An Intrusion Detection Approach Using An Adaptative Parameter-Free Algorithm

Mourad Daoudi, Mohamed Ahmed-Nacer

Electronics and Computer Science Faculty, Laboratory LSI, USTHB

BP 32 16111 El Alia, Bab-Ezzouar, Algiers, Algeria

m-daoudi@usthb.dz, anacer@cerist.dz

***Abstract*—In intrusion detection from audit security, the volume of data generated by the auditing mechanisms of current systems is very large. It is important to provide security officers with methods and tools to determine predefined attack scenarios in the audit trails. The problem is Non-deterministic Polynomial-time hard (NP-hard). Metaheuristics offer an alternative to solve this type of problems. Unfortunately, many parameters have to be tuned for any metaheuristic, and their values may have a great influence on the efficiency and effectiveness of the search of good solutions. The exploration of an optimal combination of such values may be difficult and big time consuming. Clerc et al. have defined an adaptative parameter-free algorithm, called TRIBES, issued from Particle Swarm Optimization. It is developed to solve continuous problems. In this paper, we propose to adapt TRIBES to solve our combinatorial optimization intrusion detection problem. Modifications in different mechanisms and formulae adaptations in original TRIBES are made, like in the generation process of the particles and in the displacement strategies. The experimentations results show the good behavior of our approach. Comparisons with a basic genetic algorithm are provided.**

Keywords-Metaheuristics; NP-hard Combinatorial Optimization Problems; Particle Swarm Optimization; TRIBES; Genetic Algorithm; Security Audit.

I. INTRODUCTION

An important problem encountered when dealing with metaheuristics [1], [2] is the determination of good parameters values. Indeed, many parameters have to be tuned for any metaheuristic, and they may have a great influence on the efficiency and effectiveness of the search. The optimal values for the parameters depend mainly on the problem and even the instance to deal with and on the search time that the user wants to spend in solving the problem. A universally optimal parameter values set for a given metaheuristic does not exist, and it is not obvious to define which parameter setting should be used. Moreover, in real problems, the parameters are often correlated, which makes the choice of parameters harder. The difficulties still remain when using algorithms with a lower number of parameters. These algorithms are called *adaptive* algorithms, because the information gradually collected during the optimization process are used to determine the values of the parameters.

Particle Swarm Optimization (PSO) [3], [4], [5] is a stochastic population-based metaheuristic of biological inspiration. Several adaptive methods have already been defined for PSO [6], [7], [8], [9]. A parameter free metaheuristic, called *TRIBES* [10], [11], has been developed and showed good results. It was designed to act as a black-box and the user has just to define his problem and to run the process. Such an algorithm exists among genetic algorithms [12]. Several metaheuristics of combinatorial origin are however adapted to the continuous case like in PSO and TRIBES.

The problem under study: “Security Audit Trail Analysis Problem” (SATAP) is of discrete nature [13]. Therefore, we reconsider different mechanisms in TRIBES like the generation process of a particle or the displacement strategies developed in the structural and behavioral adaptations, so that they can be used, with the definition of a distance in the search space to solve our problem.

This paper is organized as follows: after an introduction, the parameter free metaheuristic TRIBES is presented in Section II. Section III describes the “Security Audit Trail analysis problem”. Section IV is dedicated to our contribution, adaptating TRIBES to solve SATAP problem. We provide some experimentation results that show the performances of our approach in Section V, where comparisons to a basic genetic algorithm-based approach are also given.

II. TRIBES

In PSO, each particle moves in the search space and updates its velocity according to best previous positions already found by its neighbors (and itself), trying to find an even better position. This approach has been proved to be powerful but needs parameters predefined by the user, like swarm size, neighborhood size, and some coefficients, and tuning these parameters for a given problem may be, in fact, quite difficult.

In TRIBES [11], the swarm is divided in tribes of individuals. Each tribe acts as an independent swarm, i.e., each tribe has its own global behavior and explores a particular region of the search space. All the tribes exchange information about regions they are exploring. As shown in Fig.1, the swarm is presented as an interconnected network of tribes, which are themselves interconnected networks of individuals. Each particle is informed by itself (cognitive memory P, by all the other elements of its tribe, called internal informers) and if this particle is the best particle of the tribe, called a shaman, then it is also informed by the other tribes’ shamans (called external informers). The social memory, noted g, of a particle is the informer with the smaller value of the objective function.

