CHAPTER 3 RESEARCH MOTHODOLOGY

From the literature review presented in chapter 2, the Fuzzy Logic was often chosen as an approach for network intrusion detection with low research consumption. It was also robust for unknown attack detection. Thus, we are interested in the Fuzzy Logic and the Genetic Algorithm. The Genetic Algorithm can help the fuzzy logic to learn a new data/solution in changing an environment. Therefore, the fuzzy genetic algorithm is chosen for our network-based intrusion detection. Moreover, we test the performances of our intrusion detection approach with both known and unknown network data.

Chapter three is organized as follows: the overview of our IDS system is described at the beginning of the chapter. Then, section 3.1 explains how our IDS preprocesses the online data and shows examples of the online dataset. Section 3.2 shows an IDS algorithm (fuzzy genetic algorithm), section 3.3 explains the testing method and evaluation criteria of our IDS system and section 3.4 displays simulation tools.

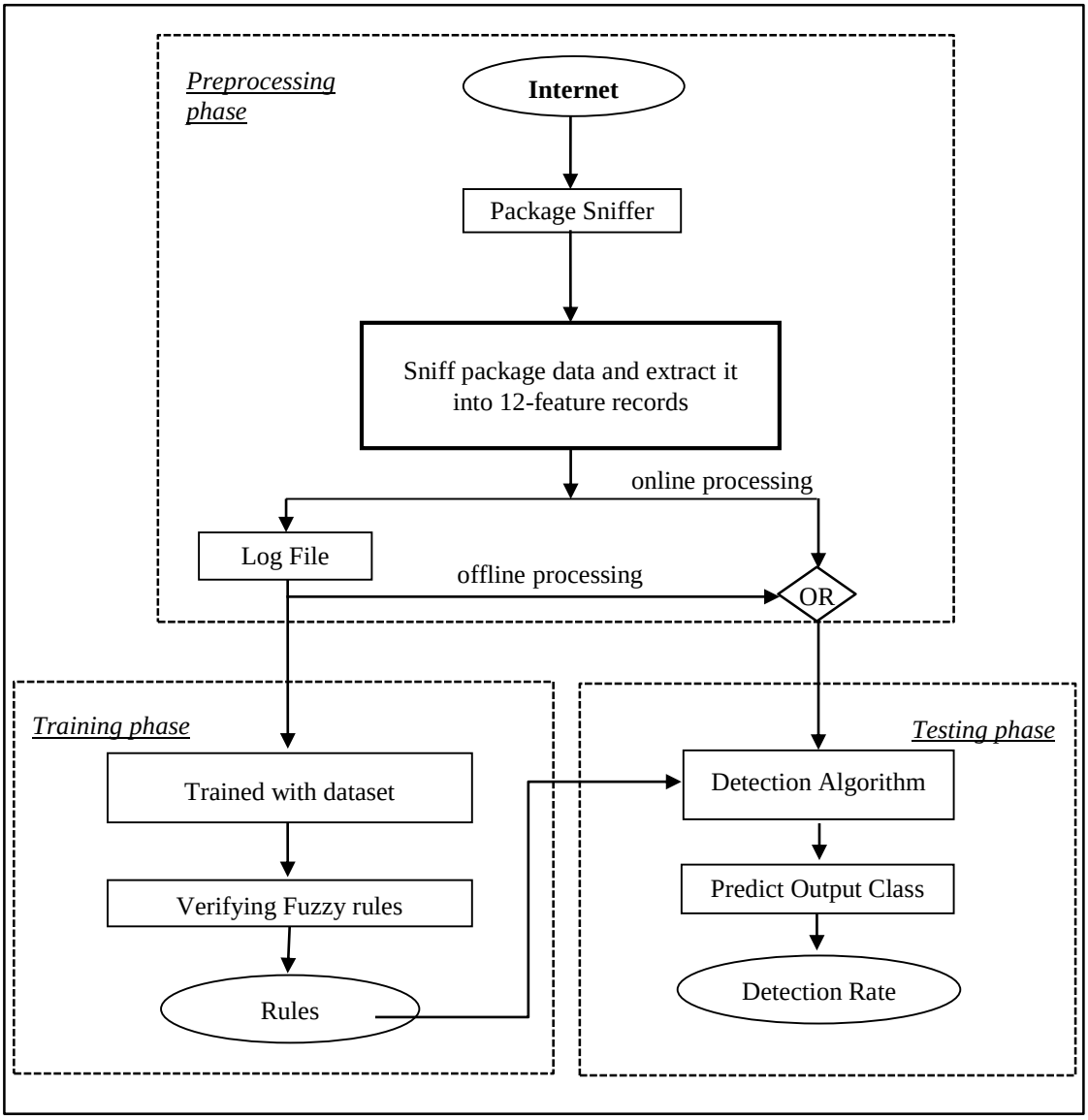

Figure 3.1 Real-time detection model

Our Real-time IDS, shown in Figure 5, consists of three phases: the pre-processing phase, the training phase and the testing phase. First, we create a network dataset by capturing the network data in CPE department of King Mongkut's University of Technology Thonburi in different time in a day for 1 month. The data packets are pre-processed using a packet header. The essential features which represent the network activity are extracted from this data. The extracted features are considered as the key-signature features, representing the main characteristics of the data. Then, the pre-processed data with the key signature extraction is sent to the training phase so that we can obtain fuzzy rules. In the training phase, the fuzzy rules are evolved by a genetic evolution concept. We can evaluate the performances of the fuzzy rules in the testing phase. Moreover, we can use the fuzzy rules to detect network attacks in an actual network environment.

3.1 Preprocessing Phase

In the pre-processing phase, we use a packet sniffer to extract network packet information as described in Komviriyavut et al [15]. This is shown in Figure 5. Each record consists of 12 data features. The features along with the data types are shown in Table 1.

