Adaptive Real-Time Network Monitoring System: Detecting Anomalous Activity with Evolving Connectionist System

Muhammad Fermi Pasha[1], Rahmat Budiarto[1], Mohammad Syukur[2], and Masashi Yamada[3]

[1] School of Computer Sciences, University of Sains Malaysia, 11800 Minden, Pulau Pinang, Malaysia
{fermi,rahmat}@cs.usm.my
[2] Faculty of Mathematics and Natural Sciences, University of Sumatera Utara, Medan 20155, Sumut, Indonesia
mhdsyukur@usu.ac.id
[3] School of Computer and Cognitive Sciences, Chukyo University, 101 Tokodachi, Kaizu-cho, Toyota, 470-0383, Japan
myamada@sccs.chukyo-u.ac.jp

Abstract. When diagnosing network problems, it is desirable to have a view of the traffic inside the network. This can be achieved by profiling the traffic. A fully profiled traffic can contain significant information of the network's current state, and can be further used to detect anomalous traffic and manage the network better. Many has addressed problems of profiling network traffic, but unfortunately there are no specific profiles could lasts forever for one particular network, since network traffic characteristic always changes over and over based on the sum of nodes, software that being used, type of access, etc. This paper introduces an online adaptive system using Evolving Connectionist Systems to profile network traffic in continuous manner while at the same time try to detect anomalous activity inside the network in real-time and adapt with changes if necessary. Different from an offline approach, which usually profile network traffic using previously captured data for a certain period of time, an online and adaptive approach can use a shorter period of data capturing and evolve its profile if the characteristic of the network traffic has changed.

Keywords: Adaptive System, Distributed Network Monitoring, Network Anomaly, Evolving Connectionist Systems.

1 Introduction

Over the past decades network technologies has grown considerably in size and complexity. More industries and organizations depend on the performance of its network to run their activities. Network problems mean less productivity and therefore it is necessary to manage and monitor the network so that it is consistently up and running. In managing a network, it is very crucial to control its performance, while the performance itself is dependent on traffic assessment, hence profiling the traffic would be one initial important aspect to be taken to secure the network.

J. Filipe et al. (Eds.): ICETE 2005, CCIS 3, pp. 113–125, 2007.

But simply profile the traffic only gives solutions for certain period of time, since as time goes the characteristic of the network traffic will change, and network administrator will need to re-profile the traffic to adapt with the changes. The main factors of these changes are nodes additions into the network, different types of software used, devices upgrades, topological changes, and different type of network access. Between these factors, nodes additions, types of access and types of software are causing the most significant changes to network traffic characteristic. For example, let say a corporate network has 100 computers connected on the first year, as the corporate grows, an additional of another 200 computers added into the network could definitely change the characteristic of the network traffic in the second year. These changes would invalidate the current profile and therefore it cannot be used to detect anomalous activity inside the network. Network administrator will need to either re-profile the traffic or manually edit the profile, which would be an arduous task to do.

We develop an adaptive and real-time network monitoring system with capabilities to profile the traffic in online lifelong mode and evolve the profile if significant traffic changes occurred. Furthermore, with the use of the profile, our system also try to detect anomalous activity inside the network in real-time.

With the ability to evolve, the system can be installed in any network without prior knowledge about the network traffic characteristics, just let the system grows as the network grows. This enabling the system to conduct distributed monitoring across different network (especially different network segment) without have to know the traffic's characteristic of each network segment in advance.

The rest of the paper is organized as follows. In section 2 we review some related work on network traffic profiling as well as network anomaly detection. Section 3 presents our connectionist engine architecture. Section 4 mentions about our system's scheme concept as a distributed application. Section 5 contains discussion on the obtained results and further analysis. Finally we summarize our conclusions and future work in section 6.