The swarm must be generated and modified automatically, by means of creation, evolution, and removal of the tribes without defining any parameters. To make this possible, rules have been set up. Each particle has a current position and a best position, and then, particles and tribes' qualifiers are defined. A particle is said to be good or neutral: good if it has just improved its best performance, neutral, if not. As for a tribe, it is said to be good, neutral or bad, the larger the number of good particles in the tribe the better the tribe itself. In TRIBES, particles are added or removed according to tribes behaviors. Structural and behavioral adaptations of the swarm may then occur.

A. *Swarm's structural adaptations*

Bad particles are removed from good tribes. The removal of a particle implies a change in the information network. All the informer particles of a removed particle are directed towards the tribe's best particle. Particles are generated by bad tribes and form a new tribe. The bad tribe will keep contact with the new tribe and will try to use it to improve its best location.

Two types of particles are generated, free particles and confined particles, given that the particle type is randomly selected.

- Free particles: The particle is randomly generated according to a uniform distribution either in the whole search space, or on a side of the search space or on a vertex of the search pace. Once the method is selected (at random), then the particle is generated as follows:

$$X_j = \mathrm{U}(x_{\min(j)}, x_{\max(j)}), j \in \{1, \dots, D\} \qquad (1)$$

$$X_j = \begin{cases} \mathrm{U}(x_{min(j)}, x_{max(j)}), j \in I \subset \{1, \dots, D\} \\ \mathrm{U}(x_{min(j)}, x_{max(j)}), j \in J \subset \{1, \dots, D\} \end{cases} \qquad (2)$$

$$X_j = \mathrm{U}(\{x_{\min(j)}, x_{\max(j)}\}), j \in \{1, \dots, D\} \qquad (3)$$

  where $\mathrm{U}(x_{\min(j)}, x_{\max(j)})$ is a real number uniformly chosen in $[x_{min(j)}, x_{max(j)}]$ and $\mathrm{U}(\{x_{\min(j)}, x_{\max(j)}\})$ is a real number uniformly chosen in the list $\{x_{min(j)}, \dots, x_{max(j)}\}$. The two sets I and J are randomly defined for each particle and determine a partition of $\{1 \dots D\}$. Generating free particles aims at provide diversity in the population.

- Confined particles: They aim at intensify research inside an interesting area. Let X_i be the best particle of the generating tribe and i_x its best informer, and let P_x and P_{ix} be the best locations of X_i and i_x. The new particle will be generated in the D-sphere of center P_{ix} and radius $|| P_x - P_{ix} ||$ such that:

$$X_{generated} = alea_{sphere}(P_{ix}, || P_x - P_{ix} ||) \qquad (4)$$

  where $alea_{sphere}(P_{ix}, || P_x - P_{ix} ||)$ is uniformly chosen in the hyper-sphere of center P_{ix} and radius $|| P_x - P_{ix} ||$.

B. *Swarm behavioral adaptations*

The second way in view of adapting the swarm to the results found by the particles is to choose the strategy of displacement of each particle according to its recent past. Between two iterations, a particle can improve its performance, denoted by **(+)**, or can deteriorate it which is denoted by **(-)**. There can be no change as well, namely a status quo that is denoted by **(=)**. Then, the choice of the strategy of displacement is made according to the two last variations as shown in the following table:

TABLE I. STRATEGIES OF DISPLACEMENT IN TRIBES

Gathered status	**Strategy of displacement**
(= +) (++)	*Local by independent Gaussians*
(+ =) (- +)	*Disturbed pivot*
(- -) (= -) (+ -) (= =)	*Pivot*

- The pivot strategy is inspired from published works as in [13]. Let p be the best position ever reached by the particle and let g be its best informer and f the objective function. Then, the displacement is determined by:

$$X = C_1 * alea_{sphere}(H_p) + C_2 * alea_{sphere}(H_g) \quad (5)$$

  where $C_1 = \frac{f(p)}{f(p)+f(g)}$, $C_2 = \frac{f(g)}{f(p)+f(g)}$, $alea_{sphere}(H_p)$ is uniformly chosen in the hyper-sphere of center p and radius $\|p-g\|$ and $alea_{sphere}(H_g)$ is uniformly chosen in the hyper-sphere of center g and radius $\|p-g\|$.

- The disturbed pivot strategy is similar to the precedent one, adding a noise.
  In practice, for each component of the last computed position, a random number b is generated using a centered Gaussian distribution with standard deviation $\frac{f(p)-f(g)}{f(p)+f(g)}$. Then the component is multiplied by $(1+b)$.

