Automatic reconfiguration in the response to network incidents by neural networks

Andrzej Olchawa, Tomasz Walkowiak
Wroclaw University of Technology, Poland
{andrzej.olchawa, tomasz.walkowiak}@pwr.wroc.pl

ABSTRACT

This paper presents the idea of network monitoring based on multi-agent techniques and automatic network reconfiguration based on Artificial Neural Networks. Authors propose to combine these two techniques, to create a intelligence entity, which will be able to make reconfiguration decision, based on discovered network incidents.

Key Words: sensors, multi-agent event monitoring, systems security, neural networks, molecules.

1. Introduction

The enormous progress in the field of computer science provides more and more business technologies, which requires a high level of reliability and stability. Moreover, security and system defence is directly related to technical, social and economic background [1, 2, 3].

Major concerns in the development and exploitation of networks is confidentiality, integrity and availability of information [4]. For these purposes, the event monitoring layer is required. Currently, events monitoring plays the conspicuous part of the system defence [5]. There are many different approaches related to event monitoring, unfortunately most of them are relatively ineffective [6]. In the aim of the efficiency enlargement except event monitoring approaches, Artificial Intelligence is introduced [7, 8].

In the paper we propose to use the Artificial Intelligence as the tool which enables reaction to incidents – automatic network reconfiguration.

The paper is structured as follows. In the second section the description of the network monitoring system structure is given. The concept of three-layer network structure is presented. Then, the idea of intelligence entity, which is able to make decisions of network reconfiguration is introduced. Finally, the proposal of realization the idea is given. The final section gives the short conclusions, and current state of work.

2. The architecture of the network monitoring system

The architecture proposed in this paper is a three-layer structure (see Fig. 1):

- low-layer (the lowest monitoring part),
- middle-layer (decision and reconfiguration monitoring part),
- high-level (the console application, layer of data presentation).

We will focus on two lower layers (low- and middle-layer). The middle-layer is a structure, which consists of the logically divided network components. Each of these logical components – which are called *Molecules* – contains a several monitored and one monitoring hosts [9]. This approach will allow us collect events from each host not as a whole monitored system, but as a small molecule. Collected data will be filtered more accurately in this way, what should reduce the quantity of

required decision (reconfiguration) rules, relating to a particular molecule.

The molecule (middle-layer) could be seen as a collection of technical services (like WebServer, DNS, FTP, etc...) and the host with an application, which supervises a particular molecule – we named it the DecisionAgent. Of course, molecules don't have to be just a logical entity, but they can be also defined by a physical division of the network (like sub-networks, for example).