Table 3.1 Twelve essential features in pre-processed data [15]

No.	Feature Description	Data Type
1	number of tcp packets	integer
2	number of tcp source ports	integer
3	number of tcp destination ports	integer
4	number of tcp fin flags	integer
5	number of tcpsyn flags	integer
6	number of tcp push flags	integer
7	number of tcpack flags	integer
8	number of tcp urgent flags	integer
9	number of udp packets	integer
10	number of udp source ports	integer
11	number of udp destination ports	integer
12	number of icmp packets	integer

The packet will be captured using Jpcap library [ref] for information extraction, the program will consider a connection between any two IP addresses (source IP and destination IP) and form a record for every 2 seconds. Then, the record will be sent to the detection phase in order to classify the attacks.

Examples of the data records where each record has 12 feature values and is labeled with its type (i.e. a normal data or an attack) can be shown as follows:

21,21,15,0,21,0,0,0,0,0,0,0, attack
169,2,90,169,0,0,0,0,0,0,0,0, attack
0,0,0,0,0,0,0,0,0,0,0,12683, attack
6,2,2,2,0,2,6,0,0,0,0,0, normal
111,2,2,0,0,2,111,0,0,0,0,0, normal
102,2,2,0,0,1,102,0,0,0,0,0, normal

3.2 Training Phase

In the detection phase, we use the fuzzy genetic algorithm as described in Fries [2]. The algorithm uses the data from a log file with the fuzzy genetic algorithm to train the rule. In this section, we will describe an idea of the fuzzy algorithm in section 3.2.1, an idea of the genetic algorithm in section 3.2.2, a methodology to encode the fuzzy rules in section 3.2.3 and the Fuzzy Genetic Algorithm in section 3.2.4.

3.2.1 Fuzzy Logic Algorithm

The fuzzy parameter is in between a range of 0-7 and the fuzzy value is between 0-1. The fuzzy rule is applied to each feature by using this set of the parameters {a, b, c, d}.

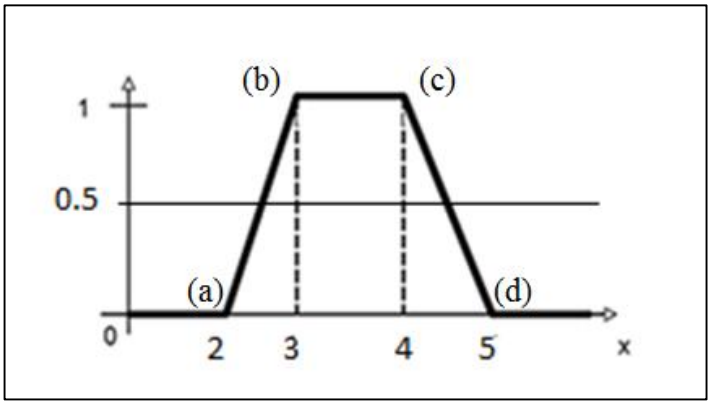

Figure 3.2 Trapezoidal fuzzy set {a=2, b=3, c=4, d=5}

From Figure 6, we can calculate the fuzzy value using these four equations:

1. if the data records between b and c, *probe =1;*
2. if the data records between a and b, $prob = \frac{attribute\_value - a}{b - a}$
3. if the data records between c and d, $prob = \frac{d - attribute\_value}{d - c}$
4. otherwise, *probe = 0;*

3.2.2 Genetic Algorithm

Procedure GA:

```
Initialize:
    Initialize population P(t)

while (not (termination condition))
    {
    Create offspring F(t)
    Evaluate offspring F(t)
    Insert offspring in the population F(t)→ P(t)
    }
```

A Genetic Algorithm uses an evolutionary method to find the best solution. Each solution is encoded into a string called *"chromosome"*. At the beginning, the chromosomes are randomly initiated. A group of the chromosomes is called *"population-P(t)"* and *"offspring-F(t)"*. In creating an offspring step, the algorithm creates a new set of the chromosome using a reproduction method as described in next paragraph. Then, it evaluates values of the offspring, and inserts the offspring F(t) to the population P(t). The new generation of the chromosomes will be created and replaced the old generation until it reaches stopping criteria. The stopping criterion in this experiment is set to a certain number, such as 5,000 generations.

There are five ways of reproduction as follows:

I. Crossover: the approach to create a new chromosome from an existing chromosome by exchanging parts of the chromosomes (genes) between two chromosomes. We use one-point crossover, i.e.

Parent	p1:	A-B-C-D-|E-F-G-H-I-J
	p2:	D-E-H-A-|B-J-G-F-I-C
Child	c1:	A-B-C-D- B-J-G-F-I-C

II. Mutation: the approach to create a new chromosome from an existing chromosome by randomly choosing the chromosome to mutate and randomly changing its gene(s). This approach is applied to avoid the GA trapping in a local optimum. In our approach, we randomly choose two genes for mutation.

Parent	p1:	D-E-**H**-A-B-**J**-G-F-I-C
Child	c1:	D-E-**F**-A-B-**A**-G-F-I-C

- At point 3rd gene H mutate to F
- At point 6th gene J mutate to A

III. Alien: the program creates a new chromosome by randomly choosing every gene in the chromosome.

IV. Elitism best chromosome: the program keeps the best chromosome from the current population. The strongest chromosome will exist in the next generation.

3.2.2 String Encoding

a. Each feature will be encoded into the string as follows:

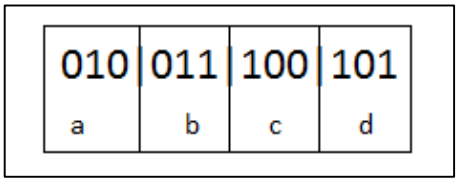

Figure 3.3 Fuzzy encoding for each feature {a=2, b=3, c=4, d=5}

b. Each chromosome refers to each feature. The records will be encoded as the chromosomes below which are series of fuzzy parameters for 12 features and the class at the end of the string.

