Communications on Applied Electronics (CAE) – ISSN : 2394-4714
Foundation of Computer Science FCS, New York, USA
Volume 3– No.7, December 2015 – www.caeaccess.org

Design and Implementation of an Improved Denial of Service (DoS) Detection System using Association Rule

Olasehinde Olayemi
Computer science Department
Federal university of
Technology, Akure. Nigeria

Olayemi Olufunke
Computer science Department
Joseph Ayo Babalola (JABU)
University, Ikeji, Osun state,
Nigeria

Aliyu E. Olubunmi
Adekunle Ajasin Unoveristy,
Akungba Akoko, Ondo State,
Nigeria

ABSTRACT
The need for effective and efficient Denial of Service (DoS) Detection System cannot be overemphasized. This position is as a result of a serious threat to the availability of internet services that limit and block legitimate users access by exhausting victim server's resources or saturating stub networks access links to the internet services instead of subverting services. Hence the need for a supervised data learning techniques known as association rule mining which has the advantage of generating explainable rules was used to build a classifier for detecting some denial of service attacks, carry out a case study on International Knowledge Discovery and data Mining (KDD '99) tools, intrusion detection dataset for benchmarking the design of the intrusion detection systems. The average classification rate for unpruned rules is 63.16% while that of the pruned rules is 96.6%. The result revealed that pruned rule sets have better classification performance than the unpruned rule set.

Keywords
Intrusion detection system (IDS); Distributed Denial of Service (Ddos); Association rule; Knowledge Discovery Database (KDD); Rule Pruning

1. INTRODUCTION
Security of information is of utmost importance to organization striving to survive in a competitive marketplace. Network security has been an issue since computer networks became prevalent, most especially now that internet is changing the face of computing. As dependency on internet increases on daily basis for business transaction, so also cyber attacks by intruders who exploit flaws in internet architecture, internet protocol, operating systems and application software to carry out nefarious activities. Massive Internet worm outbreaks such as Slammer [8] Blaster [9] and Sasser [10] have shown that a large number of hosts that goes to millions are patched lazily or are operated by security-unaware users. Such hosts can be compromised within a short time to run arbitrary and potentially malicious attack code transported in a worm or virus or injected through installed backdoors.

Distributed denial of service (DDoS) use such poorly secured hosts as attack platform and cause degradation and interruption of Internet services, which result in major financial losses, especially if commercial servers are affected [6]). DDoS attacks pose a serious threat to the availability of Internet services. DDoS attacks consume resources associated with various network elements – e.g. Web servers, routers, firewalls, and hosts which impedes the efficient functioning and provisioning of services in accordance with their intended purpose[5]. One of the main reasons why DDoS is predominant on the Internet is as a result of the design of the Internet, current Internet design focuses on effectiveness in moving packets from the source to the destination and not on security of the packets. If one party in two-way communication (sender or receiver) misbehaves, it can do arbitrary damage to its peer. No one in the intermediate network will step in and stop it, because Internet is not designed to police traffic, consequences of this policy are the presence of IP spoofing and DDoS attacks [7].

A host of research works have been carried out on intrusion detection and prevention systems, majority of these researches are either rule-based or expert-system based. Their strengths depend largely on the ability of the security personnel that develops them. The former can only detect known attack types and the latter is prone to generation of false positive alarms. This work makes a contribution using an intelligence technique known as machine learning techniques, it automatically learn from data and extract useful pattern from data as a reference for normal/attack traffic behaviour profile from existing data for subsequent classification of network traffic.

2. RELATED WORKS
Brignoli [2] worked on DDoS detection based on traffic self-similarity estimation. This approach is a relatively new approach which is built on the notion that undisturbed network traffic displays fractal like properties. These fractals like properties are known to degrade in presence of abnormal traffic conditions like DDoS. Detection is possible by observing the changes in the level of self-similarity in the traffic flow at the target of the attack.

Kevin and George [3] identified the first publicly available DDoS attacks tool, Trinoo, was based on UDP flood attack and master-slave communications (forcing "innocent" computers participate in the attack by planting in them remote-control programs). In the following years, few more tools were published – TFN (tribe flood network), TFN2K, and Stacheldraht ("Barbed wire" in German).

