Detector Generation Algorithm Based on Online GA for Anomaly Detection

[1]Dongyong Yang, [1]Jinyin Chen, [2]Matsumoto Naofumi
[1]Zhejiang University of Technology, Hangzhou, Zhejiang, China, 310014
[2]Ashikaga Institute of Technology, Ashikaga, Japan, 3268558
chenjinyin@163.com

Abstract: -Detector plays an important role in intrusion detection system in artificial immune system, which makes detector generation algorithm especially significant. Traditional NSA cannot satisfy current network demands because the affinity limit *r* is difficult to fix in prior. A novel online GA-based algorithm is come up with self-adaptive mutation probability, in which affinity limit *r* is self-adaptive. Compared with GA-based detector maturation algorithm, detectors in online GA-based algorithm evolve online during the detection process which realizes self-organization and online learning to be adaptive to dynamic network. Finally simulation results testify that TP (true positive) value and FP (false positive) value of online GA-based algorithm is much better than NSA, GA-based and IGA-based algorithms without significant algorithm complexity increase.

Key-Words: - Detector generation algorithm, GA, Online GA, Self-adaptive, Intrusion detection

1. INTRODUCTION

In recent years, the intrusion detection technique has gotten rapidly development, which plays an important role in attack detection, security check and network inspect. Intrusion detect system, inspired by biological system, was broadly adopted, especially detector generation based on T-cell generation algorithm. Negative selection algorithm (NSA), firstly brought up by Forrest, is widely used to detect changes in data/behavior patterns by generating detectors in the complementary space. In artificial immune system, negative selection algorithm is applied for self (normal) and non-self (abnormal) discrimination [1]. With the continuous development of network, intrusion detect system with traditional NSA cannot meet the demand of security standards. Various massive intrusions have taken their steps forward network and PCs. NSA has large space complexity and time complexity which mainly depends on the size of self set and detect targets. Consequently two mended negative selection algorithms are brought up, which are lineal NSA and greedy NSA [2]. Several novel NSAs are come up for various applications. A randomized real-valued NSA provides advantages such as increased expressiveness, the possibility of extracting high-level knowledge from the generated detectors, and in some cases improves scalability [3]. However because the detector representation is real-valued, the operations are more complicated than binary coded detectors [4]. Another improved NSA with an array of partial matching lengths is designed, who calculates the best affinity among trial self set [5]. Experiments prove that failure probability is much lower than traditional NSA. Cooperative intrusion detection is provided for dynamic coalition environments [6], for the reason that different types of intrusions can be detected by relevant detectors bounded to. On the other side, once the multiple detector structure is fixed, it's not flexible for changes.

In this paper, detector generation based on GA is implemented [7], simulation results of which aren't satisfying, including true positive (TP), failure positive (FP) and detection time complexity. Because fitness functions in GA and IGA-based algorithms are designed to increase matching affinity between detectors and self set. However the most important task of IDS (intrusion detection system) is to improve the performance of detection instead of indirect purpose such as matching affinity of detectors and self set. A novel online GA based T-cell

detector generation algorithm is designed and applied. Experiments results testify the high performance of online GA based algorithm on TP and FP.

2. NSA and mended nsas

NSA has successful applications in biological system, on basis of which, Forrest applied it for intrusion detection system for network security [1]. FP (false positive) is defined as $P_f = (1-P_M)^{N_R}$, where P_M is the matching probability, and N_R is the number of detectors. In order to minimize P_f, increase detectors as possible as can. However considering of practical conditions, time complexity is enhanced accordingly, which is calculated as $O(\frac{-\ln(P_f)}{P_m \cdot (1-P_m)^{N_s}} \cdot N_s)$, and space complexity is defined as $O(l \cdot N_s)$.

Lineal NSA [2] is come up to improve efficiency of NSA in aspect of relationship of detector and self set. This algorithm is targeted at r-contiguous bits matching rule, in which sting length l and continuous matching bits r are both fixed. In this situation, time complexity is lineally related with detected target that can be demonstrated as $O((l-r)\cdot N_s)+O((l-r)\cdot 2^r)+O(l\cdot N_R)$, while space complexity is described as $O((l-r)^2 \cdot 2^r)$. Taking the restriction of matching rules of lineal NSA into consideration, the application area has been limited.