2 Related Work

Many research works have been devoted to automate the process of profiling network traffic and it was quite successful in general. The approaches taken are ranging from using statistical method with K-Means clustering and approximate distance clustering (Marchette, 1999), using data mining techniques to mine the network traffic and generate the profiles in terms of rules (Pasha et al., 2004), the use of three different approaches to specifically profile network application behaviour using rough sets, fuzzy C-Means clustering and Self Organizing Maps (Lampinen et al., 2002), etc. Some of it are already applied an artificial intelligence (AI) techniques and has shown a promising result. The drawbacks of these attempts are it is done in offline mode and pertinent to the network current behaviour.

An offline mode approach is usually composed of the following steps and each step is done separately:

1. Data Collection (for certain period of time).
2. Analyze the collected data.

3. Generate (automatically) or create (manually) the profile based on the analyzed data result.

This process is often time consuming and still requires a highly specialized network administrator on deciding when is the right time to conduct the data collection phase, analyzing the data and finally automatically using AI tools or manually extract the profile from the analyzed results. When the profiles outdated, the process will need to be repeated.

There is also Bro system developed primarily by Vern Paxson from University of California at Berkeley (Paxson, 1998), which focuses on intrusion detection on high stream bandwidth. It is a Linux/Unix based application that uses its own scripting language called Bro language to write the rule policy. Although Bro comes with predefined rules for detecting common anomalous activity, network administrator is still needed to write his own additional rule to cope with the network traffic characteristic in his network.

3 Connectionist Engine

An engine that facilitates learning in online mode is modelled in a connectionist way using Evolving Connectionist Systems (ECOS). ECOS are a connectionist architecture that enables us to model evolving processes and knowledge discovery. An evolving connectionist system can be a neural network or a collection of such networks that operate continuously in time and adapt their structure and functionality through a continuous interaction with the environment and with other systems (Kasabov, 2003).

The engine consists of two modules, and each is designed for a different purpose. The first module is the adaptive module for rule creation and adaptation, and the second one is the real-time detection module for online monitoring and anomaly detection.

Figure 1 depicts our ECOS based connectionist model architecture.

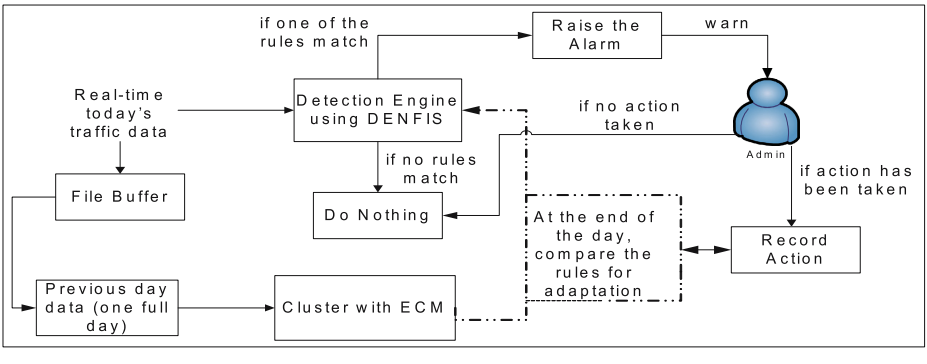


Fig. 1. ECOS based Connectionist Model Engine Architecture

3.1 Data Stream Online Clustering Using ECM

This is part of the first connectionist model module along with rule creation and adaptation functionality which are described in the next subsection. A packet capturing component is deployed to passively captured network traffic data stream in promiscuous mode. After being filtered, the data will be further clustered with one of the ECOS branches for online unsupervised clustering named Evolving Clustering Method (ECM). ECM is a fast one-pass algorithm for dynamic clustering of an input stream of data, where there is no predefined number of clusters. It is a distance-based clustering method where the cluster centres are presented by evolved nodes in an online mode (Song and Kasabov, 2001).

We had done a bit modification on the original ECM algorithm to fits our needs. Below is the algorithm:

Step 1. If it is not the first time, initialise the cluster centre C_{Cj}, $j = 1,2,\ldots,n$, that already produced before, and then go straight to Step 3. Else, go to Step 2 to start creating clusters.