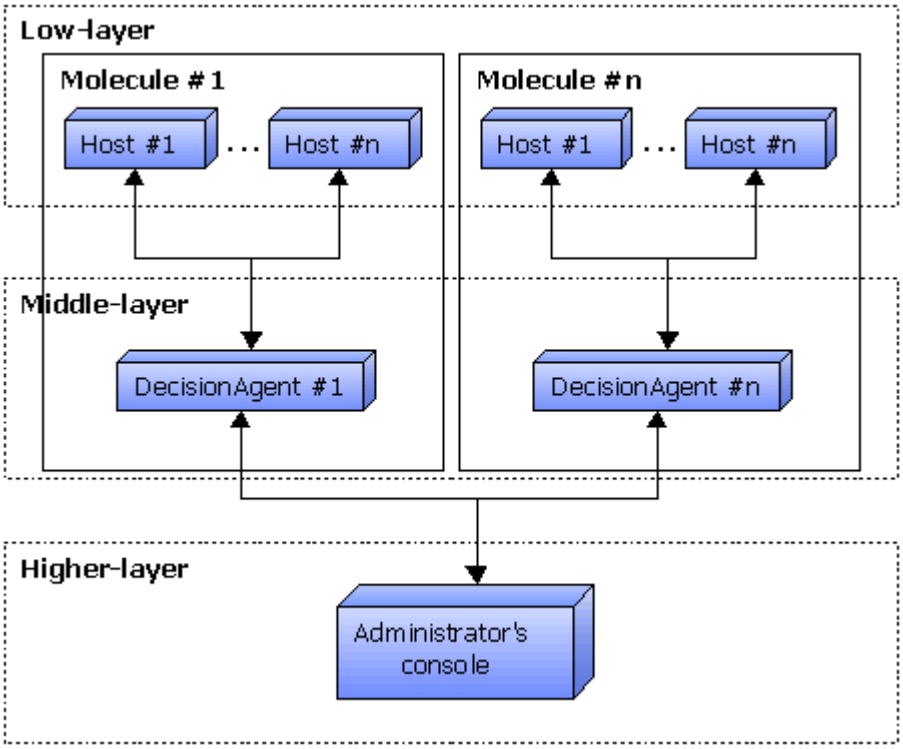

Fig. 1. The multi-layer network architecture

For the monitoring purposes, we propose to locate agents on each of the Molecule's hosts. We define an Agent as a small entity (in low-layer monitoring system), which is responsible for gaining information about the host condition, on which the agent resides. These Agents can be implemented in many different forms, from a computer applications to a small programs embedded on router/switch software. However, in this work – for a simplification, we assume it is a computer application, which is working on Linux or Windows operating system. In the simplest form, the Agent should receive some events from working on hosts services, gain information about host performance (like memory and CPU consumption) and provides these data to the higher layer, inside a molecule which contain this agent. We can reach this aim by connection an agent to the applications and libraries delivering information on a certain service. In our work, we are using applications and mechanisms such as SNORT, Statgrab, SNMP and WMI, but this is only an example. The agent should be exceptionally autonomic, and moreover, be able to gain system events from many different sources (in the application logs form, for example). To reach this aim, we design the agent which consists of independent modules, monitors host and provides information about its state. Such modules can be created independently and added into the agent (as external libraries), as the source of information on the subject of monitored host.

In the higher layer, information about network events (sent from low-layer by Agents) should be received by a monitoring host (DecisonAgent application), which de facto is a representative entity of the molecule. The responsibility of the DecisionAgent is – in

the simplest form – to collect network events from monitored hosts (in a particular molecule), making an aggregation on these information and persist in a database.

However, to introduce an element of reconfiguration to the monitoring system, we have let DecisionAgent to automatically, and remotely, change configuration of monitored hosts. To do this, we propose to design the Agent thus, it will be able to receive a feedback from DecsionAgent in the form of commands to be executed (Fig. 2.). This approach will allow us to build a special bridge responsible for the transportation of the command to the realization.

mention is also the aggregation of Agent's events.

As was written before, the Agent's events are simple messages delivered from low-layer to the middle-layer of the monitored system. This means, that we receive a large quantity of various messages, which received alone, don't bring useful information. The solution of this problem is the aggregation of these information. For example, we don't need information about CPU consumption in the single second of the time. But the same information taken in each 10 second is also useless. What we really need, is the average of the CPU consumption in the last *N* seconds, and this information, could be gained by