William et al. [11], worked on detection of DoS attacks through the polling of Remote Monitoring (RMON) capable devices. The researchers developed a detection algorithm for simulated flood-based DoS attacks that achieved a high detection rate and low false alarm rate. The detection algorithm relies not only on the raw RMON variables but also on relationships between the variables to achieve this detection rate. The researcher also indicate how the introduction of RMON2 variables and an accurate network map can be used to improve DoS detection accuracy and reduce false alarms by identifying the sources of specific DoS-related traffic. The approach is less expensive than many commercially available solutions, requiring no special purpose hardware. It is more accurate than commonly used univariate statistical approaches and it is fast, requiring only the computation of packet variables ratios and processing by a feed-forward neural network.

Yeonhee and Youngseok [12] focused on a scalability issue of the anomaly detection and introduced a Hadoop-based DDoS detection scheme to detect multiple attacks from a huge volume of traffic. Different from other single host-based approaches trying to enhance memory efficiency or to customize process complexity, the method leverages Hadoop to solve the scalability issue by parallel data processing. From experiments, it showed that a simple counter-based DDoS attack detection method could be easily implemented in Hadoop and shows its performance gain of using multiple nodes in parallel. It is expected that a signature-based approach could be well suited with Hadoop.

3. ARCHITECTURE OF THE PROPOSED SYSTEM

The structure of the proposed architecture for real time detection of DDoS intrusion detection via association rule mining is shown in Figure 1, it is divided into two phases: learning and testing. The network sniffer processed the tcpdump binary into standard format putting into learning, during the learning phase, duplicate records as well as columns with same data were expunged from the record so as to reduce operational cost. A supervised data learning techniques known as association rule mining which has the advantage of generating explainable rules was then used to build a classifier for detecting some denial of service attacks

4. ASSOCIATION RULE MINING

Following the original definition by Agrawal and Sikrant (1994) the problem of association rule mining is defined as follow:

$$let I = \{i_1, i_2, \dots\dots\dots.., i_n\} \qquad (3.1)$$

be a set of *n* binary attributes called *items.*

$$let D = \{t_1, t_{2,\dots\dots\dots\dots}, t_n\} \qquad (3.2)$$

be a set of transactions called the *database*.

Each transaction in *D* has a unique transaction ID and contains a subset of the items in *I*. A *rule* is defined as an implication of the form

$$X \rightarrow Y \text{ where } X, Y \subseteq I \text{ and } X \cap Y = \emptyset . \qquad (3.3)$$

To select interesting rules from the set of all possible rules, constraints on various measures of significance and interest can be used. The best-known constraints are minimum thresholds on support and confidence.

The *support*supp(*X*) of an itemset*X* is defined as the proportion of transactions in the data set which contain the itemset.

$$\text{supp}(X) = \text{itemset}X \,/\, Total\ no\ of\ dataset \qquad (3.4)$$

The *confidence* of a rule is defined as

$$\text{conf}(X \Rightarrow Y) = \text{supp}(X \cup Y)/\text{supp}(X). \qquad (3.5)$$

Building of a classification model using association rule required a categorized training data sets. Patterns of denial of service were extracted using this learning technique on the training sets before being subjected to testing. For example, considering sampled network traffic data in Table 3.1, the network intrusion detection could be {Udp, Sf,} ➔ {Teardrop} meaning that if udp and sf are present in network traffic data, the traffic is most probably a teardrop Dos attack .

Table 3.1: Example of traffic data

transaction ID	Protocol	service	Flag	Attacks/Label
Traffic 1	Udp	Smtp	Sf	Teardrop
Traffic 2	Tcp	Http	Sf	Smurf
Traffic 3	Udp	Http	So	Neptume
Traffic 4	Icmp	Private	Sf	Teardrop
Traffic 5	Tcp	Smtp	So	Land

4.1 Association Rules Mining Classifier Dataset

The feasibility of this approach was demonstrated using the Knowledge Discovery and Data Mining 1999 (KDD '99) dataset. This dataset was acquired from nine weeks of raw Transfer Control Protocol dump data for a local area network(LAN) simulating a typical United State Air force LAN. The dataset is made up of 41 attributes: seven (7) out of these attributes are discrete and the remaining are continuous type.

