

# Asynchronous Alert Correlation in Multi-agent Intrusion Detection Systems

Vladimir Gorodetsky, Oleg Karsaev, Vladimir Samoilov, and Alexander Ulanov

SPIIRAS, 39, 14-th Liniya, St.Petersburg, 199178, Russia  
{gor, ok, samovl, ulanov}@mail.iias.spb.su

**Abstract.** This paper presents conceptual model, architecture and software prototype of a multi-agent intrusion detection system (IDS) operating on the basis of heterogeneous alert correlation. The latter term denotes IDS provided with a structure of anomaly detection-like classifiers designed for detection of intrusions in cooperative mode. An idea is to use a structure of classifiers operating on the basis of various data sources and trained for detection of attacks of particular classes. Alerts in regard to particular attack classes produced by multiple classifiers are correlated at the upper layer. The top-layer classifier solves intrusion detection task: it combines decisions of specialized alert correlation classifiers of the lower layer and produces combined decision in order to more reliably detect an attack class. IDS software prototype operating on the basis of input traffic is implemented as multi-agent system trained to detect attacks of classes *DoS*, *Probe* and *U2R*. The paper describes structure of such multi-layered intrusion detection, outlines preprocessing procedures and data sources, specifies the IDS multi-agent architecture and presents briefly the experimental results received on the basis of DARPA-98 data, which generally confirm the feasibility of the approach and its certain advantages.

## 1 Introduction

Currently, intrusion detection task is of great concerns and the subject of intensive research ([2], [4], [10], [11], [12], [13], [14], etc.). The contemporary studies show that advanced approaches to Intrusion Detection Systems (IDS) design are focused on data fusion ideas assuming use of multiple data sources and multiple classifiers operating in various feature representation spaces with the subsequent combining of their decisions [1]. Unfortunately, several specific properties of the intrusion detection system input make the above mentioned decision combining task very difficult. Among these properties, temporal nature, high-frequency dynamics and asynchronous character of input are of the primary importance. Other important issue of IDS input that is ignored in the most of research is information ageing resulting from the temporal nature and variety of frequencies of input data streams arriving from various sources.

The paper is devoted to the *heterogeneous alert correlation* approach to intrusion detection. The introduced term denotes an approach assuming that IDS is composed of a structure of classifiers and each classifier of this structure is trained for detection of attacks of a particular class, e.g. an attack of the class either *DoS*, or *Probe*, or

*U2R*. The second assumption of the approach is that several classifiers are trained for detection of the same attack class while operating with data of various sources and/or various feature representation spaces. Each of such specialized classifiers may produce decisions of two classes: "*Alert*" regarding to "its own" attack class (e.g. "*DoS alert*", "*U2R alert*", etc.) or "*Normal*" (without producing an alert). In the second layer, alerts of the same type (if any) produced by source-based classifiers are correlated and the results are sent to the top layer. The top-layer classifier solves intrusion detection task: it combines decisions of specialized alert correlation classifiers and produces combined decision in terms of particular attack class if any.

In the rest of the paper, section 2 outlines the IDS input data model and preprocessing procedures forming various data sources (representation spaces). It describes the structure of the interacting classifiers designed for heterogeneous alert correlation and event dynamics of the IDS operation. Section 3 describes a model of data ageing used in the developed IDS software prototype while Section 4 gives detailed specification of its architecture based on multi-agent framework. This architecture is specified in the style assumed by Gaia methodology [15] that is used in development of the IDS software prototype. Section 5 outlines experimental results received through testing of the developed prototype using DARPA data [3]. Conclusion summarizes the paper contributions and intentions for future research.

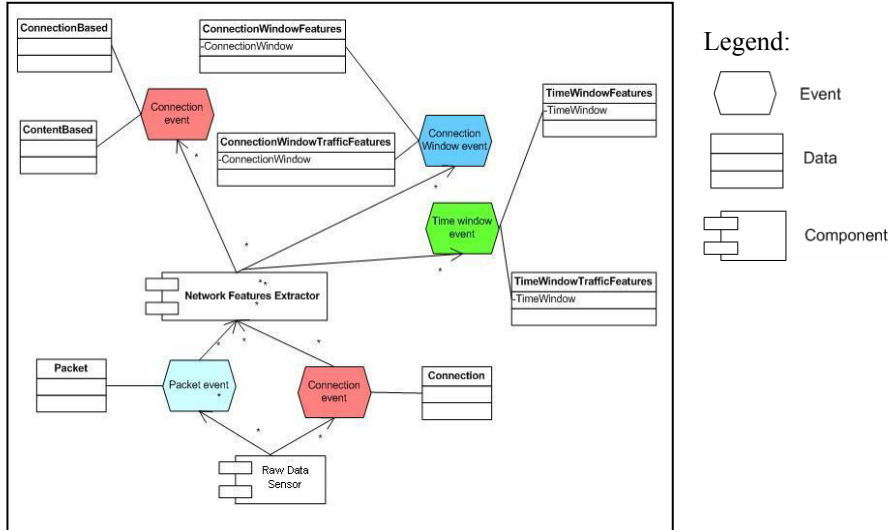
## 2 Conceptual Model of Multi-alert Correlation for Intrusion Detection

### 2.1 Input Data

The major peculiarity of IDS input data is their *temporal* nature. Indeed, input data perceived by sensors of IDS or produced by preprocessing procedures are mapped *time stamp*, which is considered as an important data attribute. Events of various data streams arrive into IDS classifiers *asynchronously*. Since averaged frequencies of various data streams are different, the data incoming to meta-level responsible for alert correlation possess finite life time, i.e. after elapsing certain time from the moment they are produced the data become of less relevance with regard to the current status of user activity and therefore less useful or useless for its assessment.

