# STATISTICAL PROFILER MODULE, FEATURE PERFORMANCE TESTING MODULE ANDFEATURE EVALUATION MODULE FORNETWORK INTRUSION DETECTION

**Nabonarayan Jha[1] & Dr. K. B. Singh[2]**

[1]Department of Mathematics, Patan Multiple Campus, Patan Dhokha, Lalitppur, T. U. Kathmandu & Research Scholar, Department of Mathematics, B R A Bihar University, Muzaffarppur, Bihar, India
[2]Department of Physics, L. S. College, Muzaffarpur, Bihar, India

**ABSTRACT**

The next processing step is the attack and normal profile generation. The task is accomplished by the *Statistical Profiler Module*. This module uses the provided attack labels to filter out the intrusive and normal behavior of each individual feature, and stores the corresponding statistical data into uniquely identifiable profiles. Each profile keeps track of the mean $\mu$ and standard deviation $\sigma$ statistics of a particular feature during the normal or intrusive stages. It is known that the features tend to have different values for different protocols. For instance, the size of the ICMP packets is expected to be smaller than the size of the TCP packets.

Thus, instead of creating a single profile for the normal behavior of a feature, in this paper our system creates individual normal profiles for each protocol that applies to the current feature.

The module also evaluates the false positives that each feature produces. The *False Positive Evaluation* sub-module is responsible for this task and the detail algorithms that it implements are described. Once the evaluation is done, the false positive predictions (i.e., $FP(f_i)$)are saved into *False Positives DB*. The database keeps for each feature $f_i$the corresponding $FP(f_i)$ value. The process of extracting profiles and false positive prediction is repeated once for each TCP dump file and tuning combinations, until all the possible combinations are exhausted.

**Keywords:** Fuzzy Logic, Database, Profiler, Statistical.

## 1. Introduction

This module is implemented as a combination of MATLAB and Java procedures. This whole process is executed once, after the *Statisical Profiler Module* has exhausted all its input data. The whole evaluation process is designed as a sequential process that consists of four tiers. Each individual tier can be executed only after the previous tiers have completely exhausted the data that they work with.For this reason, there are three temporary databases that act like buffers between adjacent tiers. The only functionality that the databases have is to store data until is needed at the next tier. As depicted in Figure 1, the first processing tier is done for each feature tuning combination.