- In local by independent Gaussians strategy, if g = $(g_1, .. g_D)$ is the particle best informer, then the displacement is determined by:

$$x_j = g_j + alea_{normal}(g_j - xj, \|g_j - x_j\|), j \in \{1, \dots, D\} \quad (6)$$

  where $alea_{normal}(g_j - x, \|g_j - x_j\|), j \in \{1, \dots, D\}$ is a point randomly chosen with a Gaussian distribution with mean $(g_j - x_j)$ and standard deviation $\|g_j - x_j\|$.

III. SECURITY AUDIT TRAIL ANALYSIS PROBLEM

A. *Introduction*

Computer security has become in recent years a crucial problem [14]. It rallies the methods, techniques and tools used to protect systems, data and services against the accidental or intentional threats, to ensure: Confidentiality; Availability, and Integrity. Nowadays, different techniques and methods have been developed to implement a security policy: authentication, cryptography, firewalls, proxies, antivirus, Virtual Private Network (VPN), and Intrusion Detection System (IDS) [15].

IDSs are software or hardware systems that automate the process of monitoring the events occurring in a computer system or network, analyzing them for signs of security problems [16-19]. The intrusion detection system was introduced by James Anderson [20], but the subject didn't have great success. After that, Denning defined the intrusion detection system models [21], where he exhibits the importance of security audit, with the aim to detect the possible violations of system security policy.

According to Intrusion Detection Working Group of IETF an intrusion detection system includes three vital functional elements: information source, analysis engine, and response component [22]. There are five concepts to classify intrusion detection Systems, which are: The detection method; The behavior on detection; The audit source location; The detection paradigm; The usage frequency.

The detection method is one of the principal characters of classification they describe the characteristics of the analyzer. When the intrusion detection system uses information about the normal behavior of the system it monitors, we qualify it as behavior-based. When the intrusion detection system uses information about the attacks, we qualify it as knowledge-based.

The Security Audit is as medical diagnosis, in order to determine the set of conditions, which may explain the presence of observed symptoms (in IDS: the recorded events in the audit trail). For this reason, expert uses specific knowledge (the scenarios of attack) based cause at an effect. The expert uses its knowledge to develop assumptions that confront the reality observed. If there are still observed symptoms than the made hypothesis made is wrong. On the other hand, if there are more symptoms than those observed in the reality, a new hypothesis more relevant must be tested [23].

B. *Specifications*

We want to achieve a system that can tell whether an intrusion has occurred or not when analyzing the trace file security audit. Among the different existing approaches to develop such a system, we consider a posterior approach based on attack scenarios recorded in the audit trail. To establish these different scenarios (different attacks or attack signatures), we first define a certain number of so-called auditable event. The selection of auditable events in a real case is left to the administrator. Thus, each attack will be defined by the number of occurrence of auditable events. The audit file for analysis will also be defined by the number of occurrence of auditable events. The temporal order of sequence of events will not be considered (in this case the system can function in a heterogeneous distributed environment where the construction of a common time is impossible). In informal terms, the problem is to find the combination of attacks that maximizes the incurred risk, while possible under number of events of each type recorded in the audit file.

In this approach, the attack scenarios are modeled as a set of couples (e, N_e), where e is the type of event and N_e is the number of occurrences of this type of event in the scenario.

Formally, SATAP can be expressed by the following [22]:

$$\begin{cases} Max \sum_{1}^{Na} R_j.H_j \\ \\ (AE\ .\ H)_i \le O_i,\ i \in \{1 \ldots N_e\} \end{cases} \tag{7}$$

where:

- N_e : the number of type of audit events
- N_a : the number of potential known attacks
- AE : a N_e x N_a attack-events matrix which gives the set events generated by each attack. AE_{ij} is the number of events of type i generated by the attack j, ($AEi\ j \geq 0$)
- R : a N_a-dimensional weight vector, where R_i ($R_i > 0$) is the weight associated with the attack i (R_i is proportional to the inherent risk in attack scenario i)
- O : a N_e-dimensional vector, where O_i is the number of events of type i present in the audit trail (O is the observed audit vector)
- H : a N_a-dimensional hypothesis vector, where $H_i = 1$ if attack i is present according to the hypothesis and $H_i = 0$ otherwise (H describes a particular attack subset).

To explain the data contained in the audit trail (i.e., O) by the occurrence of one or more attacks, we have to find the H vector which maximizes the product RxH (it is the pessimistic approach: finding H so that the risk is the greatest), subject to the constraint $(AE.H)_i \le O_i$, $1 \le i \le N_e$.