It is assumed that the input data model accounts the data streams resulting from the preprocessing of the network traffic represented in *TCP* dump. This traffic perceived by "*Data Sensor*" is further preprocessed according to the scheme presented in Fig.1. Traffic preprocessing procedures are aimed at extraction of various features resulting in creation of "secondary" data sources (feature representation spaces forming input for several source-based classifiers).

The developed traffic preprocessing procedures operate in the following order. First, events corresponding to new packets and new connections are identified. The information contained in the identified packets and connections is further processed in order to extract features and form secondary data sources. Network feature extraction procedure identifies events that indicate availability of newly arrived following data:



**Fig. 1.** Raw data streams and preprocessing procedures forming secondary data sources constituting input data streams of IDS

- (1) *Connection-related* data that are used for extraction of connection-related features forming two data sources, i.e. *ConnectionBased* and *ContentBased* data sources.
- (2) *Time window-related* data representing certain statistics averaged within sliding time window of the predetermined length and shift (in our case, *length= 5 sec.* and *shift=2 sec.*). These data are used for extraction of the features forming two secondary data sources, *TimeWindowFeatures*, and *TimeWindowTrafficFeatures*.
- (3) *Connection window-related* data representing certain statistics averaged within sliding time window containing a user-assigned number of connections (in our case, this number is equal to 20 connections and shift is equal to 1 connection). These data are used for extraction of the features forming two more secondary data sources, *ConnectionWindowFeatures*, and *ConnectionWindowTrafficFeatures*.

Traffic preprocessing procedures were developed by authors. As the input of these procedures, the DARPA data [3] are used.

**2.2 Heterogeneous Alert Correlation Structure**

The primary factor influencing on the IDS architecture is the structure of interaction of the source-based classifiers and meta-classifiers. Let us comment it by example of the structure used in the developed case study illustrated in Fig. 2.

Each data source is attached several source-based classifiers. A peculiarity of these classifiers is that each of them is trained for detection of a fixed class of attacks and produces alerts regarding corresponding attack class. That is why the alerts produced are heterogeneous, i.e. correspond to different classes of attacks. Actually, each source-based classifier solves an anomaly detection task, but each "anomaly"

alert corresponds to particular class of attacks. Thus, the IDS system in question solves intrusion detection task.

*Connection-based* data source is attached three specialized classifiers intended for detection *DNS CB*, *R2U CB* and *Probe CB* classes of attacks, i.e. these classifiers are trained to detect attacks of the classes "*Denial of Service*", *R2U* and *Probe* respectively. Each of the above connection-based data source classifiers transmit the produced decision to particular meta-classifier (see Fig.2).

*ConnectionWindowFeatures* data source forms input of two specialized classifiers, *R2U CW* and *Probe CW*, trained for detection of attacks of the classes *R2U* and *Probe* respectively. They also send their decisions to particular classifiers of the meta-level.

*ConnectionWindowTrafficFeatures* data source is attached three specialized classifiers, *R2U CWT*, *Probe CWT* and *NormalCWT* trained for detection of attacks of the classes of *R2U*, *Probe* and *Normal* activity (no attacks) respectively. They send their decisions to various classifiers of meta-level.

*Time WindowFeatures* data source forms input of three specialized classifiers, *DNS TW*, *R2U TW*, and *NormalTW* trained for detection of attacks of the classes *Denial of Service*, *R2U* and *Normal* activity (no attacks) respectively.

*Time WindowTrafficFeatures* data source is attached three classifiers, *DNS TWT*, *R2U TWT*, and *ProbeTWT* trained for detection of attack classes *Denial of Service*, *R2U* and *Probe* respectively.

At the meta-level, three specialized meta-classifiers are introduced. Each of them is responsible for combining decisions from source-based classifiers trained for detection of particular type of attack or *Normal* situation. They operate in asynchronous mode while making decision every time when an event and data from at least one source-based classifier arrives. A peculiarity of the decision making structure in question (Fig.2) is that, in it, one more decision combining layer, top layer, is used. It combines the inputs arriving from the specialized meta-classifiers thus solving the intrusion detection task.

### 2.3 Dynamics of IDS Operation

The data and event streams in the implemented IDS prototype are presented in Fig.1. Let us describe the dynamics of these streams in the process of IDS operation.

