5 SYNTHESIS COMPUTING CLASSIFIERS

The earlier chapters explain the using of decision tree, fuzzy system and Neuro-Fuzzy System classifier identifies attackers. Those methods having the own advantage and disadvantage covered in the earlier chapter, for more accuracy and removing ambiguous of the attackers, so use advantages of the discussed methods combination of the new system proposed in this chapter. The new method is the Synthesis Computing Classifier, and covered the process of the system in the chapter. The explanation divides following parts:

1. Proposed method outline
2. Type -2 Fuzzy System

Classification done using such knowledge is known as knowledge base classification. But such classification needs strong knowledge base, which sometimes become drawback of this process because of the knowledge acquisition process. Generally knowledge base is created with the help of knowledge acquired by interacting with the experts. The traditional way of knowledge acquisition is that the knowledge engineer interacts with the corresponding domain expert; write up his /her experience and knowledge in a interpretable form and then feed the entire acquired knowledge in the computer in a symbolic form such as if-then rules.

If-then rules are generate from different methods of the classifiers, but each method having own advantages and disadvantages, here problem building of the better rule system comparing with all. In building process one method is not sufficient for build better system. Then using combination of the different computing methods for build batter knowledge and identify the attacker using the knowledge. Here very important rule bounder, rule is false or true that’s depends on rule bounder. Construction rules with better bounder with combination of the different rules, different rules from the different classifiers.
5.1 Outline of the Proposed Method

Mainly motivation of the proposed method is decision boundaries are not always clear cut. That is, the transition from one class in the feature space to another is not discontinuous, but gradual.

This effect is common in fuzzy logic based classification algorithms, where membership in one class or another is ambiguous. Based on the discussed problem here required construct better rules set for classification of the attackers. Rules set bounder are very important to construct better rules set, identified problems are solving using proposed novel methodology.

The rule-base with multiple antecedent and single consequent variables is

Rule 1: if num_compromised > 0 then Class DOS
Rule 2: if num_compromised <= 0 and src_bytes > 51 and num_file_creations <= 0 then Class Normal
Rule 3: if src_bytes <= 51 and srv_count <= 1 and dst_host_same_src_port_rate <= 0.49 then Class R2L
Rule 4: if dst_host_same_src_port_ > 0.49 then Class Probing

Compactly decision tree and fuzzy system results set are different when compare with accuracy and rules bounder are different in the two methods rules set. Here proposed method final generate different bounder rules set. Final find rule set on based of the accuracy and rules generation constriction parameter.

Rules modification using following different methods:

- Mean and Standard Division parameters
- Probability threshold and Likelihood parameters.
- Fuzzy Decision Boundary
Final testing accuracy parameters identify the best approaches generation for rule construction. Next step explain build of the best rules for identify attackers.

![Diagram](image)

**Figure 5.1: Proposed outline method**

Our proposed model shows Figure 5.1, in this method using four classifier techniques those are decision tree, fuzzy, neuro-fuzzy and type-2 fuzzy logic. First method covered in chapter 3, explained about decision tree advantages and disadvantages for construct rules set generation and methodology totally in same chapter.

Next two methods are explained in chapter 4; in that discussed main use of fuzzy and neuro-fuzzy systems for identify attackers. Final method, explain in this chapter. Next of the methodology is build decision boundary for generated rules set, these concept cover in this chapter 5.2 to 5.4. Before finally step generates common rules set for IDS system based training data sets. Then finally step is classified the test data set and measure accuracy using proposed parameters and results set comparison cover upcoming chapter 6.
5.2 Type 2 Fuzzy Logic Systems

Type-1 fuzzy logic has been successful in many applications; however, the type-1 approach has problems when faced with dynamical environments that have some kinds of uncertainties.

These uncertainties exist in the majority of real world applications and can be a result of uncertainty in inputs, uncertainty in outputs, uncertainty that is related to the linguistic differences, uncertainty caused by the conditions change in the operation and uncertainty associated with the noisy data when training the FLC [11]. All these uncertainties translate into uncertainties about fuzzy sets membership functions [11]. Type-1 fuzzy Logic cannot fully handle these uncertainties because type-1 fuzzy logic membership functions are totally precise which means that all kinds of uncertainties will disappear as soon as type-1 fuzzy set membership function has been used [12].

The existence of uncertainties in the majority of real world applications makes the use of type-1 fuzzy logic inappropriate in many cases especially with problems related to inefficiency of performance in fuzzy logic control [12]. Also, interval type-2 fuzzy sets can be used to reduce computational expenses. Type-2 fuzzy systems have, potentially, many advantages over type-1 fuzzy systems including the ability to handle numerical and linguistic uncertainties, allowing for a smooth control surface and response and giving more freedom than type-1 fuzzy sets [12]. Since last decade, type-2 fuzzy logic is a growing research topic with much evidence of successful applications [13].

