A security incidents management for a CERT based on Swarm Intelligence

JOSE AGUILAR, BLANCA ABRAHAM, GILBERTO MORENO
CEMISID, Departamento de Computación
Facultad de Ingeniería, Universidad de Los Andes
La Hechicera, Mérida
VENEZUELA
aguilar@ula.ve

Abstract: - This paper proposes an incident management system based on the swarm intelligence in order to keep updated the information about computational security. Particularly, swarm intelligence is an extension of the multiagents theory, where reactive agents follow very simple rules. We propose a search and selection method based on swarm intelligence, where the agents search previous answers to incidents of security on Internet.

Key-Words: - Incident Management System, Swarm Intelligence, Security Systems, CERT, Search Algorithm, Multiagent Systems.

1 Introduction

At this moment, the security is a very important factor in all the computational environments. Since any connected device to Internet has the risk of being attacked from computers, servers, PDA's, routers, cellular telephones, etc., it is necessity that they are protected. We need quick answer to security incidents, to avoid irreversible losses [6, 8, 9]. Based on the above, has been created groups called CERT (Computational Emergencies Reponses Team), distributed to world-wide level. At the end of the years 80, the University of Carnegie Mellon created the first CERT, a group formed in their biggest part by qualified volunteers of the computer community whose main objective was to facilitate a quick answer to the problems of security that affected Internet [1, 2, 3, 4].

The present investigation has as purpose to implement the basic services of security incidents management for a CERT. For that, we have developed an application that maintains a centralized database with information about computational security incidents. The application is composed by agents that will use a search method and selection algorithms based on swarm intelligence [5]. These agents carry out searches through Internet, of incidents that have happened in other places, to know the answers to these. For the development of the distributed mechanism of search based on agents, we will use concepts derived from the work [5].

2 Theoretical Aspects

In this section we present the theoretical bases of our approach.

2.1 Computational Security Incidents

The computational security incidents are any event that is considered a threat for the security of a system [1, 3, 7]. In general, the different attacks that suffer the computational systems are known as incidents of security. These are a threat for the operability and good operation of any organization. A great variety of incidents of security exists, between which we can mention [1, 6, 8]:

- Attempts (successful or non) to gain access without authorization to a system or its data.
- Interruptions non-wished or Denial of Service. To use a system to process or to store data without authorization
- To change the characteristics of hardware, software or to install malicious software, without the knowledge of the proprietor

This way, the incidents of computer security are any event that is considered a threat for the security of a system [1, 6, 8], and they are classified in manuals, statics and automatic. The automatic ones are those software tools that, without the user's interaction, execute some operation to unbalance the operation of a computational system. Between these

types of incidents are the well-known virus, worms and Trojan. The static ones do not reproduce; between these we have: logical bombs, attacks of denial of service, among others. The manual incidents happen in an intentional way when an attacker wants to input in a computer system violating the restrictions of security that it has. Between the manual attacks we find: to scan vulnerabilities, hacking, cracking, social engineering, among others.

When a problem of security exits, it is very important that the affected organization has a quick and effective form of responding. The speed with which an organization can recognize an incident or attack, and then, the answer to the incident, will limit the caused damage dramatically and will reduce the recovery cost [4, 6, 7, 8]. At worldwide level, support groups have been created for the management of security incidents, which combat these threats and offer support to the entities assigned to them, the same as to prevent on possible attacks. These groups are denominated CERT, but they also have other names: Computer Incident Response Team (CIRT), Computer Security Incident Response Team (CSIRT), System Security Incident Response Team (SSIRT), among others. In general, these teams are coordinated and in constant collaboration with teams of different countries, experts of security and legal institutions [1, 9].

Killcrece, Kosakowsky and other (2003), in [2], outline the steps to establish an answer team to computer emergencies. Between other things, they define the processes of management of incidents and vulnerability, dissemination of watchful of security, management of tools of security, audit, intruders' detection, analysis of risks, among others (see fig. 1). This way, inside the structure of a CERT a team that takes charge of the management of incidents should exist. The functions of this team are:

- Detection and revision of reports: to revise the reports of incidents happened at the world, and to detect threats to document them.
- Analysis of the impact of damage: It is the intent to determine that it has happened, what is the impact or the damage caused, and what are the steps for the recovery that should take.
- Categorization and establishment of priorities: It is a procedure by means of which is made a revision and classification of the incidents to establish their gravities, and according to this, to assign priorities in the actions to take.
- Answer to incidents: They are the actions taken to mitigate or to solve the incident, to disseminate information of what was made, or to implement strategies so that the incident doesn't happen again.