Step 2. Create the first cluster C_1 by simply taking the position of the first example from the input data stream as the first cluster centre C_{C1}, and setting a value 0 for its cluster radius R_{U1}

Step 3. If all examples from the data stream have been processed, the clustering processes finishes. Else, the current input example, $\boldsymbol{x}_i$, is taken and the normalized Euclidean distance D_{ij}, between this example and all n already created cluster centres C_{Cj},

$$D_{ij} = \| \boldsymbol{x}_i - C_{Cj} \| \tag{1}$$

where $j = 1,2,3,\ldots, n$, is calculated.

Step 4. If there is a cluster C_m with a centre C_{Cm}, a cluster radius R_{Um} and distance value C_{C1} such that:

$$D_{im} = \| \boldsymbol{x}_i - C_{Cm} \| = min \{D_{ij}\} = min \{ \| \boldsymbol{x}_i - C_{Cj} \| \}$$

$$\text{for } j = 1,2,3,\ldots,n; \tag{2}$$

and

$$D_{im} < R_{Um} \tag{3}$$

when the current $\boldsymbol{x}_i$ belong to this cluster, then go back to Step 3.

Step 5. Find a cluster C_a (with a centre C_{Ca} a cluster radius R_{Ua}, and a distance value D_{ia} which has a minimum value S_{ia}:

$$S_{ia} = D_{ia} + R_{Ua} = min\{S_{ij}\}, j = 1,2,3,\ldots, n \tag{4}$$

Step 6. If S_{ia} is greater than 2 x $Dthr$, the example $\boldsymbol{x}_i$ does not belong to any existing cluster. Then repeat the process from Step 2 to create a new cluster.

Step 7. If S_{ia} is not greater than 2 x $Dthr$, the cluster C_a is updated by moving its centre, C_{Ca}, and increasing its radius value R_{Ua}. The updated radius R_{Ua}^{new} is set to be equal to $S_{ia}/2$ and the new centre C_{Ca}^{new} is located on the line connecting input vector $\boldsymbol{x}_i$ and the old cluster centre C_{Ca}, so that the distance from the new centre C_{Ca}^{new} to the point $\boldsymbol{x}_i$ is equal to R_{Ua}^{new}.then go back to Step 3.

A bit justification for terms "online" that we used in the clustering process, it is not like most online system terminology. In online video surveillance, for instance, every one/two second the captured image will be processed immediately to identify whether the image has something abnormal on it using some sort of image object recognition techniques. But for online network traffic clustering, the underlying data distributions in seconds or even in minutes count are not informative enough to be processed. Our system requires at least one day traffic data.

There are two types of traffic data model to be clustered. The first one is overall data (without filtering) and application based (http, ftp, icmp, ssh, NetBIOS, etc.) data. The clustering process will then performed on each model by using captured time, total packet and its size information.

3.2 Rule Creation and Evolving Procedure

Our system uses Takagi-Sugeno fuzzy rule type (Takagi and Sugeno, 1985) to forge the profile. There will be 7 resulted profiles (each profile was named the day to which the profile is referring) in the system on the first week running. These profiles are then evolved (following the procedure which will be described shortly after this) as the time goes to accurately describe each day's traffic behaviour in that particular network.

Number of rules in the profiles depends on number of clusters resulted from the clustering process. Two membership function (MF) of time (μt) and total packets (μs) are generated along with the rules. If in the future the rules were evolved, the MFs will also evolve. In creating the rule, the format is:

IF (A_{12} is T_1 OR A_{11} is T_2) AND (A_{21} is S_1) THEN *status* is C

where T_1 and T_2 defined by the fuzzy membership function μt, while S_1 defined by the fuzzy membership function μs.

The antecedents are the captured time position and the total packets obtained from the resulted cluster, while the consequent part is the state (normal, abnormal or uncertain). Figure 2 shows how the rules extracted along with both MFs (μt and μs) from the resulted clusters 2D space. Both the MFs are using triangular membership function.