Greedy NSA is another mended NSA brought up in paper [2], purpose of which is to eliminate detector set redundancy in lineal NSA, and at the same time covering more non-self space except for self space. The details aren't depicted here, and the space complexity is the same as lineal NSA, while the time complexity is less than lineal NSA as $O((l-r)\cdot 2^r \cdot N_R)$.

In NSA the most critical problem is how to define the value of r for matching rule in binary coded detector situations. In experiments, if the value of r is not defined properly in prior, the detectors cannot be generated, besides the proper value of r can only be known after several trials. So it's necessary to put forward other detector generation algorithm.

3. Three ga-based algorithms

Detector generation process can be simulated as evolution process of GA, in which detector set is simulated as population, and each detector is as an individual in whole population. The mature process can be realized by population evolution.

Binary coded rule is applied in this paper, where affinity calculation rule is hamming distance.

$$D = \sum_{i=1}^{L} \delta, where \delta = \begin{cases} 1 & if \quad ab_i \neq ag_i \\ 0 & otherwise \end{cases} \qquad (1)$$

D denotes for the sum of same bits between antibody and antigen, which are self/non-self and detector in intrusion detect system.

3.1 GA-based detector generation algorithm

Detector evolution flowchart is as follows.

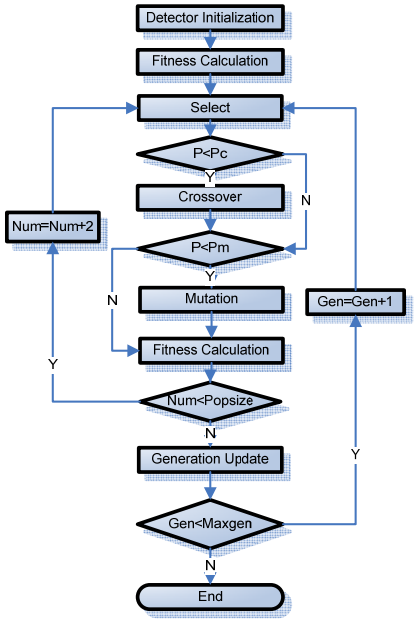

Fig. 1. Detector generation algorithm based on GA

As shown in figure 1, the progress of detector generation follows the steps of traditional GA.

Step 1: Detector initialization takes the operation of generating initial detector in random.

Step 2: Fitness of *detector(i)* is calculated according to fitness formula.

$$fitness(i)=\begin{cases}0, & \min Self=0\\ 1-\max Self, & otherwise\end{cases} \quad (2)$$

Where *minSelf* and *maxSelf* represent the minimum affinity and maximum affinity between *detector(i)* and self set. The purpose of fitness definition is to minimize the affinity between detector and self set. In order to satisfy that no self will be detected by detector set, the affinity range *r* is defined as $r=\max Self+1$. So member in self set won't be detected as non-self [8].

Step 3: Select operation adopts tournament selection strategy.

Step 4: Crossover takes one-point crossover method, in which cross point is generated randomly.

Step 5: Mutation operation introduces multi-point mutation technique. Once the mutation position is produced at random among the detector length, the bit will be flipped from '0' to '1' or from '1' to '0'. The mutation probability depends on the fitness of detector, where detector of higher fitness value takes low probability mutation.

Step 6: After fitness calculation and mutation operation, the new population may have better detector than the old population. Generation update is use to select the better half of the buffer consists of new population and old population.

Step 7: If *gen* achieves *maxGen*, then end the algorithm, otherwise transfer to step 3.

In GA-based detector generation, the population lost its diversity sharply after certain generations. In this case, the detectors are easily got local trapped. Certain operations should be employed to maintain diversity of population in evolution process.

3.2 IGA-based detector generation algorithm

In order to enhance population diversity and increas otherness among detectors, an improved fitness function is applied in IGA-based algorithm. IGA-based detector generation algorithm is aimed at solving this problem. In this way, detector fitness calculation is altered into following way.

$$fitness(i)=\begin{cases}0+\dfrac{1}{sum}, & \min Self=0\\ 1-\max Self+\dfrac{1}{sum}, & otherwise\end{cases} \quad (3)$$

Where *sum* represents for the affinity summation among all detectors in premature detector set, which is defined as formula (4).

$$sum=\sum_{i=1}^{size}\sum_{j=1,j\neq i}^{size}Affinity(i,j) \quad (4)$$

According to fitness calculation, detector fitness increases with affinity sum decreases.