Dump of the network traffic is captured by sensor, *Raw Data Sensor*. It produces primary events of two types: (1) *PacketEvent* – receiving of an IP packet and *Packet* data, and (2) *ConnectionEvent* – completion of the connection and *Connection* data. Events and data input to the component *NetworkFeatureExtractor* intended for extraction of the features from raw data and generation of the secondary events, that are (1) *ConnectionEvent* and associated arrays of the features, *ConnectionBased* and *ContentBased*; (2) *ConnectionWindowEvent* indicating completion of a time window containing given number of connections and associated arrays of the features, *ConnectionWindowFeatures* and *ConnectionWindowTrafficFeatures*; (3) *TimeWindowEvent* indicating completion of the time window of a predefined duration and associated arrays of the features, *TimeWindowFeatures* and *TimeWindowTrafficFeatures*.

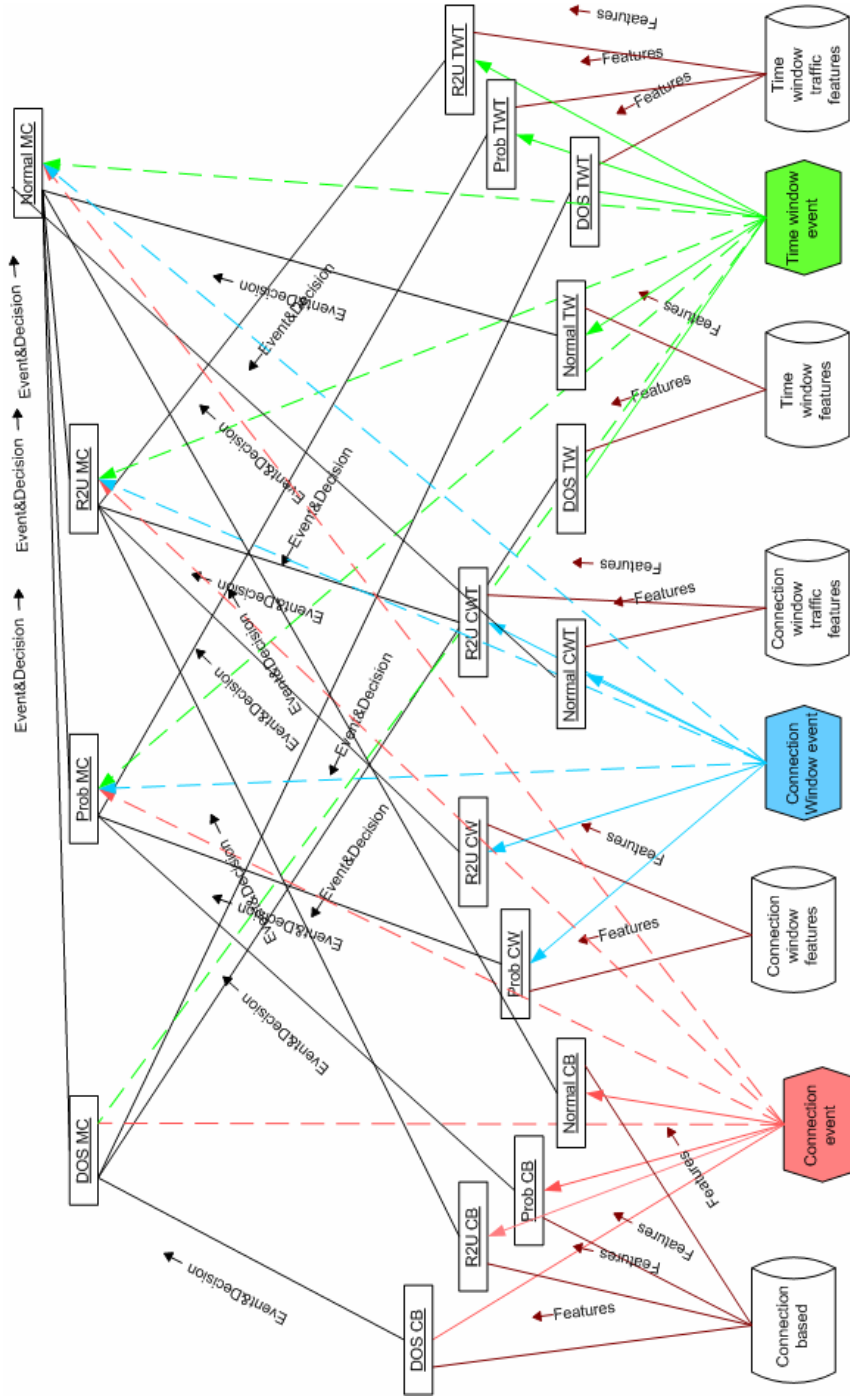


Fig. 2. The structure of decision making and decision combining implemented in the software prototype of multi-agent IDS