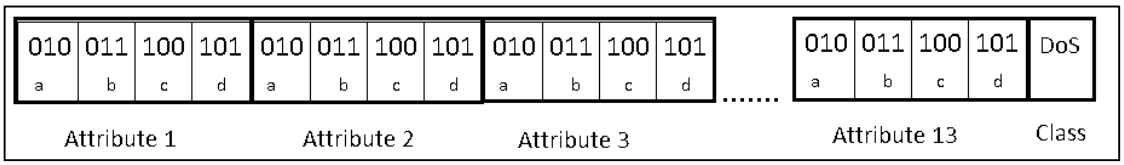

Figure 3.4 Encoding string

3.2.4 Fuzzy Genetic Algorithm

A Fuzzy Rule was developed by the genetic algorithm in order to find best fuzzy rules. The step of the algorithm is described below.

1. Initial population: Each generation has 20-50 chromosomes.
2. Finding probability: we calculate probabilities of beginning an attack for each feature, and then summarize all features in that record.
3. Classify Attack: we set the threshold = 0.5. When the total probabilities are less than 0.5, the record will be classified as normal.

```
Initial rules ();
while{
    for each record {
        for each rule{
        for each attribute{
                prob = fuzzy();
                totalprob = totalprob + prob;
            }
            If (totalprob> threshold)
            class is attack;
      else
            class is normal;
          }
compare the predicted result with actual result
find  A, B, α, and β,
    }
                Calculate fitness
    // create next generation
    Evolutionary process();
```

Figure 3.5 Fuzzy genetic algorithm pseudo code

4. Finding A, B α, and β, where
   - A is a number of attack records in the dataset.
   - B is a number of normal records in the dataset.
   - α_i is a number of correctly identified attacks (α_i) for each chromosome
   - β_i is a number of normal connections incorrectly characterized as attacks (false positive, β_i) for each chromosome
   - Summarize α for this generation and summarize β for this generation

$$\alpha = \sum_{i=1}^{n} a_i \qquad \textit{where n is a number of records in the dataset}$$

$$\beta = \sum_{i=1}^{n} \beta_i$$

5. Calculate a fitness value:
   The program will calculate a fitness value for each rule by using equation below.

$$\textit{fitness function} = \frac{\alpha}{A} - \frac{\beta}{B} [3]$$

6. Evolutionary process:
   The program will generate next generation. The next generation includes
   20% of population from the current rule that has the highest value of fitness
   30% of population from the crossover method
   20% of population from the mutation method
   30% of population from the alien method

3.3 Detecting Phase

The testing phase is a process to evaluate performances of our algorithm. In this phase, a user can select the dataset from a log-file in order to evaluate accuracy of the fuzzy rule or connect to a real network environment in order to evaluate other performances of the IDS such as resource consumption and computation time. There are three steps in the testing phase as described below.

3.3.1 Data Normalization: The system will normalize each testing record to range 0-7, the maximum and minimum bounds are imported from the training phase. If the value of a testing record is greater than the maximum, the normalized value will be 7;

3.3.2 Data Classification: We use the rules from the training phase to classify the attack class and the normal class. There are 2 types of classification processes.

a. One rule classification: the program uses one rule to classify.

b. Two-rule classification: the program uses two rules together to classify network attacks. Then, the program will compare probability of being attacked from each rule with the threshold below.

If $probability_{rule1} > threshold$ *or* $probabilityes_{rule2} > threshold)$
then classify as attack.
else
classify as normal

3.3.3 Evaluation Criteria: There are four parameters that are used to evaluate accuracy of this algorithm and are described below.

- Detection rate (DR) is the percentage of the normal and attack classes correctly classified from the total number of the data records.
- True-positive rate (TP) is the percentage of the normal class correctly classified from the total number of the data records.
- True-negative (TN) is the percentage of the attack class correctly classified from the total number of the data records.
- False-positive (FP) is the percentage that the normal data records are classified as attacks from the total number of the normal data records.
- False-negative rate (FN) is the percentage that the attacks are misclassified from the total number of the attack records.

3.4 Simulation Tools

In this work, we use simulation tools to generate attacks in close environments in order to create training datasets and also use simulation tools to test our intrusion detection system. There are 17 types of the attacks that are interesting for real-time datasets including 4 types of DoS attack type and 13 types of Probe attack type. Table 3.2 shows each name of the attacks and the simulation tools.

Table 3.2 Attack type and simulation tools [15]

No.	Data	Tools	Category
1	Smurf	Smurf.c	DoS
2	UDP Flood	Net Tools 5	DoS
3	HTTP Flood	Net Tools 5	DoS
4	Jping	Jping.c	DoS
5	Port Scan	Net Tools 5	Probe
6	Advance Port Scan	Net Tools 5	Probe
7	Host Scan	Host Scan 1.6	Probe
8	Connect	Nmap Win 1.3.1	Probe
9	SYN Stealth	Nmap Win 1.3.1	Probe
10	FIN Stealth	Nmap Win 1.3.1	Probe
11	UDP Scan	Nmap Win 1.3.1	Probe
12	Null Scan	Nmap Win 1.3.1	Probe
13	Xmas Tree	Nmap Win 1.3.1	Probe
14	IP Scan	Nmap Win 1.3.1	Probe
15	ACK Scan	Nmap Win 1.3.1	Probe
16	Window Scan	Nmap Win 1.3.1	Probe
17	RCP Scan	Nmap Win 1.3.1	Probe
18	Normal	Actual Environment	Normal

CHAPTER 4 EXPERIMENTAL RESULTS

From chapter 3, our proposed algorithm is the fuzzy genetic algorithm. In this chapter, we will demonstrate our proposed approach (fuzzy genetic algorithm) with various views. We perform two types of experiments to evaluate our IDS including Offline IDS (section 4.1) and Online IDS (section 4.2). In the offline IDS, we implement and test our system using an operating system: Ubuntu ver.12.04. We use two datasets including the KDD99 dataset (discussed in section 4.1.1) and the real-time dataset (discussed in section 4.1.2). Moreover, we test our IDS with unknown attacks, shown in section 4.1.3 and compare it with various algorithms in section 4.1.4. In the online IDS, the operating system is Windows 7. We demonstrate the efficient IDS in terms of the detection rate and resource consumption by using an actual network environment from the Computer Engineering Department of KMUTT.