Both fitness calculations of GA-based and IGA-based are supposed to optimize the target as minimizing *maxSelf*. However according to experiment results, when *maxSelf* gets minimized, the matured detector set may not get highest TP and lowest FP. Because TP and FP are not supposed to be the optimization target in this fitness calculation formula, and when TP and *maxSelf* minimization aren't consistent, the final detector set may not be the optimized one.

3.3 Online GA-based detector generation algorithm

3.3.1 Online GA-based detector generation algorithm

Aiming at the inconsistent of optimization target and TP value in evolution of GA, a novel online GA-based detector generation algorithm is put forward. TP value and FP value are regarded as the optimization target in GA, and detector with higher TP value and lower FP value is the better one. The fitness calculated as follows:

$$fitness(i)=\sum_{t=1}^{target}D_t-\sum_{t=1}^{target}E_t \quad (5)$$

Where *Di* is attack detection, *Ei* is a false-positive error and *i* is the ID of detector.

With the evolution of generation, the whole population evolves toward higher fitness values, whose TP values are higher and FP values are lower. Each generation evolves based on TP value and FP value of a round detection. The flowchart is mended as follows.

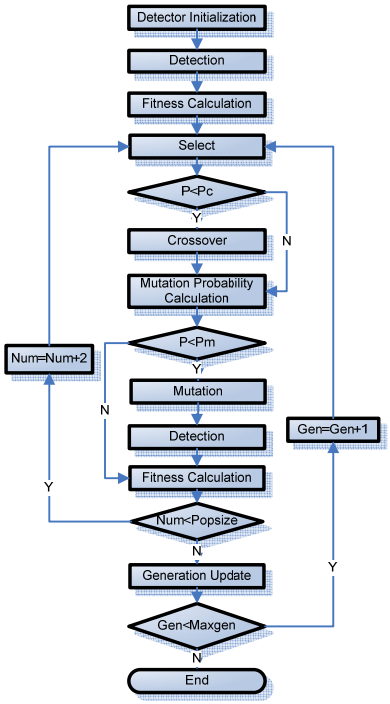

Fig. 2. Detector generation algorithm based on online GA

In *detection* block, the detector set takes detect on the target set by discriminating self and non-self, which realizes the detector mature online.

3.3.2 Self-adaptive mutation probability

Taking the population diversity into consideration, another effective way of controlling diversity is to adapt mutation probability [9-10]. In self-adaptive GA mutation probability is self-adaptive as formula (6).

$$P_m = \begin{cases} \dfrac{k_1(f_{\max} - f)}{f_{\max} - f_{avg}}, & f \geq f_{avg} \\ k_2, & f < f_{avg} \end{cases} \quad (6)$$

Where k_1 and k_2 are parameters set in prior, f_{max} and f_{avg} are maximum fitness and average fitness in each generation, and f is the fitness of detector. In this case the detector of higher fitness value f will takes low probability of mutation operation, while those of lower fitness value will much possible to take mutation operation.

3.3.3 Sample target online detection

Guaranteeing of real-time detection and the same time detector evolve, not all target can be detected for calculating fitness of detectors. A typical sample is needed as representatives. Because in actual network, all net packets are unpredictable, including safe net packets and intrusion detects. The detectors are evolved to be adaptive for detection. On basis of the above, the sample adopted here is randomly selected from target set. Every generation the sample set will be updated by new samples selected randomly from target set.

3.3.4 Self-adaptive affinity limit r

In affinity calculation, r is difficult to decide in NSA, here an adaptive method is adopted, in which r is related with maxSelf as follows.

$$r_i = \max Self_i + 1 \quad (7)$$

Where *maxSelf* is the maximum affinity value of *detector (i)* and self set. In this way, each detector has a unique affinity limit *r*. And no self number will be considered as non-self.

4. Experiments comparisons of nsa, ga-based, iga-based and online ga-based detector generation algorithms

4.1 Parameter settings and simulation data

Parameters settings in NSA.