**Table 1.** Distribution of attack classes against types of operating systems

Type of OS: <i>Redhat</i>		
Attack Class	Attack name	Number of cases
<i>Denial of Service (DoS) Attacks</i>	<i>back</i>	4
	<i>land</i>	22
	<i>pod</i>	35
	<i>smurf</i>	11
	<i>teardrop</i>	7
<i>(DoS) attacks in total</i>		79
<i>Probes attacks</i>	<i>ipsweep</i>	7
	<i>portsweep</i>	5
	<i>satan</i>	5
<i>Probes attacks in total</i>		17
<i>Remote to User (R2U) attacks</i>	<i>dict</i>	1
	<i>guest</i>	1
	<i>imap</i>	3
	<i>phf</i>	5
<i>R2U attacks in total</i>		10
<i>User to Root attacks (U2R)</i>	<i>perl</i>	5

All classifiers of the source-based layer as well as meta-classifiers of the first and top layers were trained and tested based on DARPA data [3]<sup>1</sup>. Generalized information about these data that are used for training and testing of the classifiers composing the decision structure depicted in Fig.2 is presented in Table 1.

### 3 Models of Data Ageing

According to the used alert correlation strategy, decisions of meta-classifiers are updated at any time when new input ("event") produced by some source-based classifier incomes. Let us recall that while receiving an updated decision from a source-based classifier, the meta-classifier updates its decision using the newly received decision and also on the decisions produced previously by other source-based classifiers at various time instants. The latter decisions have different "ages" and therefore different relevancies to the current computer security status. Thus, potential *data ageing* is one of the important peculi-

arities of the alert correlation system in question. Let us consider the models of data ageing.

Two data ageing models were explored. The *first* of them assumes that each data incoming to the alert correlation layer is assigned certain "age" at the moment of the computer security status update and if this "age" is less than a fixed threshold (it is individual for each data source) then the corresponding data are used in the alert correlation "as is". Otherwise, these data are assumed *missing*:

$$D_i(t_{k+1}) = \begin{cases} D(t_k), & \text{if } t_k \leq t_{k+1} \leq t_k + T_i^{Ag} \\ \emptyset, & \text{otherwise} \end{cases} .$$

where  $D_i(t)$ —stands for the decision of a base classifier associated with the  $i$ -th data source produced at time instant  $t$ ;  $t_k$  stands for the time instant at which the decision income into meta-classifier;  $T_i^{Ag}$  stands for the threshold value of life time of the decision  $D_i$  produced by the source #  $i$ ; and  $\emptyset$  stands for the missing value.

This model was experimentally investigated and the results were in full described in [5, 8]. The advantages of this model are twofold. On the other hand, this model is simple enough. On the other hand, if some sensors or data sources fail, i.e. do not

<sup>1</sup> Training and testing procedures used in design of classifiers are not considered in the paper.

produce decision in required time instant then, nevertheless, the combined decision is produced because meta-layer classifier is capable to process data with missing values. The sound algorithm solving such task is described in [6]. A drawback of this model is that it is approximate and in some cases may be too rough.

The second model of data ageing assumes that the learning mechanism has to automatically determine dependence of informative power of the decision produced by a source classifier depending on "age". More strictly, this model assumes that each input of the alert correlation classifier is assigned an additional numerical attribute  $\Delta_i(t_1, t_2)$ , where  $\Delta_i(t_1, t_2)$  is the "age" of input of  $i$ -th source-based classifier produced at the time instant  $t_1$  if it is used in alert correlation procedure of meta-layer at the time instant  $t_2$ . Thus, when  $i$ -th source-based classifier produces and sends its decision to meta-level at a time instant  $t_\beta^i$  the age of this decision is equal to zero,  $\Delta_i(t_\beta^i) = 0$ . If decision of the alert correlation classifier is produced later, at the time instant  $t_\alpha$  then the attribute  $\Delta_i(t_\alpha)$  takes value  $\Delta_i(t_\alpha) = (t_\alpha - t_\beta^i)$ .

The last model of data ageing is used in the intrusion detection system considered in this paper. It is important to note that for the model in question, no specific technique for learning of decision combining algorithm is necessary. Indeed, for this model, training and testing is a routine (but not trivial) procedure of learning based on dataset containing both binary and numerical attributes.

## 4 Architecture of Multi-agent IDS Software Prototype: An Outline

The architecture mentioned in the section title is described below in the style assumed by Gaia methodology implemented and extended within MASDK 3.0 software tool.

### 4.1 Basic Components of the Architecture

#### 1. Roles

- *DataSensor*—source of the raw data; performs raw data preprocessing, computation of the features, translation of the primary events and generation of the secondary events associated with the data source.
- *ObjectDataReceiver*—acceptor of the network level features;
- *DecisionProvider*—source of decisions regarding the computer security status;
- *DecisionReceiver*—acceptor of the decisions produced by *DecisionProviders*;
- *ObjectMonitor*—acceptor of information presenting the host security status.

#### 2. Protocols

- *DataTransmission*—the protocol transmitting features-related information;
- *DecisionTransmission*—the protocol transmitting decisions produced;
- *UpdateObjectInformation*—the protocol responsible for updating of the computer security status related information;

The aforementioned protocols are basic ones. The auxiliary ones are as follows:

- *AttackLogTransmission*—the protocol performing transmission of the attack log (the true labels of the attacks needed for the designed system testing);
- *OptionsProtocol*—the protocol performing adjusting of initial options determining the regime of the system operation.