4.1 Offline Detection

4.1.1 Fuzzy GA with KDD99 dataset

In this experiment, we use the Fuzzy Genetic Algorithm with the KDD99 dataset. The table shows a number of each attack in the KDD99 dataset from two different versions. We can see that the number of records in the KDD99 dataset have different ratios for each attack type. For example, there are many *Smurf* and *Neptune* data records and few numbers of *Land* and *Pod*. The different numbers of the records are shown in Table 4.1.

In this experiment, we reduce the original 41 features in the KDD99 dataset into 8 features including *duration, src_bytes, num_failed_logins, root_shell, num_access_files, serror_rate, same_srv_rate and srv_count* [5].

Table 4.1 Number of records of each attack in KDD99 dataset (A-full version and B-10% version contain approximately 5,000,000 records and 200,000 records respectively)

Normal	Attack Type	#of records A	#of record B
Normal	Normal		97,278
Smurf	Dos	2,807,886	280,790
Neptune	Dos	1,072,017	107201
Teardrop	Dos	979	979
Back	Dos	2,203	2,203
Land	Dos	21	21
Pod	Dos	264	264
Ipsweep	Probe	12,481	1,247
Nmap	Probe	2,316	231
Portsweep	Probe	10,413	1,040
Satan	Probe	15,892	1,589
Total		**3,924,472**	**494,021**

4.1.1.1 One-rule

First, we train our fuzzy genetic algorithm with a training dataset. Table 2 shows a result of the fuzzy genetic algorithm in a training process. The dataset from this experiment was randomly collected from 10% file version for training including 160,147 records of attack data and 39,387 records of normal network data. The dataset was reduced to 8 features as obtained [ref] and its value was normalized to be in the range of 0-7. The rule was established and evaluated by the training dataset. There were two output classes, normal class and attack class. From Table 2, the detection rate is 98.72% with the low false positive of 0.13%. Table 3 shows the fuzzy rule of this experiment. There are four parameters (a, b, c and d) for each feature. Information of the fuzzy rules is described in section 3.2.3.

Table 4.2 Experimental result from Fuzzy Genetic Algorithm with KDD99 dataset

Name	#Attack	#Normal	TP(%)	TN(%)	FN(%)	FP(%)	DR(%)
KDD99 dataset	160147	39,387	99.87	98.45	1.55	0.13	98.72

Table 4.3 Detection rule of KDD99 dataset obtained from training process

Fuzzy parameter	Features							
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
A	5	2	1	1	4	5	4	0
B	5	5	1	4	4	5	4	0
C	7	7	1	6	4	7	7	1
D	7	6	3	6	6	6	7	1

Next, we investigate in detail for each attack name used in the training dataset. In Table 4.2-1, we extract each attack from the first experiment. This experiment uses the same dataset as shown in Table 2 (160,147 of attack records and 39,387 of normal records). From the table, we can see that most attack types have the high detection rate except Back and Land with the detection rate of only 16.56% and 15.58% respectively. There is the low false negative rate of most types of the attacks except Nmap having 16.13% of the false negative rate. However, there are a lot of attacks that have the high false positive rate including Back (FP: 85.33%), Pod (FP: 84.66%), Ipsweep (FP: 6.64%), Nmap (FP: 6.3%) and Portsweep (FP: 6.4%). In summary, in the KDD99 dataset, there are some attacks that the fuzzy rule cannot distinguish them from the normal network behavior including Back, Pod, Ipsweep, Nmap and Portsweep.

These two experiments show that the KDD99 dataset have the high detection rate. However, when investigating in detail, there is misclassification in the Back attacks and the Pod attacks. The proportion of the attacks in the KDD99 testing dataset is affected by the detection rate. In this case, the detection rate is biased by Smurf and Neptune which are the main part of the whole dataset and have the high detection rate. Therefore, we cannot use only the detection rate to evaluate the IDS.

Table 4.4-1 Experimental results of Fuzzy Genetic Algorithm with KDD99 dataset

Name	Class	# Attack	Evaluation Criteria				DR(%)	Data input
			TP(%)	TN(%)	FN(%)	FP(%)		
Back	DoS	893	14.67	100.00	0.00	85.33	16.56	back + normal
Land	DoS	4	99.61	100.00	0.00	0.39	99.61	land + normal
Pod	DoS	112	15.34	100.00	0.00	84.66	15.58	: :
Smurf	DoS	113,842	99.24	99.90	0.10	0.76	99.73	: :
Teardrop	DoS	371	98.81	100.00	0.00	1.19	98.83	: :
Neptune	DoS	43,375	99.85	99.66	0.34	0.15	99.75	: :
Ipsweep	Probe	479	93.36	100.00	0.00	6.64	93.44	: :
Nmap	Probe	93	93.70	83.87	16.13	6.30	93.67	nmap+normal
Portsweep	Probe	392	93.60	100.00	0.00	6.40	93.66	: :
Satan	Probe	586	99.26	96.25	3.75	0.74	99.22	: :
Total		**160,147**						

From Table 4.4-2, we investigated the reason that the Back attack and the Pod attack have low detection rates. We found that the 8 features that we selected from the original 41 features were affected by the detection rate as shown in Table 4.4-2. The result showed that when using the original 41 features with the fuzzy genetic algorithm, the detection increased in both back attack (96.07%) and pod attack (85.05%). While using the 8 features, the detection rate decreased for the back attack (15.47%) and the pod attack (12.19%).

Table 4.4-2 Experimental results comparing different numbers of features used for Back attack and Pod attack

#Feature	TP (%)	TN (%)	FN (%)	FP (%)	DR (%)
41	95.99	99.68	0.32	4.01	96.07
8	13.56	100.00	0.00	86.44	15.47

a. Back attack

#Feature	TP (%)	TN (%)	FN (%)	FP (%)	DR (%)
41	85.01	100.00	0.00	14.99	85.05
8	11.94	100.00	0.00	88.06	12.19

b. Pod attack

Note:
Pod attack is Ping of Death attack. The attacker sends a large size of a ping packet to a victim. The victim cannot handle the ping packet that is larger than the maximum IPv4 packet size causing a system clash.