Differences between the Type 1 and 2 are showing below Figure 5.2. Type 1, or regular fuzzy sets, are mathematical tools to model vagueness and imprecision and the uncertainty, however, cannot be captured by fuzzy sets (or subsets) and Type 2 fuzzy sets, or two dimensional fuzzy sets are able to deal with both vagueness and uncertainty.
Type-1 Fuzzy System

Type-2 Fuzzy System

Figure 5.2: Differences between Type 1 and Type 2 Fuzzy Sets

All fuzzy sets are characterized by membership functions. Type-1 fuzzy sets are characterized by two-dimensional membership functions in which each element of the type-1 fuzzy set has a membership grade that is a crisp number in \([0, 1]\).

Type-2 fuzzy sets are characterized by \textit{fuzzy membership functions} that are three-dimensional. The membership grade for each element of a type-2 fuzzy set is a fuzzy set in \([0, 1]\). The additional third dimension provides additional degrees of freedom to capture more information about the represented term. Type-2 fuzzy sets are useful in circumstances where it is difficult to determine the exact membership function for a fuzzy set, which is useful for incorporating uncertainties.

Type-1 fuzzy sets handle uncertainties by using precise membership functions that the user believes capture the uncertainties. Once the type-1 membership functions have been chosen, all the uncertainty disappears, because type-1 membership functions are totally precise. However, type-2 fuzzy sets handle uncertainties about the meaning of the words that they represent by modeling the uncertainties using type-2 membership functions. Fuzzy logic systems (FLSs) which are used for representing and inferring with knowledge that is imprecise, uncertain, or unreliable consists of four main interconnected components: \textit{rules, fuzzifier, inference engine, and output processor}. 
Once the rules have been established, a FLS can be viewed as a mapping from inputs to outputs. The rules are the heart of a FLS and can be expressed as a collection of IF-THEN statements. The IF-part of a rule is its antecedent, and the THEN-part of a rule is its consequent. Fuzzy sets are associated with terms that appear in the antecedents or consequents of rules, and with the inputs to and the output of FLS. Type-1 FLSs use type-1 fuzzy sets and a FLS which uses at least one type-2 fuzzy set is called a type-2 FLS. Type-2 fuzzy sets let us model the effects of uncertainties in rule-based fuzzy logic systems (FLSs).

There are (at least) four sources of uncertainties in type-1 FLSs:

- The meanings of the words that are used in the antecedents and consequents of rules can be uncertain (words mean different things to different people)
- Consequents may have a histogram of values associated with them, especially when knowledge is extracted from a group of experts who do not all agree
- Measurements that activate a type-1 FLS may be noisy and therefore uncertain
- The data that are used to tune the parameters of a type-1 FLS may also be noisy

Mainly in Type–1 fuzzy sets membership functions are completely crisp so that reason not able to changes such uncertainties. Other side of, Type -2 fuzzy sets membership function are itself fuzzy so very easy to model such uncertainties

Type-2 FLSs are applicable when

- The data-generating system is known to be time-varying but the mathematical description of the time-variability is unknown (e.g., as in mobile communications)
- Measurement noise is no stationary and the mathematical description of the nonstationarity is unknown (e.g., as in a time-varying SNR)
- Features in a pattern recognition application have statistical attributes that are nonstationary and the mathematical descriptions of the nonstationarities are unknown
• Knowledge is mined from a group of experts using questionnaires that involve uncertain words
• Linguistic terms are used that have a nonmeasurable domain

All application areas of the Type-1 FLSs are also supporting Type 2 FLSs. And Type-2 Fuzzy sets can also be applied to non-rule based applications of fuzzy sets, if uncertainty is present. One of the specific areas in which there are no doubt lots of uncertainties present is IDS. Fuzzy logic methods are already used in IDS (already covered in chapter 4), a field that abounds in uncertainties. Rule-based FLSs that account for all kinds of uncertainties would provide better information regards attackers with decision-making flexibilities. IDS are an application where both linguistic and numerical rules will need to be developed.

In this thesis used A to represent a Type-1 fuzzy set, and the membership grade of \( x \in X \) in \( \mu_A(x) \) which is a crisp number \([0, 1]\). If \( X \) is a continuum, \( A \) can be represent as

\[
\int \circ \quad (1)
\]

The integral denotes logical union.
If \( X \) is discrete, the integral in (1) is replaced by a summation.