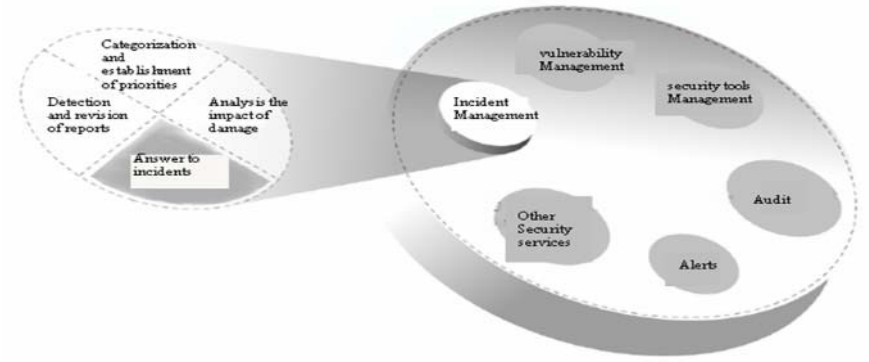

Fig. 1. A CERT System

In this work, we propose the development of an application that carries out the services of management of incidents of security for a CERT project. This application will be able to contribute with the best solution to treat an incident based on solutions found in the web. We are going to use a tool called "search agents", and they will facilitate the detection, report and answer to incidents. These agents will constantly be locating information on incidents in the web, to organize it and to store it in a database.

2.1 Collective or Swarm Intelligence (CI)

There are several works that study a behavior very interesting in a variety of animals, called "collective behavior" [5, 10]. Examples of that behavior are: a flock of birds traveling the sky, a group of ants in search of food, etc. Recently, investigations have studied that behavior, particularly how these types of animals act together to achieve collective goals, evolve, etc. The collective intelligence (CI) has been applied in different areas like telecommunications,

robotics, transport, military applications, etc. The main idea of the CI suggests that N agents in a colony cooperates mutually to achieve some goal. The agents use simple rules to govern their actions, and by means of the interactions of the group they achieve their objectives. An auto-organization type arises of the collection of actions of the group. The CI solves problems in a flexible, adaptive and decentralized way [10].

In general, in the CI models the agents are located in groups, called colonies, where they make a cooperative work. The agents process information, modulate their behavior according to stimuli, and make the best decision based on the information on the environment where they are. But the biggest challenge of these models is to make the agents to work in a collective way, where they integrate their individual activities to generate more complex and more effective results.

In the current studies of CI, the intelligent behavior frequently arises through the indirect communication among the agents. The inspiration source is the systems of insects. Individually, the insects have simple behaviors with limited memory. However, the insects carry out complicated tasks collectively with a high grade of consistency. Some examples of sophisticated behavior are: Formation of bridges; Construction and maintenance of nests; Cooperation when loading big objects; etc.

In the studied models two types of indirect communication have been identified, the first one involves a change in the physical characteristics of the environment. The construction of the nest is an example of this communication type, where an insect observes the development of the structure and it adds its ball of mud to the summit of the structure. The second is "*based on sign"*. Here, something is deposited in the environment that doesn't make any direct contribution to the task, but it is used to influence in the subsequent behavior. The indirect communication based on signs is very developed in the ants. The ants use a very volatile chemical substance, called "pheromone", to provide a sophisticated signaling system. The CI has inspired technical as collective optimization of particles, artificial systems of ants, ecological models, among other [10].

The ants can be modeled as reactive agents, and the ant colony as a Multiagents System (MAS). An agent can be defined as a computer program that acts autonomously on behalf of a person or organization. A reactive agent is an agent that emits an immediate action when receiving a sign or perceiving a state in the environment. The reactivity in the agents facilitates quick actions that do not require applying complex rules. The MAS is characterized by the interaction of several agents in the environment.

Particularly, in this work our approach is inspired on the Ants Artificial System (AAS) [10]. This model emulates the search of food in an ants colony. AAS utilizes a graph representation, where each edge (r, s) has a desirability measure, γ_{rs}, called *pheromone*, which is updated at runtime by artificial ants.