Back attack [24] is Denial of Service against Apache web server where a client requests a URL containing many backslashes. The server will try to respond to these requests until it clashes. The features that are relevant to the back attack are the following [23]:

- Feature 5 (bytes sent from source to destination)
- Feature 6 (bytes sent from destination to source)
- Feature 10 (number of "hot" indicators)
- Feature 13 (number of "compromised'' conditions)
- Feature 32 (count connections having the same destination host)

4.1.1.2 Two-rule

In this experiment, we use the fuzzy genetic algorithm to classify the KDD99 dataset in a different way in order to find a new approach to increase the detection rate. We use two different datasets that are sampled from 10-percent file version of the KDD99 dataset including 199,534 records of the training dataset and 199,514 records of the testing set.

- The training dataset contains 158,597 of DoS records, 1,550 of Probe records and 39,387 of normal records.
- The testing dataset contains 158,503 of DoS records, 1,674 of Probe records and 39,337 of normal records.

There are three steps in this experiment.

- **DoS Training Process:** we use the training dataset to train the DoS rule by focusing on only the DoS attack in the dataset (Class A). So, we group Probe and Normal into the same class (Class B) in order to find the DoS rule. The rule is shown in Table 4.6.
- **Probe Training Process:** we use the training dataset to train the Probe rule by focusing on only the Probe attack (Class A). So, we group DoS and Normal into the same class (Class B) in order to find the Probe rule. The rule is shown in Table 4.7.
- **Testing Process:** In the testing dataset, we use both DoS and Probe rules obtained previously to classify the testing dataset. The testing using these two rules is described in section 3.3.2.

Table 4.5 Experimental results of Fuzzy Genetic Algorithm with KDD99 dataset

Dataset Name	#Class A (records)	#Class B (records)	DR(%)	FN(%)	FP(%)
DoS Training process	158,597	40,937	91.93	0.21	3.91
Probe Training process	1,550	197,984	95.31	98.18	0.00
Testing process	160,177	39,337	95.88	20.45	22.85

Table 4.6 DoS rule with KDD99 dataset obtained from DoS training process

Fuzzy parameter	Features							
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
A	4	0	2	1	4	4	4	0
B	4	1	2	5	4	6	5	1
C	4	2	2	2	4	7	7	2
D	4	3	2	6	6	6	6	2

Table 4.7 Probe rule with KDD99 dataset obtained from Probe training process

Fuzzy parameter	Features							
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
A	1	2	1	2	1	0	5	3
B	2	3	2	3	1	0	5	6
C	5	5	4	3	2	0	6	1
D	6	5	6	5	7	1	6	7

4.1.2 Fuzzy GA with Real-time Dataset

In this experiment, we use the Fuzzy Genetic Algorithm with the real-time dataset. The output has two classes which are attack and normal. We collect the real-time dataset in the actual network environment in our research laboratory. It is the online network data from the Computer Engineering Department at King Mongkut's University of Technology Thonburi (KMUTT). There are 17 types of attacks (4 types are DoS and 13 types are Probe). There are two sets of the data including:

Training dataset with 14,300 records including

- 6,300 records of the attacks, consisting of 300 records of each Probe name and 600 of each DoS record)
- 8,000 records of the normal data

Testing set with 26,500 records including

- 10,500 records of the attacks (500 records of each Probe name, 1000 of each DoS record)
- 16,000 records of the normal data

4.1.2.1 One-rule

We use the Fuzzy Genetic Algorithm with the real-time dataset (Training dataset) to find a rule for classifying the normal class and the attack class, as shown in Table 4.8. Then, we use the Testing dataset to evaluate performances of a rule. It shows that the Fuzzy Genetic Algorithm can classify the real-time dataset with the high detection rate (97.97%) and the low false alarm rate (the false negative rate is 3.39% and the false positive rate is 1.14%). Table 4.9 presents parameters of the rule obtained from the training process. There are 4 parameters (a, b, c and d) for each feature, and we have twelve features of the network data.

Table 4.8 Experimental results of Fuzzy Genetic Algorithm with real-time dataset

Name	#Attack	#Normal	Evaluation Criteria				DR(%)
			TP(%)	TN(%)	FN(%)	FP(%)	
Real-time dataset	10500	16000	98.86	96.61	3.39	1.14	97.97

Next, we investigate in detail each attack name used in the training dataset. In Table 4.8, we extract each attack from the first experiment. This experiment uses the same dataset as shown in Table 4.8 (10,500 of the attack records and 16,000 of the normal records). From the table, we can see that most of the attack types have the high detection rate but UDP-flood and IPscan have the low detection rates of 89.59% and 86.89% respectively. There are three types of the attacks that have the high false negative rate including Advances Port Scan (FN: 10.20%), Connectscan (FN: 16:20%) and IPscan (FN: 16.40%).

Moreover, there are two types of the attacks that have the high false positive rates which are UDP-flooded (FP: 11.06%) and IPscan (FP: 13.01%).

