A comprehensive fuzzy logic model for feature performance assessment against network attacks

Iosif-Viorel Onut and Ali A. Ghorbani
Information Security Centre of Excellence, Network Security Laboratory
Faculty of Computer Science, University of New Brunswick
Fredericton, NB, E3B 5A3, Canada,
Email: {onut.viorel,ghorbani}@unb.ca

***Abstract*—The feature selection phase is one of the first, and yet very important, tasks to be completed during the development of any Intrusion Detection System. If this phase is neglected, the detection performance of the entire system can drop significantly, regardless of the internal detection algorithms that are used. Our research focuses on mining the most useful network features for attack detection. Accordingly, we propose a mathematical procedure that uses statistical and fuzzy logic techniques to rank the participation of individual features into the detection process. We report our experimental findings on a set of 933 features, while using 180 different tuning parameters for each feature. The experimental results empirically confirm that our feature evaluation model can successfully be applied to mine the importance of a feature in the detection process.**

I. Introduction

Internet and data fraud and has become one of the most challenging cybernetic acts that security officers around the world try to combat. The more critical and confidential the data the more appealing is for attackers to hack it. The impact of a successful attack over a bank or governmental institution can have disastrous consequences given the type of data that those organizations have. Researchers around the world constantly develop and improve Network Intrusion Detection Systems (NIDS) that are meant to combat such threats.

The design of a NIDS is a delicate process that requires the successful completion of numerous stages so that the final outcome can be considered a success. The feature selection stage is one of the first steps that needs to be addressed, and can be considered among the top most important ones. The overall performance of the NIDS will greatly suffer if at the design stage this step is not carefully considered, regardless of the detection technique, or any other algorithms that the NIDS is using[16], [14]. Despite its importance we believe that this issue is not sufficiently studied and explored by the research community.

This work intends to provide both, a hybrid method that combines statistical and fuzzy concepts for feature evaluation, as well as, a comprehensive study concerning the effectiveness of various network features in the detection process. Despite the fact that we concentrate our research on the Transport, Network, and Network Access layers of the TCP/IP Architecture Model, the proposed feature evaluation method can easily be applied to other layers of the TCP/IP Architecture Model, or OSI standard.

We analyze the performance of a set of 933 network features that are directly extracted from network packets. Furthermore, each of the studied features has various tuning parameters that play a decisive role in the evaluation process. The tuning sensitivity of a feature is factored in by studying a set of 180 different tuning values for each one of the studied feature.

Once the feature evaluation model was finalized, the main challenge was to find a good dataset for the proof of concept. This datased would need to contain the original packets captured from a real network, a fair diversity of attacks, a fair amount of data, and labels for the different type of attacks. To our knowledge, there isn't any dataset that meets our criteria and that was released after the DARPA Intrusion Detection and Evaluation dataset [4]. Thus, we consider that if we address some of the issues that exist in this dataset[11], it can be safely considered for the prof of concept. To process this high number of feature and tunings, combined with the size of the actual dataset, we used 60 processors of a Sun V60 computer cluster combined with the computational power of two computer labs consisting of 80 PCs that were provided by our university.

The rest of the paper is organized as follows, Sec. II summarizes the state of the current research as well as briefely presents some of our past contributions. The feature evaluation process is described in Sec. III. Sec. IV is entirely dedicated to the detail description of the fuzzy inference process. The experimental results are presented in Sec. V. Lastly, Sec. VI summarizes the conclusions and outlines our future work.

II. Background review

The main challenge when selecting and evaluating a comprehensive set of features is the lack of unanimously accepted feature classification schema, or feature evaluation methods in the literature. Moreover, various names are used for the same class of feature in different scientific papers. For instance, the case of single TCP connection features are referred to as *Essential Attributes*[20], *Basic TCP Features*[12], *Basic Features*[5], [6], *Basic Features of individual TCP connection*[1]; the case of connection and connectionless features is referred as *Flow Statistics*[13] and *Single Connection Dependent Features*[19] ; the case of features that capture data about the network in general are referred as *Traffic Features*[20], [1],*Derived Features*[5], [6], and *Multiple Connection Features*[19].

Additionally, empirical demonstrations showed that false correlations may exist between various features that will lead to poorer results concerning the accuracy and performance of the detection system[16], [14]. Chebrolu et al. also demonstrated that by carefully choosing only 17 of the 41 features provided in the International Knowledge Discovery and Data Mining Competition (KDD-1999) [1], the detection rate will remain the same, while the speed of the detection algorithm will experience a 50% improvement[14].

To address these concerns, we earlier proposed a feature classification schema for network intrusion detection, which aims to provide a better understanding on the types of features that can be directly extracted from network packets[18], [19]. The proposed classification schema was defined for the Transport, Network and Network Access layers of the TCP/IP architecture model, in particular for IP, TCP, UDP and ICMP protocols. The description of this schema is out of the scope of this work and is not required in any way for the understanding of the experimental results. However, we used this classification schema to select a comprehensive set of features to be analyzed. We use a total number of 933 features that are extracted by the use of our Feature Extraction Engine. This engine is capable of working both on-line, by sniffing the packets directly from the network, and off-line by reading them from a tcpdump file. The detailed description of this module can be found in our previous work[19].

There are two widely accepted feature construction techniques that are reported in the literature; namely, *time based features* and *connection based features* [6], [5], [20], [1], [19]. The *time based features* category includes those features that are computed based on a predefined time interval, whereas the *connection based features* are constructed based on a predefined connection window. Each one of this two techniques presents a different set of tunable parameters[19]. For instance, in the case of *time based features* there are the *time window size* and the *window granularity step*, whereas in the second case there are the *connection window size* and the *time to live* of a connection. It is of maximum importance to consider multiple tuning values when evaluating a feature since these values will directly influence its effectiveness in the detection process.

III. The Feature Evaluation Model

Our proposed feature evaluation model combines statistical with fuzzy techniques to mathematically extract for each feature f_i its relevance in detecting a certain type of attack. Let us define $P(f_i|\zeta_m)$ as the probability of feature f_i to detect the ζ_m attack category. The higher this value is, the more suitable feature f_i is for the detection of attack category ζ_m. For this purpose, we use our previously proposed *Feature Extraction Module*[19] to extract the set of selected features,

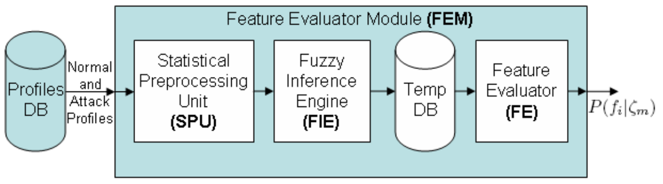
Feature Evaluator Module (FEM)
Profiles DB
Normal and Attack Profiles
Statistical Preprocessing Unit (SPU)
Fuzzy Inference Engine (FIE)
Temp DB
Feature Evaluator (FE)
$P(f_i|\zeta_m)$

Fig. 1. The underlying architecture of the proposed feature evaluation model.

and to compute for each feature f_i, intrusion ξ_j, and tuning τ_k combination the standard deviation and mean of the feature during the normal and intrusive stages. Once these are computed, the *Feature Extraction Module* will save them into a profiles database (i.e., *Profiles DB* see Fig. 1). The evaluation process consists of three main modules that incrementally extract statistical and fuzzy data about each feature in order to rank its importance (i.e., $P(f_i|\zeta_m)$). The modules are further explained in the following subsections. The Java and Matlab languages were used for their implementation.