5. EXPERIMENTAL SETUP AND RESULTS

The dataset used for building the Association classifier, consisted of 37,079 records, among which there are 99 (.267%) teardrop, 36,944 (99.64%) smurf, 20 (0.05%) pod, 15 (0.04%) Neptune and 1 (0.0027%) land connections. The testing data from the training dataset which were used to test the performance of the classifier is made up of 400 records out of which there are 98 (24.5%) teardrop, 266 (66.5%) smurf, 20 (5%) pod, 15 (3.75%) Neptune and 1 (0.025%) land. Another 300 records test data were extracted from an entirely different dataset from the training dataset, this dataset is made of 40 (13.3%) Pod, 107 (35.6%) smurf, 9 (3%) teardrop, 43(14.3%) Neptune (14.3%), 9 (3%) land, 33 (11%) apache2, 21(7%) normal, 25(8.3%) mailbomb and 8 (2.6%) snmpgetattack

The training and test dataset were pre-processed by:

i. Checking any irregularity in the data set (making sure all the data set has the same constant tuple), and
ii. Removal of attribute that has same constant value, a total of 21 columns (attributes) that has constant value were removed, leaving us with 21 columns including the classmark which are used for the training. Table 5.1 shows the 21 attributes used to build the classifier and attributes that were not used.

Table 5.1: Used and unused attributes for the classifier building

Attributes used to build the classifier	Attributes not used (Deleted attributes)
1,2,3,4,5,7,8,23,24,25,26,29,30, 32,33,34,35,36,37,38,39	6,9,10,11,12,13,14,15,16,17,18 19,20,21,22,27,28,31,40,41

5.1 Rule Generation

Association rules are usually required to satisfy a user-specified minimum support and a user-specified minimum confidence at the same time. Association rule generation is usually split up into two separate steps:

i. First, minimum support is applied to find all *frequent itemsets* in a database.

ii. Second, these frequent itemsets and the minimum confidence constraint are used to form rules.

Finding all frequent itemsets in a database is difficult since it involves searching all possible itemsets (item combinations). In order to reduce computational time for the rule generation, only the 21 attributes in table 5.1 were used for rules generation.

5.2 Rules Pruning to obtain Classifier Model

Association Generates lots of rule, most of the rules are irrelevant, hence need to prune the rule set and come up with relevant and important rules that will improve the classification process. The following steps were followed in other to prune and obtain relevant rules that will be used for the classification model:

i. All rules with confidence less than 80% were removed

ii. All duplicate and single attribute rules were removed

iii. All identical rules pointing to difference attacks were removed

5.3 Result of Implementation with Training Dataset

The initial rules generate and pruned rules were then used to classify the training set as well as testing data set. Tables 5.2, 5.3, 5.4 and 5.5 show the confusion matrix obtained Association rule mining with 20 attributes.

Table 5.2: Confusion matrix obtained from one and two attribute combination from training dataset for unpruned and pruned rules

A	N		S		P		T		L	
	UP	P	UP	P	UP	P	UP	P	UP	P
N(16)	12	14	0	0	0	0	0	0	4	2
S(264)	0	0	263	264	0	0	1	0	0	0
P(20)	0	0	19	0	0	20	0	0	1	0
T(99)	0	0	99	0	0	0	0	99	0	0
L(1)	0	0	0	0	0	0	0	0	1	1

A = Attacks, N = Neptune, S = Smurf, P = Pod, T = Teardrop, L = Land, UP= Unpruned, P= pruned

Table 5.3: Confusion matrix obtained from one, two and three attributes combination from training dataset for unpruned and pruned rules

A	N		S		P		T		L	
	UP	P	UP	P	UP	P	UP	P	UP	P
N(16)	12	15	0	0	0	0	0	0	4	1
S(264)	0	0	264	264	0	0	0	0	0	0
P(20)	0	0	19	0	1	20	0	0	0	0
T(99)	0	0	0	0	0	0	99	99	0	0
L(1)	0	0	0	0	0	0	0	0	1	1

A = Attacks, N = Neptune, S = Smurf, P = Pod, T = Teardrop, L = Land, UP= Unpruned, P= pruned

Table 5.4: Confusion matrix obtained from one, two, three and four attributes combination from training dataset for unpruned and pruned rules

A	N		S		P		T		L	
	UP	P	UP	P	UP	P	UP	P	UP	P
N(16)	14	16	0	0	0	0	0	0	2	0
S(264)	0	0	264	264	0	0	0	0	0	0
P(20)	0	0	19	0	1	20	0	0	0	0
T(99)	0	0	0	0	0	0	99	99	0	0
L(1)	0	0	0	0	0	0	0	0	1	1

A = Attacks, N = Neptune, S = Smurf, P = Pod, T = Teardrop, L = Land, UP= Unpruned, P= pruned

Table 5.5: Confusion matrix obtained from one, two, three, four and five attributes combination from training dataset for unpruned and pruned rules.

