Available online at www.elixirpublishers.com (Elixir International Journal)

Network Engineering

Elixir Network Engg. 56 (2013) 13886-13893

ISSN: 2229-712X

RIDTDM:A New Algorithm to Reduce Intrusion Detection Time Based on Data Mining

Samaneh S. Alavifar[1], Akbar Farhoodi Nejad[1] and Seyed Amirhossein Alavifar[2]

[1] Department of Computer Engineering and Information Technology, Payame Noor University,Tehran, Iran.

[2] Department of Risk Management, Bstech lim co.Tehran, Iran.

ARTICLE INFO

Article history:
Received: 31 January 2013;
Received in revised form:
21 March 2013;
Accepted: 28 March 2013;

Keywords
Intrusion detection,
Data mining,
Apriori algorithm,
Distributed systems,
Misuse detection,
Anomaly detection.

ABSTRACT

Internet connections involve various security threats to computer networks and systems. Complexity of the threats and intrusions are ever-increasing so that development of flexible methods for ensuring security of computer systems has become a serious challenge ahead of computer experts. As an advantageous solution for development of intrusion detection systems, data mining is an effective technology to detect attempts for intrusions. The present paper is a try to optimize Apriori algorithm. To this end, data are divided equally between two systems and consequently considerable improvement in abnormal pattern is obtained. This study enables IDSs to detect intrusions faster and facilitates proper reactions.

Introduction

Necessity of maintaining information security is receiving more attention along with growth of appeal for on-line services for shopping, banking and other business transaction. Centerpiece principles of information security including confidentiality, integrity and availability (CIA) guarantee authenticated and authorized access to information. Nevertheless, the principles are vulnerable to unauthorized access in large software systems. These vulnerabilities might be spotted and used by unauthorized users, when they want to access the systems. Different defensive layers are designed to avoid security failures. Some of measures in the networks are proxies, filters, firewalls, and intrusion detection systems.

Intrusion detection systems (IDS) are required due to the fact that if the intrusion passes through firewalls and anti-virus and other security criteria and entries to the system can recognize and conduct a solution to deal with it.

By definition, intrusion means: “any set of actions that attempt to compromise the integrity, confidentiality or availability of a resource"[1].

Researchers have introduced many techniques for designing intrusion detection systems in recent years [2, 3, and 4]. However, several issues have been identified regarding currently used intrusion detection systems. Statistical anomaly detection [5, 6], detection based on neural network [7, 8, and 9], detection based on data mining [2, 9, 10, and 11] and etc are some classification of currently used anomaly detection techniques. Forest et al. was used immune method to protecting computer for the first time [12]. Above all, characteristics of normal and abnormal activities must be learned by IDS [13]. Afterward, it detects traffics deviating from normal activities. Detecting anomaly is the matter of determining possibility of being flagged by established normal passage patterns [14].Basis of anomaly detection techniques are the assumed deviation of intrusive behavior from normal system procedure [15]. Ability to detect attacks whether they are detected for the first time or not is one advantage of anomaly detection systems. However, its inability to detect attacks by insiders is a disadvantage to name [16].

A strategy for effective combination of strategies of data mining and expert systems was employed for designing IDS by Sodiya [17]. In addition, combination of multiple techniques for designing IDS is a recent development, which demands further investigation.

For better coverage and more effective detection, Maumer [2] discussed improvement of intrusion system through combining data mining in WEKA environment.

Theoretical Background

Intrusion Detection Systems

A system which recognizes and handles the unauthorized usage of computer and network resources is defined as intrusion detection system (IDS). It comprise system intrusion from external users and internal user’s that non-authorized behavior. This technology was introduced to guarantee security system's capability to detect and alarm non-authorized and abnormal incidents from normal traffic [18].

IDSs designed as software hardware packages that by reviewing the packages, makes it possible to reduce the intrusion to a minimum amount. Generally speaking three functions of IDS Include: monitoring and evaluation, detection, and response.

IDSs can determine unauthorized activities almost in real-time and then Users have a chance to carry out appropriate measure for minimizing impacts of the intrusion. Also they may be host-based (HIDS) or network-based (NIDS) [19]. The NIDS is able to monitor whole computer networks through observing and analyzing network traffic.

Tele:
E-mail addresses: Sa.alavifar@gmail.com

Methods of used in Intrusion detection systems are divided into two categories: [4]:

- Anomaly detection (method to identify abnormal behavior)
- Misuse detection (Signature based detection methods)
- ***Anomaly detection method:***

In this method, a model of normal behavior is created and abnormality that may not be consistent of its pattern may be an intrusion. Designers use neural networks, Machine learning techniques and even biological safety systems to create normal behaviors patterns. Also to detect unusual behavior, normal manners should be identified and create specific patterns for them. Behaviors that follow these patterns are normal and the events that have statistical deviation from these patterns are identified as abnormal behavior.

The bad news is that they are disposed to false positives caused by unprecedented and unauthorized traffic as building a model representative of all possible normal traffic is not easy [20].