A. The Statistical Pre-processing Unit

For each feature f_i, tuning τ_k and intrusion ξ_j combination, the *Statistical Pre-processing Unit* (SPU) extracts from the *Profiles DB* the corresponding standard deviation and mean of the feature during the normal and intrusive stages. Let μ_N, σ_N represent the mean and standard deviation of feature f_i during the normal operation while configured using τ_k tuning value. Similarly, let μ_I, and σ_I represent the mean and standard deviation of feature f_i during the ξ_j attack, while configured using τ_k tuning value. Using the μ_N, σ_N, μ_I, σ_I values, the SPU computes three metrics that will help the *Fuzzy Inference Engine* (FIE) to decide upon the importance of the current feature in the detection process. Once those metrics are computed, they are passed to the FIE for evaluation.

In order to improve the quality of the feature evaluation process and to remove some of the problems (see [11]) that may exist in the dataset, the SPU will discard any f_i, τ_k, ξ_j combination with $\sigma_I = 0$. This will take care of those situations where a feature has exactly the same value during the intrusive profile. We argue that in real word this scenario is very unlikely to happen since, due to the cumulative effect, any profile would have at least a ramp-up and a rump-down sections. However, on few occasions, some of the features indeed presented a zero variance during the intrusive part and hence their contribution on detecting that particular intrusion was discarded from the analysis process.

The three metrics computed by SPU highlight the statistical differences between the distributions of the two normal and intrusive stages. Note that our basic assumption is that a feature f_i is highly effective in the detection of ξ_j attack if its value significantly changes between the normal and intrusive stages, meaning that the more different the two distributions are, the higher chances for f_i to detect that particular intrusion.

The statistical nature of a given distribution can be studied by the use of *Chebyshev's Inequality*, a well known formula named after the Russian mathematician who first proved it. The formula applies to any type of probability distribution, and states that nearly all values that belong to a given distribution are close to the mean of the distribution [21], [8]. The inequality is mathematically defined as:

$$P(|X-\mu| \geq m\sigma) \leq \frac{1}{m^2}, \qquad \forall m,\ m \subset \mathbb{R}^* \tag{1}$$

where X is a random variable, with an expected μ value and a finite standard deviation σ. The theorem states that the probability of a value X to be more than $m\sigma$ apart from the μ is less or equal than $1/m^2$. Note that if $m \to \infty$, the $P(|X-\mu| \geq m\sigma) \to 0$; in other words, the more distant a point is from the mean of the distribution, the less probable it belongs to the distribution. Finally, another remark is that for $m \geq 1$, the $P(|X-\mu| \geq m\sigma) \in [0,1]$.

Using the *Chebyshev's Inequality*, let us define $P(X|\mu,\sigma)$ as the probability of a point X to be an outlier of a distribution that has μ as mean and σ as standard deviation as follows:

$$P(X|\mu,\sigma) = \begin{cases} 1 - P(|X-\mu| \geq m\sigma) & if |X-\mu| > \sigma \\ 0 & otherwise \end{cases} \forall X \tag{2}$$

in other words, the further a point X is from the distribution defined by μ and σ, the closer the $P(X|\mu,\sigma)$ will be to one, and the higher the probability of X to be an outlier.

Using the previously described information, the statistical differences between two distributions (i.e., normal and intrusive) can be quantified if Eq. 2 is used to compute both, the probability of the normal points to be outliers of the intrusive distribution as well as the probability of the intrusive points to be outliers of the distribution captured during the normal stage. Obviously, this two-way check cannot be done for each and every point in the two distributions; however, since in a distribution most of the population lies relatively close to the mean, it would be sufficient to compute only the probability of both means to be outliers of the other population. Consequently, SPU computes the following two metrics:

Metric 1 (M_1): is defined as the probability of the normal mean μ_N to belong to the intrusive population denoted by the μ_I and σ_I:

$$M_1(\mu_N, \mu_I, \sigma_I) = P(\mu_N|\mu_I, \sigma_I) \tag{3}$$

Metric 2 (M_2)**:** is defined as the probability of the intrusive mean μ_I to belong to the population of the normal behavior:

$$M_2(\mu_I, \mu_N, \sigma_N) = P(\mu_I|\mu_N, \sigma_N) \tag{4}$$

One problem that arises when using only the two previously defined metrics is the case when both distributions are very close, but the intrusive distribution is far more spread than the normal one (i.e, $\sigma_I >> \sigma_N$). Intuitively in this case, the chances of an IDS to pick this up are high since even a simple threshold would detect the difference (one can picture this example as two bell-shape curves that have their means close, but one of the bell-shapes outruns the second one in size). Thus, a third probability which highlights this situation is needed.

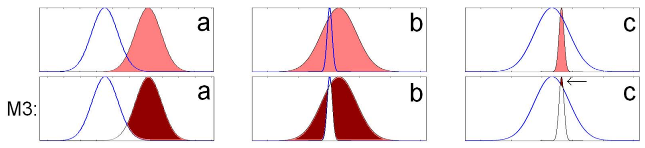

Fig. 2. Graphical visualization of the M_3 metric.

One quick remark would be that we cannot make any assumption regarding the type of the distribution that the feature generates during both the intrusive and normal stages. Thus, we cannot say that they follow a Gausian distribution (i.e., bell-shape). However, the *Central Limit Theorem* states that if a large number, n, of random samples are chosen from virtually any population the distribution of those samples will tend to a Gausian distribution as $n \to \infty$ [8]. Moreover, Walpole et al. [21] demonstrated that if $n \geq 30$ the normal approximation of the *Central Limit Theorem* is good regardless of the initial population shape, while for $n < 30$ the approximation is good only if the initial population shape is not far from a Gausian distribution.

This leads us to believe that for the purpose of defining the third metric, it is possible to visualize the intrusive (or normal) distributions as bell-shaped curves centered in the mean of the distribution and having the shape uniquely defined by the following equation:

$$bellShape(x,\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{5}$$

The first row of Fig. 2 depicts three basic cases that may happen when comparing the distributions of the normal and intrusive populations. For a better visualization, let us draw the distribution captured during the normal stage as a single bell-shape curve, and the intrusive distribution as a filled bell-shape 2D object. Consequently, the three cases are (a) both normal and intrusive distributions are relatively far from each other, (b) the normal and intrusive distributions are very close, but the $\sigma_N < \sigma_I$, and (c) the normal and intrusive distributions are very close, but the $\sigma_N > \sigma_I$.

Looking at these three examples we can infer that an IDS would probably have no problem detecting an intrusion in the first two cases (a) and (b), whereas, in case (c), the intrusion is almost invisible to the IDS, since most of intrusive population belongs to the population of the normal behavior too. It can be easily seen that in case (b) M_1 will be close to zero, while M_2 will be relatively high. However, if we 'slide' the distribution of the normal behavior even closer to the intrusive one until both means are very close (i.e, $|\mu_N - \mu_I| < max(\sigma_N, \sigma_I)$); both M_1 and M_2 will be zero since indeed none of the means would be outliers of the other population. Nevertheless, intuitively the chances of an IDS to pick this case are still very high (since $\sigma_N << \sigma_I$).

Similarly, if the same scenario is applied in case (c), M_1 and M_2 will be again zero, but the difference is that in this case and IDS wouldn't be able to detect the intrusion since $\sigma_N >> \sigma_I$.

Consequently, a third metric (i.e., M_3) is needed to grasp the difference between case (b) and (c) when M_1 and M_2 are zero. It is almost impossible to try to define a reasonable 'much smaller' and 'much greater' when comparing σ_N with σ_I. Thus, the third metric, instead of comparing the two, reports percentage wise the area that remains from the intrusion's bell-shape 2D object after removing the overlapping part of the normal shape. Last row of Fig. 2 depicts the context of this metric. It is easily seen that in case (b), only a small part of the intrusion bell-shape will be lost, while in case (c), most of the area will be lost, the only remaining part being highlighted by arrow in the figure.

