A Biologically Inspired Denial of Service Detector Using the Random Neural Network

Georgios Loukas and Gülay Öke

Electrical and Electronic Engineering, Imperial College London, UK

***Abstract*— Several of today's computing challenges have been met by resorting to and adapting optimal solutions that have evolved in nature. For example, autonomic communication networks have started applying biologically-inspired methods to achieve some of their self-* properties. We build upon such methods to solve the recent problem of detection of Denial of Service networking attacks, by proposing a combination of Bayesian decision making and the Random Neural Networks (RNN) which are inspired by the random spiking behaviour of the biological neurons. Our approach is based on measuring various instantaneous and statistical variables describing the incoming network traffic, acquiring a likelihood estimation and fusing the information gathered from the individual input features using different architectures of the RNN. The experiments are conducted using the CPN networking protocol which is also based on the RNN.**

I. Introduction

In recent years, Denial of Service (DoS) has become a predominant type of network security attack, which is relatively simple to launch, but particularly difficult to defend against. An attacker only needs to take control of a number of lightly-protected computers and order them to send simultaneously volumes of meaningless traffic to the victim. The defence methods try to decrease the effects of the overwhelming incoming traffic in a way that will not disturb the legitimate traffic that also arrives at the victim. This procedure can be facilitated if the attack is detected long before the destructive traffic build-up, which is why most comprehensive DoS defence systems need a detection mechanism to trigger the response procedure. This would not be needed in the case of an ideal response architecture with proactive qualities that would render a DoS attack impossible, but such a system has not been built to date, and proactive solutions are usually too expensive resource-wise to operate in the absence of an attack. A detection mechanism should monitor the traffic continuously and signal any developing attacks in the network, which should then trigger a response mechanism aiming to protect the network resources and maintain a satisfactory level of quality of service for the legitimate users. The success of a detection mechanism is determined by a number of factors, including its probability of correct detection of the attack, missed detection, and false alarm in the absence of an attack. Also, it should consume minimal resources and reach detection decisions quickly in real-time before the attack builds up.

In the technical literature, a large number of diverse methods have been proposed ranging from machine learning applications like neural networks [18], radial basis functions [17] and fuzzy classifiers [20] to statistical approaches employing autocorrelation functions [12], entropy and chi-square statistics [8], self-similarity properties [13] and energy distribution variance [19].

Here, we attempt to bridge these two general directions of DoS detection, machine learning and information gathered with statistical methods. We have built a system which uses several statistical features deemed in the literature as most significant for a DoS attack, and combines the individual decisions in a machine learning fashion. We present and compare six different implementations of it, which combine multiple Bayesian classifiers and the random neural network (RNN). Bayesian classifiers have been used before for DoS detection [9], but applied only on the rate of appearance of specific flags in the packets' headers, and in [22], where hypothesis testing was used on the spectral analysis of bitrate to detect only one very specific type of attack. In our work we present a more general approach which aggregates likelihood estimation of heterogeneous statistical features and combine them in a biologically inspired neural network structure.

The random neural network (RNN) introduced by Gelenbe in [2] is an alternative neural network model based on the spiking behaviour of the biological neuron instead of the classical approaches which assume analog transmission of signals. In this paper, we exploit the capability of the RNN to model the excitatory and inhibitory interactions among its inputs for the case of malicious incoming traffic in a network. In fact, the experiments we conduct also exploit a RNN-based networking protocol, the Cognitive Packet Network (CPN) [15], which is an autonomic Quality of Service (QoS)-driven routing protocol. In CPN each flow specifies the QoS metric that it wishes to optimise, and data payload is carried by source routed "dumb packets" (DPs), while "smart packets" (SPs) and "acknowledgment packets" (ACKs) gather and carry control information which is used for decision making. In CPN, each flow specifies its QoS requirements in the form

of a QoS "goal" and SPs associated with each flow constantly explore the network and obtain routing decisions from network routers based on observed relevant QoS information. In our experiments we use the CPN to ensure that the traffic arrives to their destination quickly using the optimal routes.