### 3. Agent classes and roles to perform

The agent classes introduced in the IDS architecture and allocated the roles they have to perform are as follows:

*NetLevelAgent*—an agent class performing the *DataSensor* role intended for raw data preprocessing and extraction of the events and secondary features;

*BaseClassifiers*—an agent class assigned the *DecisionProvider* role performing source-based classification; it produces decisions when it receives an event from "its" source. This class is inherited by several subclasse that are as follows:

- *DOS\_CB*: produces decisions when it receives the event *ConnectionEvent* using *ConnectionBased* features; it is trained to detect the *DoS* attack class;
- *DOS\_TW*: produces decisions when it receives the event *TimeWindowEvent* and *TimeWindowFeatures* features; it is trained to detect *DoS* attack class;
- *DOS\_TWT*: produces decisions after receiving *TimeWindowEvent* event and *TimeWindowTrafficFeatures* features; it is trained to detect *DoS* attack class;
- *Prob\_CB*: produces decisions after receiving *ConnectionEvent* event and *ConnectionBased* features; it is trained to detect attacks of the class *Probes*;
- *Prob\_CW*: produces decisions after receiving the *ConnectionWindowEvent* event and *ConnectionWindowFeatures* features; it is trained to detect attacks of the class *Probes*;
- *Prob\_TWTr*: produces decisions after receiving *TimeWindowEvent* event and *TimeWindowTrafficFeatures* features; it is trained to detect attacks of the class *Probes*;
- *R2U\_CB*: produces decisions after receiving *tConnectionEvent* event and *ConnectionBased* features; it is trained to detect the attacks of the class *R2U*;
- *R2U\_CW*: produces decisions after receiving the *ConnectionWindowEvent* event and *ConnectionWindowFeatures* features; it is trained to detect attacks of the class *R2U*;
- *R2U\_CWT*: produces decisions after receiving the *ConnectionWindowEvent* event and *ConnectionWindowTrafficFeatures* features; it is trained to detect the attacks of the class *R2U*;
- *R2U\_TWT*: produces decisions after receiving the *TimeWindowEvent* event and *TimeWindowTrafficFeatures* features; it is trained to detect attacks of the class *R2U*.

*Metaclassifiers*:—an agent class performing the roles *DecisionReceiver* and *Decision-Provider*; it is responsible for combining decisions produced by its child classifiers (Fig.2). It is replicated into the following instances:

- *DOS\_MC*: an agent instance of the *Metaclassifier* class correlating alerts of the source-based classifiers trained for detection of *DoS* attack class;
- *Prob\_MC*: an agent instance of the class *Metaclassifier* correlating alerts of the source-based classifiers trained for detection of *Probes* attack class;

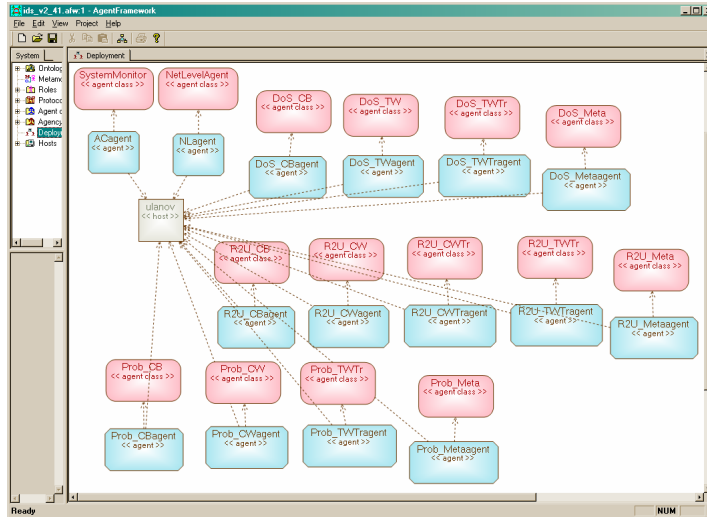


Fig. 3. IDS MAS agency configuration

- *R2U\_MC*: an agent instance of the class *Metaclassifier* correlating alerts of the source-based classifiers trained for detection of *R2U* attack class;
- *Normal MC*: an agent instance of the *Metaclassifier* class combining alerts arriving from the meta-classifiers correlating alerts of particular attack classes;

*SystemMonitor*—an agent class assigned the role *ObjectMonitor*; it provides visualization of the information about security status of the host depending on time.

The instances of the above agents are structured according to the conceptual heterogeneous alert correlation structure depicted in Fig.2. The above mentioned components represented graphically in Fig.3 determine configuration of the agents of the implemented multi-agent IDS.