Metric 3 (M_3): is defined as the percentage of the area that remains from the intrusion's bell-shape 2D object after removing the overlapping part of the normal shape as follows:

$$M_3(\mu_N, \sigma_N, \mu_I, \sigma_I) = 1 - \frac{area(\mu_I, \sigma_I) \bigcap area(\mu_N, \sigma_N)}{area(\mu_I, \sigma_I)} \tag{6}$$

where $\bigcap$ is defined as the intersection of two areas and $area(\mu, \sigma)$ is:

$$area(\mu, \sigma) = \int_{-\infty}^{+\infty} (bellShape(x, \mu, \sigma))\, dx \tag{7}$$

Note that in the worse case scenario, the value for this metric will be zero, while in the best case scenario maximum value would be one. M_3 essentially highlights the difference between the normal and intrusive standard deviations while keeping track of the means as well. Thus, the overall outcome of M_3 would be similar with the current situation if the bell-shape representation for defining the area in Eq. 7 would be changed with any other kind of shape (e.g., triangular, rectangular). The only restrictions on the shape would be that it needs to be centered in the mean, it needs to be symmetrical, and needs to be defined between $[\mu - \alpha\sigma, \mu + \alpha\sigma]\ \forall \alpha \in \mathbb{R}, \alpha > 0$. Furthermore, in the simplest case, the shape could be change with a segment, and the area with a length, as long as the previous conditions are kept.

B. The Fuzzy Inference Engine

The next building block of the FEM is the Fuzzy Inference Engine (FIM) (see Fig. 1). This module performs two simultaneous tasks for each set of inputs that it receives (i.e., M_1, M_2,and M_3). Firstly, it computes a unique value (between 0 and 1) representing the statistical significance of feature f_i in detecting attack ξ_j based on τ_k tuning factor. Let $P(f_i|\xi_j, \tau_k)$ denote this output probability. Secondly, as a side effect it also determines if the current inputs are worthwhile to be taken into account or not. Once the computation is done, if $P(f_i|\xi_j, \tau_k)$ is high enough (i.e., it needs to be *acceptable*, *good* or *excellent*, see Sec. IV) then the value is recorded into a temporary database (i.e., *Temp DB*). On the other hand, if the FIE decides that the current inputs are to be discarded, nothing is further recorded; given that it is not recommended to use f_i for detecting attack ξ_j while using τ_k tuning. The detail description of this module is presented in Sec. IV.

C. The Feature Evaluator

The aim of this sub-module is to ultimately provide a final ranking for each of the studied feature based on the data stored in the *Temp DB* database. Let $P(f_i|\zeta_m)$ represent the ability of a feature f_i to detect attacks that belong to ζ_m category. This probability is computed by the Feature Evaluator (FE) for each f_i, ζ_m combination. To do this, once FIE computes all the $P(f_i|\xi_j, \tau_k)$ probabilities, the FE module goes over the whole database and identifies all the related $P(f_i|\xi_j, \tau_k)$ probabilities to ζ_m category (i.e., $P(f_i|\xi_j, \tau_k)$ is related to the ζ_m category if intrusion ξ_j belongs to the ζ_m attack category).

Recall that every item that is stored in the *Temp DB* is labeled by the fuzzy engine as being statistically significant for attack detection. Consequently, for each f_i, ζ_m combination there are three main attributes that are extracted by the FE as follows:

Number of attacks detected by the feature: Represents the number of attacks within ζ_m attack category that f_i has the potential to identify. Let $\mathcal{N}(f_i|\zeta_m)$ represent this number, defined as follows:

$$\mathcal{N}(f_i|\zeta_m) = count(\xi_j), \quad \forall f_i, \zeta_m, \exists \xi_j \mid \xi_j \in \zeta_m \tag{8}$$

where $count(\xi_j)$ represents the number of ξ_j attacks. This property is relevant since the more attacks a certain feature has potential to detect, the more multifunctional that feature is, and the less features the detection engine would need, which ultimately reduces to a lower computational power required.

Feature sensitivity to tunings: This property is very important to compute since in real word it is very hard to decide a perfect tuning factor for each network. Thus, the less sensitive a feature is to the tuning process, the more robust and consistent the detection mechanism will be on different networks and tunings. Let $\mathcal{T}(f_i|\zeta_m)$ represent the average number of tunings τ_k that f_i can be tuned with, so that it has potential in detecting ξ_j attacks from ζ_m category; defined as follows:

$$\mathcal{T}(f_i|\zeta_m) = \overline{\mathcal{T}(f_i|\xi_j)}, \quad \forall f_i, \zeta_m, \exists \xi_j \mid \xi_j \in \zeta_m \tag{9}$$

where $\mathcal{T}(f_i|\xi_j)$ is the number of tunings that f_i can safely use to detect attack ξ_j; defined as:

$$\mathcal{T}(f_i|\xi_j) = count(\tau_k), \quad \forall f_i, \xi_j, \exists \tau_k \tag{10}$$

Average fuzzy performance: This property is very important since it shows the potential of feature f_i in detecting an attack category ζ_m based on the proposed fuzzy criteria. Let $\mathcal{F}(f_i|\zeta_m)$ represent this value and be defined as:

$$\mathcal{F}(f_i|\zeta_m) = \overline{\mathcal{F}(f_i|\xi_j)}, \quad \forall f_i, \zeta_m, \exists \xi_j \mid \xi_j \in \zeta_m \tag{11}$$

where $\mathcal{F}(f_i|\xi_j)$ is the average performance of f_i against ξ_j attack; defined as:

$$\mathcal{F}(f_i|\xi_j) = \overline{P(f_i|\xi_j, \tau_k)}, \quad \forall f_i, \xi_j, \exists \tau_k \tag{12}$$

Consequently, for each f_i, ζ_m combination, the FE module extracts the previously presented attributes. However, the final goal is to compute a unique final probability $P(f_i|\zeta_m)$ as defined at the beginning of this section. The main difficulty of combining $\mathcal{N}(f_i|\zeta_m)$, $\mathcal{T}(f_i|\zeta_m)$, and $\mathcal{F}(f_i|\zeta_m)$ in a single value is that they are semantically different and a simple average won't make sense. Moreover, while $\mathcal{F}(f_i|\xi_j) \in [0, 1]$, the other two attributes are not normalized. However, if all the attributes are normalized, then, a more suitable solution for computing the final ranking would be to compute the norm of the vector formed by the three attributes in a 3D space. Thus, $P(f_i|\zeta_m)$ is defined as:

$$P(f_i|\zeta_m) = \left|\left[\frac{\mathcal{N}(f_i|\zeta_m)}{max(\mathcal{N})}, \frac{\mathcal{T}(f_i|\zeta_m)}{max(\mathcal{T})}, \mathcal{F}(f_i|\zeta_m)\right]\right|_2, \quad \forall f_i, \zeta_m \tag{13}$$

where $|\mathbf{x}|_2$ is the norm of vector $\mathbf{x}$, $max(\mathcal{N})$ is the maximum value $\mathcal{N}$ that a feature reaches, and $max(\mathcal{T})$ is the maximum value $\mathcal{T}$ that a feature reaches.

IV. The Fuzzy Inference Engine

The main decision engine, that computes the relevance of a feature in attack detection, is implemented using fuzzy logic concepts. Fuzzy logic was first introduced by L.A. Zadeh in the early seventies [23], [22], [24] as one multivariate logic that allows a more human-like way of developing computer programs. By the use of fuzzy theory the programmer can define fuzzy sets that are designed to relax the crisp separation between different sets (e.g., Boolean logic), transforming that into a more gradual boundary. In conjunction, fuzzy rules can also be used over the fuzzy variables to define the logistics of the desired outcome; for this purpose a fuzzy inference methodology is used. Mamdani [10], [9] and Surgeno [15] are just only two examples of methodologies, Surgeno being a special case of Mamdani. However, Mamdani engine is by far the most popular FIE in the literature, also widely considered as a standard.