II. Selecting the Input Features

The task of DoS is a pattern classification problem, where the observed traffic is to be classified as normal or attack traffic. The Bayesian Decision theory is a basic approach used in pattern recognition problems. It assumes the availability of probabilistic descriptions of the underlying features of a problem and aims to find a decision rule which would minimise the risks encountered by the decision taking process [6]. For a two-category classification problem, let us assume we can measure an observation value x for a certain feature, and we have to decide whether the observed data point falls into the normal (w_N) or DoS (w_D) category. The practical utilisation of Bayesian classifiers in the two-category classification problem entails evaluating the likelihood ratio $\Lambda(x) = \frac{f(x|w_D)}{f(x|w_N)}$ and comparing it with a threshold T, x is assigned to category w_D if $\Lambda(x) > T$, or to w_N otherwise.

For any pattern classification problem, the selection of useful and information bearing input features constitutes a significant part of the solution. In our scheme we have used the features which capture both the instantaneous behaviour and the longer-term statistical properties of the traffic, and are easily measurable without high computational cost. Since the goal of the attacker is to deny or degrade the service for legitimate users by overwhelming either the processing or the networking resources of a victim network, a DoS detection mechanism should not further aggravate this condition with considerable overhead. Being able to measure them quickly is also a factor, since the faster detection decisions are taken the easier it is for the defence mechanisms to counter the attack.

- ***Bitrate***. An unexpectedly high rate of incoming traffic is the most conspicuous indicator of a flooding DoS attack. Similar measurements, such as the number of packets per flow are often used in detection mechanisms [11].
- ***Increase in Bitrate***. Another obvious characteristic of DoS attacks is the sudden and sustained rate of increase of the bitrate of the incoming traffic. For example, flooding attacks start with a long period of increasing bitrate, while in pulsing attacks there are consecutive periods of increasing and decreasing bitrate.
- ***Entropy***. It has been reported in the technical literature that the entropies of normal internet traffic and DoS traffic differ significantly [8]. In this work, we compute the entropy of the value of the incoming bitrate at the nodes we monitor according to [1]: $E = -\sum_i f_i log f_i$, where f_i are the histogram values obtained for the bitrate, as explained in Section IV. This is expected to yield a higher value when the probability distribution expands over a wider range of values, indicating an increase in uncertainty.
- ***Hurst Parameter***. Another statistical attribute which exhibits different behaviour for normal and attack traffic is the self-similarity. Hurst parameter is an indicator of the self similarity of traffic and can be used in DoS detection. For example, Xiang et al. [13] use the variations of the Hurst parameter of the number and the size of packets to detect attacks. In our approach we compute the actual value of the Hurst parameter for the incoming bitrate, for which we have used the (R/S) analysis, as described in [14]. If x is the bitrate of the incoming traffic, n is the observation time, and N is the total number of observation points, then (R/S) is given by:

$$(R/S)_N = \frac{\max\limits_{1\le n\le N} \sum\limits_{n=1}^{N}(x-\bar{x}) - \min\limits_{1\le n\le N} \sum\limits_{n=1}^{N}(x-\bar{x})}{\sqrt{\dfrac{\sum\limits_{n=1}^{N}(x-\bar{x})^2}{N}}}$$

  The Hurst parameter and $(R/S)_N$ are related by $(R/S)_N = cN^H$, which for the selected value $c = 1$ becomes $H = log_N((R/S)_N)$.
- ***Delay***. A natural consequence of high bitrate and building up of congestion is the increase in the packet delays. Still, to our knowledge it has not been used before as an attack indicator. For the fastest and least invasive way to detect changes in the delays, the node we monitor sends constantly packets at a very low rate to all its direct neighbours. By measuring the average round trip time (RTT) for the acknowledgments to return, we have a clear indication of the congestion near the node.
- ***Delay Rate***. As with bitrate, depending on the type of the attack and for its whole duration, the packet delays are expected to undergo significant changes. We are not aware of existing work using the change of the delay as a detection feature, but we consider it a natural next step.