Because finding H vector is NP-complete, the application of classical algorithm is impossible where N_a equals to several hundreds.

The heuristic approach that we have chosen to solve that NP-complete problem is the following: a hypothesis is made (e.g., among the set of possible attacks, attacks i, j and k are present in the trail), the realism of the hypothesis is evaluated and, according to this evaluation, an improved hypothesis is tried, until a solution is found.

In order to evaluate a hypothesis corresponding to a particular subset of present attack, we count the occurrence of events of each type generated by all the attacks of the hypothesis. If these numbers are less than or equal to the number of events recorded in the trail, then the hypothesis is realistic.

To derive a new hypothesis based on the past hypothesis, several approaches have been proposed by researchers, including Genetic Algorithms [22-25] and Biogeography Based Optimization [26], [27].

IV. SECURITY AUDIT TRAIL ANALYSIS USING TRIBES ALGORITHM

The approach aims to determine if the events generated by a user correspond to known attacks, and to search in the audit trail file for the occurrence of attacks by using a heuristic method because this search is an NP–complete problem. The goal of the heuristic used is to find the hypothesized vector H that maximizes the product R*H, subject to the constraint $(AE.H)_i \leq O_i$,$1\leq i \leq N_e$, where R is a weight vector that reflects the priorities of the security manager, AE is the attack-events matrix that correlates sets of events with known attacks, and N_e the number of types of audit events.

TRIBES is an optimum search algorithm issued from PSO. A swarm is divided into tribes of artificial particles. Each particle changes its position, moving from a position to an other. These positions are strings of length l coding a potential solution to the problem to be solved. We are in a situation where the coding of positions is immediate since the solution of the problem is expressed specifically in the form of a binary sequence. A position is considered as a series of N_a integers with values 0 or 1. Each position is a particular instance of the vector H. In other words, the element i in the position will be 1 if the position is a solution in which the attack i is declared present. Otherwise, this element takes the value 0. We note that the sum of the component elements in a position indicates the number of attacks that are detected.

The TRIBES-based method should return a binary N_a – vector $H = (H_1, \ldots, H_{Na})$, where the value $H_i = 1$ indicates the presence of the attack i in the audit file O and 0 its absence. As we have to solve a maximization problem, the best solution is associated with the hypothesis H of larger value of the selective function

$$F(H) = R*H = \sum_{1}^{Na} R_i\, H_i \qquad (8)$$

which represents the total risks incurred by the system under surveillance.

In addition, as we deal with a constrained problem, any solution to the problem must verify the inequalities: $(AE\times H)_I \leq O_i$ avec $0<i<N_e$. Thus, we have to eliminate the solutions that do not comply, setting to zero the value of the corresponding objective function (that is a harmony in which each note has a zero value). There combination process is repeated until a solution satisfying the constraints is generated. Recall that R_i is the weight of the attack i, that is the risk incurred to the system if the attack is not detected. For simplicity, the R_i value is taken equal to 1 for all i, i=1 … N_a. In this case, the objective function F resumes in computing the number of detected attacks.

We recall here that TRIBES is initially defined to solve continuous problems. The problem under consideration in our paper is an NP Hard combinatorial problem. Therefore adaptations of TRIBES are needed to solve our SATAP problem.

Let us define first a distance in our particular search space of binary N_a-vectors. We consider the well known Hamming distance, well suited to our problem. If *x* and *y* are two positions, then we note: *dist(x, y).*

After these specifications, we are now able to present the different changes that occur in the structural and behavioral adaptation mechanisms in our approach.

A. *Swarm's structural adaptations*

As defined in original TRIBES, the swarm is an interconnected network of tribes of different size, which are themselves interconnected networks of particles. In intra-tribe communication, each particle is informed by all the other elements of its tribe, and in inter-tribe communication, links are specified at their generation; each new tribe keeps contact with its generating tribes.

Because of the discrete solutions search space, the generation process of the free particles and confined particles seen in Section II is modified.

- The free particles are binary random vectors, randomly generated in the whole search space. If $X=(x_1,.., x_D)$ denotes the position of a particle, x_i is 1 or 0, indicating that the ith attack is detected or not.
- Let x be the best particle of the generating tribe and i_x its best informer, and let P_x and P_x be the best locations of x and i_x. Then, the confined particle, *Xnew* is such that: $dist(p_{ix}\ ,\ Xnew\ ) \leq d$ where $d=dist(p_{ix}\ ,\ p_x)$. To do so, we generate randomly an integer *N* between *0* and *d*, and then proceed randomly to *N* changes in P_x. The obtained new binary vector is the current position of the created new particle.