For the purpose of this work we use a Mamdani fuzzification method with min-min-max as the operators for the premise conjunction, implication and aggregation steps respectively, and center of gravity method for the defuzzification process. The following subsections describe the reasoning behind the input and output membership functions, as well as the rules that the FIE uses for creating the final $P(f_i|\xi_j, \tau_k)$ value.

A. Determining the input fuzzy membership functions

One of the most important part of the fuzzy modeling is building the fuzzy membership functions. These functions mathematically represent the meanings (e.g., low, high) of a linguistic variable. There are several techniques for building the fuzzy membership functions such as direct rating, polling, interval estimation, reverse rating, membership function exemplification, pairwise comparison[17], [2], to name a few. However, it is important to note that membership functions are subjective and context dependent [22], [24], and for their creation one has to use expert knowledge from the given domain. Consequently, we use a variant of the interval estimation method for determining the fuzzy membership functions of M_1 and M_2 metrics, and a combination of estimation method with membership exemplification for M_3 metric.

Furthermore, we choose to work with sigmoidal functions when defining each set of the input variables, since having a smoother transition between different values better models the linguistic terms than, for instance, a trapezoidal or triangular shaped function. The general form of a sigmoidal function is $f(x, s, c) = 1/(1 + e^{s(x-c)})$, where s and c are the slope and center of the curvature. Two points on the curvature are sufficient to determine its slope and center. Let us consider those two points as (x_1, y_1) and (x_2, y_2). It can be easily demonstrated that the center and slope of the sigmoidal function that passes through the previously two mentioned points are:

$$\begin{cases} c = \frac{x_1 ln\left(\frac{1-y_2}{y_2}\right) - x_2 ln\left(\frac{1-y_1}{y_1}\right)}{ln\left(\frac{1-y_2}{y_2}\right) - ln\left(\frac{1-y_1}{y_1}\right)} \\ s = -\frac{ln\left(\frac{1-y_1}{y_1}\right)}{x_1 - c} \end{cases} \quad \forall\ (x_1, y_1), (x_2, y_2) \tag{14}$$

Consequently, two points are sufficient to define the corresponding sigmoidal function of each semantic term. Since the absolute 0 and 1 values can not be chosen for the y coordinate, we use instead 0.01 and 0.99 values for an ascending curve, and 0.99 and 0.01 for a descending one.

Determining the membership functions for M_1 and M_2: Recall that the first two inputs in the FIE engine express the probability of a point to belong to a certain distribution. Thus, even though semantically the two metrics target two different aspects of the feature's potential to detect a certain attack; they can be fuzzyfied using the same membership functions.

We split the zero to one interval of the M_1 and M_2 into three fuzzy sets as follows: *not-an-outlier*, *suspicious*, and *outlier*. These three fuzzy sets can be defined based on expert knowledge. For this purpose, the *The Empirical Rule* states that about 95.44%, 99.73%, 99.994%, and 99.99994% of the objects of a population will lie within 2σ, 3σ, 4σ, and 4σ from the mean of the population, respectively. If these factors of σ are used in Eq. 2, the $P(X|\mu, \sigma)$ will become 0.750, 0.889, 0.936 and 0.960, respectively. We can safely consider a point as *not-an-outlier* if is within 3σ from the mean of the distribution, in other words $P(X|\mu, \sigma) \in [0, 0.889]$. However, as $P(X|\mu, \sigma)$ is approaching 0.889, our confidence in the previous statement decreases, especially after two standard deviations (i.e., 0.750). Furthermore, it can also be assumed that if a point is above five standard deviations from the mean it is a very good candidate for being an outlier,

TABLE I
THE MEMBERSHIP FUNCTIONS FOR THE M_1 AND M_2 METRICS

Membership	**ascending**		**descending**		**ascending**		**descending**	
functions	x_1	x_2	x_1	x_2	c	s	c	s
not-an-outlier	-	-	0.75	0.89	-	-	0.82	-66.12
suspicious	0.75	0.89	0.96	1.00	0.82	66.12	0.98	-229.76
outlier	0.96	1.00	-	-	0.98	229.76	-	-

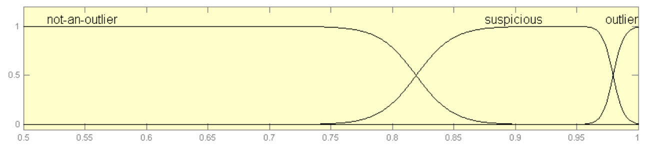

Fig. 3. The membership functions for the M_1 and M_2 metrics - graphical detail

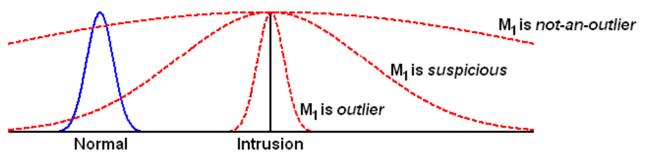

Fig. 4. Different cases when M_2 is outlier.

and thus, we expect the $P(X|\mu,\sigma) \in [0.936, 1]$. Finally, whatever is left from the interval range (i.e., $P(X|\mu,\sigma) \in [0.889, 0.936]$) can be labeled as being *suspicious*. The description of the membership functions is shown in Table I and Fig. 3.

Determining the membership functions for M_3: The nature of M_3 metric is different than M_1 and M_2, and as a consequence it needs to be studied separately. In this case, we use the following three linguistic terms *poor*, *mediocre*, and *good* to express the amount of uncovered intrusive area by the normal behavior. Intuitively speaking, a 0.5 coverage or more of the intrusion shape leads to a higher overlap between the normal and intrusive behavior, which leads to a poor detection chance for the IDS. Furthermore, it is safe to consider that after 0.5 coverage the degree of confidence on the previous assertion starts decreasing until it reaches a point where we can safely say that the coverage is not poor anymore. Let us consider this point to be 0.75 (i.e., mid way between 0.5 and 1). Similarly, it can be assumed that starting at around 0.8 coverage we start considering the M_3 metric as having a good outcome. The confidence in this statement reaches a maximum point at 1. Finally, the uncovered part of the two previous assertion is considered to be *mediocre*. Table II and Fig. 5 show the detailed description of the membership functions.

B. Defining the fuzzy rules

The fuzzy rules describe in fuzzy terms the core functionality of the inference engine. This gives us the advantage of defining in linguistic terms the logistics behind our feature evaluation criteria. The general format of a fuzzy rule is: "IF (x_1 is S_1 and x_2 is S_2 ... and x_n is S_n) THEN (y_1 is Q_1 and y_2 is Q_2 ... and y_m is Q_m)", and consists of a *antecedent* block which is situated between the IF and

TABLE II
THE MEMBERSHIP FUNCTIONS FOR THE M_3 METRIC

Membership	**ascending**		**descending**		**ascending**		**descending**	
functions	x_1	x_2	x_1	x_2	c	s	c	s
poor	-	-	0.50	0.75	-	-	0.63	-36.76
mediocre	0.50	0.75	0.80	1.00	0.63	36.76	0.90	-45.95
good	0.80	1.00	-	-	0.90	45.95	-	-

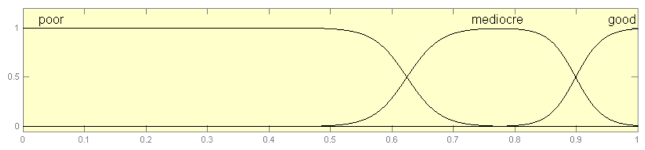

Fig. 5. The membership functions for the M_3 metric - graphical detail

THEN parts of the rule, and an *consequent* block which follows the THEN part. Recall that the output of FIE module semantically models one *linguistic variable* that quantifies the statistical significance of feature f_i in detecting the attack ξ_j based on τ_k tuning factor (i.e., $P(f_i|\xi_j,\tau_k)$). Defining the fuzzy rules and determining the number of fuzzy *terms* that will define the final output of the FIE module is done in parallel.