- ***misuse detection method:***

In this technique, the pre-made intrusion signatures are maintained as rule [21, 22]. So that each pattern contains of different types of intrusions and when happened one of them, the occurrence of intrusion will be announced. In these ways, detector has a database from signatures and attack patterns, and attempts by examining the network traffic find similar pattern to what is stored in its database. Misuse detection methods are only able to detect the known intrusions, and in the case of new attacks at the network, they cannot identify them. The benefit of this approach is that the model is accurate in detecting intrusions that their patterns are given identical to the system.

Although misuse detection method that using rules, in addition to permanent modification, is not enough to deal with new kinds of attacks. Another negative point is easy access to toolkits on the web that permits invaders to design new malware and employ malware polymorphism. So many speculations have been raised that signature based antivirus and IDSs need to be updated regularly [23].

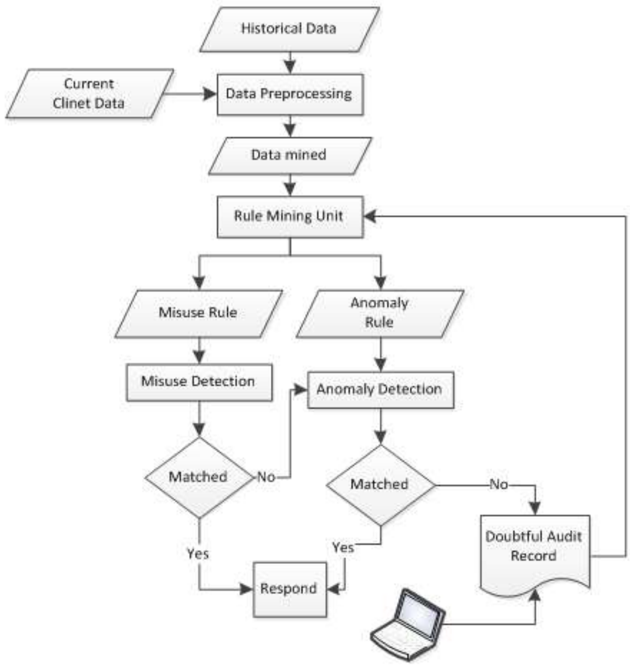

Figure 1- Intrusion detection framework based on Data mining

In addition, the volume of data is huge and there are more situations that network managers can't able to investigate all reports of activity in the network, so they are needed to summarize the reports and looked for the expert for suspect case. On the other hand, misuse-based systems usually fail to spot new or zero-day invasions [20]. A general solution for the both systems was introduced by Jing Xu et al; they use anomaly detection, which identifies patterns not conforming to a historic norm. [24].

According to what was mentioned, the researchers have used data mining techniques in intrusion detection systems to derive new patterns and suspected to intrusion, automatically from the available data [25, 26].

Further, Maumer et al. had designed Intrusion detection system based on data mining techniques that contains two engine: anomaly detection engine and missus detection engine, that they were working together as serial to control users by collecting activities in the first time and judge based on data mining that what behavior is right and what is destructive behavior. Main parts of mentioned framework are data collecting and preprocessing module, association rule mining module, intrusion detection analyzing module, as illustrate in Figure 1.

The key problem with this explanation is that proposed algorithm was tested with increasing minimum support and minimum confidence, so critical data may be lost [2].

Data Mining Technology

In order to realize that how data mining can be used in intrusion detection systems, data mining must be explained first. By definition, Data mining is a set of techniques by which they can discover hidden knowledge in data, ie dynamic model, through data obtained. So far, many algorithms have been introduced at the International Conference on Data Mining (ICDM) attempted to find algorithms that are well-known and powerful algorithms in which C4.5, k-Means, SVM, Apriori - , EM, Page Rank, AdaBoost, kNN, Naive Bayes, and CART [27]. Data mining algorithms were selected as the top ones. Apriori algorithm is one such algorithm that this paper is a kind of developing on Apriori algorithm.

The Basic Apriori Algorithm

One of major scopes of data mining is association rule mining [28]. It is aimed to find association relation among a group of items. Mining association rule covers two sub-problems including detect all frequent item-groups founded more commonly that a minimum support threshold; and utilization of the frequent item-groups for developing association rules [29]. The first sub-problem mentioned, has a major role in association rules mining. When a set with higher frequency is detected in the database, strong association rules may be developed directly. The core algorithm is described in what follows [30]. On the other hand Apriori algorithm is known as two sub-processes including Apriori-gen () and subset (). Apriori-gen () develops candidate and uses Apriori property (all non-empty subsets of frequent items groups should also be frequent) for omitting the candidates of the non-frequent sub groups. It is essential for Apriori property that all non-empty sub groups of frequent items be frequent. To obtain the frequent item groups a two-step process in employed: join and prune actions.

a) Joining step

A group of candidate (k-item sets) is generated by gathering L_{k-1}. The group of candidate is marked by C_k.

b) Pruning step

There are frequent and non-frequent Items in C_k, though all the frequent k-item sets are member in C_k. To count the candidate in C_k the whole database is scanned [31].

Proposed algorithm

There are limitations on traditional Intrusion detection system such as: poor adaptability, lack of extensibility, high modeling cost slow updating speed, inability to detect novel attacks, etc.