Informally, the AAS works as follows. Each ant generates a complete tour by choosing the nodes according to a probabilistic state transition rule; ants prefer to move to nodes that are connected by short edges, which have a high pheromone presence. Once all ants have completed their tours, a global pheromone updating rule is applied: a fraction of the pheromone evaporates on all edges, and then each ant deposits an amount of pheromone on edges which belong to its tour in proportion to how short this tour was. Then, we continue with a new iteration of the process.

The state transition rule used by ant system is given by the equation (1), which gives the probability with which ant k in city r chooses to move to the city s while building its t^{th} tour (transition probability from node r to node s for the k^{th} ant) [10]:

$$P_{rs}^{k}(t)=\begin{cases}[\gamma_{rs}(t)]^{\alpha}[\eta_{rs}]^{\beta} \Big/ \sum_{u\in J_r^k}[\gamma_{ru}(t)]^{\alpha}[\eta_{ru}]^{\beta} & \text{If } s\in J_r^k \\ 0 & \text{Otherwise}\end{cases} \qquad (1)$$

Where $\gamma_{rs}(t)$ is the pheromone at iteration t, η_{rs} is the inverse of the distance between city r and city s ($d(r,s)$), $J_k(r)$ is the set of nodes that remain to be visited by ant k positioned on node r and, β and α are two adjustable parameters which determine the relative importance of trail intensity (γ_{rs}) versus visibility (η_{rs}).

In AAS, the global updating rule is implemented as follows. Once all ants have built their tours, pheromone (that is, the trail intensity) is updated on all edges according to the equation [5, 10]:

$$\gamma_{rs}(t) = (1-\rho)\gamma_{rs}(t-1) + \sum_{k=1}^{m}\Delta\gamma_{rs}^{k}(t) \qquad (2)$$

Where ρ is a coefficient such that $(1-\rho)$ represents the trail evaporation in one iteration

(tour), *m* is the number of ants, and $\Delta\gamma_{rs}^{k}(t)$ is the quantity per unit of length of trail substance laid on edge (r, s) by the k^{th} ant in that iteration:

$$\Delta\gamma_{rs}^{k}(t)=\begin{cases}1/L_k(t) & \text{If edge (r,s)} \in \text{tour completed by ant k}\\ 0 & \text{Otherwise}\end{cases}$$

Where $L_k(t)$ is the length of the tour performed by ant *k* at iteration *t*. Pheromone updating is intended to allocate a greater amount of pheromone to shorter tours. Pheromone placed on the edges plays the role of a distributed long-term memory; this memory is not locally within the individual ants, but is distributed on the edges of the graph. The general algorithm is summarized as follows:

- Place the *m* ants randomly on the nodes of the AS graph
- Repeat until system convergence
  - For i=1, n
    - For j= 1, m
      - Choose the node s to move to, according to the transition probability (equation 1)
      - Move the ant m to the node s
  - Update the pheromone using the pheromone update formula (equation 2)

The time complexity of AS is $O(t*n^2*m)$, where *t* is the number of iterations done until the system convergence, and *n* the number of nodes to be visited.

Different versions to improve the classic AS have been proposed [10]. Two of them are the ant-density and ant-quantity algorithms. They differ in the way the trail is updated. In these models each ant lays its trail at each step, without waiting for the end of the tour. In the ant-density model a quantity Q of trail is left on edge (r, s) every time an ant goes from *r* to *s*; in the ant-quantity model an ant going from *r* to *s* leaves a quantity Q/d(r,s) of trail on edge (r, s) every time it goes from *r* to *s*.

3 Our System

We propose an algorithm based on CI that will be applied for the search of incidents. Equally, we propose a standard of the information of security incidents to be store in a XML file.