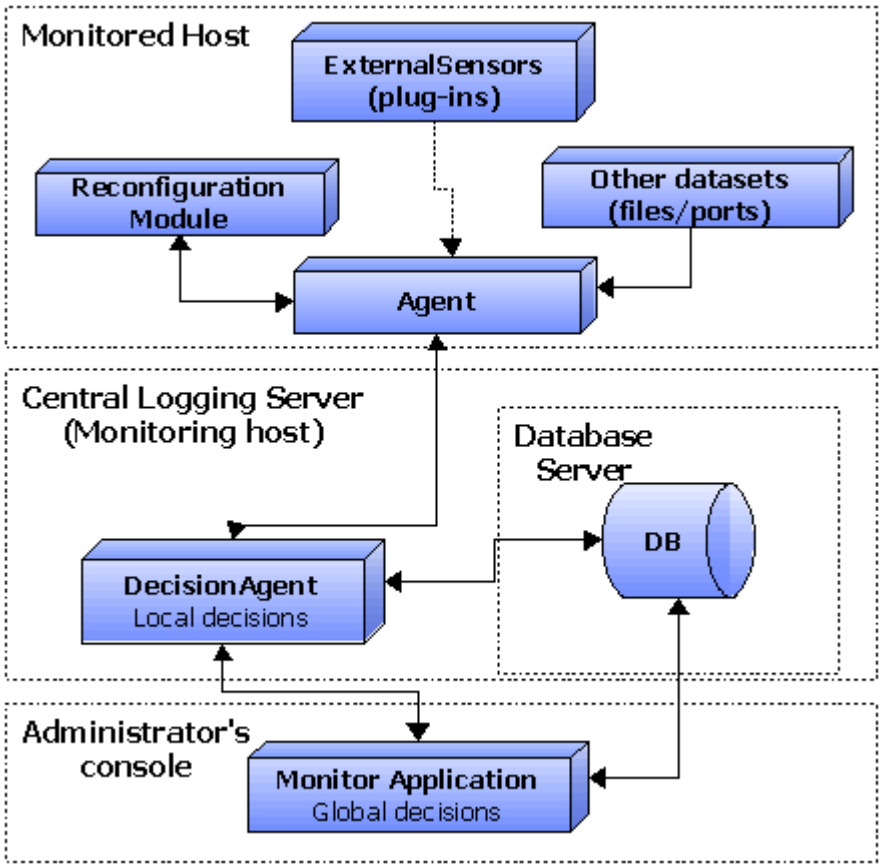

Fig. 2. The architecture of the network monitoring system

3. The decision Agent

The main aspect of this paper, is the consideration of an intelligent entity, which could automatically generate decisions based on network events provided by agents. We'll describe mentioned entity, but before that, worth introducing the data aggregation. To be more clear, we should prepare a set of aggregation rules, which will process the events and generate events on more higher level allowing a process of making decisions on system reconfiguration.

As a solution of this problem, we propose to use an element of Artificial Intelligence (AI) - the Artificial Neural Networks. Using characteristic capabilities of neural networks (like approximation and data organization is time-consuming and requires many tests before we will obtain satisfying results. First of all, we should digitize a message received from an agent, and then, deliver the content of this