On the other hand, when we have amount of data in network, Apriori algorithm demands massive I/O operation and CPU resource for data processing. For large volume of data there are still massive resources even with Apriori algorithm for reducing frequency item sets. So in that situation we are confronting with bottleneck and single point of failure.

In addition, detection rate of abnormal behaviors in IDSs is very important that with increasing network traffics and dimensions, Becomes more palpable and low referred to this issue. Our object in this paper is design a smart intrusion detection system based on data mining that divide processing between two or more systems that result in preventing bottleneck in the system and increase the speed of the intrusion detection system.

In this system, historical data and current data are collected and after preprocessing, they are appropriately divided between the two systems, each system create Rule sets separately and thus analyze the user's activities on their first time entrance, so IDS can more quickly determine what behavior normal and what is aggressive and destructive behavior. Then response with comprehensible form will be send to the system Manager to adopt an appropriate decision. In this RIDTDM Minimum support and Min confidence is given to the system. Two separate systems have been simulated as virtual machines on a system. An overview of the proposed system is shown in Figure 2.

As figure illustrates, data mining are performed by two parallel systems in the intrusion detection system. Proposed structure consists of three main parts:

1. Data collection and preprocessing module network
2. Balanced data distribution module between two systems in order to obtain rules
3. Intrusion detection analysis module

In the first module at first, historical data and current data from network users will be collected and preprocessed to acceptable for IDS's input data.

Then, in the second module, determined that items of database are including what 1-Itemsets? After extracting them, calculate their number and divided by two.

So half of them (1 - Itemset) are sent to the first distributed system and the second half went to the second system. The first system, gets all 1-Itemset compounds from the main Database and the second system also repeats this scenario, after that each systems select their composition from database, and removes them from that, after this stage, each system separately extract their rule sets. Finally, what remains in the database are the items that are common between two systems and must be extract their rule sets.

In RIDTDM, the optimum state is when that the divided data between distributed systems haven't any common items. In this case the after dividing the data between the two systems, no data remains into the database to be processed, so the algorithm will be completed.

The medium state is that, data are divided into three groups' equivalent, and one-third of data are shared to the first distributed system and one-third of them went to the second system and the rest are common in two systems.

The worst case of RIDTDM is when all data are common in distributed systems, so in that case, the data cannot be divided into discrete systems and extract their rule sets separately. Thus in order to avoid increasing computational time of algorithm, we have set the condition that selects 10 items randomly from database and if 8 of them are common in tow systems, the algorithm will jump to basic Apriori algorithm.

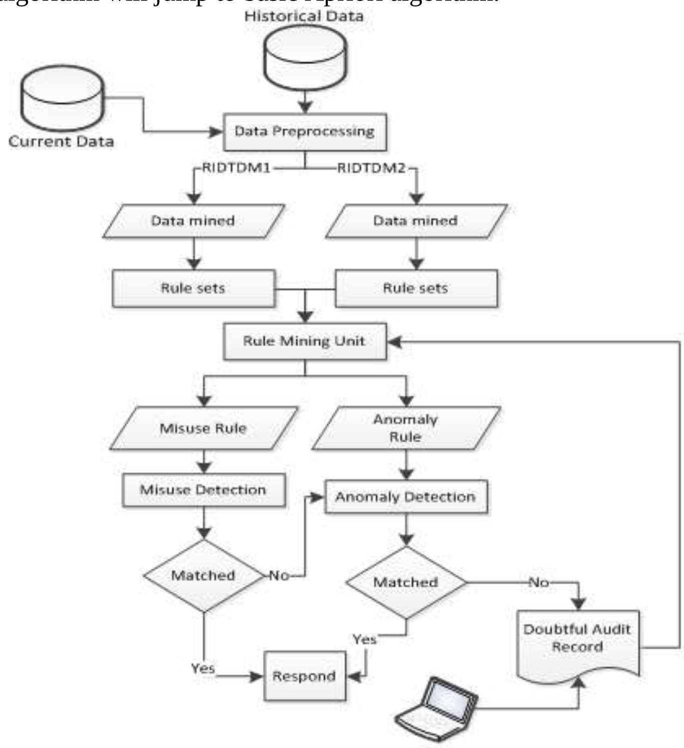

Figure 2 - Proposed structure for IDS based on Data mining

The Flow Chart Of Proposed Algorithm

RIDTDM helps us to increase time of extracting rule sets from data base, so that's flowchart and algorithm is as follows:

1. Find frequent 1 – Itemsets.
2. 1-Itemsets are sorted and their number of equal to n.
3. n should be divided by 2, the first half of 1-itemset will be send to first system and the second half will be send to second system.
4. Select 10 items randomly from the database, if more than 8 selected elements contain 1-itemsets of both of two systems, we are in the worth case and the basic Apriori algorithm runs and the algorithm finishes. Otherwise, the next step is executed.
5. In this step, each system separately, perform the following steps and extract Apriori rule sets:

a. C_1 contains single element sets that come from previous step.
b. Prune infrequent candidates and determine the frequent ones.
c. While there are frequent item sets, create item sets with one item more and determine the frequent more.
d. Generate association rules.
e. Remove related Itemsets from main data base.