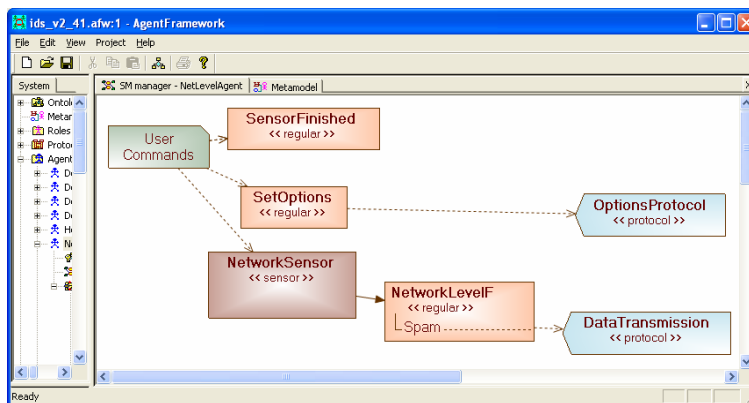


Fig. 4. Model of behavior of the agent class *NetLevelAgent*

## 4.2 Agent Classes Behavior Specification

Behavior of each agent class is specified in two layers. At the upper layer, the structure of the interaction of the state machines representing particular variants of the agent class behavior, which correspond to different agent services<sup>2</sup>, is specified. At the lower layer, each such state machine is specified in details. Correspondingly, let us describe some of the agent class services distinguishing upper and lower layers.

### 4.2.1 *NetLevelAgent* Agent Class

The basic services of this agent class are the followings (Fig.4):

- *NetworkSensor*—provides monitoring of the network traffic and generation of the primary events associated with this data source. In other words, it is responsible for dispatching of input events and sequencing of its preprocessing;
- *NetworkLevelF*—provides computation of the connection-based features and generation of the secondary events;
- *Spam* – provides forwarding of the events and feature values to the source-based classifiers.

Interface of the options of the adjustment of the *NetworkSensor* service is shown in Fig.5.

Let us describe state machines implementing the services of the *NetLevelAgent* agent class. An example of the state machine corresponding to the *NetworkLevelF* service is presented in Fig.6. In the state *Get\_Input\_Data* the newly arrived data are analyzed. After detection of the type of the data arrived, the latter are processed according to their type: the packet data are processed in *Process\_Packet* state, while connection-associated data are processed in the *Process\_Connection* state. After this,

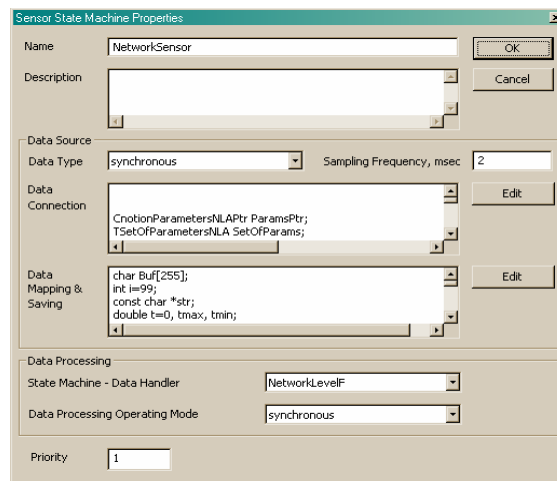


Fig. 5. Options of the service *NetworkSensor* adjusting

<sup>2</sup> Term "*agent service*" is used in multi-agent technology to denote an agent's functionality.

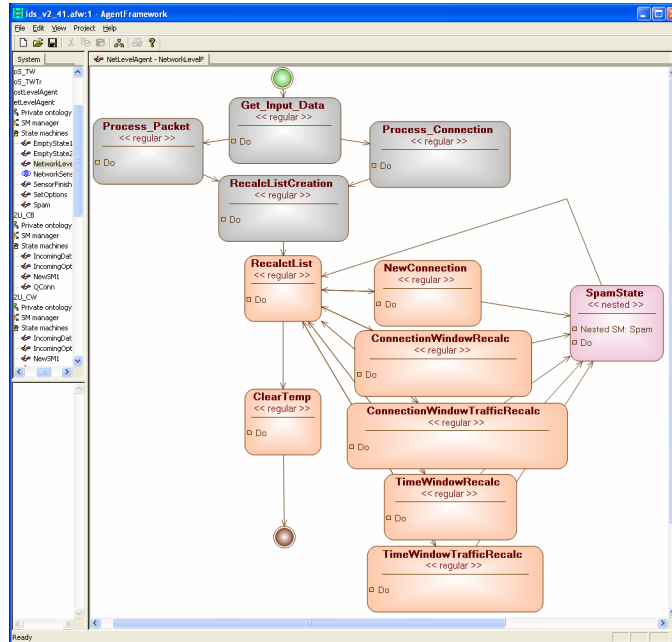


Fig. 6. State machine-based specification of the *NetworkLevelF* service

the list of all events, both primary and secondary, is formed (in the state *RecalcList-Creation*). Then, for each event stored in the aforementioned list, computation and updating of the features with the subsequent call of the service *SpamState* is carried out. This is done in the state *RecalcList*. In turn, the service *SpamState* performs forwarding the computed feature values to the source-based classifiers associated with the respective events in the above mentioned list.