3.1 Standard of the information about security incidents

The system will make use of files that characterize to the incidents, and in our work this files will have a XML format. These files will contain static information (name, description, type, discovery date, etc.) and dynamic information (the platforms that it affects, level of danger, level of damage, symptoms, etc.) about the incidents. Also, a variable is included that is associated to the level of versatility of the incident, call "pheromone" (see Fig. 2).

```
<?xml version="1.0" ?>
<!-- <?xml-stylesheet type="text/css" href="imagen.css" ?>
-->
<incident-listing>
  <incident>
    <general_information>
          <incident_id>04</incident_id>
          <incident_name>W32.Stration.</incident_name>
          <incident_type>Gusano</incident_type>
          <date_discovered>15-12-2006</date_discovered>
          <date_updated>15-12-2006</date_updated>
          <affected_platform>Windows 9X, Windows NT,
Windows Server 2003, Windows XP</affected_platform>
          <desc_full>W32.Stration.EL@mm is a worm that
spreads by emailing itself to other computers.</desc_full>
     <incident_wild_level>Baja</incident_wild_level>
     <incident_damage_level>Medio</incident_damage_level>
     <distribution_level>Medio</distribution_level>
    </general_information>
    <detalles>
          <infection_method> </infection_method>
          <recomendations> </recomendations>
          <other>
            <distribution_method> </distribution_method>
            <efects> </efects>
            <more> </more>
          </other>
    </detalles>
    <solution>
          <removal> </removal>
          <other>
            <protected> </protected>
            <know> </know>
      </other>
    </solution>

<url_homepage>http://www.symantec.com/home_homeoffice/secur
ity_response/writeup.jsp?docid=2006-121511-2140-
99</url_homepage>
    <pheromone>1</pheromone>
  </incident>
</incident-listing>
```

Fig. 2. XML File with the description of the incidents

The labels used by the XML file were selected because they are those that better adapt to the characteristics of the incidents of security (see table 1).

Table 1. Labels of the XML file

<incident_listing>	It indicates the beginning of the list of incidents.
<incident>	It indicates the beginning of the information of the incident of computer security.
<general_information>	It contains general information of the incident. Inside the range of this label they are the labels incident_id, incident_name, incident_type, date_discovered, date_updated and affected_platform.
<incident_id>	It specifies the identification for each incident.
<incident_name>	It contains the information of the name of the incident.
<incident_type>	It contains the different types of incidents of computer security, such as: virus, worms, spyware, spam, among others.
<date_discovered>	It indicates the date of when was discovered this incident for the first time.
<date_updated>	It indicates the date of the last time in which the information of the incident was modified.
<affected_platform>	It specifies the computer platforms or operating systems in which the incident causes effect.
<desc_full>	It contains descriptive and detailed information of the incident.
<method_infected>	It specifies the method of incident infection, that is, as the incident carries out the infection of a computer system.
<sintoms>	It details the most common symptoms that present the computer systems when they are affected by this incident.
<method_distribution>	It specifies the method of propagation of the incident.
<efects>	It details the effects due to the presence of the incident in a computer system.
<solution>	Inside this label several labels are presented that specify forms of solving the incident. It contains t he labels: removal, protected and know.
<protected>	It presents information on as preventing to a system of being attacked by this incident.
<incident_damage_level>	It specifies the level of damage that can cause the incident.
<source>	It specifies the source where the information was obtained about the incident.
<pheromone>	This label contains information used by the search agents

3.2 Our CI Algorithm

3.2.1 Model of Selection

The steps that compose the selection process are the following (see fig. 3):

- Each agent travels across a path, independent to the followed for the rest of the agents, for the repositories looking for incidents according to the requested characteristics.
  - The trajectory of each path will consist on visiting, randomly, 1, 2 or N repositories that are those that contain the incidents and its possible solutions.
  - At the end of the trajectory for each one of the path, the agent will have accumulated a group of incidents, called "cc", that are candidates to be selected.
- Finally, each agent carries out the selection of the incident that better is adapted, from the group of incidents "cc" found.

After each agent completes the selection process, each one of them will proceed to upgrade the pheromone of the selected incidents. The decision on inserting the incidents to the database will be in charge of the administrator of the Incidents Management System. The general premises are:

- It is supposed that each incident has a XML file that characterizes it.
- It is assumed that the initial requirements are perfect and one has exact information of the incidents that are wanted to select. This information is associated to XML files, which contain the characterization of the incidents.

The equation to establish the grade of correspondence between an ideal incident and an incident pre-selected comes given for:

$$X_{lji} = 1 + \sum H_i \; - \; \sum N_{lj}$$

Where X_{lji} is the grade of correspondence among the ideal incident i, and the incident j, located in the repository l. $\sum H_i$ identifies the characteristics that should have the incident i that we is looking for, for example: the platforms that it affects, its symptoms, among others. On the other hand, $\sum N_{lj}$ represents the characteristics, similar to those wanted, of the incident j pre-selected (candidate). While nearer at 1

is the value X_{lji}, j and i are more similar. Each agent will evaluate a group of incidents of security, for each required incident, considering:

- The ideal value of $X_{lj}i$ is 1, the agent will choose incidents whose correspondence between the wanted incident and the found incident is near to that value.
- The quantity of pheromone $Y_{lj}(t)$ related to each incident j, located in the repository l, will be upgraded after the administrator chooses the incidents to enter in the database.