Detector Size	6
Self Set Size	S1 = 8
	S2 = 32
r	[6,11]

Parameters settings in GA, IGA and online GA.

Population Size	6
Maximum Iteration	2000
Crossover Probability	0.5
Mutation Probability	self-adaptive
One-point crossover	
Multi-point mutation	

Self set is shown as follows.

Pattern	S1	S2
1111**********	2	4
****1111********	2	4
********1111****	2	4
**********1111	2	4

Self set is altered in dynamic during detector maturation process. Firstly set 1 is fixed as self set and then set 2 is added into self set. In this case, the adaptive performance of generation algorithms is testified and compared. Each simulation runs for 20 times and average values are recorded.

4.2 Simulation results

The simulation target in this experiment is binary string from 0x0000 to 0xffff exclusive of members in self set, which are supposed to be intrusions. TP values of detector generation algorithm based on NSA are shown in table 1.

Table 1. Average TP value of NSA for set1, set2 and dynamic set1, set2

r	Set 1	Set 2	Set 1+2
6	0	0	0
7	0.753	0	0
8	0.745	0.678	0.621
9	0.698	0.655	0.617
10	0.622	0.603	0.560
11	0.398	0.408	0.410

From the simulation results, several conclusions can be summarized.

(1) When r=6, it's impossible to generate matured detectors because affinity limit is too small that no available detector can be matured. Almost all of the randomly generated detectors have larger affinity than 6, which makes no detector available for detection. With the value of r gets larger, TP turns lower, in other word, the detector can only detect a small part of intrusions.

(2) We can see that it is especially important to fix a proper value for *r* in prior of detection, on which TP mainly depends. However it is difficult to evaluate *r* in various network systems, which makes an obvious disadvantage of NSA.

Simulation results of set 1 based on various algorithms are taken and compared which shown in following figures.

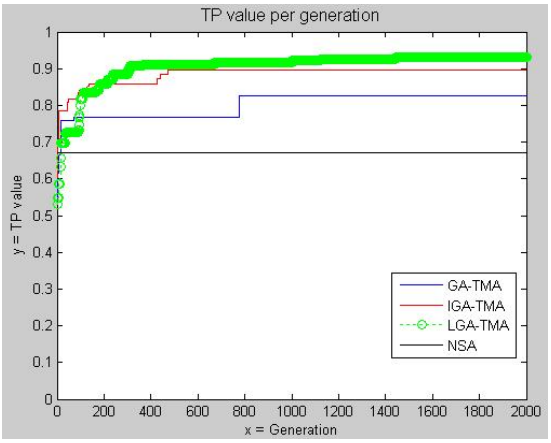

Fig. 3. TP value comparison of NSA, GA-based, IGA-based and online GA-based detector generation algorithm on set 1.

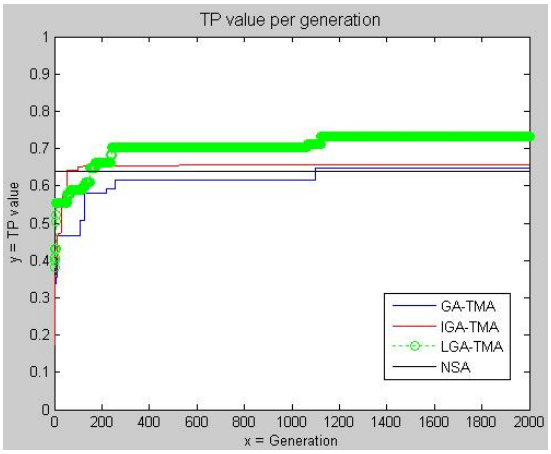

Fig. 4. TP value comparison of NSA, GA-based, IGA-based and online GA-based detector generation algorithm on set 2.

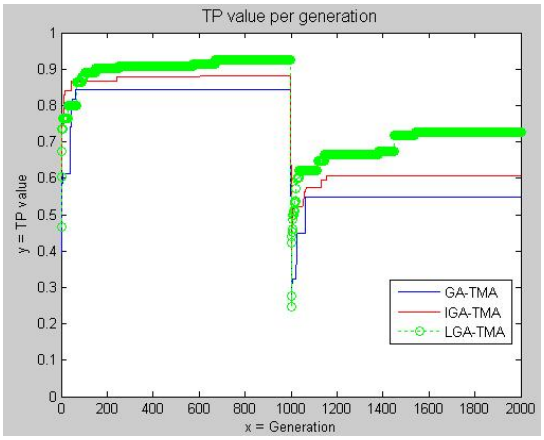

Fig. 5. TP value comparison of NSA, GA-based, IGA-based and online GA-based detector generation algorithm on set 1 and set 2 alternations in dynamic.