III. Offline Statistical Information Gathering

The probabilistic description of the network is acquired in the statistical information gathering phase which mainly consists of two steps. First, the probability density function (pdf) values are obtained for both normal and attack traffic and then the likelihood ratios are calculated based on the pdfs. At each victim candidate of the network, the incoming traffic is analysed offline to collect this statistical information. Estimates of probability density functions in the form of histograms for both normal and attack traffic are computed

for each of the input features described in Section II. The pdfs are denoted by $f_{feature}(x|w_N)$ and $f_{feature}(x|w_D)$, where *feature* is replaced by bitrate, increase in bitrate (bit acceleration), entropy, Hurst parameter, delay and delay rate respectively, x is the measured value of the feature from the available traffic data, w_N denotes the normal traffic and w_D the DoS traffic.

In the second step, the probability density function estimates obtained above for each input and for both traffic types are used to compute the likelihood ratios $l_{feature}$ of each feature: $l_{feature} = \frac{f_{feature}(x|w_D)}{f_{feature}(x|w_N)}$. These likelihood ratios are later used in real-time by the decision taking mechanism (Section IV). Likelihood ratios, actual values and quantised actual values (histogram category values) of the features are used also in the training of RNNs.

IV. Detection decision

The statistical information collected about the network off-line is utilised during the decision taking process, which comprises two steps. In the first step, decision for each feature is given individually, and the individual decisions are then combined in an information fusion step to yield a final outcome for the state of the traffic. The numerical values of the features are measured in real-time and a likelihood value for each feature is computed. Then, these values are aggregated in a higher-level decision taking step which we realized by employing a feedforward (f-RNN) and a recurrent (r-RNN) architecture of the RNN, with input the individual likelihood values, histogram category values and actual values, for a total of six different implementations of our generic detection technique.

The random neural network (RNN), proposed by Gelenbe [2] is a computational paradigm, inspired by the random spiking behaviour of the biological neurons. The RNNs are computationally efficient structures and they represent a better approximation of the true functioning of a biophysical neural network, where the signals travel as spikes rather than analog signals. The strong analogy between queuing networks and the RNN make it a powerful tool for dealing with problems where excitation and inhibition among problem inputs are prevalent. The RNN has been successfully applied in various problems, including image processing [4], pattern recognition [21], and optimisation [3].

In the RNN, neurons exchange positive and negative impulse signals, with unit amplitude, which represent excitation and inhibition respectively. Neurons accumulate signals as they arrive and positive signals are cancelled by negative signals. Neurons may fire if their potential is positive, to send signals either to other neurons or outside the network. In RNN a signal may leave neuron i for neuron j as a positive signal with probability $p^+(i,j)$, as a negative signal with probability $p^-(i,j)$, or may depart from the network with probability $d(i)$, where $p(i,j) = p^+(i,j) + p^-(i,j)$ and $\sum_j p(i,j) + d(i) = 1$. Positive and negative weights are computed by:

$$w^+(j,i) = r(i)p^+(i,j) \geq 0$$
$$w^-(j,i) = r(i)p^-(i,j) \geq 0$$

where $r(i)$ is a Poisson firing rate, with independent identically exponentially distributed interimpulse intervals:

$$r(i) = \sum_j w^+(i,j) + w^-(i,j)$$

The weights w may be interpreted in a way analogous to the weights in artificial neural networks (ANNs), but they actually represent excitatory and inhibitory signal emission rates.

The steady state probability that the neuron i is excited is defined by $q_i = lim_{t\to\infty} Pr[k_i(t) > 0]$ which is computed to be $q_i = \frac{N(i)}{D(i)}$, where

$$N(i) = \sum_j q_j w^+(j,i) + \Lambda(i)$$
$$D(i) = r(i) + \sum_j q_j w^-(j,i) + \lambda(i)$$

with $\Lambda(i)$ and $\lambda(i)$ denoting the rates of exogenous excitatory and inhibitory signal inputs into neuron i, respectively.

RNNs can be designed in both feedforward and recurrent architectures (f-RNN and r-RNN). In our work we have implemented both architectures for the fusion of inputs to compare the results.

The f-RNN architecture we implemented consists of an input layer of six nodes, a hidden layer with twelve nodes and an output layer with two nodes. Each output node stands for a decision; attack or not. The final decision about the traffic is determined by the ratio of the two output nodes. The r-RNN structure we have designed consists of two layers, an input layer with twelve nodes and an output layer with two nodes. In the input layer, there are two nodes for each input variable; one for the excitatory signals and one for the inhibitory signals. Each node sends excitatory signals to nodes of the same type and inhibitory to the rest. At the output layer, one node sums up the excitatory and a second the inhibitory signals. Just as in the feedforward case, the decision is given by computing the ratio of the two output nodes.