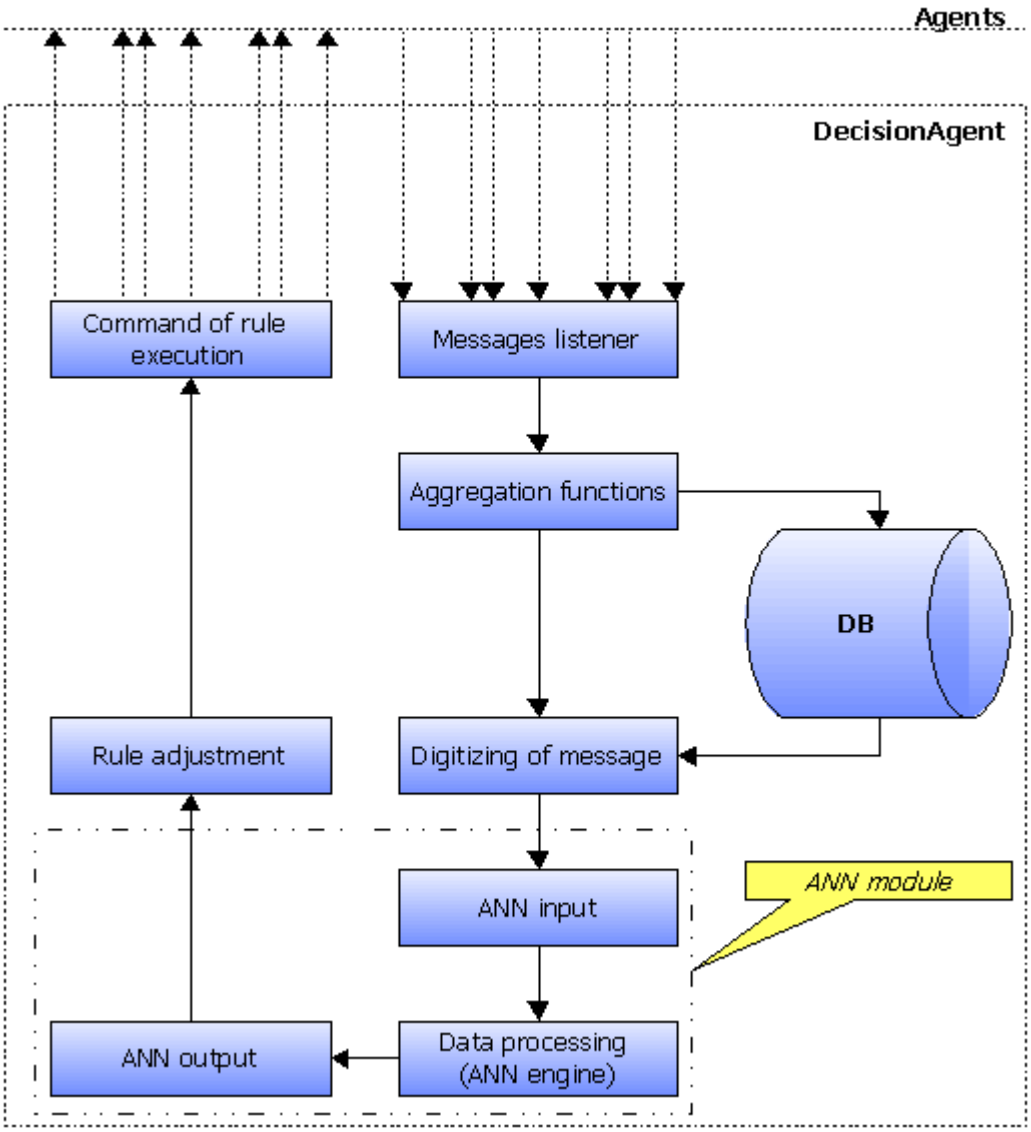

Fig. 3. DecisionAgent architecture

generalization [10]), we should be able to create an independent entity, which the main aim could be the automatic network reconfiguration.

We are trying to design mechanism based on ANN, which will be able to recognize a problem, classify it and take a proper action (reconfigure the system). The most difficult task relating to this aim, is an arrangement of the input data format for the neural network. Especially, when we have to deal with many different network events. As usually, the process of the input message as a set of numbers into the neural network. Because the DecisionAgent will have to deal with events, which are carrying many various information, the most important thing is to design a generic format of rules (input data) for the neural network. Such rule could be representative form of the state of monitored host and related with the concrete decision. In this way, each of event becomes an input vector pattern, and at the same time, creates a learning vector for the ANN.

Now, we try to introduce an example of problem, which probably could be solved by DecisionAgent, without interference of network administrator. Let's imagine, that we have a molecule which consist of two identical application servers (Tomcats, for example). And assume following situation: somehow CPU consumption increase to 80-90%, the performance of services falls considerably and is incapable of serving clients. This is the most simple situation, and obvious solution is to redirect the movement to the second application server. To do this automatically, we have to launch the Agent on monitored application server, which will provide information about CPU performance to DecisionAgent. On the other side, DecisionAgent should received information of this event, interpret properly (make digitalisation), aggregated in time, and put it on the his neural network input. The result should be a decision chosen from some set of decisions. For example, a message presenting this situation (sent by Agent) could looks thus:

<agentName> <IP> <time> <attribute> <value>

Table 1 contains description of these message fields.

We can digitize this message because, we know exactly what and where we are monitoring. When we do this, we'll obtain the *X* - input vector for neural network. The result of ANN work will fit the *D* - rule for the rules set.

The last thing which we have to take under the consideration, is the teaching of DecisionAgent. Of course, we need an expert (network administrator) for this purpose, which will arrange the set of data mentioned above. Moreover, the expert should take the part in the initial bevel of the system work, and be able to add new elements or make contingent modifications of rules sets. However, after these actions we should obtain the system able to make own decisions for network reconfiguration (at least such simple ones).