B. *Swarm's behavioral adaptations*

1) Pivot strategy: The pivot method concerns particles with bad behavior in the last two iterations. The method given in the continuous case is still maintained in its principle. However, the new position is determined from *p* et *g* using the distance **dist** to create their neighborhoods in place of the two hyper-spheres H_p and H_g. We recall that *p* is the best position of the particle and *g* the best position of the informers of the particle, and f the objective function.

The proposed procedure is:

a) Compute the distance dist(p, g). Let d its value.

b) Generate two uniformly distributed random integer numbers u_1 and u_2 in (0, d)

c) Generate two uniformly distributed random Na-vectors X and Y such that: dist(X, p) = u_1 and dist(Y, g) = u_2

d) Compute the attraction coefficients:

$$c_1 = \frac{f(\vec{p})}{f(p)+f(g)} \text{ and } c_2 = \frac{f(g)}{f(p)+f(g)}$$

e) For each element i, $0 \leq i \leq N_a$, of the new position vector Xnew, generate a random number u_3 , from a Bernoulli distribution of probability c_1.

If u_3 =1, then Xnew(i)=X (i)

If u_3 =0, then Xnew(i)=Y(i)

2) Disturbed Pivot strategy: The disturbed pivot method concerns particles with medium performances in the last two iterations. The proposed method preserves the original disturbed pivot.

The proposed procedure is :

a) Generate a position X' using the adapted Pivot method seen above.

b) Generate a random number *b* from a centered Gaussian distribution with standard deviation: $\frac{f(p)-f(g)}{f(p)+f(g)}$

c) Associate a number of k components of *X* to b,

d) Choose randomly k elements in *X***.** Let $X_{i1},\ldots, X_{ik}$ these elements.

If $X_{ij} = 1$ then *Xnew*(ij)=0, and

If $X_{ij} = 0$ then *Xnew*(ij)=1.

3) Local strategy with independent Gaussians: In this strategy, the principle is to intensify the search around the best informer $\vec{g}$. So, the neighborhood $\vec{g}$ will be determined using the distance d defined earlier. Then, a neighbor is generated from a Gaussian distribution as follows:

a) Compute the distance dist(X, g), *X* is the current position.

b) Generate an integer random number k from a centered Gaussian distribution with standard deviation dist(X, g).

c) Choose randomly k elements in *X*. Let $X_{i1},\ldots, X_{ik}$ these elements.

If $X_{ij} = 1$ then Xnew(ij) = 0, and

If X_{ij}=0 then Xnew(ij)=1.

Remark: The steps given earlier in TRIBES and not redefined in the present section, still remain unchanged.

V. EXPERIMENT RESULTS

A. Adapted-TRIBES approach

During the simulations, all the attacks actually present in the analyzed audit file must be known in advance. Thus, the events corresponding to one or more attacks are included in the observed audit vector O.

Ratios TPR, FPR, Accuracy and Precision [28] are used to evaluate our approach intrusion detection quality, with:

- TPR (True positive rate): TP / (TP+FN)
- FPR (False positive rate): FP / (TN+FP)
- Accuracy: (TN+TP) / (TN+TP+FN+FP)
- Precision: TP / (TP+FP)

where True negatives (TN) as well as true positives (TP) correspond to correct intrusion detection: that is, events are successfully labeled as normal and attacks, respectively. False positives (FP) refer to normal events being predicted as attacks; false negatives (FN) are attack events incorrectly predicted as normal events.

To evaluate our approach, many tests were performed using Attack-Events matrices of different sizes. All results are obtained as the average of 10 executions carried out for the same data and same number of injected attacks.

The following figures show the quality of our intrusion detection approach.

Reported results in Fig. 1 concern tests performed on an attack-events matrix of size (28x24) issued from [23], with 24 attacks and 28 types of events, and 15 attacks are injected.