For the implementation of the RNNs, we have used software developed in [25]. For both f-RRN and r-RNN we have used three different types of input:

- **Likelihood ratios**. The likelihood ratios $l_{feature}$ for the six input features are obtained by measuring their real-time values and resorting to the likelihood ratio values stored during the offline statistical information gathering phase described in section III.

- **Histogram categories**. These refer to the histogram intervals used to estimate the pdfs described in section III and they essentially quantise the available values.
- **Actual values**. For the sake of comparison we have also investigated the case in which the actual values of the input features are directly fed in the two types of RNN.

V. Experimental Evaluation

We have implemented and evaluated our detection mechanism on a networking testbed in our laboratory, which consists of 46 nodes connected with 100 MBits/sec links. We chose a specific node to play the role of the victim while the rest of the nodes send traffic to it according to a variety of datasets that we tried. In our experiments, we utilised traffic traces of DoS attacks designed both in our laboratory and by other academic sources: (i) attack traffic of our design slowly increasing, (ii) attack traffic downloaded from [24] representing rapid flood, and (iii) attack traffic downloaded from the same source representing a pulsing attack, with traffic rates reciprocating between very low and very high values. We have used two types of RNN architecture, f-RNN and r-RNN, with three types of input for each feature: likelihood ratios, quantised histogram category values and actual values. Each experiment lasts 120 sec. In the attack cases, to illustrate the difference in the traffic and to see graphically the operation of the detection mechanism, we start with normal traffic on which we superimpose attack traffic for the time period between 50s and 100s. The last 20s the network returns to its normal operation, as the attack sources stop sending traffic to the victim. We used a sampling time of 2 sec.

Table 1 summarises the performance results of the detection mechanism in terms of average correct detection, false alarm rates and detection delays, while figures 2-4 show the real-time detection decisions as time progresses. The y axis in these graphs is in logarithmic scale with the detection metric being the ratio of the two output RNN nodes as described in section IV. The decision threshold over which an attack is signalled is the RNN output ratio of 1. The closer the detection ratio is to 1 the less certain the mechanism is of its detection decision.

Figures 2-4 correspond to the three attack datasets that we used. All three implementations of the RNN detect the attacks quickly and have minimal missed detections and false alarms. The r-RNN with actual values had high detection rates, no false alarms and the lowest detection delay for all datasets, but also provided the lowest degree of certainty for its decisions by yielding values close to 1. The use of the quantised values of histogram categories appear to improve the correct detection rate even further, but at the expense of a few false alarms, while the likelihood value methods for both f-RNN and r-RNN performed at about the same level in both respects, but took a little longer to detect the pulsing attack (dataset3).