#### 4.2.2 Alert Correlation Agent Classes

In general, the agent classes mentioned in the subsection title are the same; they only differ (1) in their rule bases used for alert correlation, (2) in the lists of the source-based classifiers forming their inputs and also (3) in the lists of the receivers of the

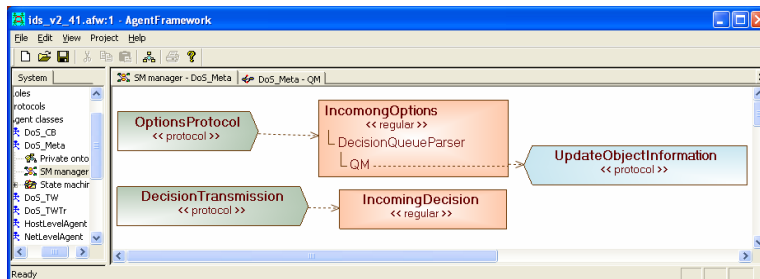


Fig. 7. Services of the agent classes responsible for meta-classification

decisions produced by alert correlation agents. The basic services of these agent classes structured as it is depicted in Fig.7 are the followings:

- *IncomingDecision*—service responsible for processing of the incoming decisions of the child classifiers of the lower layer;
- *IncomingOption* – service responsible for adjusting of the agent class options;
- *DecisionQueueParser* – service responsible for processing of the incoming decisions stored in the queue;
- *QM*—service implementing alert correlation (meta-classification functionality).

Detailed specification of the state machines implementing the aforementioned services is omitted due to the lack of the paper space.

#### 4.2.3 Source-Based Classifier Agent Class

The basic services of the *Source-based classifier* agent class are as follows:

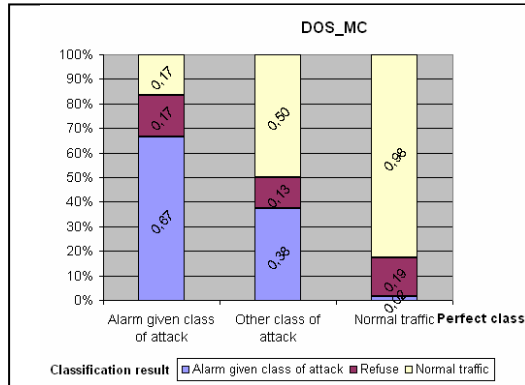
- *IncomingData*—service implementing the incoming events and data processing;
- *IncomingOption*—service responsible for adjusting of the agent class options;
- *ConnQueueParser*—service responsible for processing of the incoming decisions stored in queue (*Connection*-based, *Windows*-based);
- *QConn* – service responsible for producing decisions (*Alert* or *Normal*).

Like all the services, the aforementioned ones are specified and implemented in terms of state machines, whose description is omitted. due to lack of the paper space.

## 5 Experimental Results

The multi-alert correlation IDS MAS designed according to the above described principles and architecture was implemented using MASDK 3.0 platform providing support of the MAS technology [7]. All the classifiers composing the proposed homogeneous alert correlation structure were trained using VAM [9] and GK2 [6] algorithms. The resulting system as a whole was tested using DARPA data [3].

Some testing results are illustrated in Fig.8. These figures present information about performance quality (probabilities of correct classifications and probabilities of false alarms and signal missing) of the alert correlation classifiers dealing with inputs produced by the source-based classifiers trained for detection of attacks of particular classes. At that, data used in training procedures as "counter class" include basically normal traffic. But, if, for a source-based classifier, the difference between the sums of the weights of rules voting in favor of *Alert* and *Normal* decision is less than a selected threshold (it is computed for each particular classifier experimentally in testing procedure) then the classifier refuses to classify input data. Analysis proved that as a rule, such kind of situation actually corresponds to some other class of attacks. Fig.8 illustrates the performance quality of the alert correlation meta-classifier destined for detection of the *DoS* class of attacks. It illustrates graphically the probability distributions of correct alert detection and various types of errors. An important observation is that even if the source-based classifiers operate not very precise, at the meta-layer, where the decisions of the particular source-based classifiers are combined, the



**Fig. 8.** Evaluation of the performance quality of the *DOS\_MC* meta-classifier

with multiplicity and heterogeneity of data sources to be taken into account, several other specific features of the intrusion detection system input are critical to fill in the gap between existing models used in IDS and reality. Among these features, temporal nature, high-frequency dynamics and asynchronous nature of input are of the primary importance. These factors result in the necessity to account such an important issue as information ageing caused by the fact that input data streams arrive in IDS with various averaged frequencies and asynchronously.

The input data model considered in this paper takes into account the aforementioned factors. For such model of IDS input, the paper proposes an approach called heterogeneous alert correlation. The major idea of the approach is to organize IDS system as a structured set of interacting classifiers dealing with data received from various data sources. The first layer of this structure is composed of classifiers operating with inputs of particular data sources. Each of them is trained for detection of attacks of a fixed class (in the developed IDS software prototype, the attack classes *DoS*, *Probe*, and *U2R* are considered). Each of such specialized classifiers produces decisions of two types: "Alert" in regard to the particular class of attacks (e.g. "*DoS alert*", "*U2R alert*", etc.) or "Normal". A peculiarity of such classifiers operation is that they produce decisions in different time instants. These decisions asynchronously arrive at the second layer responsible for correlation of the alerts produced by the first layer classifiers trained for detection of the attacks of the same class. In turn, the results of the alert correlations produced by the specialized classifiers of the second layer are asynchronously forwarded to the top layer. The top-layer classifier solves intrusion detection task: it combines heterogeneous alerts of specialized alert correlation classifiers and combines them producing decision in terms of particular attack class.