The first remark when constructing the rules is that M_2 is probably the most important metric of all the three metrics, since it statistically quantifies the probability of intrusion's mean (i.e.,μ_I) to be part of the normal behavior. Since, the system uses three *literals* to characterize the M_2 metric, there can be identified three main cases as follows (a) M_2 is *outlier*, (b) M_2 is *suspicious*, and (c) M_2 is *not-an-outlier*. Once the μ_I is considered an *outlier* of the normal behavior (i.e., case (a)) it is most certainly that the feature is heavily influenced by the current attack and it should be rewarded. However, the closer μ_I gets to the normal behavior (i.e., case (b) followed by case (c)), the lesser the current feature is influenced by intrusion ξ_j.

Furthermore, M_1 can also take three values as follows (1) *outlier*, (2) *suspicious*, and (3) *not-an-outlier*. Thus, we have 9 basic cases showed in Table III:

Let us further analyze case (a), Fig 4 depicts this example. Recall that M_2 depends on three factors as follows: μ_I, μ_N,

TABLE III
DIFFERENT CASES THAT APPEAR WHEN ANALYZING THE VALUES OF M_1 AND M_2.

case	M_2	M_1	**ranking**	**literals used**
a_3	outlier	not-an-outlier	1	excellent
a_2	outlier	suspicious	2	good
a_1	outlier	outlier	3	acceptable
b_3	suspicious	not-an-outlier	2	good
b_2	*suspicious*	*suspicious*	**depends on M_3 (3/-1/-2)**	acceptable/ unacceptable/ ridiculous
b_1	*suspicious*	*outlier*	**depends on M_3 (3/-1/-2)**	acceptable/ unacceptable/ ridiculous
c_3	*not-an-outlier*	*not-an-outlier*	**depends on M_3 (3/-1/-2)**	acceptable/ unacceptable/ ridiculous
c_2	not-an-outlier	suspicious	**-1**	unacceptable
c_1	not-an-outlier	outlier	**-2**	ridiculous

σ_N; thus, when the "M_2 is *outlier*" value is computed, μ_I, μ_N, σ_N are known. Consequently, the only variable that does not have a known value is σ_I. Note that the normal behavior is depicted by a solid bell-shape curve, while the intrusive behavior is depicted by different dashed bell-shape curves. As the standard deviation of the intrusive behavior increases, the bell-shape widens, and as a consequence the first metric M_1 changes its value from *outlier*, *suspicious*, to a *not-an-outlier* case since the normal behavior is gradually consumed by the intrusive one. We claim that the bigger the standard deviation of the intrusion gets, the better are the chances of detecting that attack, since in this case it is guaranteed that the mean of the intrusion will be an outlier and as a consequence most of the intrusion population will be too. Thus, we rank the case of "M_1 is *not-an-outlier*" higher than the case when "M_1 is *suspicious*", and higher than the case when "M_1 is *outlier*". Note that in all of these three cases (i.e., a_1, a_2, and a_3) the feature is heavily influenced by the attack.

The same reasoning is applied for the cases (b) and (c), except that as M_2 becomes *suspicious* and *not-an-outlier*, the confidence that the feature is influenced by attack ξ_j decreases, and thus, we need to use the third metric as help. However, as seen in Table III, we can safely infer that in both c_2 and c_3 cases the intrusive behavior is almost entirely contained into the normal one, and thus the feature becomes useless. The ranking column in Table III shows the final ranking of all nine cases, while the last column uses fuzzy terms to describe each of the fuzzy *terms* that will linguistically describe the output variable.

Once Table III is established, the rules of the fuzzy engine can be easily extracted, and are further listed in Table IV. Note that the only difference is the last rule which state that if M_3 is *poor* (i.e., all the intrusion is contained inside the normal behavior), then the final output of the FIE needs to be heavily punished.

C. Defining the membership functions for the output

Last step is to define the membership function for the output of the FIE module. Each membership function models one *linguistic variable* that quantifies the statistical significance of feature f_i in detecting the attack ξ_j based on the τ_k tuning factor (i.e., $P(f_i|\xi_j, \tau_k)$). The number of fuzzy *terms* used to describe the output is known, as well as their semantic order, that is *excellent*, *good*, *acceptable*, *unacceptable*, and *ridiculous*.

Next, each linguistic variable needs to be further characterized based on its impact on the *universe of discourse*; in our case each linguistic variable is defined by a fuzzy set that is *normal*, *convex*, *non distinct* having *partially overlapping* with the adjacent sets (see [7] for details on the used terminology). Once the number and order of the *terms* is established, the interval of the output variable is split into that many equal slices. We choose to have the output of the FIE in the $[0, 1]$ interval. Next, for each of the semantic terms a triangular membership function is selected

TABLE IV
THE LIST OF RULES USED IN THE FUZZY INFERENCE PROCESS OF THE FIE MODULE

no	**case**	**antecedent**	**consequent**
1	a_3	IF (M2 is outlier) AND (M1 is not-an-outlier)	THEN detection is excellent
2	a_2	IF (M2 is outlier) AND (M1 is suspicious)	THEN detection is good
3	a_1	IF (M2 is outlier) AND (M1 is outlier)	THEN detection is acceptable
4	b_3	IF (M2 is suspicious) AND (M1 is not-an-outlier)	THEN detection is good
5	b_2	IF (M2 is suspicious) AND (M1 is suspicious) AND (M3 is good)	THEN detection is acceptable
6	b_2	IF (M2 is suspicious) AND (M1 is suspicious) AND (M3 is mediocre)	THEN detection is unacceptable
7	b_2	IF (M2 is suspicious) AND (M1 is suspicious) AND (M3 is poor)	THEN detection is ridiculous
8	b_1	IF (M2 is suspicious) AND (M1 is outlier) AND (M3 is good)	THEN detection is acceptable
9	b_1	IF (M2 is suspicious) AND (M1 is outlier) AND (M3 is mediocre)	THEN detection is unacceptable
10	b_1	IF (M2 is suspicious) AND (M1 is outlier) AND (M3 is poor)	THEN detection is ridiculous
11	c_3	IF (M2 is not-an-outlier) AND (M1 is not-an-outlier) AND (M3 is good)	THEN detection is acceptable
12	c_3	IF (M2 is not-an-outlier) AND (M1 is not-an-outlier) AND (M3 is mediocre)	THEN detection is unacceptable
13	c_3	IF (M2 is not-an-outlier) AND (M1 is not-an-outlier) AND (M3 is poor)	THEN detection is ridiculous
14	c_2	IF (M2 is not-an-outlier) AND (M1 is suspicious)	THEN detection is unacceptable
15	c_1	IF (M2 is not-an-outlier) AND (M1 is outlier)	THEN detection is ridiculous
16	**n/a**	IF M3 is poor	THEN detection is ridiculous

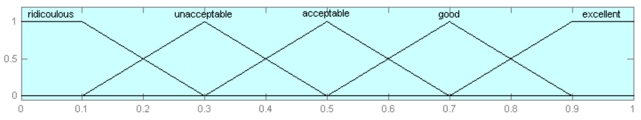

Fig. 6. The membership functions for the output of the FIE - graphical detail

that will be centred in the middle of its corresponding interval and have the right and left extremes extending half way into adjacent intervals. Furthermore, another common procedure in the fuzzy expert systems is to use trapezoidal membership functions for the left and right most terms. In this way, more weight is given to the extreme terms of the linguistic variable. Figure 6 graphically depicts the membership functions.