The transition equation that calculates the probability that an agent k selects an incident j, located in the repository l, of among a group cc_{ik} of possible incidents to select, is [5]:

$$P_{lj}^{ik}(t) = \left[Y_{lj}(t)\right]\left[X_{lj}^{i}\right] / \sum_{rsn \in cc_{ik}} \left[Y_{rsn}(t)\right]\left[X_{rsn}^{i}\right] \qquad (3)$$

Where, $Y_{lj}(t)$ represents the quantity of pheromone for the incident j that has been found by the agent k in the repository l.

It is important to notice that the value of the probability $P_{lj}^{ik}(t)$ could be different for two agents evaluating a same incident, since this depends on the group of incidents cc_{ik} that each agent has found.

As it was said previously, each agent k deposits a quantity of pheromone $\Delta Y_{lj}(t)$ in each one of the found incidents, inverse product of X_{lji} and of the evaluation R_{lj} that has been given to the found incident by this agent.

$$\Delta Y_{lj}(t) = (X_{lj}^{k} R_{lj})^{-1}$$

In our proposition, it is necessary to apply a positive and negative feedback. This last one is made through the rate of evaporation of pheromone, either in the incident selected by the user or not. In general, the positive feedback and negative is carried out according to the next equation [5]:

$$Y_{lj}(t) = (1-\alpha) * Y_{lj}(t) + \Delta Y_{lj}^{k}(t)$$

The initial quantity of pheromone of the incidents is assumed as a random number. α is a constant that controls the rate of evaporation. Finally, to determine the incident "i" to be selected, we define the following transition rule:

$$S^{*}_{ki} = \begin{cases} \arg\max_{r \in cc_{ik}} \{P_{rsn}^{ki}\} \\ J \quad \text{(Randomly)} \end{cases} \qquad (4)$$

where the value of P_{rsn}^{ik} comes given by the equation 3 (see fig. 3). This way, S^{*}_{ki} is the incident i selected by the agent k from a group of incidents cc_{ik}, or it is an incident J selected randomly.

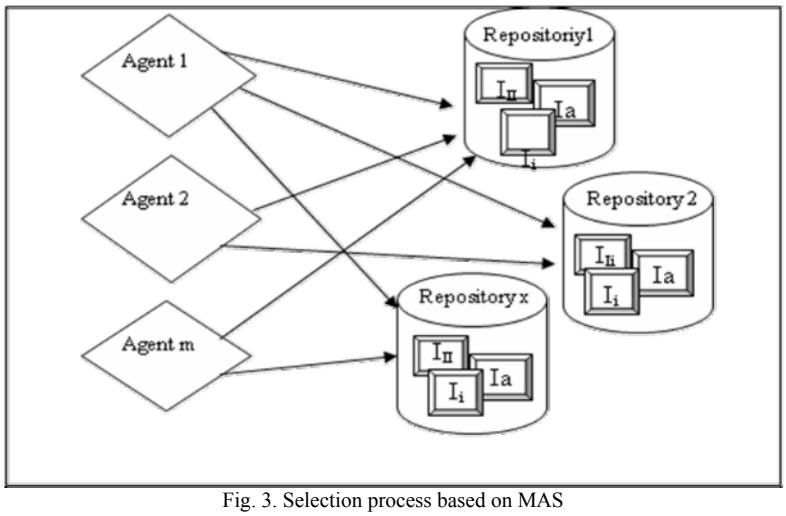

Fig. 3. Selection process based on MAS

3.2.2 Macro Algorithm

1. Definition and identification of the wanted profile of the incidents. To create k selection agents.
2. Each agent carries out the selection of the required incidents using the procedure "selection".

3. To upgrade the pheromone for each incident of the set cc_{ik} .
4. To select an incident of among the selection made by the different agents, and to enter it to the database of incidents.

Procedure of "selection":

1. Search of similar incidents to the incident required i (these incidents conform to the group cc_{ik})
2. To select one of them using the equation 4.