6. In This step main database is analyze, if data has remained there, it can be concluded that the items remained that was shared in the first and second systems,

so for remaining items for each of the database, do the below actions:

a. Whether this item has been added to the shared memory already? If yes, check the next Item; else go to the next step

b. Determined that this item is made up from what 1-itemsets and save them in the array is called strtemp

c. All components of this array should be computed and save it on a list name lst_ShMemory.

d. r = 0

e. if r less than the number of lst_ShMemor, go to the step f, else check the next Item

f. Whether this element of the array is existed in the lst_ShMemory [r] the shared memory or not? If yes, go to the step g, else check the next Item

g. Does this element exist in the first system? If yes, go to the next step else check second system (step i)

h. Take the element from the first system and then put it in the shared memory then increase r

i. If element exists in the second system? If yes, go to the step h; else go to step k

j. Take the element from the second system and then put it in the shared memory then increase r

k. remove item from main data base and put it in the shared memory.

l. r = r +1

m. Is there any items in the main database? If Yes, go to the next step n, else exit (step 7).

n. check the next Item in the database

o. Extract the rule sets from shared memory

7 - End.

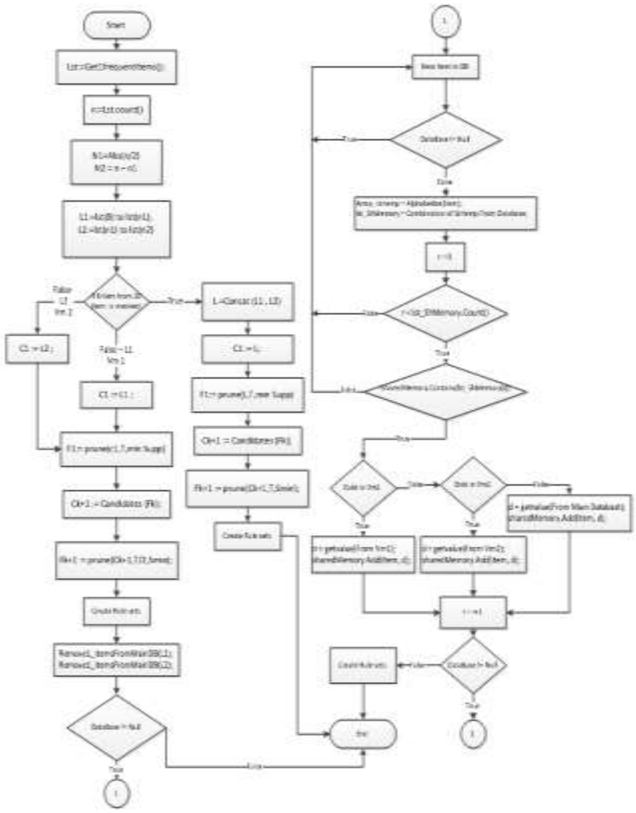

Figure 3 – The Flow Chart of Proposed Algorithm

It is worth mentioning that if we have a number of common data, some of the mining rules are repeated, however decreases the time.

Samples tested showed that the response of the RIDTDM and a priori Basic rules are the same.

Result and Discussion

The simulation of this algorithm was done in Visual C #. Net 2008 and system with specifications Intel Core 2 Duo 2.Ghz. (T6400) and 2 GB DDR2 Ram and 320GB SATA HDD, windows 7 32 bit.

The Number of each item in the database has been created by the randomized algorithm to test the value of 11,000 up to one million variables. The value of Minsupport and Minimum confidence could be changed dynamically in the simulated program. We have assumed in all states Minimum support = 1% and Minimum confidence = 2%.

Optimum State:

As noted above, the optimum situation for RIDTDM is that when the number of 1-Itemsets is divided between two distributed systems and they can select their Items from the main database and after this deviation there aren't any shared items in data base, so each systems separately extracts their rule sets. By this method execution time dividing between two systems and detection rate of a malicious pattern will be reduced.

The table1 shows the production data and combination of them:

Both of algorithms (basic Apriori and RIDTDM) run with Minsupport = 1 % , Min confidence = 2%.

The results obtained from execution both of algorithms can be compared in table 2.

As shown in Figure 4, when the number of data increases, we will save time in RIDTDM. So division of tasks and parallelism of execution is useful.

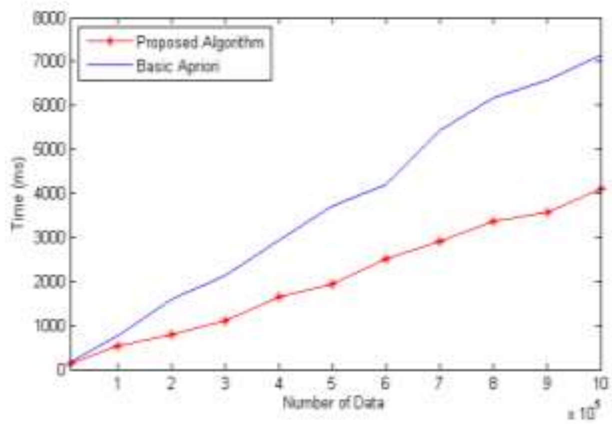

Figure 4- comparing execution time in the optimum state

Middle State

In the middle state, related items from both of systems and Items that common between them would be equal. Table 3 presents the data sets that generated for the middle state.