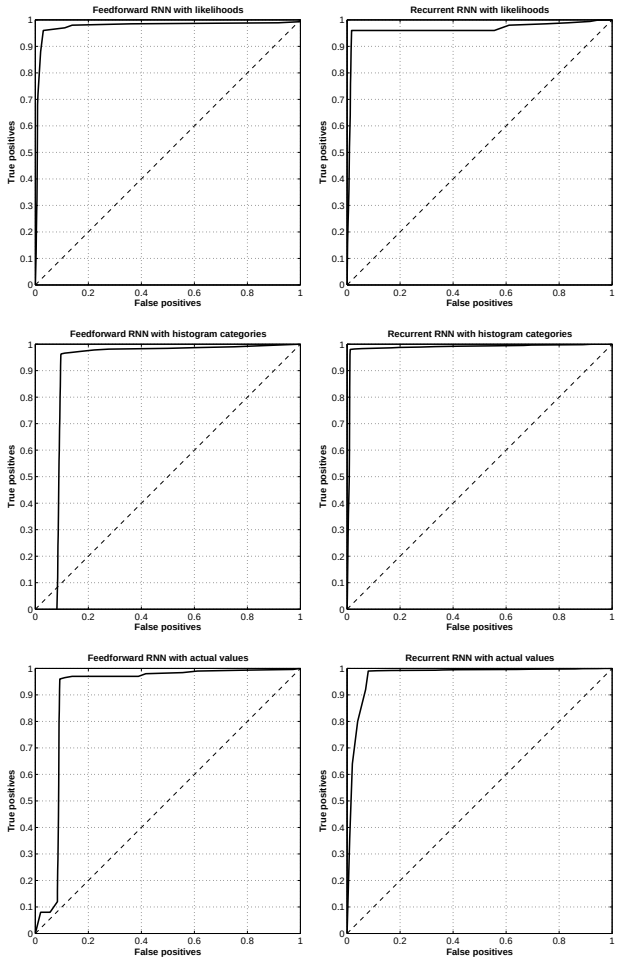

Fig. 1. ROC curves for Dataset2 (attack)

We have also plotted some representative ROC curves for the six variations, based on one of the datasets (Fig. 1). The ROC curves are graphs of false positives versus true positives, to which we have referred in this paper as false alarms and correct detections respectively. In an ideal system, the ROC curve should rise immediately from (0,0) to (0,1) and continue to (1,1). We see this behaviour in all six variations, with a slight advantage of the r-RNN with histogram categories and f-RNN with likelihood values.

Looking at the overall performance of our detection mechanism, it emerges that apart from the f-RNN with actual values implementation, all methods are quite powerful detectors of attack traffic. In general, the r-RNN implementations are observed to be more accurate and slightly faster.

VI. Conclusions

We have described the design of a generic DoS detection scheme which employs multiple Bayesian classifiers and the

	False Alarm			Correct detection			Detection Delay (s)		
Detection implementation	Dataset1	Dataset2	Dataset3	Dataset1	Dataset2	Dataset3	Dataset1	Dataset2	Dataset3
f-RNN likelihood values	0.17	0.11	0.08	0.96	0.96	0.84	2	0	12
f-RNN histogram categories	0.03	0.11	0.03	0.92	1	0.80	4	0	8
f-RNN actual values	1	1	1	1	1	1	0	0	0
r-RNN likelihood values	0.06	0.11	0.03	0.96	0.96	0.80	2	0	12
r-RNN histogram categories	0.06	0.06	0.06	0.96	1	0.88	2	0	8
r-RNN actual values	0	0	0	0.92	1	0.84	2	0	6

TABLE I

COMPARISON OF THE DIFFERENT DETECTION IMPLEMENTATIONS, IN TERMS OF FALSE ALARM AND CORRECT DETECTION RATES

biologically inspired RNNs. We first select input features to capture both the instantaneous behaviour and the longer-term statistical properties of the traffic and in an offline information gathering step we obtain the probability density function estimates and evaluate likelihood ratios. Then, during the decision taking step, we measure the features of the incoming traffic to reach detection decisions according to each feature. These are combined into an overall detection decision using both feedforward and recurrent architectures of RNN. The experiments we conducted showed that our mechanism is able to detect DoS threats in a timely fashion and can very quickly identify the end of an attack, at least for the range of attacks that we investigated.

REFERENCES

[1] Shannon C.E. and Weaver W. (1963) *The Mathematical Theory of Communication*. University of Illinois Press.

[2] Gelenbe E. (1993) Learning in the Recurrent Random Neural Network. *Neural Computation*, **5**, pp. 154-164.

[3] Gelenbe E., Koubi V., and Pekergin F. (1993) Dynamical random neural network approach to the traveling salesman problem. *in: Systems, Man and Cybernetics, 1993. 'Systems Engineering in the Service of Humans', IEEE Conference Proceedings*, **2**, pp. 630-635.

[4] Cramer C., Gelenbe E., and Bakircioglu H. (1996) Low bit-rate video compression with neural networks and temporal subsampling. *Proc. IEEE*, **84(6)**, pp. 1529-1543.

[5] Haykin S. (1999) *Neural Networks A Comprehensive Foundation*, Prentice-Hall Inc., pp. 143-146.

[6] Duda R.O., Hart P.E. and Stork D.G. (2001) *Pattern Classification*, John-Wiley and Sons, pp. 20-214.

[7] Morein W.G., Stavrou A., Cook D.L., Keromytis A.D., Mishra V. and Rubenstein D. (2003) Using graphic Turing tests to counter automated DDoS attacks against Web servers. *Proc. 10th ACM Int'l Conference on Computer and Communications Security (CCS '03)*, pp. 8-19.