Table 1. Description of message filed

Name of parameter	Meaning of parameter
agentName	The Agent's name
IP	The address of monitored host
time	The time of checking of the state
attribute	Name of attribute (CPU in our case)
value	Value of attribute (CPU consumption in our case)

4. The architecture of the decision Agent

According to description presented before, the DesicionAgent is an intelligence entity, which is able to make his own reconfiguration decision, based on network events provided by Agents. This means, that the architecture of DecisionAgent should consists of elements composition, such as (Fig. 3.):

- messages listener,
- aggregation functions and digitizing message modules,
- input data module (for ANN),
- structure of neural network and data processing,
- output data module (from ANN),
- rule adjustment module,
- command execution module.

Messages listener module – the module, which is responsible for receiving message from low-layer monitoring and the message organization in the objects form.

Aggregation functions module – this module will received prepared events objects. In the next step, this should execute the set of functions, which are responsible for the data pre-filtering and proper aggregation of the pre-filter work results. The system should spacious reconfiguration of rules, based on which the aggregation will be made.

Digitizing of message module – this part is responsible for digitizing of message, which means, that each part of message will be represented by a number. As in the aggregation case, this module also should allows on the maximum of reconfiguration, to give the possibility change of digitizing rules for user/administrator.

- output data – this part is responsible for providing results of neural network work.

In other words, the ANN module, contains all elements relevant with the delivery input data into neural network, returning output data as results, and at last, all

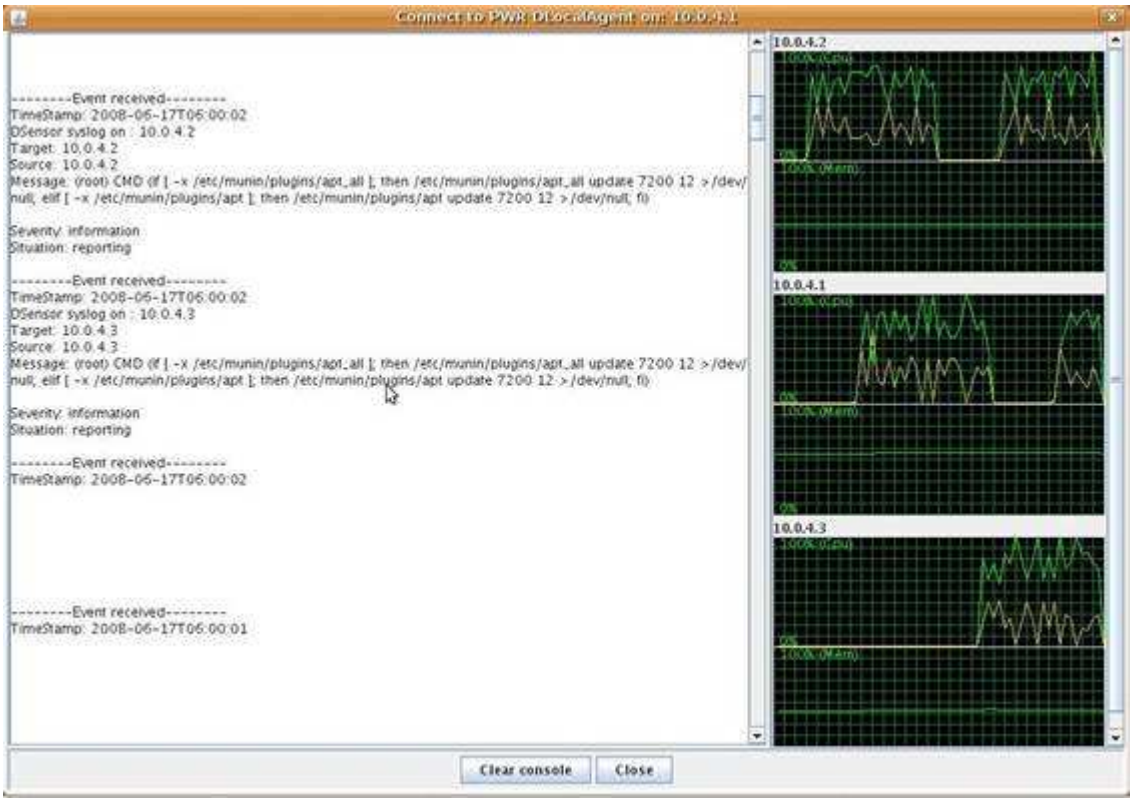

Fig. 4. Screen-shot: console application. Left side presents information received from Sensors, right side shows current performance of monitored machines.