As table illustrate, algorithm was simulated with small number of data, from 1000 to 23000. So we found that before 7000 data Items, the results of the implementation does not better than Apriori algorithm and after that, give us better response, so we found that the threshold is required to run the

RIDTDM. This threshold depends on the number of shared data Items, combination of them, the number of items and the number of characters constituting the first and second system. Of course with a fixed amount of combinations and so on and increasing number of data items, RIDTDM gives better time execution.

Table 4, presents the results obtained from simulation of RIDTDM and basic Apriori algorithm

Table 4 – results respond tims of execution RIDTDM for middel state

ID		Number of records	users respond time in Basic Apriori (ms)	users respond time in RIDTDM (ms)
1	R1	1000	94.0695	125.1165
2	R2	3000	109.2748	133.5430
3	R3	5000	138.1604	153.6126
4	R4	7000	136.3711	199.0339
5	R5	9000	165.6745	151.0636
6	R6	11000	223.6346	192.5958
7	R7	13000	204.4048	177.8535
8	R8	15000	230.7569	185.0712
9	R9	17000	304.6085	209.6073
10	R10	19000	246.6253	214.6893
11	R11	21000	275.0392	227.7095
12	R12	23000	324.9578	278.0352

The following table shows data set for simulation:

As shown in Figure 5 after 5000 items, RIDTDM works better than original algorithm and extracts rule sets in less time than basic Apriori algorithm.

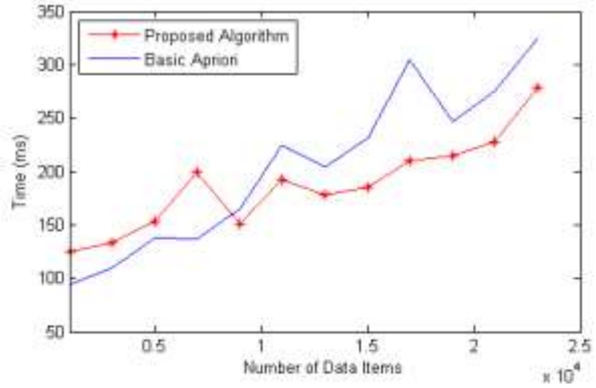

Figure 1 - comparing execution time in middle state

After that, RIDTDM simulated with more data, as table 5 illustrates:

As can be seen from the table bellow, the results obtained from the executing basic Apriori algorithm and RIDTDM:

From the graph bellow we can see that, the execution time of RIDTDM significantly better than the basic Apriori algorithm.

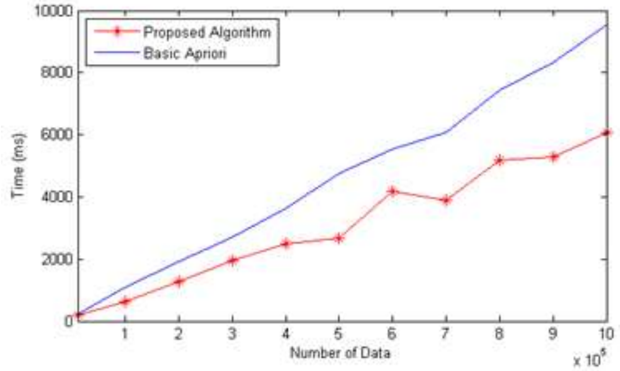

Figure 2 - comparing execution time in middle state with larg number of data

The worst state:

In the worst state, all of data are shared in distributed systems. Stochastic data for simulation algorithms are shown in table 7:

It can be seen from the data in Table 8 that the results are significantly closer together.

Table 8 – results respond tims of execution RIDTDM for middel state

ID		Number of records	users respond time in Basic Apriori (ms)	users respond time in RIDTDM (ms)
1	T01	100,000	995.7993	970.1798
2	T02	200,000	2327.4702	2094.282
3	T03	300,000	3085.3526	2997.0497
4	T04	400,000	3853.2398	3651.8143
5	T05	500,000	4842.6089	4759.401
6	T06	600,000	6537.7391	6419.5643
7	T07	700,000	7480.174	7391.4784
8	T08	800,000	8167.8546	7822.3957
9	T09	900,000	8451.8927	8255.2398
10	T10	1000,000	10351.6453	9985.6283

Figure 7 shows the results obtained from the execution of data set in basic Apriori and RIDTDM.

