspect, we can basically summarize Forrest *et al.*'s work in the following steps: (1) First randomly generate detectors, whose

sum of recognition region can cover the whole problem space; (2) Use negative selection algorithm to delete those detectors that will detect "self"; (3) Then use the remaining detectors to detect "non-self"; if one of these detectors detect something, then it must be something "non-self"; and represent something abnormal had occurred.

We think the shortcomings of previous work lie in: (1) Negative selection algorithm has exponential time complexity. So it will not be practical when the problem space gets larger. Kim and Bentley (2001) and Harmer *et al*.(2002) proved this empirically; (2) The assumption in Forrest's negative algorithm that self-set is distributed randomly in the whole space, does not apply to most real life problems; (3) The negative selection algorithm only uses one scale detector (in this paper, scale equals to detector's recognition distance).

Two assumptions

In most real self/non-self discrimination problems, the self set is not distributed randomly. We think most of them are based on the following two assumptions: (1) Self set are not randomly distributed in the space, but are instead aggregated in the space; (2) Self set only occupies a very small region of the whole space.

For example, in network intrusion application, Heberlein e*t al*.(1990) discovered that in Network security Monitor's (NSM) test for two months, for all the possible datapaths (the whole space) in the LAN, only 0.6% were included in the normal datapaths not subject to frequent changes. The same rule was observed in the UNIX process (Forrest *et al*., 1996). That is to say, a user's normal behaviors (featured by UNIX process and their sequences) are stable. It is also the reason why we can use anomaly detection techniques in network intrusion detection.

With the above two assumptions in mind, combined with the idea of wavelet's multi-resolution analysis, we designed a multi-level negative selection (MNS) algorithm, different from previous negative selection algorithms called one-level negative selection algorithms. The basic idea of MNS algorithm is to use different level detectors from the beginning for integrating clonal selection process into negative selection process. Our clonal selection process uses local optimization techniques yielding more precise recognition in the local recognition space of the old detectors as shown in Fig.4.

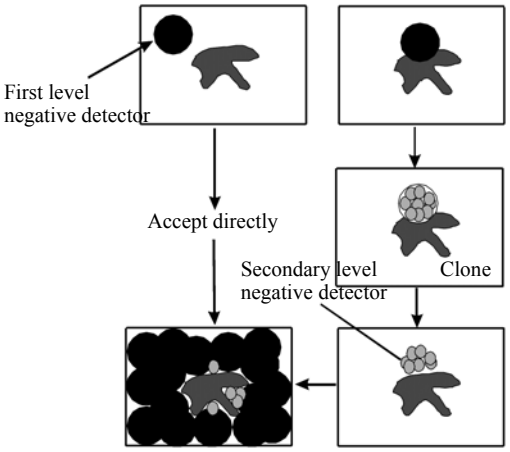

Fig.4 Multi-level negative selection algorithm

1. First generates big scale detectors (with bigger recognition distance). These detectors must guarantee that the sum of their recognition space covers the whole space. They can be generated randomly or by other method, such as simulated annealing algorithm (SA).

2. These detectors undergo negative selection process. If any one of them fails to pass negative selection, they will not be deleted directly as in one-level negative selection algorithm, and will then undergo clonal selection process to generate new detectors again, but with smaller recognition distance; the sum of these new detectors' recognition space must cover the recognition space of the old detector. Again these new detectors enter a second round of negative selection process, for those that failed to pass negative selection. The above process is repeated again until the detectors' recognition distance reaches a threshold, which can guarantee the precision we need. So the detectors set we finally got is composed of different scales.

Compared with one-level negative selection algorithm, multi-level negative selection algorithm can solve the following problems (Certainly it is based on previous two assumptions):

1. Since it is hard to describe exactly the normal behavior's distribution of real life applications mathematically, so we have not come out with MNA's time complexity. But if it follows the previous two assumptions, we think the time complexity of

our algorithm is dramatically reduced as compared to Forrest's algorithm.

2. Under the same detection error percentage, the number of detectors we need is far less than the number of detectors if we use single scale detector in one-level negative selection algorithm, because large scale detector can cover more space than small scale detector. And it is easy for us to imagine that in the space closer to that of normal behavior, we can get more small detectors as shown in Fig.4.

3. We can reduce the so-called hole region as much as we want. Holes exist only because the boundary of normal behavior is not consistent, so there must be some region this detector cannot cover while using single scale detector. But MNA can generate corresponding scale detectors according to the boundary. This is depicted in Fig.4. This method to reduce holes is much easier and reliable than D'haeseleer's method, and the whole process is completely self-adaptively.

4. Clonal selection process in this algorithm is easy to implement and is theoretically sound. In MNA, clonal selection process plays a clear role, which is to generate new small scale detectors that can cover their parent detector's recognition space, so more precise recognition can be achieved.

5. It is easy to sum up the information we collected when abnormality is detected. Just like normal behavior, a certain kind of abnormal behavior is also aggregated in the space, for example, a kind of network intrusion. Using this multi-resolution idea, we can learn about the intrusion's features from different scale.