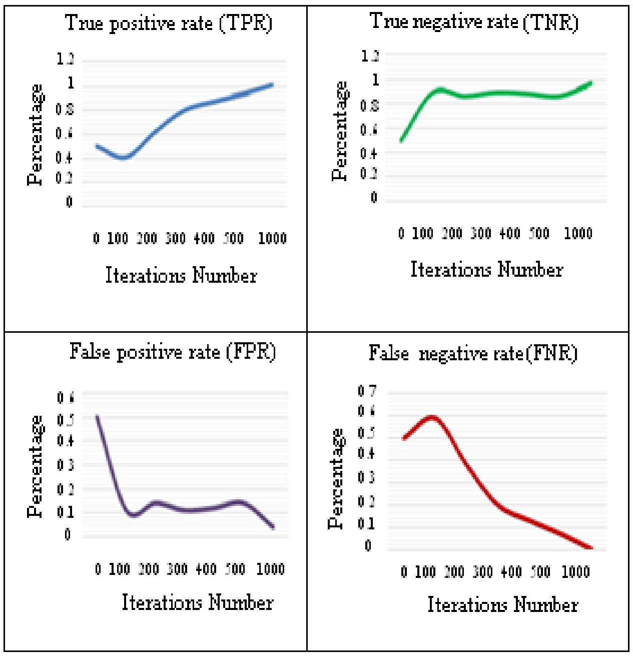

Figure1. Intrusion Detection Quality Measures: (AE: 24x28)

Fig. 2 shows results of tests performed on larger data, randomly generated, with an attack-events matrix of size (100x200), and 200 injected attacks.

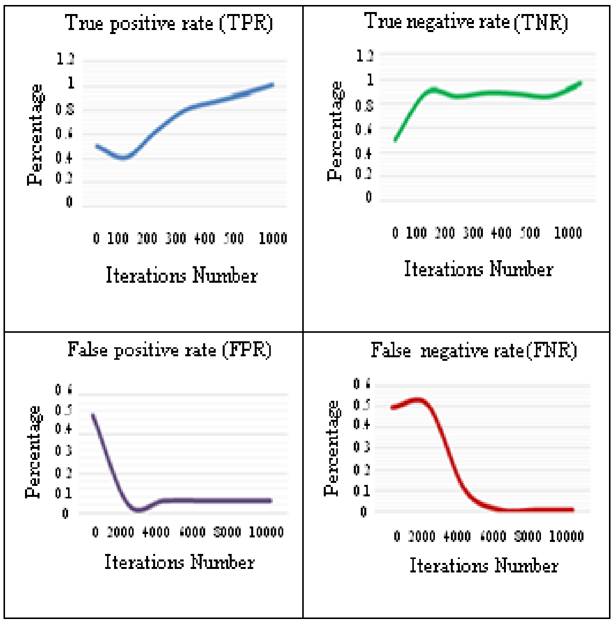

Figure 2. Intrusion Detection Quality Measures: (AE: 100x200)

We observe that after a certain number of generations, all injected attacks are detected, and no false attack (TPR= 1 and FPR =0). In addition, the number of attacks injected has no influence on these results. Indeed, we observe in both figures Fig. 1 and Fig. 2 that all attacks are detected (TPR=1 and FPR=0) after 1000 iterations (respectively 10000). Further, no false positive nor false negative attacks are detected (FNR.= FNR = 0) after 1000 iterations (respectively 10000).

B. TRIBES vs. Genetic Algorithms

An intrusion detection approach using Genetic Algorithms (GA) has been developed and tested in order to compare the results with those obtained when using our Tribes based approach.

A genetic algorithm handles a group formed of a population of individuals, of constant size, initially randomly generated. This constant size induces a competition between individuals representing the potential solutions to the problem at hand. The population evolves in successive generations. The strongest individuals survive and reproduce to create new individuals, while the others are gradually disappearing. To enable this change in population, genetic operators have been defined such as selection, crossover and mutation. However, to evaluate the different individuals in a population and allow differentiating between a "strong" individual and a "weak" one, a so-called selective function (fitness) is used, to associate a value on each individual in the population.

Results reported in Fig. 3 concern a (28x24) - Attack-Events matrix issued from [23].