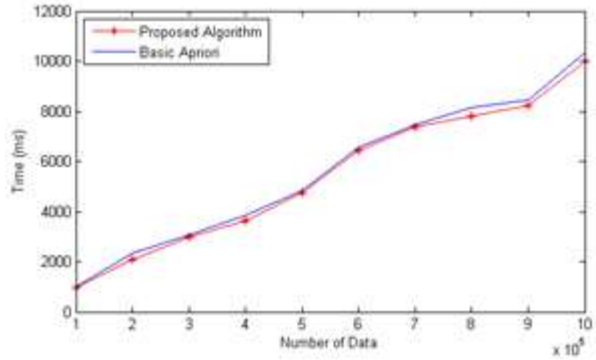

Figure 7- comparing execution time in worst state

The Objective Function:

Analyzing obtained graphs, help us to predict objective function of RIDTDM that depends on the values are listed in the following table:

The objective function has direct correlation with the number of items in the database (x), as a result when they are increasing extracting rule sets get more time. Also it has direct correlated with Combination of Items sets it means if the item sets are single-member or multi-member, execution time is low or high.

We can guess that the objective function has inverse correlation with minimum confidence, minimum support. Because if they are decreasing then we should spending more time to extract rule sets. Also the formula has inverse relation with systems processing power. Obviously if the processing power systems are low, producing rule sets needed much time. Another factor that is effective in the objective function is the number of the distributed systems, in this paper considered two distributed systems and processing time is divided by two systems.

Considering the facts and evaluate the results obtained from the simulation that are plotted in the following chart.

Table 1 - data set for optimum state

Item	The Number Of Items										
ab	1092	7888	1885	7556	27024	48734	94861	4352	108750	55409	136397
abc	1095	3920	16620	38479	40840	51670	65425	86830	35718	38048	60687
ac	813	13157	29584	7278	28542	47895	10957	88839	106996	35155	88948
bc	578	8819	30471	45340	18977	53179	37385	41294	74945	10995	20675
a	1323	11357	295	21047	39018	50159	966	91406	1916	135556	71247
b	1820	1157	22382	28140	38006	11910	12354	91517	10205	93460	79579
d	1577	10972	18726	19007	46204	50998	84338	34045	32369	136569	45815
f	1199	8739	12262	15948	23076	60479	81539	47656	47680	22713	95790
ef	544	13414	3102	33796	32540	32544	15899	37946	88020	52805	81362
df	523	3052	16547	39128	46255	18788	78639	25223	38110	138161	116020
def	184	7678	33795	42008	33456	24408	96257	102678	98704	106044	58456
de	252	9847	14331	2273	26062	49236	21380	48214	156587	75085	145024
Sum	11000	100000	200000	300000	400000	500000	600000	700000	800000	900000	1000000

Table 2 – results respond tims of execution RIDTDM for optimum state

ID		Number of records	users respond time in Basic Apriori (ms)	users respond time in RIDTDM (ms)
1	T01	11000	158.5775	122.3838
2	T02	100000	746.7924	537.6041
3	T03	200000	1604.2269	774.3091
4	T04	300000	2146.6793	1106.4256
5	T05	400000	2947.1033	1659.5965
6	T06	500000	3702.4505	1948.6688
7	T07	600000	4213.7930	2506.8524
8	T08	700000	5435.7824	2917.9934
9	T09	800000	6180.7481	3355.4125
10	T10	800000	6580.6719	3580.6764
11	T11	1000000	7160.6074	4117.8822

Table 3 - data set for middel state with small number of data

Item	The Number Of Items											
ab	62	134	421	929	567	1050	1055	121	856	1024	834	2570
a	101	71	350	929	956	594	250	1905	1047	189	921	743
abc	85	289	407	49	607	142	1651	1253	1582	376	2488	2402
bc	112	271	129	357	424	1317	530	431	1805	3108	1856	1540
ae	40	176	752	86	530	345	1338	151	2175	1223	2896	987
bd	130	440	68	908	561	1542	1733	2331	2530	2857	1185	2170
aef	106	270	359	203	902	234	1036	1689	2150	1670	2824	1555
bcf	137	366	349	392	400	1179	1112	945	1454	933	1030	1297
d	101	149	430	334	1043	1500	1363	502	1620	3197	412	2328
de	19	255	414	717	948	1594	650	2256	389	375	2618	3085
df	46	328	698	996	719	421	1432	1232	640	1582	2300	2548
def	61	253	624	1102	1342	1081	850	2185	752	2466	1636	1775
Sum	1000	3000	5000	7000	9000	11000	13000	15000	17000	19000	21000	23000

Table 5 - data set for middel state with larg number of data

Item	The Number Of Items										
ab	1050	13573	32733	13210	67950	36307	77704	32124	23212	19344	85993
a	594	12371	14056	1097	2510	15819	6277	95513	52639	3567	56819
abc	142	4717	23929	26283	30207	41666	48268	17926	65958	78338	132427
bc	1317	959	2844	44385	56762	16424	63707	101393	50229	115655	62625
ae	345	4139	15060	33963	20011	17289	79909	94365	10020	44039	98518
bd	1542	10849	6024	39859	32794	56797	80248	49907	79784	51996	61707
aef	234	6528	25535	15678	23663	41234	53781	18213	124686	35807	60794
bcf	1179	5386	17135	33232	57371	34405	38512	19217	82194	107012	142966
d	1500	10438	15327	41780	14913	77195	34440	121480	99064	68460	122346
de	1594	14470	12349	10696	9587	64747	56760	75002	75344	126911	74537
df	421	7258	21552	18284	74950	27835	11872	25957	63421	125533	36199
def	1081	9312	13456	21533	9282	70282	48522	48903	73449	123338	65069
Sum	11000	100,000	200,000	300,000	400,000	500,000	600,000	700,000	800,000	900,000	1,000,000