The strategies for evolving procedures are quite straightforward. It consists of the following:

1. Compare the newly resulted profile with last week profile, and mark all the changes.
2. Check for generated Warning Alarm (WA) or Critical Alarm (CA) and recorded actions by the administrator and match every event with the previously marked profile changes (how this alarms created will be described in the next subsection).
3. If an action is taken by the administrator, keep the old rules unchanged. Else, evolve the rules by:

$$A_{VJn}^{new} = A_{VJn}^{old} \pm \| A_{VJn}^{current} - A_{VJn}^{old} \| \quad (5)$$

where n = sets of profile changes event without action, and each is calculated such that A_{VJ} are the first antecedent if $_V$ =1 (in which $_J$ = 1 & 2) and the second antecedent if $_V$ =2 (in which $_J$ = 1).

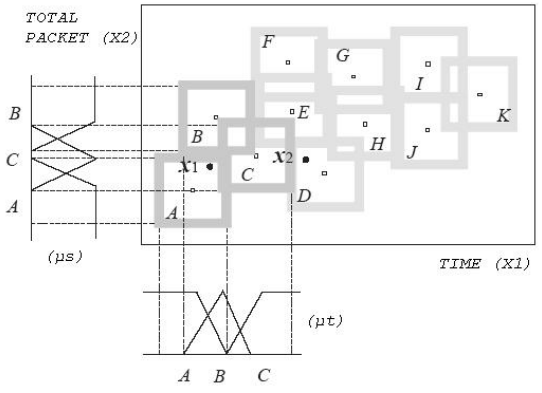


Fig. 2. Rule Extraction Scheme from 2D Space of the Resulted Clusters

Network administrator is also can evaluate and modify the resulted profiles if necessary. This would avoid black box symptoms like most connectionist system do (especially neural network based system) and network administrator can derive why such alarms was raised based on the profile on that day.

3.3 DENFIS for Detecting Anomalous Activity

The real-time detection module uses a modified Dynamic Evolving Neuro-Fuzzy Inference Systems (DENFIS) model engine for real-time anomalous activity detection. DENFIS uses Takagi-Sugeno inference method and utilises ECM in partitioning the input space (Kasabov and Song, 2002).

The engine will process every 5 minutes traffic data streaming in real-time to detect anomaly. Our modified DENFIS model engine was deployed using profiles created by the first connectionist module. It utilises alarm trigger mechanism specifically developed for the engine.

In process of adapting its structure, the profiles are inserted and adapted at the same time into the engine at the beginning of each day. If new rules were inserted or old rules were adapted, the MFs shape will also adjusted to evolve accordingly through gradient descent algorithm and a method given by (Purvis et al., 1999).

The system performs network-based anomaly detection. At presents, our system is not intended to detect signature based intrusion detection. We only focus on analyzing the traffic flows inside the network and detecting anomalous activity which was unusual. With this methodology, the system can detect Denial of Service (DoS) attack by analyzing the http traffic, detecting network device (such as router, hub, switch, etc.) failure by checking an extremely low traffic, detecting internet-worms which propagate through open file shares by analyzing NetBIOS packets statistics, detecting Novel NetWare server down (if applicable) by analyzing a low IPX type of packets, and a possibilities of other network traffic flows types of anomaly.

When the system is installed in new network, it will then assume the current traffic is normal and the engine will start to evolve under supervision of the network administrator. Basically there are two types of alarms that the system will generate.

The first one is the less important called Warning Alarm (WA) and the second one is Critical Alarm (CA) which requires an urgent immediate attention from network administrator.