A	N		S		P		T		L	
	UP	P	UP	P	UP	P	UP	P	UP	P
N(16)	14	16	0	0	0	0	0	0	2	0
S(264)	0	0	264	264	0	0	0	0	0	0
P(20)	0	0	19	0	1	20	0	0	0	0
T(99)	0	0	0	0	0	0	99	99	0	0
L(1)	0	0	0	0	0	0	0	0	1	1

A = Attacks, N = Neptune, S = Smurf, P = Pod, T = Teardrop, L = Land, UP= Unpruned, P= pruned

The results in Tables 5.2, 5.3, 5.4 and 5.5 were obtained from classification of training data set with raw unpruned and pruned rule sets, from the tables, the degree of accuracy of classification of smurf attack with the unpruned rule sets ranges between 99.6% to 100% while Pod attacks classification could not be classify correctly by the classification model, about 95% pod attacks were classified as smurf attacks and the rest were classified as Pod. 98% of teardrop and 100% of land attacks were also correctly classified, 20% of Neptune attacks were classified correctly with unpruned rules generated from one, two and three attributes, unpruned rules generated from four and five network attributes improved the classification of Neptune attacks to 65%.

The pruned rule sets has a better classification performance than the unpruned rule sets, all the attacks except Neptune and pod recorded 100% correct classification, 90% of pod attacks were correctly classifier with single attributes rules and 100% correctly classified with two and more attributes rule sets. Neptune attacks recorded 93% classification with combination of 2 and 3 attributes rule set and 100% with combination of 2,3,4 and 5 attributes rules set.

Pruned rules set has an higher classification performance than the unpruned rules set as shown in Table 5.6

Communications on Applied Electronics (CAE) – ISSN : 2394-4714
Foundation of Computer Science FCS, New York, USA
Volume 3– No.7, December 2015 – www.caeaccess.org

Table 5.6: `Summary of the Percentages of Correctly Classified Attacks with unpruned and pruned rules set

	N (%)		S (%)		P (%)		T(%)		L (%)	
	UP	P	UP	P	UP	P	UP	P	UP	P
1, 2	75	87.5	99.6	100	0	100	0	100	100	100
1,2, 3	75	93.75	100	100	5	100	100	100	100	100
1,2,3, 4	87.5	100	100	100	5	100	100	100	100	100
1,2.3.4, 5	87.5	100	100	100	5	100	100	100	100	100

N = Neptune, S = Smurf, P = Pod, T = Teardrop, L = Land, UP= Unpruned, P= pruned

5.4 Result of Implementation with Test Dataset

The association rule classifier was tested with test data that did not belong to the same network with the training dataset, there are three (3) (Appache, Mail bomb, Snmpget attacks in the test data that were not present in the training set. Figures 5.7a, 5.7b, 5.8a, 5.8b, 5.9a, 5.9b, 5.10a and 5.10b shows the confusion matrix table obtained from the association rule classification of the test data.

Table 5.7a: confusion matrix obtained from one and two attribute combination from test dataset (unprune rules)

	P	S	T	N	L	N	A	M	S	U
P (40)	**0**	0	0	0	40	0	0	0	0	0
S (107)	0	**107**	0	0	0	0	0	0	0	0
T (9)	0	0	**7**	0	2	0	0	0	0	0
N(44)	0	0	0	**0**	44	0	0	0	0	0
L (9)	0	0	0	0	**9**	0	0	0	0	0
N (22)	0	3	0	3	0	**16**	0	0	0	0
A (34)	0	0	0	0	34	0	**0**	0	0	0
M(26)	0	0	0	0	26	0	0	**0**	0	0
S(8)	0	8	0		0	0	0	0	**0**	0

P = Pod, N = Neptune, S = Smurf, P = Pod, T = Teardrop, L = Land, A= Apache2, M= Mailbomb, S= Snmpget, U= Unknown