Table 4.9 Detection rule of real-time dataset obtained from training process

Fuzzy parameter	Features											
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
A	4	0	0	0	0	4	3	0	0	6	0	1
B	4	1	1	1	1	4	3	1	6	6	2	1
C	6	4	2	4	3	5	3	1	7	7	7	0
D	6	7	3	6	7	7	4	2	7	7	7	7

Table 4.10 Experimental results of Fuzzy Genetic Algorithm with real-time dataset

Dataset name	Type	# attack	Evaluation Criteria				DR (%)	Data input
			TP(%)	TN(%)	FN(%)	FP(%)		
HTTPflooded	DoS	1,000	99.64	96.50	3.50	0.36	99.46	httpflooded+normal
Jping	DoS	1,000	99.98	100.00	0.00	0.02	99.98	jping+normal
Smurf	DoS	1,000	99.98	100.00	0.00	0.02	99.98	: :
UDPflood	DoS	1,000	88.94	100.00	0.00	**11.06**	**89.59**	: :
Ackscan	Probe	500	99.97	100.00	0.00	0.03	99.97	ackscan+normal
AdvancePortscan	Probe	500	100.00	89.80	**10.20**	0.00	99.69	: :
Connectscan	Probe	500	99.86	83.80	**16.20**	0.14	99.38	: :
Finscan	Probe	500	97.42	100.00	0.00	2.58	97.5	: :
Hostscan	Probe	500	100.00	97.00	3.00	0.00	99.91	: :
IPscan	Probe	500	86.99	83.60	**16.40**	**13.01**	**86.89**	: :
Nullscan	Probe	500	99.01	96.00	4.00	0.99	98.92	: :
Portscan	Probe	500	99.98	100.00	0.00	0.02	99.98	: :
RCPscan	Probe	500	98.63	99.00	1.00	1.38	98.64	: :
Synscan	Probe	500	99.35	95.80	4.20	0.65	99.24	: :
UDPscan	Probe	500	97.52	100.00	0.00	2.48	97.59	: :
Winscan	Probe	500	99.98	100.00	0.00	0.02	99.98	: :
XmasTree	Probe	500	99.11	99.40	0.60	0.89	99.12	: :

4.1.2.2 Two-rule

In Table 4.11, we use two rules to classify the network dataset as we perform in section 4.1.1 but change the dataset to the real-time dataset. In this experiment, we use the training dataset to create rules and use the testing dataset to test in a testing process. The result is shown in Table 4.11. We can increase detection rate to 95.88% with low false positive rate. In Tables4.12 and 4.13 present Probe rule and DoS rule obtained from the training process, respectively.

Table 4.11 Detection rate of real-time dataset from using two rules of Fuzzy Genetic Algorithm

Dataset name	#C-Attack	#C-Normal	DR(%)	FN(%)	FP(%)
Training DoS process	2,400	11,900	91.93	30.69	1.48
Training Probe process	3,900	10,400	95.31	10.53	2.34
Testing process	10,500	16,000	95.88	6.28	2.70

#C-Attack (considered as attack) is number of records that is trained as attack.
#C-Normal (considered as normal) is number of record that is trained as normal.

Table 4.12 Probe rule of real-time dataset from training process

Fuzzy parameter	Features											
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
A	5	5	0	0	3	3	2	0	3	1	0	4
B	6	5	1	2	3	5	4	1	3	2	1	5
C	6	6	6	3	4	6	7	2	3	2	4	5
D	7	6	6	7	6	7	7	6	3	4	7	5

Table 4.13 DoS rule of real-time dataset from training process

Fuzzy parameter	Features											
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
A	6	0	4	2	0	4	5	2	1	4	2	0
B	6	1	4	2	1	4	5	2	6	4	2	1
C	6	1	4	2	1	4	5	2	7	5	2	7
D	6	2	4	2	1	3	5	2	7	6	2	7

4.1.3 Fuzzy GA with Unknown Detection

In this experiment, we consider detecting unknown attacks with three different algorithms (Decision Tree Algorithm, Naïve Bayes Algorithm and Fuzzy Genetic Algorithm$_{2\ rules}$). We use 26,500 data records from the real-time dataset including 16,000 records of the normal dataset and 10,500 records of the attack dataset. The number of the records in each attack type is shown in Table 14. In Table 14, seven test cases are used in this experiment which are T1, T2, ..., T7. For each test case, 13 attack types as well as the normal network data are provided in the training dataset, while the other 3 attack types are used as an unknown testing dataset for our Fuzzy Genetic Algorithm. For example, in the first test case, we use the training set which does not have Advances Port Scan, Ack Scan and Xmas Tree. Then we use these three types of the attacks for the testing dataset. Moreover, we test each type. The output classes are attack and normal.

Table 4.14 Seven test cases with unknown data types

No.	Data Type	Category	#Record	T1	T2	T3	T4	T5	T6	T7
1	Normal Activity	Normal	10,500							
2	Smurf	DoS	1,000			√				
3	UDP Flood	DoS	1,000				√		√	
4	HTTP Flood	DoS	1,000		√					√
5	Jping	DoS	1,000					√		
6	Port Scan	Probe	500			√				
7	Host Scan	Probe	500				√			
8	Connect	Probe	500			√				
9	SYN Stealt	Probe	500					√		
10	FIN Stealt	Probe	500					√	√	√
11	UDP Scan	Probe	500				√			
12	Null Scan	Probe	500		√					
13	IP Scan	Probe	500		√					
14	Window Scan	Probe	500							
15	RCP Scan	Probe	500						√	√
16	Adv Port Scan	Probe	500	√						
17	Xmas Tree	Probe	500	√						
18	ACK Scan	Probe	500	√						

Table 4.16 shows that **Decision Tree Algorithm** has the low detection (9.59%-26.17%). It has less than 1% of the false negative rate in the test cases 4, 5, 6 and 7, but has the high false positive rate in every test case (about 90%). **With Naïve Bayes Algorithm**, the high detection rates are obtained in the test cases 1, 2, 3, 5 and 7, with the detection rates of 91.11%, 90.35%, 93.11%, 90.90% and 93.17% respectively. However, this algorithm still has the high false alarm rate in every test case. **Fuzzy Genetic Algorithm** has the high detection rate. Its lowest detection rate is the test case 4 with 92.17%. It also has the low false positive rate of 0.25%-3.24% and the low false negative rate of 2.40%-61.15%. From this experiment, we can see that the Fuzzy Genetic Algorithm is the most robust algorithm for the unknown detection comparing with the Decision Tree algorithm and the Naïve Bayes algorithm.

4.1.4 Intrusion Detection with various Approaches

In this experiment, we compare various algorithms for intrusion detection with the KDD99 dataset and the real-time dataset. The algorithms include Decision Tree Algorithm, Naïve Bayes Algorithm and Fuzzy Genetic Algorithm$_2$ rules. There are two output classes which are normal and attack.