The mechanisms for triggering the alarms are divided into two categories, alarms generation in peak time and in off-peak time. We are applying an extended standard working hours adopted by most companies and organizations (7 am to 9 pm) as the peak time and the rest are defined as off-peak time. The details are described as follows:

- Peak Time:
  We apply an extended period of commonly used working ours adopted by most companies and organization. This will enable categorizing an overtime shift as peak time when applicable. In this peak-time the alarms will be carefully triggered and mostly only WAs are raised. Events that fall under WA is unusual overall traffic flows which may come from node additions, different software types used other than its usual, plateau behaviour caused by traffic reaching environmental limits, etc. CA can also possible to be raised in this time period but only in extreme cases.
- Off-Peak Time:
  Most of CAs are generated under this category. CA is triggered for an event that is really needs an attention and the likeliness to be false is small. Events that fall under CA are network devices failure, Novel NetWare server down, internet worms, and some common DoS attack at night. A raised WA at this period will most likely not get an attention from network administrator. So even if WA is raised, the system then automatically adapts the profile with the changes.

The standard mechanism procedure in raising an alarm (both WA and CA) is expressed by $\forall$ Traffic $\exists$ x, where x is set of traffic which breaks the threshold. In this case, different events will have different threshold value. Following are more details mechanism for triggering alarms for based on the events (only some are presented):

- CA for network device failure:
  If incoming traffic at 5 minutes interval below 500 packets (all packets), Then there was a device failure either in the router if the system is monitoring the whole network, or in the switch or hub if the system is monitoring one network segment and a CA is raised.
- WA for unusual traffic flows caused by node additions:
  If incoming traffic at 3 pm is 3000 packets assuming according to the profile normally at that time only around 2000 packets, then a WA is raised.
- CA for an extreme unusual traffic flows:
  If incoming traffic at 3 pm is 12000 packets or 6 times bigger than normal traffic, then CA is raised.

Logs to record all the generated alarms are deployed to track the system's performance over time and help the administrator understand how and to what direction the system grows.

4 Distributed Application Scheme

Our system employs a distributed client-server application in monitoring the network. The idea of having the system works that way is to enable monitoring large network especially a large corporate network which is separated in two different buildings from one location. It also designed to enabling monitor multiple network segments in a switched network.

A managed and distributed network monitoring scheme employed by the system is enabling us to reduce the complexities of analyzing enormous amount of traffic at once to detect anomaly. Instead of having one profile for the whole network, it will create multiple profiles for each network segment and thus it would be easier to locate the source of problem. Figure 3 pictured the system's distributed architecture.

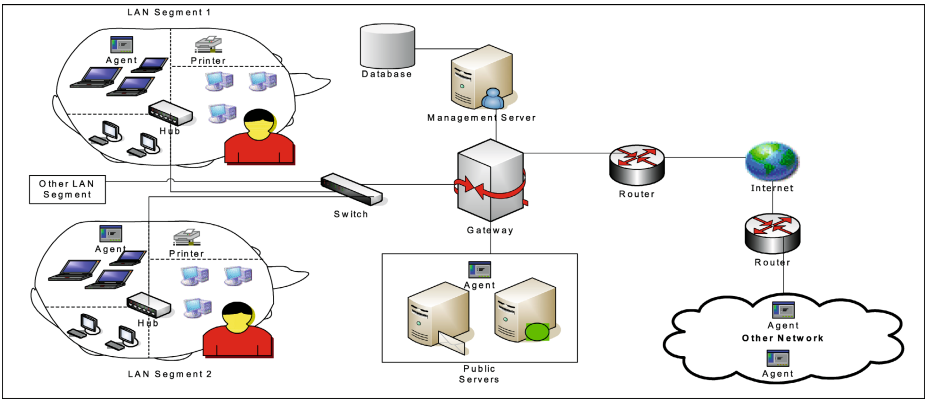


Fig. 3. Distributed Managed Client-Server Architecture

The system is not platform dependent since it is implemented using Java 2 Technologies and make it possible to have the server installed in a Windows based machine and have its agent resides on any Linux, Mac, or Sun Solaris based machine spreads in a different network segment.

Furthermore, the connectionist model of the system's engine are implemented using MATLAB and even though it is not using Java 2 Technology, it is fully integrated into the system.

4.1 Management Server

The management server can be installed in the gateway or resides in special machines dedicated for it. It has the abilities of performing the following task:

- Managing all the daily profiles from different network segment.
- Archiving all past traffic from different network segment in the databases.
- Authenticate an agent to use the profile for the segment where the agent was running.
- Add new agents including network segment information where the agent intends to monitor it.