[8] Feinstein L., Schnackenberg D., Balupari R. and Kindred D. (2003) Statistical Approaches to DDoS Attack Detection and Response. *Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX'03)*.

[9] Noh S., Lee C., Choi K. and Jung G. (2003) Detecting Distributed Denial of Service (DDoS) Attacks through Inductive Learning. *Lecture Notes in Computer Science*, **2690**, pp. 286-295.

[10] Gelenbe E., Gellman M and Loukas G (2004) Defending networks against denial of service attacks. In E.M. Carapezza, editor, *Proceedings of the Conference on Optics/Photonics in Security and Defence (SPIE)*, Unmanned/Unattended Sensors and Sensor Networks, **5611**, pp. 233-243.

[11] Kim M., Na H., Chae K., Bang H. and Na J (2004) A Combined Data Mining Approach for DDoS Attack Detection. *Lecture Notes in Computer Science*, **3090**, pp. 943-950.

[12] Li M. (2004) An Approach to Reliably Identifying Signs of DDOS Flood Attacks Based on LRD Traffic Pattern Recognition. *Computers and Security*, **23**, pp. 549-558.

[13] Xiang Y., Lin Y., Lei W.L. and Huang S.J. (2004) Detecting DDOS attack based on Network Self-Similarity. *IEE Proceedings in Communication*, **151**, pp. 292-295.

[14] Cajueiro D.O. and Tabak B.M. (2004) The Hurst Exponent over Time:Testing the Assertion That Emerging Markets Are Becoming More Efficient. *Physica A*, **336**, pp. 521-537.

[15] Gelenbe E., Lent R. and Nunez A. (2004) Self-aware networks and QoS. *Proceedings IEEE*, **92(9)**, pp. 14781489.

[16] Gelenbe E., Gellman M. and Loukas G. (2005) An autonomic approach to denial of service defence. *In Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, pp. 537-541.

[17] Gavrilis D. and Dermatas E. (2005) Real-time detection of distributed denial-of-service attacks using RBF networks and statistical features. *Computer Networks*, **48**, pp. 235-245.

[18] Jalili R., Imani-Mehr F., Amini M. and Shahriari H.R. (2005) Detection of Distributed Denial of Service Attacks Using Statistical Pre-processor and Unsupervised Neural Networks. *Lecture Notes in Computer Science*, **3439**, pp. 192-203.

[19] Li L. and Lee G. (2005) DDoS Attack Detection and Wavelets. *Telecommunication Systems*, **28:3(4)**, pp. 435-451.

[20] Wei W., Dong Y., Lu D. and Jin G. (2006) Combining Cross-Correlation and Fuzzy Classification to Detect Distributed Denial-of-Service Attacks. *Lecture Notes in Computer Science*, **3994**, pp. 57-64.

[21] Teke A. and Atalay V. (2006) Texture Classification and Retrieval Using the Random Neural Network Model. *Computational Management Science*, **3(3)**, pp. 193-205.

[22] Chen Y. and Hwang K. (2006) Collaborative detection and filtering of shrew DDoS attacks using spectral analysis. *Parallel Distrib. Comput.*, **66**, pp. 1137-1151.

[23] Gelenbe E. and Loukas G. (2007) Self-Aware Approach to Denial of Service Defence. *Computer Networks*, 51, pp.1299-1314.

[24] UCLA CSD packet traces: http://www.lasr.cs.ucla.edu/ddos/traces /public/usc/.

[25] H. Abdelbaki: Matlab simulator for the RNN, http://www/cs/ucf.edu/ ahossam/rnnsim.