There are four datasets used in this experiment, where the two datasets from the KDD99* and the two datasets are collected on-line recently from an actual network environment.

- KDD99* Training dataset with 20,000 data records sampling from 10% file version.
- KDD99* Testing dataset with 50,000 data records sampling from 10% file version.
- Real-time Training dataset with 14,300 data records
- Real-time Testing dataset with 26,500 data records.

(*The KDD99 dataset was reduced into 8 features as shown in section 4.1.1)

Table 4.15 Experimental results with unknown attack type with real-time dataset

Test Case	Unknown Attacks	Decision Tree[7]						Naïve Bayes						Fuzzy Genetic					
		DR (%)		FP(%)		FN(%)		DR (%)		FP(%)		FN(%)		DR (%)		FP(%)		FN(%)	
1	Advance Port Scan	6.67	9.59	96.25	96.25	0.00	28.07	90.95	91.11	9.09	9.08	8.00	6.80	99.44	99.46	0.26	0.26	10.20	3.53
	Ack Scan	6.67		96.25		0.00		91.19		9.09		0.00		99.75		0.26		0.00	
	Xmas Tree	4.12		96.25		84.20		90.81		9.09		12.40		99.74		0.26		0.40	
2	HTTP Flood (DoS)	22.30	26.17	82.56	82.56	0.00	4.00	93.12	90.35	7.22	7.21	1.50	29.15	98.15	97.10	1.81	1.81	2.50	11.65
	IP Scan	19.47		82.56		15.60		93.00		7.22		0.00		97.15		1.81		36.20	
	Null Scan	19.93		82.56		0.40		92.59		7.22		13.60		98.08		1.81		5.40	
3	Smurf (DoS)	9.44	14.16	96.22	96.22	0.00	2.85	93.21	93.11	7.21	7.21	0.00	4.40	98.48	98.09	0.54	0.54	17.30	12.90
	Port Scan	6.70		96.22		0.00		93.01		7.21		0.00		99.48		0.54		0.00	
	Connect Scan	6.35		96.22		11.40		92.47		7.21		17.60		98.97		0.54		16.80	
4	UDP Flood	9.44	14.41	96.23	96.23	0.00	0.55	35.99	36.94	61.96	61.96	96.80	71.90	93.10	92.17	1.16	1.16	98.70	61.15
	Host Scan	6.62		96.23		2.20		39.82		61.96		3.00		98.79		1.16		2.80	
	UDP Scan	6.69		96.23		0.00		37.16		61.96		91.00		97.53		1.16		44.40	
5	Jping (DoS),	9.43	14.44	96.23	96.23	0.00	0.15	90.83	90.90	7.21	7.21	40.50	24.25	96.91	96.79	3.24	3.24	0.70	3.00
	Syn Scan,	6.67		96.23		0.60		92.90		7.21		3.40		96.54		3.24		10.60	
	Fin Scan	6.68		96.23		0.00		92.62		7.21		12.60		96.86		3.24		0.00	
6	UDP Flood,	9.44	14.47	96.23	96.23	0.00	0.00	55.07	57.21	43.06	43.05	74.90	40.70	93.59	93.51	0.64	0.64	98.70	53.30
	RCP Scan,	6.69		96.23		0.00		58.24		43.06		0.40		99.08		0.64		10.00	
	Fin Scan	6.69		96.23		0.00		57.87		43.06		12.60		99.21		0.64		5.80	
7	Http Flood,	9.44	14.47	96.23	96.23	0.00	0.00	93.15	93.17	7.21	7.21	1.10	3.80	97.16	97.11	2.96	2.96	1.00	2.40
	RCP Scan,	6.69		96.23		0.00		92.99		7.21		0.40		96.90		2.96		7.60	
	Fin Scan	6.69		96.23		0.00		92.62		7.21		12.60		97.13		2.96		0.00	

Table 4.16 Number of KDD99 data records in training dataset and testing dataset

KDD99 Dataset		
Attack Name	**Training**	**Testing**
Normal	3,919	9,851
Back	84	241
IPsweep	45	134
Land	5	3
Neptune	4,419	11,062
Nmap	12	22
Pod	13	29
Portsweep	32	88
Satan	67	137
Smurf	11,359	28,348
Teardrop	45	85
Total	**20,000**	**50,000**

Real-time Dataset		
Attack Name	**Training**	**Testing**
Normal	8,000	16,000
Smurf	600	1,000
UDP Flood	600	1,000
HTTP Flood	600	1,000
Jping	600	1,000
Port Scan	300	500
Host Scan	300	500
Connect	300	500
SYN Stealt	300	500
FIN Stealt	300	500
UDP Scan	300	500
Null Scan	300	500
Adv Port Scan	300	500
Xmas Tree	300	500
ACK Scan	300	500
Total	**14,300**	**26,500**

Table 4.17 Results from various detection algorithms

Dataset	Decision Tree	Naïve Bay	Fuzzy GA	
			2 rules	1 rule
KDD99 dataset	83.19	95.94	79.77	**99.77**
Real-time dataset	**99.71**	99.17	97.3	98.86

(a) True Positive rate

Dataset	Decision Tree	Naïve Bay	Fuzzy GA	
			2 rules	1 rule
KDD99 dataset	**98.84**	98.64	98.77	98.28
Real-time dataset	98.75	88.31	93.72	96.61

(b) True Negative rate

Dataset	Decision Tree	Naïve Bay	Fuzzy GA	
			2 rules	1 rule
KDD99 dataset	1.16	**1.36**	1.23	1.72
Real-time dataset	**1.25**	11.69	6.28	3.39

(c) False Negative rate

Dataset	Decision Tree	Naïve Bay	Fuzzy GA	
			2 rules	1 rule
KDD99 dataset	16.81	4.06	98.52	**0.23**
Real-time dataset	**0.29**	0.83	2.71	1.14

(d) False Positive

In Table 17, the highest true positive rate in the KDD99 dataset is the Fuzzy Genetic Algorithm with 1 rule (99.77%) and the Naïve Bayes Algorithm (95.94%). Moreover, every algorithm has high values of the true negative rate with the low false negative rate. In addition, the false positive rates from the different algorithms are different. The false positive rate in the Fuzzy Genetic Algorithm with 1 rule is as low as 0.23% while the Decision Tree Algorithm gives 16.81%.