- View all the logs for any alarmed events.
- Re-cluster specific traffic from the archived using its own ECOS module.
- View each profile if the network administrator wants to seek for explanation.

A comprehensive database is also deployed using MySQL to store all the system logs and archiving all past traffics.

When an alarm is triggered, the status of the agent will change to "Warning" for WA and "Urgent Attention" for CA. An email and sms will also be sent to the network administrator email and mobile phone respectively. Figure 4 is the running example of the Management Server.

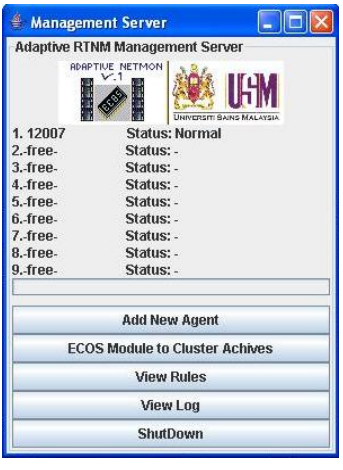


Fig. 4. Management Server GUI

4.2 Monitoring Agent

The Agent performs basic network monitoring functionality adopted by most network monitoring application available nowadays in addition to its ECOS based module which enables the Agent to detect anomalous activity and evolve its profile over time. It performs passive monitoring and a connectionless communication with the Management Server so that it is not adding more workload to the network.

Some of its cardinal functionalities among other functions are outlined as follows:

- It performs real-time traffic data stream capturing in a 10/100MB Ethernet network with promiscuous mode.
- Filter the data before further processed.
- Evolves its detection engine structure over time.
- Connect to the Management Server to adapt the profiles if necessary.
- Detects anomalous activity specifically in the network segments where the Agent was running.
- Generate WA and CA.

- Real-time statistical analyzer based on different perspective (application layer statistic, network layer statistic, and protocol based statistic).
- Basic packets decoding functionality.

A friendly graphical user interface is designed to help new user (especially network administrator).understand how to use the system better. Figure 5 is a running example of the Agent's GUI.

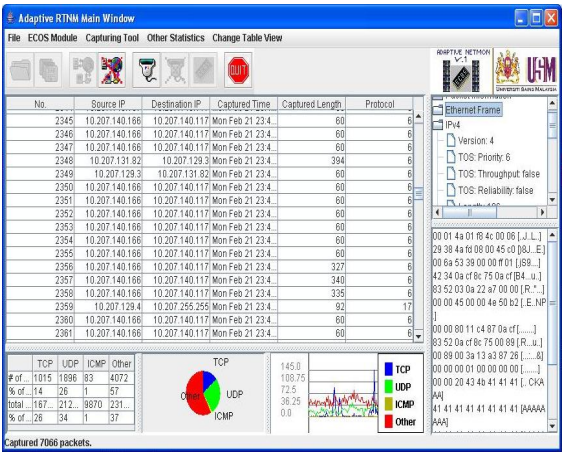


Fig. 5. An Agent GUI

5 Results and Analysis

The system was tested at our School of Computer Sciences' network for the duration of 2 months. The Monitoring Agent was installed on different network segments in our school's switched network. Our current preliminary results shows that the system is able to evolve as expected.

In Figure 6 we can see that the resulted profile has two clusters which differ significantly with others. At the time of running a WA is raised and as we (the network administrator) chooses not to take any action for that WA, the profile was adapted in next week at the same day as depicted in Figure 7.

Basically network administrator involvement in adapting the system is more as a guidance to which way the system should grow. Without the presence of network administrator the system is still able to evolve but if the attacker knows the behaviour and the evolving mechanism of the system, they can fool the system to adapt into the wrong way. This is where the hand of network administrator is needed.