A Type-2 fuzzy set in \( X \) is \( \tilde{A} \), and the membership grade of \( x \in X \) in \( \tilde{A} \) is \( \mu_{\tilde{A}}(x) \), which is type-1 fuzzy set whose domain in \([0, 1]\). The elements of the domain of \( \mu_{\tilde{A}}(x) \) are called Primary memberships of \( x \) in \( \tilde{A} \) and the memberships in \( \mu_{\tilde{A}}(x) \) are called Secondary Memberships of \( x \) in \( \tilde{A} \). Secondary membership defines the possibilities for the primary membership function. The membership grade of any \( x \in X \) in \( \tilde{A} \) can be represented as

\[
\int_{[0,1]} \circ \quad (2)
\]
When the secondary membership functions are type-1 interval sets, the type-2 set is called an interval type-2 set. Interval set-2 are simplest kind of type-2 and [10] presents fast algorithms to compute the output of a type-2 FLS which uses interval type-2 sets. Type-2 fuzzy logic systems and type-1 fuzzy logic systems are similar rule based system design in term of the structure and components but type-2 has itself some extra output process component that is type-reducer before defuzzification. The type-reducer reduces output type-2 fuzzy sets to type-1 fuzzy sets then the defuzzifier reduces it to a crisp output.

### 5.2.1 Components of the Type-1 Fuzzy System

Main components of a type-2 fuzzy system are:

- Fuzzifier
- Rules
- Inference Engine
- Output Processor

**Fuzzifier**

Fuzzifier maps crisp inputs into type-2 fuzzy sets by evaluating the crisp inputs \( x = (x_1, x_2, \ldots, x_n) \) based on the antecedents part of the rules and assigns each crisp input to its type-2 fuzzy set \( \tilde{A}(x) \) with its membership grade in each type-2 fuzzy set.

**Rules**

A fuzzy rule is a conditional statements in the form of IF-THEN where it contains two parts, the IF part called the antecedent part and the Then part called the consequent part.
Inference Engine

Inference Engine maps input type-2 fuzzy sets into output type-2 fuzzy sets by applying the consequent part where this process of mapping from the antecedent part into the consequent part is interpreted as a type-2 fuzzy implication which needs computations of union and intersection of type-2 fuzzy sets and a composition of type-2 relations by using the extended sup-star composition for type-2 set relations. The inference engine in a Mamdani system maps the input fuzzy sets into the output fuzzy sets then the defuzzifier converts them to a crisp output. The rules in Mamdani model have fuzzy sets in both the antecedent part and the consequent part. For example, the \( i \)th rule in a Mamdani rule base can be described as follows:

\[
R^i : \text{IF } x_1 \text{ is } \tilde{A}^i_1 \text{ and } x_2 \text{ is } \tilde{A}^i_2 \text{ .. and } \tilde{A}^i_p \text{ then } y \text{ is } \tilde{O}^i
\]

Output Processor

There are two stages in the output process:

Type-Redcuer:
Type-reducer reduces type-2 fuzzy sets that have been produced by the inference engine to type-1 fuzzy sets by performing a centroid calculation [10].

Defuzzifier:
Defuzzifier maps the reduced output type-1 fuzzy sets that have been reduced by type-reducer into crisp values exactly as the case of defuzzification in type-1 fuzzy logic systems.
5.2.2 Operation on Type-2 Fuzzy Set

The membership grades of type-2 sets are type-1 sets; therefore, in order to perform operations like union and intersection on type-2 sets, we need to be able to perform t-conform and t-norm operations between type-1 sets. This is done using Zadeh’s Extension Principle and also proven in using a representation method without having to use the Extension Principle.

Type 2 Fuzzy Set supports following operators
Binary operation
Negation operation
5.3 *Common Rule Set Generation*

An information system is a pair $A = (U, A)$ where $U$ is a non-empty, finite set called the Universe and $A$ – a non empty finite set of attribute.

Elements of $U$ are called objects and interpreted eg: any variable, rule character etc.

Here consider a special case of information system called decision table. A decision table is an information system of the form

$$A = (U, A \cup \{d\}) \quad \text{--------- 5.8}$$

Where $d$ set not in $A$ is a distinguished attribute called the decision. The element are of $A$ is called conditions.

Each rules Tables are called training sets of example in machine learning.

Here $d(U)=\{k : d(s) = k \text{ for some } s \in U\}$ is called the rank of $d$ and denoted by $r(d)$.

One can use approximate decision rule instead of optimal decision rules to construct the classification algorithm for decision table rules $A$ from the generated rules from different classifiers. In common rule generation method with algorithms for synthesis of optimal decision rules from a given decision rules list of tables. Next, compute approximate rules from already calculated optimal decision rules.