NETWORK INTRUSION DATA SET AND EXPERIMENT RESULTS

We used MNA to solve network intrusion detection problems. Our experiment used the same data as Dipankar used. Intrusion's features are selected the same they do it. Dipankar used two methods to detect intrusion. One of them is negative selection. They used genetic algorithm to generate detectors in negative selection.

Real intrusion data was obtained from the Lincoln Laboratory of the Massachusetts Institute of Technology. The data represented both normal and abnormal information collected in a test network, where simulated attacks were performed. These data was used to test the performance of the intrusion detection system. The data set (corresponding to the year 1999) contained complete weeks with normal data (not mixed with attacks) (DARPA, 1999), which provided enough samples to train our detection system.

The test data set was composed of network traffic data (tcpdump, inside and outside the network traffic), audit data (bsm), and file systems data. They used only the outside tcpdump network data for a specific computer (e.g., hostname: marx) and then they applied the tool tcpstat to get traffic statistics. They used the first week's data for training (attack free) and the second week's data for testing, which included some attacks, some of which were network attacks while the others were inside attacks. Only the network attacks were considered for our testing. These attacks are described in Table 1.

Table 1 Second week attacks description

Day	Attack name	Attack type	Start	Duration
1	Back	DOS	9:39:16	00:59
2	Portsweep	PROBE	8:44:17	26:56
3	Satan	PROBE	12:02:13	02:29
4	Portsweep	PROBE	10:50:11	17:29
5	Neptune	DOS	11:20:15	04:00

Three parameters were selected to detect some specific type of attacks. These parameters were sampled each minute (using tcpstat) and normalized. Table 2 lists six time series S_i and T_i for training and testing, respectively. The set of normal descriptors was generated from a time series $R=\{r_1, r_2, \ldots, r_n\}$ in an overlapping sliding window fashion

$$S=\{(r_1,\ldots,r_w),(r_2,\ldots,r_{w+1}),\ldots,(r_{n-w+1},\ldots,r_n)\}$$

Table 2 Datasets and parameters used

Name	Description	Week	Type
$S1$	Number of bytes per second	1	Training
$S2$	Number of IP packets per second	1	Training
$S3$	Number of ICMP packets per second	1	Training
$T1$	Number of bytes per second	2	Test
$T2$	Number of IP packets per second	2	Test
$T3$	Number of ICMP packets per second	2	Test

where w is the window size. In general, from a time series with n points, a set of $n–w+1$ of w-dimensional descriptors can be generated. In some cases, they used more than one time series to generate the feature vectors. In those cases, the descriptors were put side by side in order to produce the final feature vector. For instance, if we used the three time series $S1$, $S2$, and $S3$ with a window size of 3, a set of nine-dimensional feature vectors was generated. Dipankar used window size of 1 and 3. But we only used window size of 1 in our experiment.

Instead of using binary encoding in the negative selection algorithm, like Dipankar, our approach used real-valued representation to characterize the self/non-self space. Detectors were generated by MNA.

For simplicity of description, we used two-dimensional space as example. Higher dimensional space is easy to expand. Suppose a detector's coordinates are (x,y), its recognition distance is dis, then its recognition space is a square with the following 4 points: $(x–dis,y–dis)$, $(x–dis,y+dis)$, $(x+dis,y–dis)$, $(x+dis,y+dis)$. If this detector fails to pass negative selection, it will clone into the 4 next level detectors: $(x–dis,y–dis)$, $(x–dis,y+dis)$, $(x+dis,y–dis)$, $(x+dis,y+ dis)$. These four detectors' recognition distances are all changed to $dis/2$. They can cover the old detector's recognition space. Then these 4 detectors undergo negative selection again. Those that fail to pass negative selection clone again, generate next level detectors. This process continues until the detector's recognition distance reaches some threshold that can guarantee a certain error we want. This is depicted by Fig.5.

In our experiment, detectors were designed like these: detector $(x, y, z, distance)$, x, y and z corresponding to $S1$, $S2$ and $S3$. That is to say, window size is 1. This time the detector's recognition space is a cube. At the beginning, we define the detector as (0.5,0.5,0.5,0.5); its recognition space is the whole space. Using first week's data, if this detector cannot pass negative selection, it will clone into 8 new detectors, whose recognition distance is 0.25, and recognition space is 1/8 that of the old detector. We define $distance<0.01$ as the stop condition in the experiment. That is to say, the detectors we generated were composed of five-level detectors with the following recognition distance: 1/4,1/8,1/16,1/32,1/64. In another aspect, we can see, in this way, these detectors' recognition regions do not overlap.

We got totally 120 detectors in the end. The sum of their recognition space was 0.999573, which means normal behavior only accounts for 0.000427 (because the total space accounts for 1, 0.000427= 1–0.999573) in the space. This proves our previous two assumptions.

Then we used these detectors to detect attacks in the second week data. If the data is recognized by a detector, then these data's abnormal degree is set as the detector's recognition distance. If the data are not recognized by any detector, these data's abnormal degree is set as 0. The reason why we give data's abnormal degree in this way is that we can see clearly that as behavior gets closer to self, its corresponding detector's recognition distance gets smaller as shown in Fig.5. Fig.6 shows second week data's abnormal degree sequences.