3.3 Our System

The system has a database that stores the gathered information of the repositories of incidents. In general the system will allow:

- To carry out the search of information of incidents of security located in web repositories using search agents and selection.
- To speed up the answer process to incidents.
- To manage a database with information of incidents.

The components of our system are (see fig. 4):

Interface: The interface receives the client's petitions and invokes the agents and the database management.

Search agents: They will look for the incidents that fulfill the specifications given by the user in the XML files on the web repositories. In the same way, they will select the incidents closer to the user's requirements, using our CI algorithm explained previously.

Database Management: it carries out the exchanges of data between the system and the client. It will manage a **local database** designed to store data of incidents.

XML Repository of incidents: They are the repositories of incidents based on XML files for the description of the incidents.

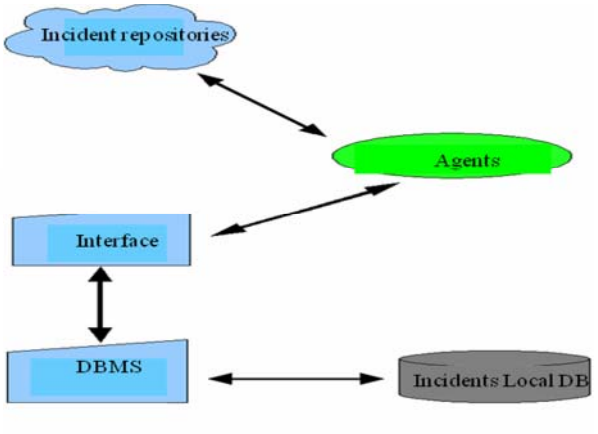

Fig. 4. Our System

In our system a client can select one of the next options (see fig. 5):

- To look for incidents in the repositories (see fig. 6).
- To register incidents in the local database (see fig 7).
- To modify incidents in the local database.
- To consult incidents in the local database (see fig. 8).
- To eliminate incidents in the local database.