From the TP value comparison, it clearly shows that online GA-based algorithm gets best TP value in all three simulation experiments. In figure 5, changes happen in generation 1000, because the set 2 is added into self set, which makes TP drop suddenly. And NSA isn't suitable for this situation for the reason that it's static, self set of which cannot be altered, while other three algorithms are capable of dealing. Online GA-based algorithm has better performance on TP value compared with NSA is because the detectors in former algorithm evolve during the detection process while detectors in NSA are matured once regardless of detection performance, which is one of main advantages belong to online GA-based algorithm. When compared with GA-based and IGA-based algorithm, detectors in online GA-based algorithm have learning ability contributing to its fitness calculation function. Because fitness function in online GA-based algorithm is aimed at improving TP value and decreasing FP value directly while fitness function in GA-based and IGA-based algorithm don't have, which makes online GA-based algorithm has better performance than the other two.

FP is another index for evaluating performance of detector generation algorithm. FP value of NSA is recorded in table 2.

Table 2. Average FP value of NSA for set1, set2 and dynamic set1, set2

r	Set 1	Set 2	Set 1+2
6	0	0	0
7	1.068E-4	0	0
8	1.068E-4	2.121E-4	2.152E-4
9	8.392E-5	2.120E-4	2.075E-4
10	5.035E-5	1.511E-4	1.221E-4
11	3.662E-5	8.230E-5	8.392E-5

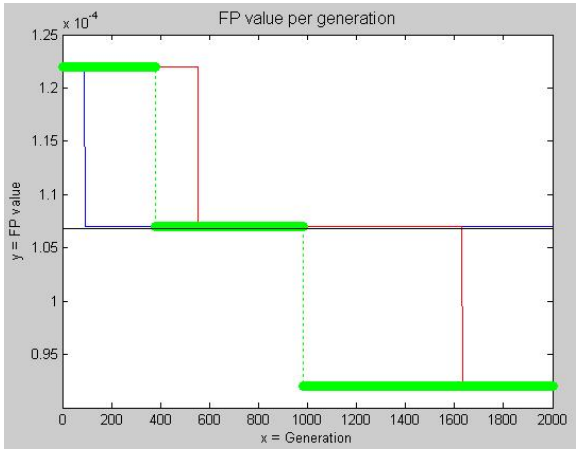

Fig. 6. FP value comparison of NSA, GA-based, IGA-based and online GA-based detector generation algorithm on set 1.

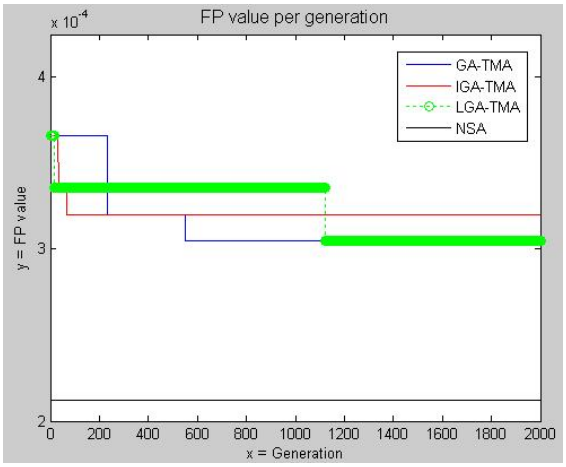

Fig. 7. FP value comparison of NSA, GA-based, IGA-based and online GA-based detector generation algorithm on set 2.