Table 6 – results respond tims of execution RIDTDM for middel state

ID		Number of records	users respond time in Basic Apriori (ms)	users respond time in RIDTDM (ms)
1	T01	11000	223.6346	192.5958
2	T02	100,000	1091.9594	610.9677
3	T03	200,000	1913.0008	1288.3028
4	T04	300,000	2692.4124	1954.0700
5	T05	400,000	3647.9996	2490.3666
6	T06	500,000	4751.4616	2685.4508
7	T07	600,000	5533.8471	4167.2568
8	T08	700,000	6083.8594	3891.0978
9	T09	800,000	7426.0485	5169.0579
10	T10	900,000	8342.4277	5294.5612
11	T11	1000,000	9563.7927	6089.1487

Table 7 - data set for worst state

Item	The Number Of Items									
af	12289	4014	19488	4025	44635	37337	71494	109215	65781	65819
cd	432	17352	23321	35000	54374	52098	17631	107686	101096	32435
ae	9662	11228	8313	30042	54780	71394	75579	49037	106289	72837
bd	14700	13712	23969	38442	64518	1350	9568	46853	138141	25606
ce	13206	22796	5884	49409	68576	79208	73752	30689	86208	179059
bf	3149	23885	12254	52040	38224	96	11073	88464	24927	29969
ad	8060	12453	53122	50459	10144	16208	60958	12778	21295	142830
cf	7430	3742	48371	46142	8927	50078	78160	88354	102654	94650
abd	12558	5444	7827	31733	39198	73130	84937	84864	55395	44959
ade	4552	33310	18309	26250	73770	54799	41714	95672	14021	155349
bdf	6207	33657	35306	32142	32218	94259	68269	15836	149913	67905
ace	7755	18407	43836	4316	10636	70043	106865	70552	34280	88582
Sum	100,000	200,000	300,000	400,000	500,000	600,000	700,000	800,000	900,000	1,000,000

Table 9 - data set for middel state

Proposed time function	T(x)
The number of Items	x
processing power of the distributed systems	ρ
Minimum support	mc
Minimum confidence	ms
0.007231 (-0.001199, 0.01566)	α
Combination of data Items	c
1.013 (-4.983e+006, 4.983e+006)	β
The number of distributed systems	n

Table 10 – Analysis of the obtained function

Xi	Lower f(Xi)	f(Xi)	Upper f(Xi)	df(Xi)/dX	d2f(Xi)/dX2
11000	-1733.97	45.1248	1824.22	0.00396075	3.54E-09
109900	-22521.1	442.888	23406.9	0.0040513	3.62E-10
208800	-45018.6	844.962	46708.5	0.00407692	1.92E-10
307700	-57501.7	1248.99	59999.7	0.00409248	1.31E-10
406600	-92213.7	1654.31	95522.3	0.00410369	9.91E-11
505500	-114072	2060.62	118194	0.00411248	7.99E-11
604400	-118562	2467.71	123497	0.0041197	6.69E-11
703300	-161810	2875.46	167561	0.00412584	5.76E-11
802200	-166947	3283.78	173514	0.00413117	5.06E-11
901100	-248574	3692.59	255959	0.00413589	4.51E-11
1.00E+06	-230179	4101.84	238382	0.00414012	4.06E-11

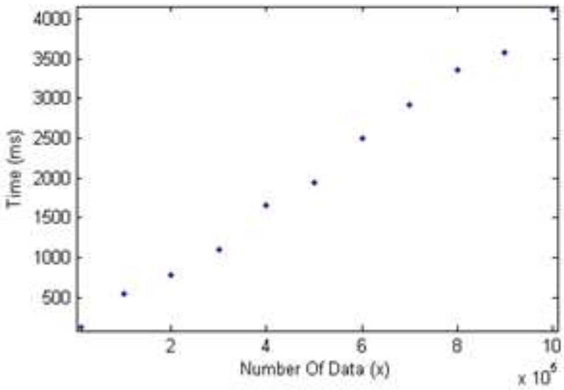

Figure 3- plotting time function

The results obtained from the preliminary analysis of plotting are shown in equation 1, which presents the closest function time is T(x).

Equation 1- Time Function

$$\mathbf{T(x)} = \frac{\alpha x^{\beta+\exp x} + c}{(mc + ms + n + p)}$$

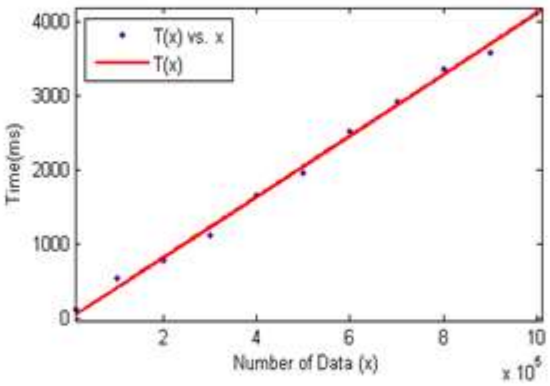

Figure 9 – drawing function time

Table 10, provides the Analysis of the obtained function from RIDTDM.