Fig. 5. Main Screen

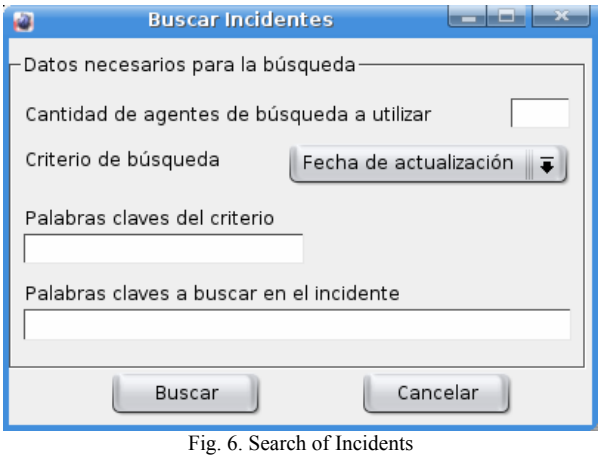

Fig. 6. Search of Incidents

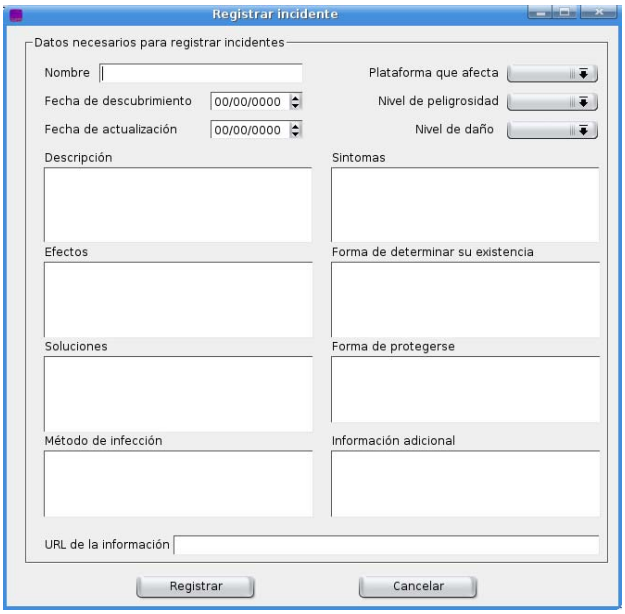

Fig. 7. To register Incidents

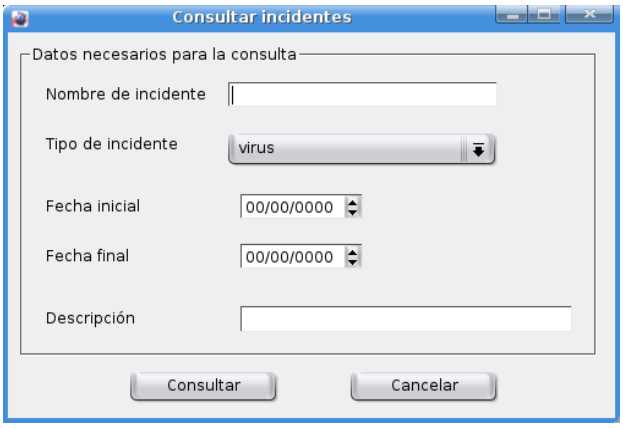

Fig. 8. To consult about an Incident in the Local Database

4 Experiments

4.1 Description of the experiments

The tests are carried out with three (3) repositories that store information of incidents, gathered of internet sources like panda [11], symantec [12] and mcafee [13]. We execute the system in 4 serial opportunities, varying the number of agents, and looking for (2) incidents explained in the table 2. For this experiment, the assignment of the repositories is random.

Table 2. Description of required incident

Incident to look for #1	
Name	Exploit-ANIfile
Platform that affects	Windows
Presented symptoms	System crashing unexpectedly
Incident type	Trojan
Descriptive words	exploit to microsoft windows kernel ANI file parsing vulnerability
Incident to look for #2	
Name	MalwareAlarm
Platform that affects	XP
Presented symptoms	An icon appears in alarm form in the tray of the system
Incident type	Spyware
Descriptive words	it is a program of type adware that tries to deceive the user

The order of execution of the system, together with the parameters assigned in each case, can see in the table 3.

Table 3. Cycle of Executions

Incident 1 and 2	**#of agents to use**
First Execution	2
Second Execution	3
Third Execution	4
Fourth Execution	6

4.2 Results Analysis

Before presenting the results, the used nomenclature is described next:

Arch/Prob/Pher/Rep = Arch indicates the name of the file selected by the agent, Prob corresponds to the selection probability, Pher the quantity of deposited pheromone and Rep indicates the repository where it was located.

Previously to the experiments, we are carried out tests to check the behavior of the agents, and this way to know if the system has not functional faults. In these tests the profiles of wanted incidents were specified (technical and functional specifications), verifying their presence in some of the repositories. At the end of this process we verify if the profiles selected by the agents were the previously identified. These tests verify that our system well. All the obtained results are shown in [14]. Here alone we will analyze the experiments of the table 2. The results are shown in the table 4.

Table 4. Results of the search for the experiment

Execution #1		
	Incident 1	Incident 2
	Arch/Prob/Pher/Rep	*Arch/Prob/Pher/Rep*
agent #1	3.xml/1/4.78585/1	Not found
agent #2	Not found	9.xml/1/14.5226/2
Execution #2		
	Incident 1	Incident 2

	Arch/Prob/Pher/Rep	*Arch/Prob/Pher/Rep*
agent #1	3.xml/1/6.48242/1	Not found
agent #2	Not found	9.xml/1/8.3831/2
agent #3	Not found	Not found
Execution #3		
	Incident 1	Incident 2
	Arch/Prob/Pher/Rep	*Arch/Prob/Pher/Rep*
agent #1	3.xml/0.88609/14.8786/1 7.xml/0.06834/2.88003/1 9.xml/0.04556/1.72975/1	Not found
agent #2	Not found	9.xml/1/7.89194/2
agent #3	Not found	Not found
agent #4	Not found	Not found
Execution #4		
	Incident 1	Incident 2
	Arch/Prob/Pher/Rep	*Arch/Prob/Pher/Rep*
agent #1	3.xml/0.851272/16.3263/1 9.xml/0.090265/9.86651/1 7.xml/0.036149/3.03881/1 4.xml/0.022311/1.28494/1	Not found
agent #2	Not found	9.xml/0.7978/10.6697/2 3.xml/0.1010/1.65539/2 7.xml/0.1010/5.7619/2
agent #3	3.xml/0.712473/12.1563/1 9.xml/0.215285/19.4443/1 7.xml/0.044204/3.86004/1 4.xml/0.028037/3.34125/1	Not found
agent #4	Not found	7.xml/0.5103/8.65146/2 9.xml/0.4269/10.6699/2 3.xml/0.0627/1.02155/2
agent #5	3.xml/0.489468/9.04667/1 9.xml/0.391458/18.5159/1 4.xml/0.067266/3.42173/1 7.xml/0.067266/4.03210/1	Not found
agent #6	9.xml/0.4334/4.72142/1 3.xml/0.4235/4.14648/1 4.xml/0.0800/4.69445/1 7.xml/0.0629/4.77927/1	Not found