In novel method is based on the notion of consistency of decision rule. The original optimal rule is reduced to an approximate rule with coefficient of consistency exceeding a fixed threshold.

Let $A = (U, A \cup \{d\})$ be a decision rule table and $r_0 \in \mathcal{R U L}(A)$. The approximate rule based on rule $r_0$ is computed by the following algorithm.
Algorithm 1 Approximate rule synthesis (by descriptor dropping)

Input:
1. decision table $A = (U, A \cup \{\text{d}\})$,
2. decision rule $r_0 \in \text{RULE}(A)$,
3. threshold of consistency $\mu_0$ (e.g. $\mu_0 = 0.9$).

Output: the approximate rule $r_{\text{app}}$ (based on rule $r_0$).

Method:
- Calculate the coefficient of consistency $\mu_A(r_0)$
- If $\mu_A(r_0) < \mu_0$ then STOP (in this case no approximate rule).
- $\mu_{\text{max}} = \mu_A(r_0)$ and $r_{\text{app}} = r_0$.
- While $\mu_{\text{max}} > \mu_0$ do
  - begin
    - $\mu_{\text{max}} = 0$
    - For $i = 1$ to the number of descriptors from Pred($r_{\text{app}}$) do
      - begin
        - $r = r_{\text{app}}$
        - Remove $i$-th descriptor from Pred($r$).
        - Calculate the coefficient of consistency $\mu_A(r)$ and $\mu = \mu_A(r)$.
        - If $\mu > \mu_{\text{max}}$ then $\mu_{\text{max}} = \mu$ and $i_{\text{max}} = i$.
      - end
    - If $\mu_{\text{max}} > \mu_0$ then remove $i_{\text{max}}$-th conditional descriptor from $r_{\text{app}}$.
  - end
  - Return $r_{\text{app}}$.

The time and space complexity the algorithm 1 is $O(l^2.m.n)$ and $O(C)$.

The approximate rules, generated by the above method, can help to extract more important rules from decision rules table. By applying approximate rules instead of optimal rules one can slightly decrease the quality of classification of objects from the training set but expect in return to receive more general rules with the higher quality of classification for new objects and more and new attacks easy identify use the above methods.
5.4 **Strength of Rule Set**

Next step in common rules set, after using above approximation algorithm also even having some conflict using rules that’s means set rules related to the common class. Main about rule weight discussed in chapter 3, but that’s covered only for single decision tree, that’s not possible applicable in this section (common rules set). So, presented in this section several methods for constructing the measure called strength of rule set. The strength of rule set is a rational number belonging to [0, 1] representing the important of the sets of decision rules relative to consider tested objects.

Let us assume that:

- $W= (W, A \cup \{d\})$ is a universal decision rules.
- $A = (U, A \cup \{d\})$ is a given different decision classifier table rules
- $u_t \in W$ is a tested object,
- $Rul(X_j)$ is a set of all calculated basic decision rules for $A$, classifying objects to the decision class $X_j$
- $MRul(X_j, u_t)$ sub set equal $Rul(X_j)$ is asset of all decision rules from $Rul(X_j)$ matching tested objects $u_t$.

Defined and used server measures for the rule set $MRul(X_j, u_t)$ depending on the number of rules from this set matching tested objects, the number of objects supporting decision rules from this set and the stability coefficient of rules.

1. A simple rule of decision rule is defined by

$$
Simple \ Strength \ (X_j, u_t) = \frac{\left( \begin{array}{c} 
\cdot \\
\cdot 
\end{array} \right)}{\left( \begin{array}{c} 
\cdot \\
\cdot 
\end{array} \right)}
$$

2. A maximum Strength of decision rule set is defined by
Maximal Strength \((X_j, u_t) = \max_{r \in M} Rul(X_j, u_t)\)

3. A global Strength of decision rule set is defined by
\[
\text{Global Strength} (X_j, u_t) = \left( \frac{\{ \cdot \} \cap \{ \} \cap \{ \} = \{ \}}{\{ \} = \{ \}} \right)
\]

4. A Stability Strength of decision rule set is defined by
\[
\text{Stability Strength} (X_j, u_t) = \max_{r \in M} Rul(X_j, u_t) \left\{ \{ \} \} \}
\]

5. A Global Stability Strength of decision rule set is defined by
\[
\text{Global Stability Strength} (X_j, u_t) = \left( \{ \cdot \} \cap \{ \} \cap \{ \} = \{ \} \cap \{ \} = \{ \}
\]

Here used above strength rules applied constructing classification algorithms of classification. Applied rules order to classifier dependent on the strength of the rules, so will solved based this point conflict. Results and accuracy of the rules set discussion will cover at section 6.3.