V. Experimental results

To validate our feature evaluation procedure, as a proof of concept, we concentrate on the IP, TCP, UDP, and ICMP protocols. We use our previously proposed feature classification schema[19] to select a comprehensive set of 933 features. Next, the experimental analysis is carried out using the DARPA dataset[4], after some of the issues regarding the dataset are addressed (see Sec. III-A). There are five main attack categories that exist in the dataset: Denial of Service (DoS), Probing, Remote to Local (R2L), User to Root (U2R) and Data attacks [4]. However, we report our results on DoS and Probing attacks only, since Data attacks represent the intention of the attacker, U2R attacks are invisible to network level features, and R2L attacks are application level attacks

TABLE V
THE TOP 15 FEATURES BY $P(f_i|\zeta_m)$, WHERE ζ_m INCLUDES ALL THE DOS AND PROBING INTRUSIONS IN THE DATASET.

no.	*Unique ID*[1]	$P(f_i\|\zeta_m)$			
		Overall	**Conn.Based**	**TimeBased1**	**TimeBased2**
1	DFMxB_54_TCP	**1.40**	1.09	**1.40**	**1.40**
2	DFMxOS_56_TCP	**1.40**	1.26	1.38	**1.40**
3	DFMxG_29_TCP	**1.38**	**1.38**	0.50	1.14
4	DFMxOS_26_TCP	**1.38**	1.32	1.34	**1.38**
5	DFMxB_04_TCP	**1.34**	**1.34**	1.20	1.13
6	DFMxB_64_TCP	**1.32**	1.31	1.31	**1.32**
7	DFMxOD_25_TCP	**1.32**	1.27	1.26	**1.32**
8	DFMxB_49_TCP	**1.31**	**1.31**	0.95	0.85
9	DFMxOS_10_TCP	**1.31**	0.90	1.15	**1.31**
10	DFMxB_47_TCP	**1.31**	**1.31**	1.17	1.24
11	DFMxOS_04_TCP	**1.30**	0.88	1.14	**1.30**
12	DFMxOD_63_TCP	**1.30**	1.25	1.24	**1.30**
13	DFMxOD_56_TCP	**1.29**	1.22	1.23	**1.29**
14	DFMxB_07_TCP	**1.28**	1.18	**1.28**	1.26
15	DFSxUS_08_TCP	**1.27**	**1.27**	0.35	0.72

[1](See the Glossary Table VIII)

TABLE VI
THE TOP 15 FEATURES BY $P(f_i|\zeta_m)$, WHERE ζ_m INCLUDES ALL THE DOS INTRUSIONS IN THE DATASET.

no.	*Unique ID*[1]	$P(f_i\|\zeta_m)$			
		Overall	**Conn.Based**	**TimeBased1**	**TimeBased2**
1	DFMxB_01_TCP	**1.52**	1.16	1.46	**1.52**
2	DFMxB_09_TCP	**1.52**	1.25	1.46	**1.52**
3	DFMxB_05_TCP	**1.51**	1.16	1.45	**1.51**
4	DFMxB_11_TCP	**1.51**	0.91	1.45	**1.51**
5	DFMxOS_56_TCP	**1.49**	1.27	1.46	**1.49**
6	DFMxB_07_TCP	**1.49**	1.37	1.42	**1.49**
7	DFMxOD_14_TCP	**1.47**	1.14	1.40	**1.47**
8	DFMxOD_04_TCP	**1.47**	1.14	1.40	**1.47**
9	DFMxOS_10_TCP	**1.47**	1.21	1.38	**1.47**
10	DFMxOS_04_TCP	**1.46**	1.19	1.36	**1.46**
11	DFMxOD_08_TCP	**1.46**	1.16	1.39	**1.46**
12	DFMxB_04_TCP	**1.46**	1.30	1.37	**1.46**
13	DFMxOD_23_TCP	**1.45**	0.92	0.73	**1.45**
14	DFMxOS_13_TCP	**1.45**	1.35	1.33	**1.45**
15	DFMxOD_56_TCP	**1.45**	1.19	1.37	**1.45**

[1](See the Glossary Table VIII)

that are implemented through legitimate connections from the network point of view.

In our experiments we use two implementation methods for the *time based features*, and one method for the *connection based features*. In the case of *time based features*, the first method uses a circular list to store the actual value of the feature during the specified time interval, (see [19]) while the second one uses exponential smoothing to store a weighted value of the feature (see [3]). The former method has the advantage of keeping the exact value of the feature, but it is resource expensive, whereas, the second method has the advantage of consuming a small amount of resources, but the disadvantage of loosing part of the information due to the decay factor. Hereafter we refer to these two methods as *TimeBased1* and *TimeBased2*, respectively. For the *connection based features*, we use the implementation method described in our previous work[19]. We will refer to this method as *ConnectionBased*.

TABLE VII
THE TOP 15 FEATURES BY $P(f_i|\zeta_m)$, WHERE ζ_m INCLUDES ALL THE PROBING INTRUSIONS IN THE DATASET.

no.	*Unique ID*[1]	$P(f_i\|\zeta_m)$			
		Overall	**Conn.Based**	**TimeBased1**	**TimeBased2**
1	DFSxUS_08_TCP	**1.45**	**1.45**	0.71	0.68
2	DFMxB_54_TCP	**1.44**	0.99	**1.44**	1.44
3	DFMxOS_68_TCP	**1.42**	1.41	**1.42**	1.42
4	DFMxB_63_TCP	**1.42**	1.41	**1.42**	1.42
5	DFMxOD_18_UDP	**1.42**	0.90	**1.42**	0.97
6	DFMxB_19_UDP	**1.42**	0.59	**1.42**	1.40
7	DFMxB_17_UDP	**1.42**	0.59	**1.42**	1.40
8	DFMxB_02_UDP	**1.42**	1.21	**1.42**	0.96
9	DFMxOS_20_UDP	**1.42**	0.83	**1.42**	1.40
10	DFMxOD_22_UDP	**1.42**	1.21	**1.42**	1.40
11	DFMxOD_05_UDP	**1.42**	1.21	**1.42**	0.95
12	DFMxOS_02_UDP	**1.42**	1.21	**1.42**	0.95
13	DFMxB_29_UDP	**1.41**	0.71	**1.41**	1.38
14	DFMxOD_35_UDP	**1.40**	0.66	**1.40**	1.38
15	DFMxOD_13_TCP	**1.40**	**1.40**	0.94	0.86

[1](See the Glossary Table VIII)

We use a set of 180 tuning values for each of the studied features as folows: for *time based features* we use a set of 30 values that range in the $[1, 3600]$seconds and a set of 6 different values that range in the $[1, 20]$seconds for the *time window size* and *window granularity step*, respectively. Similarly, for *connection based features* we use a set of 30 values in the range $[3, 200]$ connections and a set of 6 different values that range in the $[20, 1800]$ seconds for the *connection window size* and *time to live*, respectively.

Tables V, VI, and VII report the top 15 features based on our evaluation criteria that mostly contribute to the attack detection. Note that a feature f_i (e.g., "number of packets") is semantically different when implemented using *TimeBased1*, *TimeBased2*, and *ConnectionBased* (e.g., "number of packets in the last x seconds", "number of packets in the last x seconds with a decay factor", and "number of packets in the last x connections"). Consequently we treat it as three distinct features. However, after sorting the top 15 features, for comparison purposes we also report their overall performance when are implemented using the other two implementation techniques (e.g., if a feature performs best with *TimeBased1* technique, we write in bold in the table, but we also report its performance using *TimeBased2* and *ConnectionBased* features).