Table 5.7b: confusion matrix obtained from one and two attribute combination from test dataset (prune rules)

	P	S	T	N	L	N	A	M	S	U
P (40)	40	0	0	0	0	0	0	0	0	0
S (107)	0	**107**	0	0	0	0	0	0	0	0
T(9)	0	2	**7**	0	0	0	0	0	0	0
N (44)	0	0	0	44	0	0	0	0	0	0
L (9)	0	0	0	0	9	0	0	0	0	0
N (22)	0	0	0	2	0	**20**	0	0	0	0
A(34)	0	1	0	1	0	0	**0**	0	0	32
M (26)	0	10	0	0	0	0	0	**0**	0	16
S(8)	0	3	0		0	0	0	0	**0**	5

P = Pod, N = Neptune, S = Smurf, P = Pod, T = Teardrop, L = Land, A= Apache2, M= Mailbomb, S= Snmpget, U= Unknown

Table 5.8a: confusion matrix obtained from one, two and three attribute combination from test dataset (unpruned rules)

	P	S	T	N	L	N	A	M	S	U
P (40)	**0**	40	0	0	0	0	0	0	0	0
S (107)	0	**107**	0	0	0	0	0	0	0	0
T(9)	0	0	**9**	0	0	0	0	0	0	0
N (44)	0	0	0	**0**	44	0	0	0	0	0
L (9)	0	0	0	0	**9**	0	0	0	0	0
N(22)	0	2	0	3	0	**17**	0	0	0	0
A (34)	0	0	0	0	34	0	**0**	0	0	0
M(26)	0	0	0	0	26	0	0	**0**	0	0
S(8)	0	8	0		0	0	0	0	**0**	0

P = Pod, N = Neptune, S = Smurf, P = Pod, T = Teardrop, L = Land, A= Apache2, M= Mailbomb, S= Snmpget, U= Unknown

Table 5.8b: confusion matrix obtained from one, two and three attribute combination from test dataset (pruned rules)

	P	S	T	N	L	N	A	M	S	U
P (40)	**38**	2	0	0	0	0	0	0	0	0
S(107)	0	**107**	0	0	0	0	0	0	0	0
T(9)	0	0	**9**	0	0	0	0	0	0	0
N(44)	0	0	0	44	0	0	0	0	0	0
L(9)	0	0	0	0	**9**	0	0	0	0	0
N(22)	0	0	0	2	0	**20**	0	0	0	0
A(34)	0	1	0	1	0	0	**0**	0	0	32
M(26)	0	10	0	0	0	0	0	**0**	0	16
S(8)	0	3	0		0	0	0	0	**0**	5

P = Pod, N = Neptune, S = Smurf, P = Pod, T = Teardrop, L = Land, A= Apache2, M= Mailbomb, S= Snmpget, U= Unknown

Table 5.9a: confusion matrix obtained from one, two, three and four attribute combination from test dataset (unpruned rules)

	P	S	T	N	L	N	A	M	S	U
P (40)	0	40	0	0	0	0	0	0	0	0
S(107)	0	107	0	0	0	0	0	0	0	0
T(9)	0	0	9	0	0	0	0	0	0	0
N(44)	0	0	0	10	34	0	0	0	0	0
L (9)	0	0	0	0	9	0	0	0	0	0
N(22)	0	2	0	3	0	17	0	0	0	0
A(34)	0	0	0	0	34	0	0	0	0	0
M (26)	0	0	0	0	26	0	0	0	0	0
S(8)	0	0	8		0	0	0	0	0	0

P = Pod, N = Neptune, S = Smurf, T = Teardrop, L = Land, A= Apache2, M= Mailbomb, S= Snmpget, U= Unknown

Table 5.9b: confusion matrix obtained from one, two, three and four attribute combination from test dataset (pruned rules)

	P	S	T	N	L	N	A	M	S	U
P (40)	38	2	0	0	0	0	0	0	0	0
S(107)	0	107	0	0	0	0	0	0	0	0
T(9)	0	0	9	0	0	0	0	0	0	0
N(44)	0	0	0	44	0	0	0	0	0	0
L(9)	0	0	0	0	9	0	0	0	0	0
N(22)	0	0	0	2	0	20	0	0	0	0
A(34)	0	1	0	1	0	0	0	0	0	32
M(26)	0	10	0	0	0	0	0	0	0	16
S(8)	0	3	0		0	0	0	0	0	5