ANN module – this module provide a mechanism of identifying events. As we mentioned previously, it will be a neural network. In this case, we propose to divide ANN module on 3 dependent parts:

- input data – responsible for conversion of aggregated objects to a input vector for the neural network; the result of this part work should be a set of numbers,
- data processing – this is the whole structure of the neural network, which will used during DecisionAgent teaching as also identification process,

processes relevant with teaching and identifying of neural network. Of course, to acquire this aim, ANN module should allows to design and configure each of mentioned above parts. Especially, the configuration of neural network structure should be extremely elastic.

Rule adjustment module – is a simple module responsible for match results deliver by ANN module to the one or more from the rules set.

Rule execution module – this is the last step of DecisionAgent work. This module is responsible for arrangement of message defining a execution command.

5. Conclusion and further work

This paper presents the architecture of the monitoring and reconfiguration system. At present, we are in the implementation stage. We have already implemented the lowest monitoring part (agents). Currently, we are working on DecisionAgent, what should allow us to make some tests and distinguish, which structure of the neural network is the best for this type of a usage. This part is almost ready, except the decision module. Moreover, we designed and implemented the administrative console to be able to make some tests. The console is working as the administrator's tool, which communicates with the DecisionAgent in order to provide information about conditions of monitored hosts in a given molecule. The example of console work, represents the picture (Fig. 4.). This screen-shot, presents main application window. The left side of the window displays information about collected monitoring events. On the right side, we see graphs, which show the current state of monitored machines. We are able to see CPU and Memory consumption – in this specific example. The result of our work, will be the fully configurable monitoring system, which hopefully will allow to adapt itself to many different network structures, and will be able to automatically make reconfiguration decisions, based on information provided by network components.

Acknowledgement

Work reported in this paper was sponsored by a EU grant DESEREC IST-2004-026600, (years: 2006-2008) "Dependability and Security by Enhanced Reconfigurability" and grant SPUB 208/6.PR UE/2006/7 from the Polish Ministry of Science and Higher Education.

References

[1] McCarthy, L.: Symantec Internet Security Threat Report. Tech. rep., Symantec, Inc., September 2003.

[2] McHugh, J.: Intrusion and Intrusion Detection. In: International Journal of Information Security, pp. 14–35, vol. 1, num. 1, 2001.

[3] Woda, M. Walkowiak, T.: Multi agent event monitoring system. In Proc. of the 3rd International Conference on Information Technology ICIT 2007. Al-Zaytoonah University of Jordan, Amman, Jordan, May 9-11, 2007.

[4] Woda, M., Walkowiak T.: Agent based approach to events monitoring in Complex Information Systems, Securware, pp.397-402, 2008 Second International Conference on Emerging Security Information, Systems and Technologies, 2008.

[5] Bace, R.G: Intrusion Detection. Macmillan Technical Publishing, New York, NY 2000.

[6] Idris, N.B., Shanmugam, B.: Artificial Intelligence Techniques Applied to Intrusion Detection. IEEE Indicon 2005 Conference, India, Chennai, 2005.

[7] Garcia, R.C. Copeland, J.A.: Soft Computing Tools to Detect and Characterize Anomalous Network Behaviour, IEEE World Congress, pp. 475-478, 2000.

[8] Dickerson, J. E., Juslin, J., Dickerson J.A. and Koukousoula, O.: Fuzzy Intrusion Detection. In the Proceedings of North American Fuzzy Information Processing Society 2001 (NAFIPS 2001), Vancouver, Canada, July 25th, 2001.

[9] Israel, M., Borgel, J. ,Cotton, A.: Heuristics to Perform Molecular Decomposition of Large Mission-Critical Information Systems, Securware, pp.338-343, 2008 Second International Conference on Emerging Security Information, Systems and Technologies, 2008.

[10] Bischop, Ch.: Neural Networks for Pattern Recognition. Clarendon Press Oxford, 1996.