Table 1 shows the simple statistic record on how many rules created in each day profile, counts for how many rules evolved in each day profile, and how many actions taken by the network administrator for any WA and CA events happened in the network for a two months period. It can be seen that working day profile (Monday-Friday) has more rules compare to off working day (Saturday and Sunday) since the network has more traffics at that days.

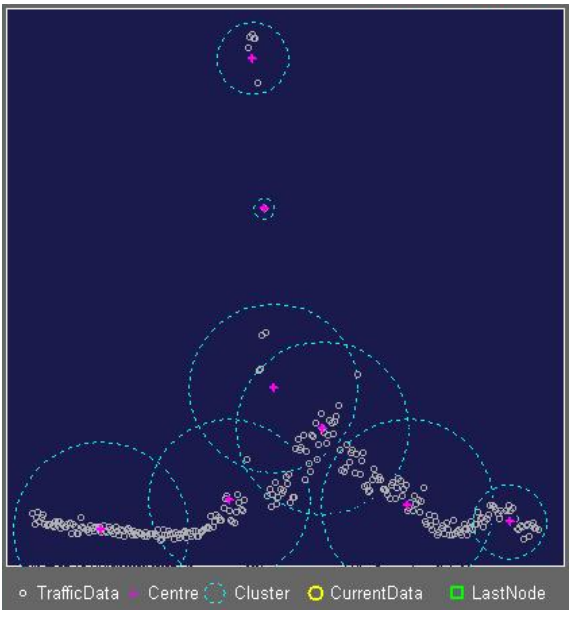


Fig. 6. Clusters Produced For Thursday, 27 January

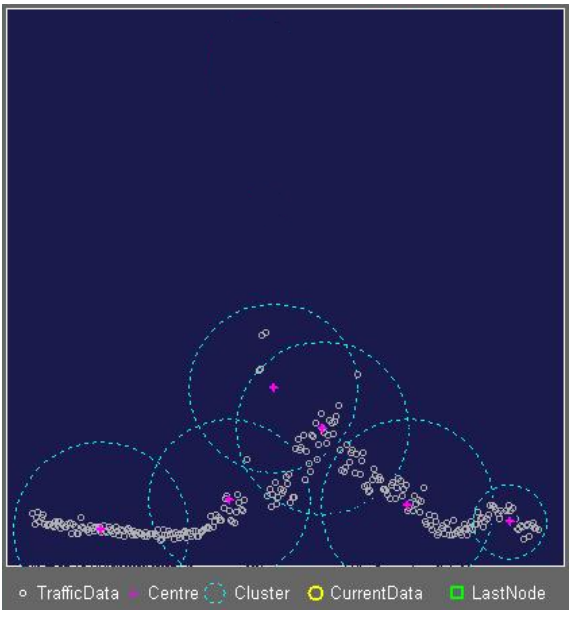
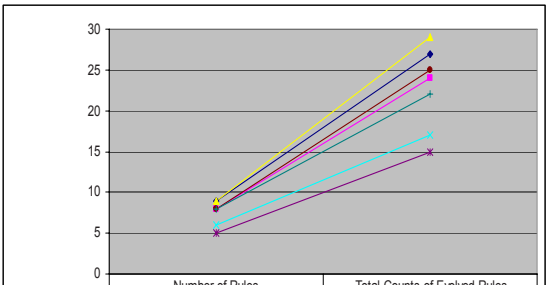


Fig. 7. Clusters Produced For Thursday, 03 February

Table 1. Counts for an Evolving Process within the System



	Number of Rules	Total Counts of Evolved Rules
Monday	8	24
Tuesday	9	27
Wednesday	8	25
Thursday	8	22
Friday	9	29
Saturday	6	17
Sunday	5	15

The rules on working day profile have evolved 58 times in average, or around 14 times each week. Actually more rules were evolved in the first week to adapt the initial profile with actual traffic in the particular network. The first week run was an important adaptation made by the system to the network traffic characteristic where the system is attached. This statistic was made based on a counter that we put on the system to record all the necessary events on the system. As such, no weekly statistic, which can show the number of times rules evolved after the initial rules, are recorded.