The results of the experiment demonstrate that several agents, in the same execution, present similar results in the evaluation of the found incidents. On the other hand, the agent 1 of the third execution, and the agents 1, 3, 5 and 6 of the fourth execution (see table 4), visited alone a repository (the Repository 1: McAfee) to look for to the Incident #1.

Most of the agents when looking for the incidents #1 and #2 carried out the selection of the same group of incidents, being the incidents with more selection probability those described by the "3.xml" and "9.xml" files, respectively. However, the obtained results of the evaluation made by these agents on these incidents differ among the executions, because when concluding a search and selection process in an execution of the program, all the evaluated incidents related to the required incidents are subjected to the process of evaporation of their pheromones, and, the selected incidents receive a certain quantity of pheromone that will depend on the evaluation (considered random for the case of tests).

Another important point is that the selected incidents by most of the agents belong to a specific repository; this is because a given incident is not registered in several repositories. If that was the case, the selection was carried out in all the repositories where the incident was found.

4 Conclusions

The creation of a repository of incidents is required for a CERT that can end up becoming a standard for exchange and publication of information of incidents of computer security. To try to achieve this standardization we have proposed the use of XML files that contain detailed information of the characteristics of the incidents. Thanks to the generality of the search and selection algorithm of security incidents, this can be used in the search and selection of other elements or computer resources. When many agents are used in the program, we could not obtain good results. In the same way, the use of few agents could not produce the effect of cooperation.

Acknowledgment
This work has been supported in part by CDCHT-ULA under grant I-820-05-02-AA.

References:

[1] US-CERT. (https://forms.us-cert.gov/report/).
[2] Killcrece G., Kosakowsky K., Ruefle R., Zajicek M. *Organizational models for computer security incidents response teams (CSIRT's),* Technical Report No. IA-230, Carnegie Mellon Software Engineering Institute. 2003.
[3] Alberts C., Dorofee A. *Managing Information Security Risks*, Addison Wesley, 2002.
[4] Killcrece G., Kosakowsky K., Ruefle R., Zajicek. *State of the practice of Computer Security Incident Response Team (CSIRT's).* Technical Report No. IA-233, Carnegie Mellon Software Engineering Institute. 2003.
[5] Abraham B. Aguilar J. Bastidas J. Selection algorithm using Artificial Ant Colonies, WSEA

Transactions on Computers, Vol. 5, No. 10, 2006, pp. 2197-2203.

[6] Tahar, R. Benferhat, S. Towards Handling Dynamic Security Policies, *WSEA Transactions on Computers,* Vol. 1, No. 2, 2006, pp. 162-167.

[7] Trcek, D. Using System Dynamics for Managing Risks in Information Systems Denis, *WSEAS Transactions on Information Science & Applications*, Vol. 5, No. 2, 2008, pp. 175-180.

[8] Aris, S. Arshad, N. Mohamed, A. Conceptual Framework on Risk Management in IT Outsourcing Projects, *WSEAS Transactions on Information Science & Applications*, Vol. 5, No 4, 2008, pp. 816-831.

[9] Gao, K. Xi, L. Li, J. A Security Architecture Model for Mobile Computing, *WSEA Transactions on Computers*, Vol. 6, No. 3, 2007, pp. 494-499.

[10] Bonabeau, E. Dorigo, M.Theraulaz, M. *Swarm intelligence: from natural to artificial systems,* Oxford Editions, 1999.

[11] Panda Software. http://www.pandasoftware.com/virus_info/encyclopedia/.

[12] Symantec Corporation. http://www.symantec.com/enterprise/security_response/threatexplorer/threats.jsp.

[13] McAfee Inc. http://us.mcafee.com/virusInfo/default.asp?id=calendar.

[14] Aguilar, J. Abraham, B. Moreno, G., Latorre, E. *Un Sistema de Gestión de Incidentes de Seguridad,* Technical Report No. 6-2008 CEMISID, Universidad de los Andes. 2008.