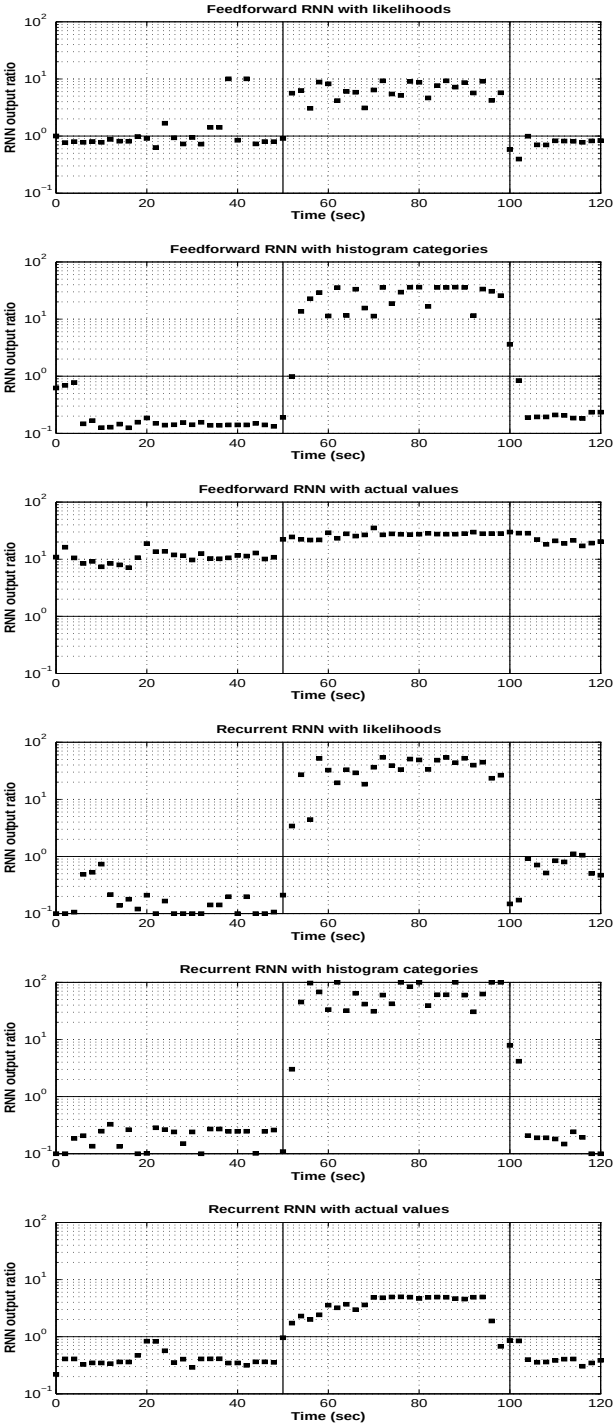

Fig. 2. Detection results for Dataset1 (attack)