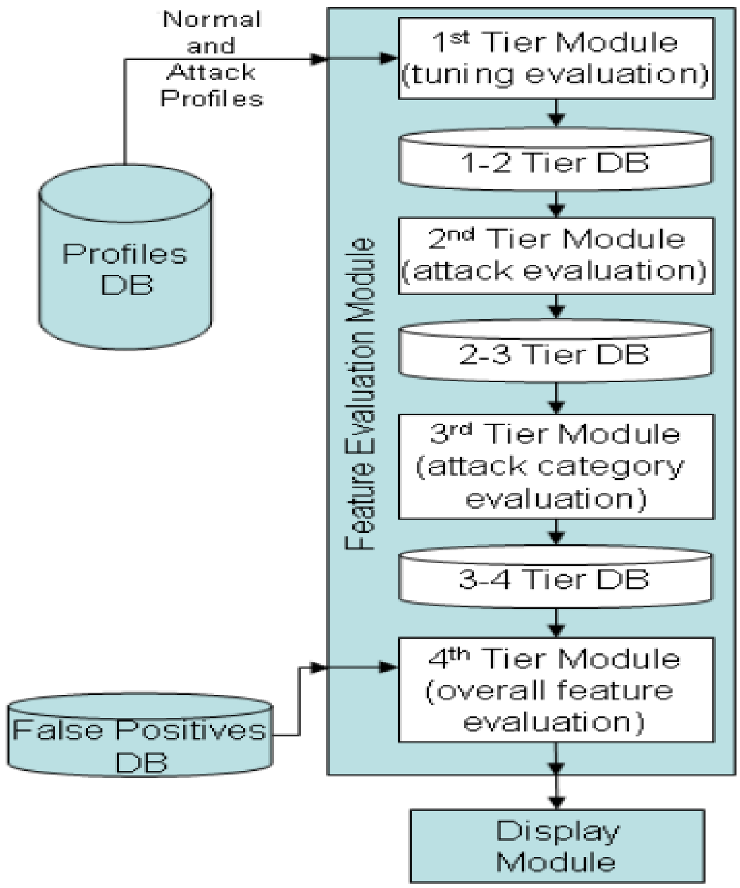


Figure 1: The overall view of the *Feature Evaluation Module* block diagram.

This module implements the previously presented algorithm for *Fuzzy evaluation of* $f_i$ *against* $\xi_j$ *attack while using* $\tau_k$ *tuning*. We use MATLAB Fuzzy Logic Toolbox to implement the fuzzy inference engine. Next, after all the possible combinations are exhausted, the second tier starts its processing stage for each individual attack-feature combination. This second module implements the algorithmdepicts just a part of the Figure 1, and was introduced here for convenience purposes). When all the possible combinations are exhausted, the third tier starts evaluating each feature against all the defined attack classes. The fourth and final tier uses both, the information provided by the antecedent tier, as well as the information stored in the *False Positives DB*.

## 2. Feature Performance Testing Module

The purpose of this module is to empirically evaluate the performance of each individual feature in the detection process given a set of attacks and corresponding tunings. The module is depicted in Figure and implements two functions. The first one is an anomaly detection module that mines the data for possible intrusions, and the second function performs the assessment on the alerts that the anomaly detection module produces. Once that is done, it reports the final performance to the Display Module.We choose to work with a very simplistic threshold-based anomaly detection algorithm that uses a lower and upper control limit to define the boundaries of the normal values. The two thresholds are computed during the training phase and are set on both sides of the normal population mean at five $\sigma$. During the detection phase, an anomaly is signaled for each individual point that exceeds the two boundaries.For this purpose, the data from each dataset has been divided into 2 parts one for training and one for testing. The training part consists of 80% of the normal data whereas the testing part consists of the rest 20% plus all the intrusions in the datasets.

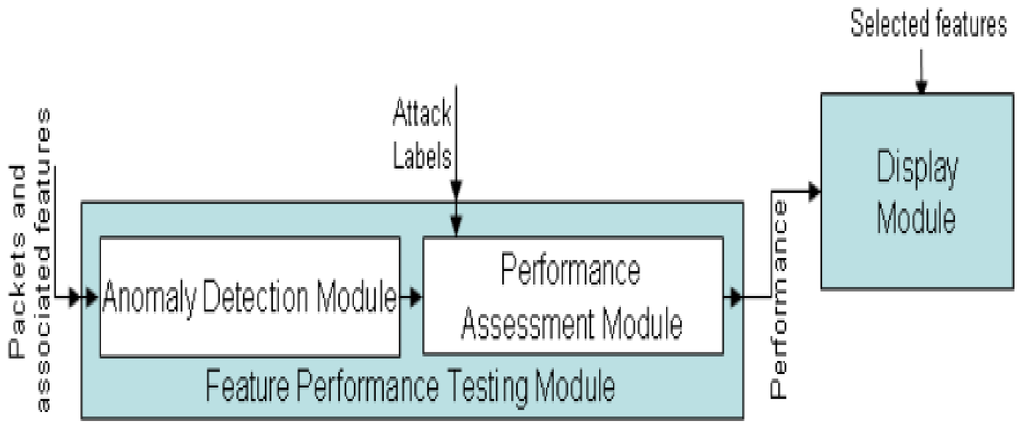


Figure 2: The overall view of the block diagrams for the *Feature Performance Testing Module* and *Display Module*.

The detection results are computed for each individual tuning value and feature. The average of those individual runs is further reported. For statistical significance, the best and worst cases are excluded.