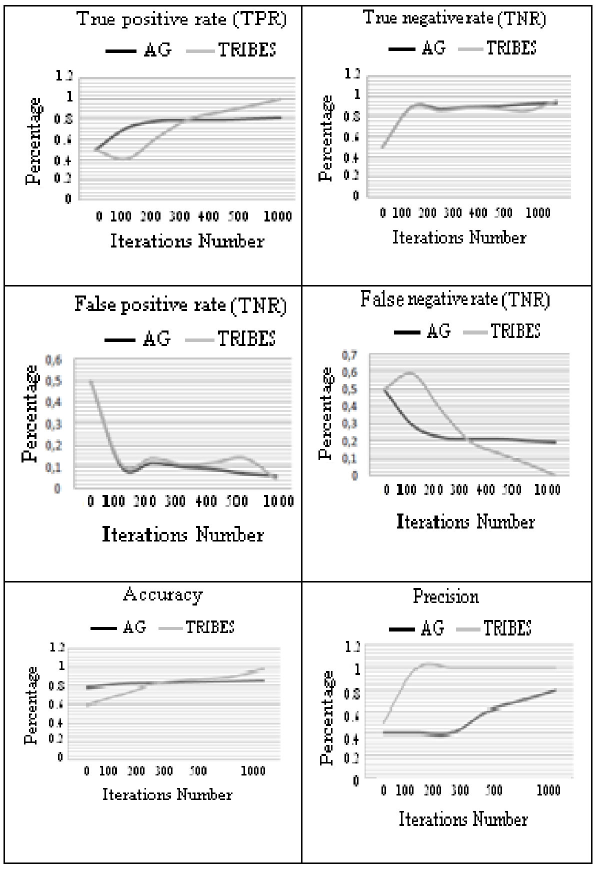

Figure 3. Tribes vs. AG: Intusion Detection Quality Measures (AE-matrix: 24x28)

The number of injected attacks is 15. We observe that all attacks are detected in both approaches. We note that GA detected the attacks after more than 1000 iterations, and obtained more false negatives.

Table 2 shows the different parameters involved in GA and their values obtained through simulation.

TABLE II. GENETIC ALGORITHM PARAMETERS SET

Population Size	500
Generation Number	1000
Mutation rate	0.01
Crossover rate	0.5

Now, let us consider the execution time for both methods. Tests are performed on the randomly generated attack-events matrix of size (100x200), and 200 injected attacks (used in section A above). Results are reported in Fig. 4.

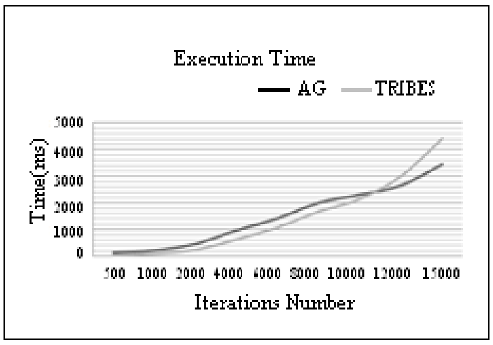

Figure 4. TRIBES vs. AG: Intrusion Detection Execution Time (AE-matrix: 100x200)

We observe that TRIBES-based method presents higher execution time with the number of iterations.

5. CONCLUSION AND FUTURE WORK

Security Audit trail Analysis can be accomplished by searching audit trail logs of user activities for known attacks. The problem is a combinatorial optimization problem NP-Hard. Metaheuristics offer an alternative for solving this type of problem when the size of the database events and attacks grow. We proposed to use an adaptative parameter-free algorithm as detection engine. Originally conceived to solve continuous problems, we had to we reconsider different mechanisms in TRIBES like the generation process of a particle or the displacement strategies developed in the structural and behavioral adaptations, so that they can be used, with the definition of a distance in the search space to solve our NP-Hard combinatorial optimization SATAP problem.

Experimental results of simulated intrusions detection are given. The effectiveness of the approach is evaluated by its ability to make correct predictions. It proved to be effective and capable of producing a reliable method for intrusion detection.

Comparisons with Genetic Algorithms inspired approach are provided, showing for our approach a good behavior. However, these systems are usually developed for predefined environments and do not offer a solution to some network characteristics such as changes in behavior of users and services, the increasing complexity and evolution of the types of attacks that they may be subject, the speed of attacks that can occur simultaneously on several machines, etc.

REFERENCES

[1] E. Talbi, Metaheuristics from Design to Implementation, John Wiley Sons, 2009.

[2] J. Dreo, A. Petrowski, P. Siarry, and E Taillard, Metaheuristics for difficult optimization, Eyrolles, Paris, 2003.

[3] J. Kennedy and R. C. Eberhart, "Particle Swarm Optimization", IEEE International Conference on Neural Networks, vol. 4, pp. 1942-1948, Perth, Australia, 1995.

[4] M. Clerc, Particle Swarm Optimization. Hermes Science, 2005.

[5] M. Clerc, Particle Swarm Optimization, International Scientific and Technical Encyclopedia, 2006

[6] V. Miranda and N. Fonseca, "Evolutionary self-adapting Particle Swarm Optimization", INESC, Oporto, Portugal, 2002.