As a general remark, intuitively all the reported features seem to be able to capture DoS and Probing attacks, which makes us believe that our proposed feature evaluation model functions properly. However, the main outcome here is that the features are ranked using a deterministic method rather than an intuitive one based on expert knowledge.

When comparing *TimeBase1*, *TimeBased2*, and *ConnectionBased* techniques for the detection of DoS and Probing attacks, it appears that the time based approaches outperform the *ConnectionBased* method. This is expected since both DoS and Probing attack categories in the used dataset are fast attacks. This is also in agreement with what was previously believed that connection based features are more suitable for

TABLE VIII
GLOSSARY FOR TABLES 5 TO 7, ALPHABETICAL ORDER.

Unique ID	**Tab.**[1]	**A**[2]	**Description**[3]
DFMxB_01_TCP	6	sd	number of TCP connections between SrcIP and DstIP hosts
DFMxB_02_UDP	7	sd	number of UDP connections between SrcIP and DstIP hosts
DFMxB_04_TCP	5, 6	sd	number of TCP connections created by SrcIP using any port to connect to the DstIP host on PSrcPort
DFMxB_05_TCP	6	sd	number of TCP connections created by SrcIP using any port to connect to the DstIP host on any port but PSrcPort
DFMxB_07_TCP	5, 6	sd	number of TCP connections created by SrcIP using any port to connect to the DstIP host on any port but PDstPort
DFMxB_09_TCP	6	sd	number of TCP connections created by DstIP using any port to connect to the SrcIP host on any port but PSrcPort
DFMxB_11_TCP	6	sd	number of TCP connections created by DstIP using any port to connect to the SrcIP host on any port but PDstPort
DFMxB_17_UDP	7	sd	number of UDP connections created by DstIP using any port to connect to the SrcIP host on any port but PSrcPort
DFMxB_19_UDP	7	sd	number of UDP connections created by DstIP using any port to connect to the SrcIP host on any port but PDstPort
DFMxB_29_UDP	7	sd	number of UDP packets sent by SrcIP to DstIP
DFMxB_47_TCP	5	sd	number of TCP packets sent by SrcIP to DstIP with reset flag
DFMxB_49_TCP	5	sd	number of TCP packets sent by SrcIP to DstIP with finalize flag
DFMxB_54_TCP	5, 7	sd	number of TCP packets received by SrcIP from DstIP with reset flag
DFMxB_63_TCP	7	sd	number of already open TCP connections that match new SYN packets sent by SrcIP to DstIP
DFMxB_64_TCP	5	sd	number of TCP connections received by SrcIP from DstIP that do not start with SYN flag
DFMxG_29_TCP	5	n	number of TCP partially open connections that use CSrcPort (the 3-way handshake is not finished)
DFMxOD_04_TCP	6	d	number of TCP connections that connect to DstIP host
DFMxOD_05_UDP	7	d	number of UDP connections that connect to DstIP host
DFMxOD_08_TCP	5	d	number of TCP connections created by DstIP host using any port to connect to any other host using on any port but DstPort
DFMxOD_13_TCP	7	d	number of TCP connections created by any host using any port to connect to DstIP host on SrcPort
DFMxOD_14_TCP	6	d	number of TCP connections created by any host using any port to connect to DstIP host on any port but SrcPort
DFMxOD_18_UDP	7	d	number of UDP connections created by any host using any port to connect to DstIP host on any port but DstPort
DFMxOD_22_UDP	7	d	number of UDP connections created by any host using any port to connect to DstIP host on any port but SrcPort
DFMxOD_23_TCP	6	d	number of TCP partially open connections created by DstIP host
DFMxOD_25_TCP	5	d	number of TCP connections created by DstIP host which end with reset flag
DFMxOD_35_UDP	7	d	number of UDP packets received by DstIP
DFMxOD_56_TCP	5, 6	d	number of TCP packets received by DstIP with reset flag
DFMxOD_63_TCP	5	d	number of TCP packets sent by DstIP with reset flag
DFMxOS_02_UDP	7	s	number of UDP connections created by SrcIP
DFMxOS_04_TCP	5, 6	s	number of TCP connections that connect to SrcIP
DFMxOS_10_TCP	5, 6	s	number of TCP connections created by any host using any port to connect to SrcIP on any port but DstPort
DFMxOS_13_TCP	6	s	number of TCP connections created by any host using any port to connect to SrcIP on the SrcPort
DFMxOS_20_UDP	7	s	number of UDP connections created by SrcIP using any port to connect to any other host on any port
DFMxOS_26_TCP	5	s	number of TCP connections that connect to SrcIP and are reset
DFMxOS_56_TCP	5, 6	s	number of TCP packets received by SrcIP with reset flag
DFMxOS_68_TCP	7	s	number of already open TCP connections that match new SYN packets sent by SrcIP to any other host
DFSxUS_08_TCP	5, 7	c	number of TCP packets sent by SrcIP to DstIP with reset flag

[1] The list of tables where the feature appears.
[2] The following abbreviations are used: "c", "sd", "s", "d", and, "n", for the following abstractions: *Current Connection*, *SrcIP and DstIP Hosts*, *SrcIP Host*, *DstIP Host*, and *the Network*, respectively.
[3] The feature description. Note that some descriptions use "DstPort" and "SrcPort" to refer to the destination and source port of the current connections, while "PSrcPort" and "PDstPort" refer to the source and destination ports of the current packet

TABLE IX
THE SUMMARY OF FEATURE'S IMPACT OVER THE MAIN TYPES OF INTRUSIONS THAT EXIST IN THE DATASET.

	All attacks				**DoS**				**Probing**			
Abstraction (α)	$\overline{P(\alpha\|\zeta_m)}$	$\overline{\mathcal{F}(\alpha\|\zeta_m)}\%$	$\overline{\mathcal{T}(\alpha\|\zeta_m)}\%$	$\overline{\mathcal{N}(\alpha\|\zeta_m)}\%$	$\overline{P(\alpha\|\zeta_m)}$	$\overline{\mathcal{F}(\alpha\|\zeta_m)}\%$	$\overline{\mathcal{T}(\alpha\|\zeta_m)}\%$	$\overline{\mathcal{N}(\alpha\|\zeta_m)}\%$	$\overline{P(\alpha\|\zeta_m)}$	$\overline{\mathcal{F}(\alpha\|\zeta_m)}\%$	$\overline{\mathcal{T}(\alpha\|\zeta_m)}\%$	$\overline{\mathcal{N}(\alpha\|\zeta_m)}\%$
Current Connection	0.81	*.72*	*.28*	*.11*	0.81	*.70*	*.31*	*.11*	0.81	*.73*	*.20*	*.12*
SrcIP & DstIP Hosts	1.02	*.80*	*.49*	*.31*	1.06	*.80*	*.48*	*.41*	0.88	*.74*	*.38*	*.17*
SrcIP Host	0.98	*.76*	*.48*	*.33*	1.02	*.76*	*.47*	*.39*	0.87	*.72*	*.34*	*.23*
DstIP Host	0.95	*.74*	*.43*	*.32*	0.98	*.74*	*.42*	*.37*	0.83	*.70*	*.31*	*.21*
The Network	0.80	*.67*	*.33*	*.21*	0.79	*.66*	*.30*	*.22*	0.78	*.66*	*.32*	*.18*
Average	**0.92**	**.74**	**.40**	**.26**	**0.93**	**.73**	**.40**	**.30**	**0.83**	**.71**	**.31**	**.18**

the detection of stealthy attacks.