Two theoretical problems should be solved to implement the described approach: (1) development of data ageing model; and (2) development of specific techniques to train alert correlation classifiers to make decisions based on asynchronous input. In the developed IDS prototype the solutions proposed by the authors in previous research are used [5, 8]. This approach was implemented within multi-agent IDS

quality of the *DoS* attack detection is increased. The same is valid for other alert correlation classifiers.

## 6 Conclusion

Though intrusion detection task is being a subject of intensive research during at least the last decade, it remains to be a problem; many important issues and peculiarities of this task have not been investigated in depth. One of the remarkable drawbacks of the existing approaches is simplified modeling of input data used in development of IDS. Indeed, along

dealing with three classes of attacks, *DoS*, *Probe* and *U2R*, and operating with input traffic. Architecture of the prototype and some experimental results are outlined.

The intended directions for future research will concern enrichment of the developed structure of interacting classifiers by the learning capabilities.

## Acknowledgement

This research is supported by grant #1993P of European Office of Aerospace R&D and Russian Foundation for Basic Research (grant 04-01-00494a).

## References

1. Bass, T.: Intrusion Detection and Multisensor Information Fusion: Creating Cyberspace Situational Awareness. *Communication of the ACM*, Vol. 43(4) (2000) 99–105
2. [http://www.ll.mit.edu/IST/ideval/data/1998/1998\\_data\\_index.html](http://www.ll.mit.edu/IST/ideval/data/1998/1998_data_index.html)
3. Cuppens, F., Mieke, A.: Alert correlation in a cooperative intrusion detection framework. *IEEE Symposium on Research in Security and Privacy* (2002)
4. Debar, H., Wespi, A.: Aggregation and Correlation of Intrusion-Detection Alerts. *RAID 2001*, LNCS 2212 (2001) 85–103
5. Gorodetsky, V., Karsaev, O., Samoilov, V.: On-Line Update of Situation Assessment: Generic Approach. *International Journal of Knowledge-Based & Intelligent Engineering Systems*. IOS Press, Netherlands, 2005 (Accepted for publication)
6. Gorodetsky, V., Karsaev, O., Samoilov, V.: Direct Mining of Rules from Data with Missing Values. *Studies in Computational Intelligence*, T.Y.Lin, S.Ohsuga, C.J. Liao, X.T.Hu, S.Tsumoto (Eds.). *Foundation of Data Mining and Knowledge Discovery*, Springer (2005) 233–264
7. Gorodetsky, V., Karsaev, O., Samoilov, V., Konushy, V., Mankov, E., Malyshev, A.: Multi-Agent System Development Kit. R.Unland, M.Klusch, M.Calisti (Editors). "Multi-Agent Technology and Software Tools", Whitestein Publishers. Accepted for publication (2005)
8. Gorodetsky, V., Karsaev, O., Samoilov, V.: On-Line Update of Situation Assessment Based on Asynchronous Data Streams. *8th International Conference on Knowledge-Based Intelligent Information & Engineering Systems*, LNAI, Vol. 3213, Springer (2004) 1136–1142
9. Gorodetski, V., Skormin, V., Popyack, L.: Data Mining Technology for Failure Prognostics of Avionics, *IEEE Transactions on Aerospace and Electronic Systems*. Volume 38, # 2 (2002) 388–403
10. Lazarevic, A., Ertöz, L., Kumar, V., Ozgur, A., Srivastava, J.: A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection. *3rd SIA Conference on Data Mining*, San Francisco, CA (2003)
11. Morin, B., Debar, H.: Correlation of Intrusion Symptoms: An Application of Chronicles. *RAID 2003*, LNCS 2820, Springer-Verlag (2003) 94–112
12. Pietraszek, T.: Using Adaptive Alert Classification to Reduce False Positives in Intrusion Detection, *RAID 04*, LNCS volume 3224 (2004) 102–124
13. Song, T., Ko, K., Alves-Foss, J., Zhang, C., and Levitt, K.: Formal Reasoning About Intrusion Detection Systems, *RAID 04*, LNCS volume 3224 (2004) 278–295
14. Valdes, A., Skinner, S.: Probabilistic Alert Correlation. W. Lee, L. Me, and A. Wespi (Eds.): *RAID 2001*, LNCS 2212, Springer-Verlag (2001) 54–68
15. Wooldridge, M., Jennings, N.R., Kinny, D.: The Gaia Methodology for Agent-Oriented Analysis and Design. *Journal of Autonomous Agents and Multi-Agent Systems*, Vol. 3. No. 3 (2000) 285–312