[7] X.F. Ye, W.J. Zhang and Z.L. Yang, "Adaptive Particle Swarm Optimization on Individual Level", Proc. International Conference on Signal Processing (ICSP), Beijng, China, IEEE Press, 2002, pp. 1215-1218.

[8] W. Zhang, Y. Liu and M. Clerc, "An adaptive PSO algorithm for real power optimization", Proc. APSCOM (partI), 2003, pp. 302-307, IEEE Press. 2003.

[9] K. Yasuda and N. Iwasaki.. _Adaptive Particle Swarm Optimization using velocity information of swarm_, Proc. IEEE Conference on System, Man and Cybernetics, 2004, pp. 3475-3481, IEEE Press.

[10] M. Clerc, TRIBES - Un exemple d'optimisation par essaim particulaire sans paramtres de contrle. In: OEP 2003, Paris, France 2003.

[11] Y. Cooren, M. Clerc, and P. Siarry, "Initialization and Displacement of the Particles in TRIBES, a Parameter-Free Particle Swarm Optimization Algorithm", Springer, Studies in Computational Intelligence, 2008, vol. 136, pp. 199-219.

[12] H. Sawaih and S. Kizus, "Parameter-free Genetic Algorithm inspired by disparity theory of evolution", Congrès PPSN V : Parallel problem solving from nature. International conference 1998 , vol. 1498, pp. 702-711.

[13] L. Mé, Audit de sécurité par algorithmes génétiques. Thèse de Doctorat de l'Institut de Formation Supérieure en Informatique et de Communication de Rennes, 1994.

[14] B. C. Tjaden, Fundamentals of secure computer systems. Franklin and Beedle & Associates, 2004.

[15] E. Cole, R. K. Krutz, and J. Conley, Network security bible, Wiley Publishing, 2005.

[16] R. Bace and P. Mell, NIST Special publication on intrusion detection systems, 2001.

[17] P. Roberto and V. Luigi, Intrusion detection systems, Springer, Advances in Information Security, 2008, vol. 38, ISBN: 978-0-387-77265-3.

[18] H. Debar, M. Dacier, and A. Wespi, "A revised taxonomy for intrusion detection systems". Annales des Telecommunications, 55(7-8), 2000.

[19] R. G. Bace, Intrusion detection systems. Mac Millan Technique Publication, USA, 2000.

[20] J. Anderson, Computer security threat monitoring and surveillance. Technical Report 79F296400, James Anderson, Co., Fort Washington, PA, 1980.

[21] D. Denning, "An intrusion detection model", Proc.of the 1986 IEEE Symposium on Security and Privacy, pp. 118-131, 1987.

[22] A. Ahmim, N. Ghoualmi, and N. Kahya, "Improved off-line intrusion detection using a genetic algorithm and RMI". Proc. of the International Journal of Advanced Computer Science and Applications (IJACSA), vol. 2, no. 1, 2011.

[23] L. Mé, "GASSATA, A genetic algorithm as an alternative tool for security audit trails analysis", Proc. of the 1st International Workshop on the Recent Advances in Intrusion Detection (RAID 98). Louvain-la-Neuve, Belgium, pp. 14–16, 1998.

[24] P. A. Diaz-Gomez and D. F. Hougen, "Improved off-line intrusion detection using a genetic algorithm". Proc. of the seventh International Conference on Enterprise Information Systems, pp. 66-73, 2005.

[25] P. A. Diaz-Gomez and D. F. Hougen, "A genetic algorithm approach for doing misuse detection in audit trail files, Proc. of International Conference on Computing (CIC-2006), pp. 329-335, 2006.

[26] M. Daoudi, A. Boukra, and M. Ahmed-Nacer, "A biogeography inspired approach for security audit trail analysis. Journal of Intelligent Computing, vol. 2, no. 4, December 2011.

[27] M. Daoudi, A. Boukra, and M. Ahmed-Nacer, "Security audit trail analysis with biogeography based optimization metaheuristic". Proc. of the International Conference on Informatics Engineering & Information Science: ICIES 2011, A. Abd Manafet al. (Eds.): ICIEIS 2011, Part II, CCIS 252, pp. 218–227, 2011.© Springer-Verlag Berlin Heidelberg; 2011.

[28] S. X. Wu and W. Banzhaf, "The use of computational intelligence in intrusion detection systems: A review". Elsevier, Applied Soft Computing 10, 2010, pp. 1-35, doi: 10.1016/j.asic.009.06.015.