From the table, we can see that the Decision Tree can classify the real-time dataset better than other algorithms (highest values of true positive (99.71%) and true negative (98.75%); lowest values of false negative (1.25%) and false positive (0.29%)). However, the Fuzzy Genetic Algorithm with 2 rules has the same rate of true positive as the Decision Tree Algorithm but has the lower false negative rate which is 93.72%.

4.2 Online Detection

In online IDS, we would like to test performances of IDS in term of CPU Consumption, Memory Consumption and Network Consumption.

4.2.1 Experimental Setting:

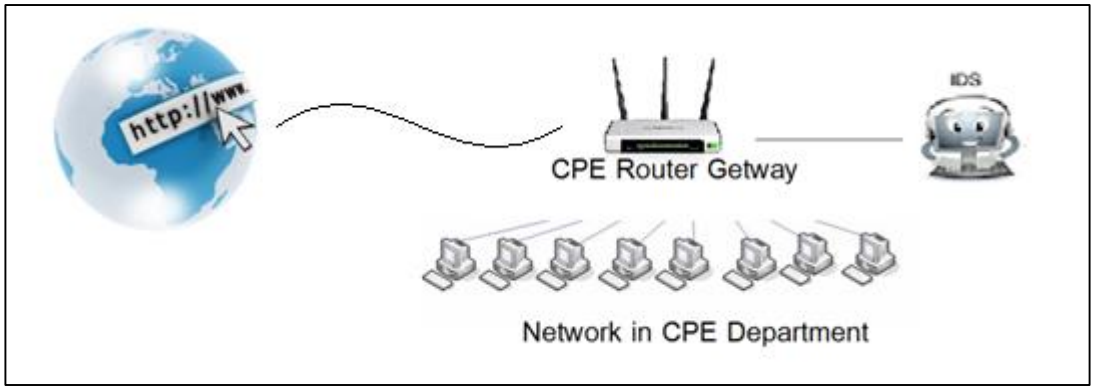

Figure 4.1 Real-time network environments

In this experiment, we monitor every packet in the CPE Department of KMUTT in both in and out of the network gate way. (The speed of the traffic is between 5-100 Mbit/sec). We connect our IDS to a gateway router using a mirror port during 12.30 pm. - 17.30 pm. on April 24, 2013. The IDS computer used Intel® Core™ i7-3770k CPU@ 3.5GHz 3.90 GHz RAM 8 GB Windows 7 Ultimate 67-bit with Network Interface card: Atheros AR8151 PCI-E Gigabit Ethernet Controller (NIDS 6.20).

In Table 4.18, there are 52,564,018 packets during the experimental time. It consists of 47,822,054 packets of TCP, 4,634,052 packets of UDP and 107912 of ICMP. In our real-time IDS, the system preprocessed these network data into 1,201,208 records. Moreover, our IDS classifies 1,519 records into the attack class. The CPU Consumption is between 7-14% while using only 2-2.5 GB of memory.

4.2.2 Experimental Result

Table 4.18 Experimental result from CPE network environment

TCP	UDP	ICMP	Total	Attack	Normal	CPU	Memory
47,822,054	4,634,052	107,912	1,201,208	1,519	1,199,689	7-14%	2-2.5 GB
Total: 52,564,018 packets							

Note:

TCP: Number of TCP packets
UDP: Number of UPD packets
ICMP: Number of ICMP packets
Total: Total number of records after preprocessing
Attack: Number of records that was detected as attack
Normal: Number of records that was detected as Normal
CPU: CPU Consumption
Memory: Memory Consumpt

CHAPTER 5 CONCLUSION

In this thesis, we proposed the fuzzy genetic algorithm to detect DoS and Probe attacks in both offline and online network environments. Our IDS can detect the attack in the real-time network environment. We began by evaluating accuracy of the fuzzy genetic algorithm in an offline dataset. The offline dataset includes a benchmark dataset (KDD99 dataset which was reduced to 8 features), and our real-time dataset. The result showed that the fuzzy genetic algorithm offered the high detection rate with the low false alarm rate on both datasets.

In addition, we explored in detail for each attack name in the dataset. We found that there were two attacks in the KDD99 dataset, namely Back and Pod, that were misclassified with the fuzzy genetic algorithm, while the algorithm could detect all attack types in our real-time dataset. From previous study, we have learned that the detection rate could be biased by the dataset. Therefore, we had to consider the false alarm rate and the proportion of the dataset. Next, we evaluated our fuzzy genetic algorithm by comparing with other algorithms considering both datasets. The accuracy of the fuzzy genetic algorithm is close to the results obtained from the decision tree algorithm.

We also compared our fuzzy genetic algorithm with other algorithms for detecting unknown attacks. We used only the real-time dataset to evaluate with seven test cases. Each of the training sets contained 13 attack types while the other 3 attack types were used as an unknown testing dataset. The results showed that the fuzzy genetic algorithm was the most robust algorithm for the unknown attack detection.

In the online network environment, we used our IDS to monitor the real-time traffic in the CPE Department of KMUTT (IDS PC spec: Intel® Core™ i7-3770k CPU@ 3.5GHz 3.90 GHz RAM 8 GB Windows 7 Ultimate 67-bit and Network Interface Card is Atheros AR8151 PCI-E Gigabit Ethernet Controller) in order to demonstrate performances of our IDS. The speed of the traffic was between 5-100 Mbit/sec. Our IDS consumed less than 14% of CPU resources while using only 2.5 GB of memory. In the real-time detection, our IDS could raise an alarm message within 2-3 seconds. This did not affect the PC performance when other applications were running.