## 3. Conclusions

The *Performance Assessment Module* receives the generated alerts from the *Detection Module* and compares them with the true attack labels that it has access to. The module computes four main evaluation functions as follows:

1. **The number of detected intrusions:** This value represents the number of actual attacks detected by the current feature during the testing phase.
2. **The number of misclassified intrusions:** This value represents the number of attacks that the current feature misclassifies.
3. **True positive rate:** This value represents the percentage of correctly classified intrusions over the total number of intrusions that the current feature produces while in the testing phase.
4. **False positive rate:** This value represents the percentage of normal data incorrectly classified as intrusion by the current feature while in the testing phase.

## References

[1] K. Das, *The development of stealthy attacks to evaluate intrusion detection systems*, Master's thesis, MIT Department of Electrical Engineering and Computer Science, June 2000.

[2] M. Dash, K. Choi, P. Scheuermann, and H. Liu, *Feature selection for clustering - a filter solution*, Data Mining, 2002. ICDM 2002.Proceedings.2002 IEEE International Conference on (2002), 115{122.

[3] M. Dash and H. Liu, *Feature selection for clustering*, PADKK '00: Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Current Issues and New Applications (London, UK), Springer-Verlag, 2000, pp. 110{121.

[4] M. Dash, H. Liu, and J. Yao, *Dimensionality reduction of unsupervised data*, Tools with Artificial Intelligence, 1997. Proceedings., Ninth IEEE International Conference on, no. 3-8, November 1997, pp. 532{539.

[5] P.A. Devijver and J. Kittler, *Pattern recognition a statistical approach*, Prentice Hall International, 1982.

[6] J. Doak, *An evaluation of feature selection methods and their application to computer security*, Tech. Report CSE-92-18, University of California at Davis, 1992.

[7] P. Dokas, L. Ertoz, V. Kumar, A. Lazarevic, J. Srivastava, and P. Tan, *Data mining for network intrusion detection*, Proceedings of NSF Workshop on Next Generation Data Mining (Baltimore, MD), November 2002, pp. 21{30.

[8] P. Domingos, *Context-sensitive feature selection for lazy learners*, (1997), 227{253.

[9] J. G. Dy and C. E. Brodley, *Feature subset selection and order identification for unsupervised learning*, ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning (San Francisco, CA, USA), Morgan Kaufmann Publishers Inc., 2000, pp. 247{254.

[10] L. Ertoz, E. Eilertson, A. Lazarevic, P.N. Tan, P. Dokas, V. Kumar, and J. Srivastava, *Detection of novel network attacks using data mining*, In ICDM Workshop on Data Mining for Computer Security (DMSEC) (Melbourne, FL), Nov. 19 2003, pp. 30{39.
[11] Chapman Flack and Mikhail J. Atallah, *Better logging through formality applying formal specification techniques to improve audit logs and log consumers*, Proceedings of Recent Advances in Intrusion Detection, 3rd International Symposium, (RAID 2000) (Toulouse, France) (H. Debar, L. M, and S.F. Wu, eds.), Lecture Notes in Computer Science, Springer-Verlag Heidelberg, October 2000, pp. 1{16.
[12] S. Forrest, S. Hofmeyr, A. Somayaji, and T. Longstaff, *A sense of self for unix processes*, Proceedings of the 1996 IEEE Symposium on Security and Privacy (Los Alamitos, CA), IEEE Computer Society Press, 1996, p. 120128.
[13] J.M. Garibaldi and R.I. John, *Choosing membership functions of linguistic terms*, Proceedings 2003 IEEE International Conference on Fuzzy Systems, 2003, pp. 578 { 583.
[14] Anup K. Ghosh, Christoph Michael, and Michael Schatz, *A real-time intrusion detection system based on learning program behavior*, Proceedings of Recent Advances in Intrusion Detection, 3rd International Symposium, (RAID 2000)(Toulouse, France) (H. Debar, L. M, and S.F. Wu, eds.), Lecture Notes in Computer Science, Springer-Verlag Heidelberg, October 2000, pp. 93{109.