In this testing phase, the system detects some device failure (especially our switch) happened in some network segment. A simple DoS attack that we simulate was also successfully detected by the engine. As a whole, the system performance was satisfactory.

6 Conclusion

This work is an important step to build a complete evolvable network monitoring system. By understanding the network traffic characteristic, plenty can do to keep on eye on the network, these includes detecting more anomalous activity such as current worms, DDoS attack, and other advanced network intrusion.

Working in offline mode, the integrity of the data in the capturing process can be argued. While capturing the data for duration of (let say) two months, it cannot completely represent the normal traffic happened in the network. A case might happened for some heavy traffic captured in those two months period was because the corporate where the network resides was in an intensive work load to launch a product. This thing will affect the integrity of the resulted profile and might lead to be more burdens for network administrator to recapture the traffic, reevaluate the traffic, and finally re-profile the traffic.

Our system still needs improvements in many ways for future works. Currently we are improving the structure of the connectionist model by proposing new methods. By having these methods which specifically designed for network traffic data, the results can be more accurate and the system can grow from scratch. We also try to consider implementing a signature based intrusion detection engine to improve the detection engine's performance. Lastly, our future work will also to add in an intelligent module to automate an action as responses for such an alarmed event to prevent network down, which in turn will reduce the dependency of the network from network administrator's presence when an event which requires an immediate attention or response happened at late night.

References

Pasha, M.F., Budiarto, R.: Developing Online Adaptive Engine for Profiling Network Traffic using Evolving Connectionist Systems. In: NCEI'04, Conference on Neuro-Computing and Evolving Intelligence 2004. Auckland, New Zealand (2004)

Kasabov, N.: Evolving Connectionist System: Methods and Applications in Bioinformatics, Brain Study and Intelligent Machines, 1st edn. Springer-Verlag, London (2003)

Pasha, M.F., Budiarto, R., Sumari, P., Osman, A.: Data Mining and Rule Generation in Network Traffic using Fuzzy Clustering Techniques. In: M2USIC'04, MMU International Symposium on Information and Communications Technologies. Putrajaya, Malaysia (2004)

Lampinen, T., Koivisto, H., Honkanen, T.: Profiling Network Application with Fuzzy C-Means Clustering and Self Organizing Map. In: First International Conference on Fuzzy System and Knowledge Discovery: Computational Intelligence for the E-Age. Singapore (2002)

Song, Q., Kasabov, N.: ECM, A Novel On-line, Evolving Clustering Method and its Applications. In: ANNES'01, Fifth Biannual Conference on Artificial Neural Networks and Expert Systems (2001)

Marchette, D.: A Statistical Method for Profiling Network Traffic. In: Workshop on Intrusion Detection and Network Monitoring. USA (1999)

Paxson, V.: Bro: A System for Detecting Network Intruders in Real-Time. In: 7th USENIX Security Symposium. USA (1998)

Kasabov, N., Song, Q.: DENFIS: Dynamic Evolving Neuro-Fuzzy Inference System and its Application for time-series prediction. IEEE Trans. Fuzzy System 10(2), 144–154 (2002)

Purvis, D., Kasabov, N., Benwell, G., Zhou, Q., Zhang, F.: Neuro-Fuzzy methods for environmental modelling. System Research and Information Systems 8(4), 221–239 (1999)

Barford, P., Plonka, D.: Characteristics of Network Traffic Flow Anomalies. In: ACM Internet measurement Workshop ACM SIGCOMM'01. San Francisco, USA (2001)

Sureswaran, R.: Network Monitor. In: Conference of Asia Pasific Advance Network. Penang, Malaysia (2001)

Degioanni, L., Risso, F., Varenni, G., Viano, P.: WinPcap: The Free Packet Capture Architecture for Windows. (2003) In HYPERLINK http://winpcap.polito.it

Takagi, T., Sugeno, M.: Fuzzy Identification of Systems and its Application to Modeling and Control. IEEE Trans. System, Man. and Cybernetics 15(1), 116–132 (1985)