From this figure, we can discover 5 attacks clear-

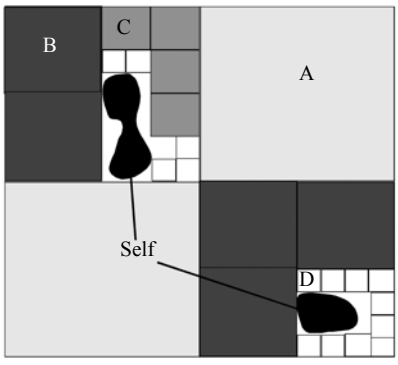

Fig.5 Multi-level negative selection algorithm (A, B, C, D denote different level detector respectively)

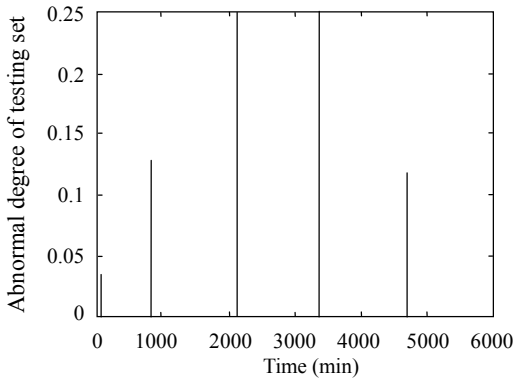

Fig.6 Indicates the abnormal degree in the testing set detected by generated detectors

ly. Compared with the results of Dipankar, our result was better than the results of both of his detection methods. And in the same time, we used only window size of 1 (three-dimensional feature) while Dipankar used window size of 3 (nine-dimensional feature).

CONCLUSION

In this paper, we introduced multi-level negative selection algorithm, which compared to one-level negative selection algorithm, improved detector generation efficiency dramatically. After analyzing this algorithm from many aspects, we applied it to network intrusion detection and achieved very good results.

In another aspect, this algorithm is universally applicable. It can be applied to any space and distance. The only condition it requires is that our previous two assumptions hold in the self/non-self discrimination problem. And we think in most of cases it will.

Our future research will include experiments with other intrusion detection data (both online and offline), and other application domains.

References

DARPA, 1999. Intrusion Detection Evaluation. http://www.ll.mit.edu/IST/ideval/index.html.

Dasgupta, D., Gonzalez, F., 2002. An immunity-based technique to characterize intrusions in computer networks. *IEEE Trans on Evolutionary Computation*, **6**(3):281-291.

de Castro, L.N., Von Zuben, F.J., 1999. Artificial Immune Systems: Part I–Basic Theory and Applications. Technical Report–RT DCA 01/99, FEEC/Univ. Campinas, Campinas, Brazil. http://www.dca.fee.unicamp.br/~lnunes/immune. html.

Denning, D., 1987. An intrusion-detection model. *IEEE Trans. Software Eng.*, **13**:222-232.

D'haeseleer, P., Forrest, S., Helman, P., 1996. An Immunological Approach to Change Detection: Algorithms, Analysis and Implications. Proceedings of the 1996 IEEE Symposium on Research in Security and Privacy, IEEE Computer Society Press, Los Alamitos, CA.

Forrest, S., Perelson, A., Allen, L., Cherukuri, R., 1994. Self-Nonself Discrimination in A Computer. Proc. of the IEEE Symposium on Research in Security and Privacy, Oakland, CA, USA, p.202-212.

Forrest, S., Hofmeyr, S.A., Somayaji, A., 1996. A Sense of self for UNIX Processes. Proceedings of the 1996 IEEE Symposium on Research in Security and Privacy, IEEE Computer Society Press, Los Alamitos, CA.

Forrest, S., Hofmeyr, S., Somayaji, A., 1997, Computer immunology. *Communications of the ACM*, **40**(10):88-96.

Harmer, P.K., Williams, P.O., Gunsch, G.H., Lamont, G.B., 2002. An artificial immune system architecture for computer security applications. *IEEE Transactions on Evolutionary Computation*, **6**(3):252-280.

Heberlein, L.T., Dias, G.V., Levitt, K.N., Mukherjee, B., Wood, J., Wolber, D., 1990. A Network Security Monitor. Proceedings of the IEEE Symposium on Security and Privacy, IEEE Press.

Hofmeyr, S.A., 1999. An Immunological Model of Distributed Detection and Its Application to Computer Security. Ph. D. Dissertation, University of New Mexico.

Hunt, J.E., Cooke, D.E., 1996. Learning using an artificial immune system. *Journal of Network and Computer Applications*, p.189-212.

Kim, J., Bentley, P.J., 2001. Evaluating Negative Selection in An Artificial Immune System for Network Intrusion Detection. Genetic and Evolutionary Computation Conference 2001 (GECCO-2001), San Francisco, p.1330-1337.

Perelson, A.S., Weisbuch, G., 1997. Immunology for Physicists. *Rev. of Modern Physics*, **69**(4):1219-1265.