P = Pod, N = Neptune, S = Smurf, T = Teardrop, L = Land, A= Apache2, M= Mailbomb, S= Snmpget, U= Unknown

Table 5.10a: confusion matrix obtained from one, two, three, four and five attribute combination from test dataset (unpruned rules)

	P	S	T	N	L	N	A	M	S	U
P(40)	40	0	0	0	0	0	0	0	0	0
S (107)	0	107	0	0	0	0	0	0	0	0
T (9)	0	0	9	0	0	0	0	0	0	0
N (44)	0	0	0	10	34	0	0	0	0	0
L (9)	0	0	0	0	9	0	0	0	0	0
N (22)	0	2	0	3	0	17	0	0	0	0
A (34)	0	0	0	0	34	0	0	0	0	0
M (26)	0	0	0	0	26	0	0	0	0	0
S(8)	0	8	0		0	0	0	0	0	0

P = Pod, N = Neptune, S = Smurf, T = Teardrop, L = Land, A= Apache2, M= Mailbomb, S= Snmpget, U= Unknown

Table 5.10b: confusion matrix obtained from one, two, three, four and five attribute combination from test dataset (prune rules)

	P	S	T	N	L	N	A	M	S	U
P(40)	38	2	0	0	0	0	0	0	0	0
S (107)	0	107	0	0	0	0	0	0	0	0
T (9)	0	0	9	0	0	0	0	0	0	0
N (44)	0	0	0	14	0	0	0	0	0	0
L (9)	0	0	0	0	9	0	0	0	0	0
N (22)	0	0	0	2	0	20	0	0	0	0
A (34)	0	1	0	1	0	0	0	0	0	32
M (26)	0	10	0	0	0	0	0	0	0	16
S(8)	0	3	0		0	0	0	0	0	5

P = Pod, N = Neptune, S = Smurf, T = Teardrop, L = Land, A= Apache2, M= Mailbomb, S= Snmpget, U= Unknown

The results in tables 5.7a, 5.8a, 5.9a, 5.10a and 5.11a were obtained from classification of test data set with the unpruned rules. From the tables, Pod attacks were classified as Teardrop and smurf attacks. Smurf and Teardrop attacks were 100% and 88% classified correctly respectively, all Neptune attacks were wrongly classified as Land attacks, all Land attacks were correctly classified, between 84% of Normal traffic were classified correctly. 94% and 6% of Apache attack were classified as Land and Neptune attacks respectively. All Smnpget attacks were either classified as smurf or Teardrop attacks.

The results in tables 5.7b, 5.8b, 5.9b, 5.10b and 5.11b were obtained from classification of test data set with the raw pruned rules. 2, 3, 4,and 5 attributes. Pod, Smurf Teardrop, Neptune and Land attacks were classified correctly, while Apache, mailbomb and snmpget attacks were classified as either Unknown, Smurf or Teardrop attacks. Table 5.12 shows the summary of all the correctly classified attacks.

Table 5. 12: Summary Correctly Classified Attacks from the Test Dataset

Table	P (%)	S (%)	T (%)	N (%)	L (%)	N (%)	A (%)	M (%)	S (%)
4.7b	0	96	0	22.5	77.7	91	0	0	0
4.8b	0	100	88	100	100	91	0	0	0
4.9	95	100	100	100	100	91	0	0	0
4.10b	95	100	100	100	100	91	0	0	0
4.11b	95	100	100	100	100	91	0	0	0

P = Pod, N = Neptune, S = Smurf, T = Teardrop, L = Land, A= Apache2, M= Mailbomb, S= Snmpget, U= Unknown

All the attacks present in the test dataset which were not used for training of the association rule classifier were classified as other attacks in test data and unknown attacks with unpruned rule, the pruned rule respectively. Tables 5.13 and 5.14 shows how they were classified.

Table 5.13: Classification of Attacks not Present in the Test Data (unpruned Rule).

	Pod	**Smurf**	**Teardrop**	**Neptune**	**Land**
Apache			1(2.9%)	1(2.9%)	34((100%)
Mailbomb					26(100%)
Snmpget			8(100%)		

Table 5.14: Classification of Attacks not Present in the Test Data (prune Rule).