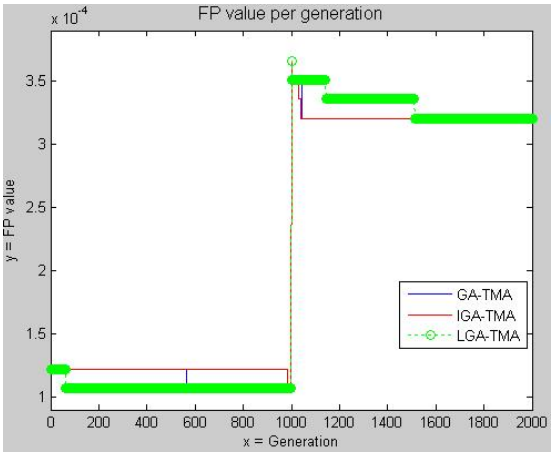

Fig. 8. FP value comparison of NSA, GA-based, IGA-based and online GA-based detector generation algorithm on set 1 and set 2 alternations in dynamic.

Compared with NSA, GA-based algorithms have much lower FP in intrusion detection based on self set 1. When self set is Set 2, FP value of NSA is the lowest among the four. Because NSA is a static generation algorithm which cannot deal with self change in dynamic. Figure 8 testifies that online GA-based algorithm achieves the lowest FP value among the three.

In aspect of CPU time cost, online GA-based algorithm has acceptable larger time complexity because it has complicated operations compared with other three algorithms. The time cost is listed in table 3, which is counted by second.

Table. 3. CPU time cost for algorithms

Algorithm	Set 1	Set 2	Set 1+2
NSA	0.109	0.109	0.125
GA-based	0.156	0.312	0.265
IGA-based	0.234	0.391	0.344
Online GA-based	2.121	4.010	4.658

5. Conclusion

In this paper, online GA-based detector generation algorithm is brought up whose fitness is ameliorated compared with GA and IGA-based algorithm. Fitness function of online GA-based algorithm is designed to evolve the whole population of detectors toward the direction of TP increase and FP decrease while the other two GA-based algorithms don't have. Online GA-based algorithm is compared with traditional NSA, GA-based and IGA-based algorithm in intrusion simulation. Experiment results prove that the novel online GA-based algorithm achieves best performance among these algorithms. However there is an acceptable shortcoming of online GA-based algorithm is that it cost more CPU time considering of complex operations and online check. On the whole, online GA-based detector generation algorithm provides a novel thought that detectors evolve online during detection process with self-adaptive affinity limit *r*, which is adaptive for online learning. Further research is still needed for cut short of CPU time.

6. References

[1] Forrest S, Perelson A S, Allen L, et al. Self non-self discrimination in a computer[A]. In Proceedings of the 1994 IEEE Symposium on Research in Security and Privacy[C]. Los Alamos CA IEEE Computer Society Press, 1994.

[2] D'haeseller P, Forrest S. An immunological approach to change detection: Algorithm, analysis and implication. In: Proc. of the IEEE Symp. On Research in Security and Privacy. Oakland: IEEE Computer Society Press, 1996.110-119.

[3] Fabio Gonzalez, Dipankar Dasgupta, Luis Fernando Nifio, A randomized real-valued negative selection algorithm, In Proceedings of the 2nd International Conference on Artificial Immune Systems, pp 261-272, Edinburgh, UK, September 2003.

[4] Ji Z, Dasgupta D. Real-valued negative selection algorithm with variable-seized detectors[C], Genetic and Evolutionary Computation. Seattle: IEEE Press, 2004: 287-298.

[5] Wenjian Luo, Xin Wang, Ying Tan, Xufa Wang, A novel negative selection algorithm with an array of partial matching lengths for each detector, PPSN IX, LNCS 4193, pp. 112-121, 2006.

[6] Mohammad Reza Ahmadi, Davood Maleki, A co-evolutionary immune system framework in a grid environment for enterprise network security, SSI 2006, November 9th, pp. 1136-1143, 2006.

[8] Jungan Chen, Feng Liang, Dongyong Yang, Dynamic negative selection algorithm based on match range model. LNCS, Volume 3809, 2005.

[9] Y. J. Cao, Q. H. Wu, Convergence analysis of adaptive genetic algorithms, Genetic algorithm in engineering systems: Innovations and applications, 2-4 September 1997, Conference publication No. 446, IEEE, 1997.

[10] Katja L, Rainer B, Tansu A, Achim M, Sahin A, A cooperative AIS framework for intrusion detection, IEEE Comminication society subject matter experts for publication in ICC 2007 proceedings, 2007, pp. 1409-1416.