The overall statistics on all the studied features (not only the ones displayed in Tables V, VI, and VII) are summarized in Table IX. In this table all the features are grouped based on the network abstraction that they refer to. We use five main abstraction types as follows: *Current Connection*, *SrcIP and DstIP Hosts*, *SrcIP Host*, *DstIP Host*, and *the Network*[19]; to refer to a feature that is defined based on the current connection, all the connections between SrcIP and DstIP, all the connections between SrcIP and the network, all the connections between DstIP and the network, and all the connections in the network, respectively (also highlighted for each individual feature in the Glossary, Table VIII).

It is easily seen that the most suitable features in detecting the DoS and Probing attacks are the ones that take into account all the connections between SrcIP and DstIP hosts (i.e., *SrcIP and DstIP Hosts*); followed by all the features that take into consideration all the connections between the SrcIP and the network (i.e., *SrcIP host*). This is also reasonable since usually the initiator of a connection (i.e., SrcIP) is usually the attacker, while the target of a connection (i.e., DstIP) is usually the victim. This is also the main reason why *DstIP Host* does not perform well in detecting the probing attack. Basically in this case the only type of probing attacks these features can identify are the vertical scanning ones; whereas, the features that are defined based on the *SrcIP Host* can identify the horizontal ones too. Finally, as expected, the features that target the whole network (e.g., number of packets) come last based on our evaluation criteria.

VI. CONCLUSION AND FUTURE WORK

This work concentrates on the feature selection stage that any IDS must undergo during its design. We argue that this phase is among the most important ones, since a poor feature selection will lead to an inefficiency of the overall performance of the IDS. We propose a mechanism that combines fuzzy logic with statistical analysis, which allows us to study the effect of various intrusions and tunings over a certain features. The proposed feature evaluation model can be successfully applied to any type of feature not only numerical but also categorical (that can be easily converted into numerical ones).

Our experimental evaluations concentrate on the Transport, Network, and Network access layers of the TCP/IP Architecture model. We use our previously proposed feature classification schema[19] to select a set of 933 features that, we believe, cover a comprehensive set of features that can be extracted from IP, TCP, UDP, and ICMP protocols. As proof of concept, we use the DARPA[4] dataset for evaluating our model. Even though, the dataset is old, to our knowledge, there is not any other dataset that meets our criteria and was released after the DARPA. Thus, (in the case of DoS and Probing attacks,) we believe, that if we address some of the concerns about this dataset, it constitutes a good starting dataset to work with. We use a set of 180 tuning values for each of the studied feature and report our findings on DoS and Probing attacks.

Overall, our experimental results show that the evaluation model highlights features that have potential in detecting the DoS and Probing attacks; however, the novelty of our results is that this potential is deterministically quantified.

Our future work will concentrate not only on expanding the studied feature to the application layer, but also finding a more recent dataset to further evaluate the proposed feature evaluation model. We also intend to study other attack types and report the related features. Furthermore, one aspect that we would like to tackle is the importance in the detection process of multiple features working together. Finally, another interesting area of research would be to study and propose the most suitable feature tunings. Here the challenge is that different networks have different optimal tuning values.

Acknowledgment

The authors graciously acknowledge the funding from the Atlantic Canada Opportunity Agency (ACOA) through the Atlantic Innovation Fund (AIF) and through grant RGPN 227441 from the National Science and Engineering Research Council of Canada (NSERC) to Dr. Ghorbani.

References

[1] KDD 99. The fifth international conference on knowledge discovery and data mining. http://kdd.ics.uci.edu, Website, Last accessed Octomber 2005.

[2] J. L. Verdegay A. Sancho-Royo. Methods for the construction of membership functions. volume 14, pages 1213–1230, 1999.

[3] Jake D. Brutlag. Aberrant behavior detection in time series for network monitoring. In *LISA '00: Proceedings of the 14th USENIX conference on System administration*, pages 139–146, Berkeley, CA, USA, 2000. USENIX Association.

[4] DARPA. Darpa intrusion detection and evaluation dataset 1999. http://www.ll.mit.edu, Website, Last accessed February 2006.

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

[6] 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 *In ICDM Workshop on Data Mining for Computer Security (DMSEC)*, pages 30–39, Melbourne, FL, Nov. 19 2003.

[7] J.M. Garibaldi and R.I. John. Choosing membership functions of linguistic terms. In *Proceedings 2003 IEEE International Conference on Fuzzy Systems*, pages 578 – 583, 2003.

[8] H.R. Gibson. Elementary statistics. In *Wm. C. Brown Publishers, Dubuque, Iowa*, 1994.

[9] E.H. Mamdani. Advances in the linguistic synthesis of fuzzy controllers. In *International Journal of Man-Machine Studies*, volume 8, pages 669–678, 1976.

[10] E.H. Mamdani and S. Assilian. An experiment in linguistic synthesis with a fuzzy logic controller. In *International Journal of Man-Machine Studies*, volume 7, pages 1–13, 1975.

[11] John McHugh. Testing intrusion detection systems: A critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory. In *Proceedings of te ACM Transactions on Information and System Secrity*, volume 3, pages 262–294, November 2000.

[12] A. N. Zincir-Heywood P. Lichodzijewski and M.I. Heywood. Dynamic intrusion detection using self-organizing maps. In *Proceedings of the 14th Annual Canadian Information Technology Security Symposium - CITSS*, page Soft copy only, May 2002. http://www.sdl.sri.com/projects/emerald/live-traffic.html.

[13] A. Powers. Behavior-based ids: Overview and deployment methodology. White Paper Lancope, Inc., 3155 Royal Drive, Building 100, Alpharetta, Georgia 30022, USA, http://www.lancope.com, 2003.

[14] J. P. Thomas S. Chebrolu, A. Abraham. Hybrid feature selection for modeling intrusion detection systems. In *Proceedings of Lecture Notes in Computer Science*, volume 3316, pages 1020–1025, October 2004.

[15] M. Sugeno. *Industrial applications of fuzzy control*. Elsevier Science Pub, 1985.

[16] A.H. Sung and S. Mukkamala. Identifying important features for intrusion detection using support vector machines and neural networks. In *Proceedings of the IEEE Symposium on Applications and the Internet*, pages 209–216, January 2003.

[17] I. B. Turksen. *An Ontological and Epistemological Perspective of Fuzzy Set Theory*, chapter 3 Measurement of Membership. Elsevier Science, December 2005.

[18] Onut-Iosif Viorel and Ali A. Ghorbani. Toward a feature classification scheme for network intrusion detection. In *Proceedings of the 4th Annual Communication Networks and Services Research Conference*, pages 24–25, May 2006.

[19] Onut-Iosif Viorel and Ali A. Ghorbani. A feature classification scheme for network intrusion detection. In *Proceedings of the International Journal of Network Security*, volume 5, pages 1–15, July 2007.

[20] S. J. Stolfo W. Lee and K. W. Mok. Mining in a data-flow environment: Experience in network intrusion detection. In *Proceedings of the 5 International Conference on Knowledge Discovery and Data Mining*, pages 114–124, 1999.

[21] R.E. Walpole. In Macmillan, editor, *Elementary Statistical Concepts, Second edition*, 1983.

[22] L.A. Zadeh. Fuzzy algorithms. In *Information and Control*, volume 8, pages 338–353, 1965.

[23] L.A. Zadeh. Outline of a new approach to the analysis of complex systems and decision processes. In *IEEE Transactions on Systems, Man, and Cybernetics*, volume 3, pages 28–44, Jan. 1973.

[24] L.A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning, parts 1, 2, and 3. In *Information Sciences*, volume 8,9, pages 199–249,301–357,43–80, 1975.