	Pod	Smurf	Tear drop	Neptune	Land	Unknown
Apache		2(6%)				32(94.1%)
mailbomb		10(38.5%)				16(61.5%)
Snmpget		3(62.5%)				5(62.5%)

6. CONCLUSION

In this paper, we propose a technique, an association rule based algorithm, developed for mining known-patterns. It provides methods of improving intrusion detection systems to ascertain the degree of accuracy using two types of datasets. It also reduces computational time for the rule generation by excluding constant value in the KDD dataset used for rule generation. It finally prunes the rule set and come up with relevant and important rules that will improve the classification process.

Communications on Applied Electronics (CAE) – ISSN : 2394-4714
Foundation of Computer Science FCS, New York, USA
Volume 3– No.7, December 2015 – www.caeaccess.org

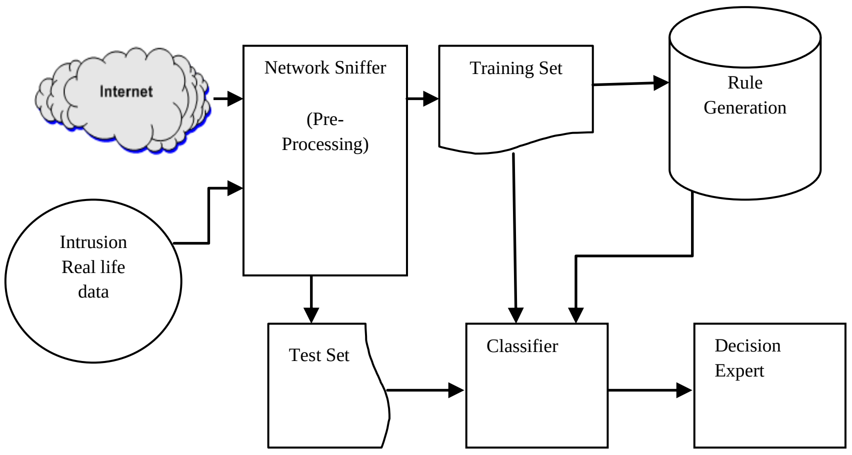

Fig. 3.1 Architecture of Association Rule Intrusion Detection System

7. REFERENCES

[1] Agrawal R. and Sikrant R., (1994), "Fast Algorithms for mining association rules", In *Proceedings of the 20th VLDB Conference,* Santiago, Chile.

[2] Brignoli, D (2008) "DDoS detection based on traffic self-similarity" Publisher: University of Canterbury. Computer Science and Software Engineering

[3] *Kevin H. and George W.* (2001) "A White Paper on Trends in Denial of Service Attack Technology"

[4] CERT Coordination Center,https://resources.sei.cmu.edu/asset_files/.../**2001_**019_001_52491.pd Accessed 2014

[5] Kihong P. and Heejo L.(2000), " Proactive Approach to Distributed DoS Attack and Prevention using Route-Based Packet Filtering" Network Systems Lab, Department of Computer Sciences, Purdue University,West Lafayette, IN 47907, December 3, 2000

[6] Larry W. (2007) "Toward The Development of a Defensive Cyber Damage and Mission Impact Methodology" Thesis, Presented to the Faculty Department of Systems Engineering and Management Graduate School of Engineering and Management, Air Force Institute of Technology march 2007

[7] MirkovicJ.and Peter R.(2004) "A taxonomy of DDos attack and DDos defense mechanisms". ACM SIGCOMMComputer Communication Review, Volume 34 Issue 2, April 2004

[8] Moore D., (2003) Inside the slammer worm - Security & Privacy Magazine, IEEE25 January 2003

[9] Symantec Security Response, (2003) http://www.symantec.com/security_response/writeup.jsp ?docid=2003-081113-0229-99&tabid=2. Accessed 2014.

[10] Symantec Security Response, (2004) http://www.symantec.com/security_response/writeup.jsp ?docid=2004-050116-1831-99&tabid=2. Accessed 2014

[11] William W. Streilein, David J. Fried, Robert K. Cunningham (2005) "Detecting Flood-based Denial-of-Service Attacks with SNMP/RMON" MIT Lincoln Laboratory

[12] Yeonhee Lee, Youngseok Lee (2011)"Detecting DDoS Attacks with Hadoop