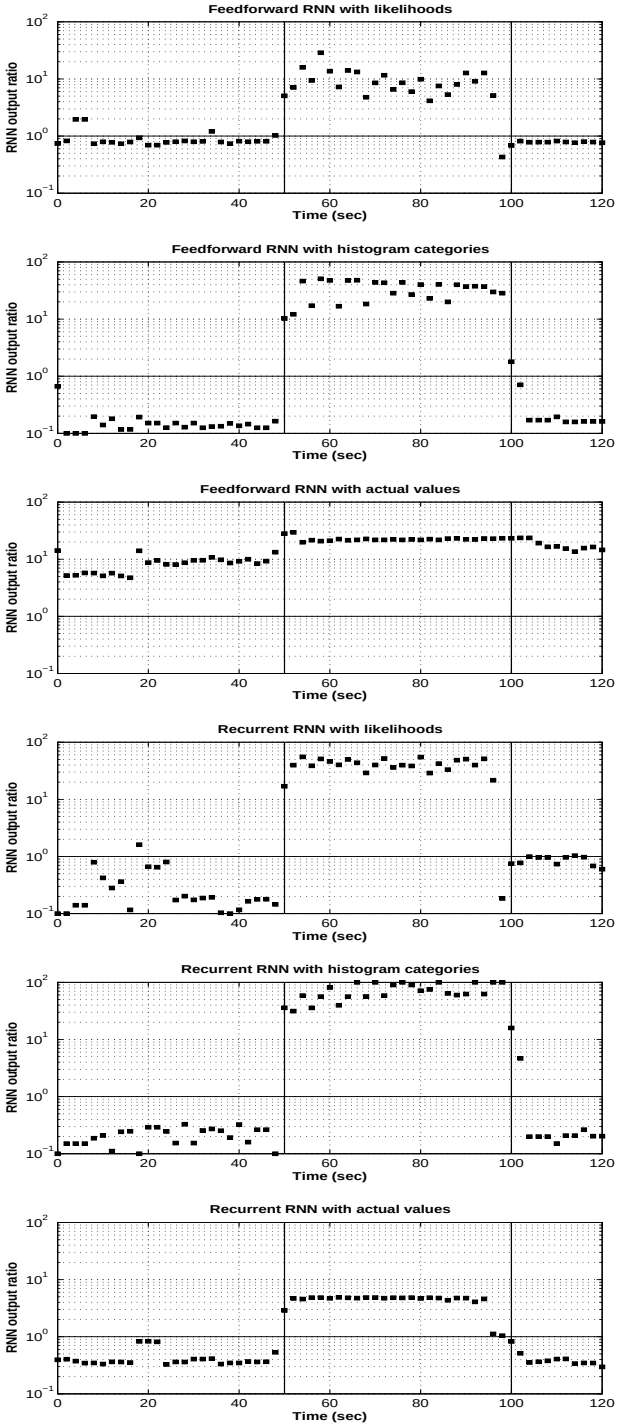

Fig. 3. Detection results for Dataset2 (attack)

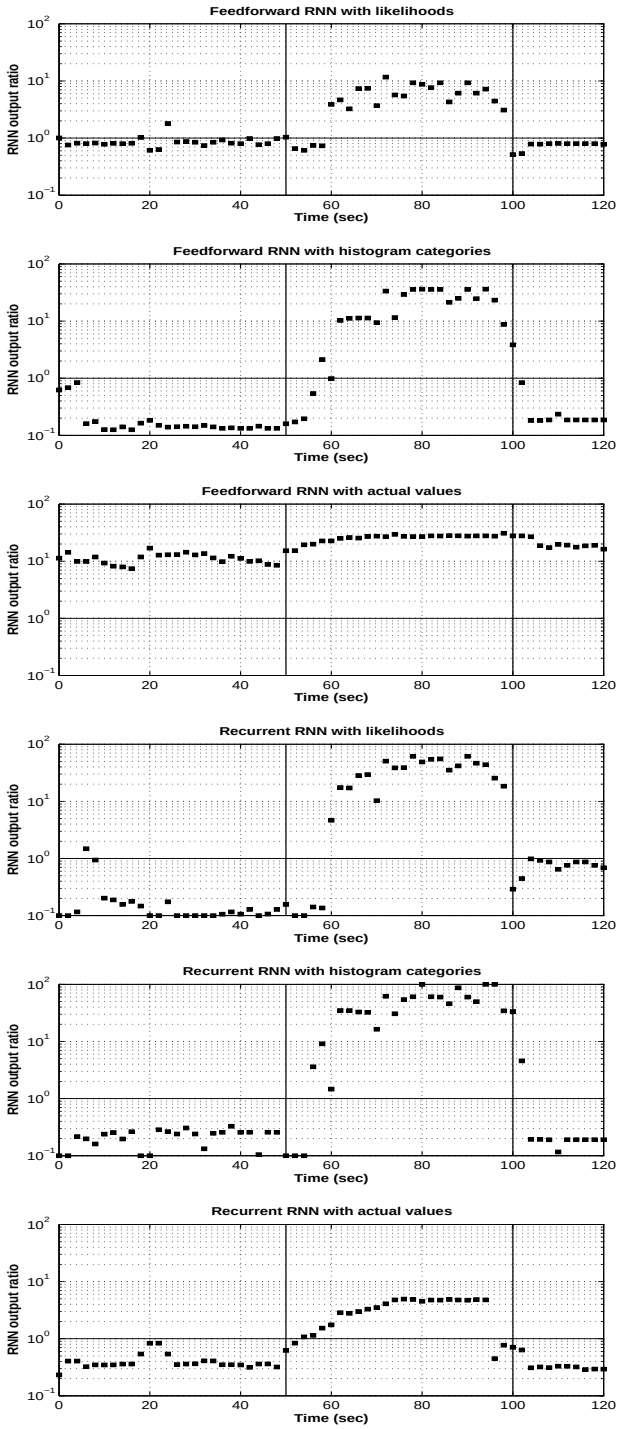

Fig. 4. Detection results for Dataset3 (attack)