Conclusion

The present study was designed to determine the effect of distributing execution of intrusion detection system with data mining. In this investigation, the aim was to assess more speed to finding intrusion and increase saving time of reaction to the cyber threats because with the rapid development of Internet and network technologies, security issues also wants to highlight.

By the plotting of this function, we will have the following diagram in figure 9, that it closest diagram to the worst case.

References

Heady, R., Luger, G., Maccabe, A., and Servilla, M. The architecture of a network level intrusion detection system. Technical report, Computer Science Department, University of New Mexico, August 1990.

M. N. Mohammad, N. Sulaiman, and O. A. Muhsin, “A novel intrusion detection system by using intelligent data mining in weka environment,” *Procedia Computer Science*, vol. 3, no. 0, pp. 1237-1242, 2011.

H. Alanazi, R. M. Noor, B. Zaidan *et al.*, “Intrusion Detection System: Overview,” *arXiv preprint arXiv:1002.4047*, 2010.

Y. Bai, and H. Kobayashi, “Intrusion Detection System: Technology and Development,” in Proceedings of the 17th International Conference on Advanced Information Networking and Applications, 2003, pp. 710.

S.-H. Kim, and J. R. Wilson, “A discussion on ‘Detection of intrusions in information systems by sequential change-point methods’ by Tartakovsky, Rozovskii, Blažek, and Kim,” *Statistical Methodology*, vol. 3, no. 3, pp. 315-319, 2006.

Anderson J. P., et al., “Detecting Unusual Program Behavior Using the Statistical Components of NIDES”, Computer Science Laboratory SRI-CSL-95-06, 1995.

S. Dhopte, and N. Tarapore, “Design of Intrusion Detection System using Fuzzy Class-Association Rule Mining based on Genetic Algorithm,” *IJCSI International Journal of Computer Science*, vol. 9, no. 3, pp. 508-514, 2012.

E. Corchado, and Á. Herrero, “Neural visualization of network traffic data for intrusion detection,” *Applied Soft Computing*, vol. 11, no. 2, pp. 2042-2056, 2011.

G. Wang, J. Hao, J. Ma *et al.*, “A new approach to intrusion detection using Artificial Neural Networks and fuzzy clustering,” *Expert Systems with Applications*, vol. 37, no. 9, pp. 6225-6232, 2010.

J. J. Davis, and A. J. Clark, “Data preprocessing for anomaly based network intrusion detection: A review,” *Computers & Security*, vol. 30, no. 6–7, pp. 353-375, 2011.

S.-Y. Wu, and E. Yen, “Data mining-based intrusion detectors,” *Expert Systems with Applications*, vol. 36, no. 3, Part 1, pp. 5605-5612, 2009.

G.W. Dekker, M. Pechenizkiy and J.M. Vleeshouwers, “Predicting Students Drop Out: A Case Study”, Proceedings of

2nd International Conference On Educational Data Mining, Cordoba, Spain, July 1-3, 2009, pp 41-50

A. Din Riad and I. Elhenawy and A. Hassan and N. Awadallah, “Data Visualization Technique Framework for Intrusion detection” *IJCSI International Journal of Computer Science,* vol. 8, no. 1, pp. 508-514, 2011.

Y. Bai and H. Kobayashi, “Intrusion Detection Systems: Technology and Development,” Proceeding of the 17th International Conference on Advanced Information Networking and Applications (AINA’03), 2003.

W. Xuren, H. Famei, and X. Rongsheng, “Modeling Intrusion Detection System by Discovering Association Rule in Rough Set Theory Framework,” International Conference on Computational Intelligence for Modeling Control and Automation, and International Conference on
Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC’06), 2006.

I. Khalkhali, R. Azmi, M. Azimpour and M. Khansari, “Host-based Web Anomaly Intrusion Detection System, an Artificial Immune System Approach,” *IJCSI International Journal of Computer Science,* vol. 8, no. 2, pp. 11-24, 2011.

Sodiya, A.S., Longe, H.O.D. and Akinwale, A.T. (2004), “A new two-tiered strategy to intrusion detection”, Information Management & Computer Security, Vol. 12 No. 1, pp. 27-44.

Vikram Chopra, Sunil Saini and A. Choudhary, “A Novel Approach for Intrusion Detection,” *IJCSI International Journal of Computer Science,* vol. 8, no. 2, pp. 294-297, 2011.

S. Boubaker Ourida, “Implementation of an Intrusion Detection System,” *IJCSI International Journal of Computer Science,* vol. 9, no. 1, pp. 420-424, 2012.

R. Bace and P. Mell. Intrusion detection systems. Technical Report Special Publication 800-31, National Institute of Standards and Technology (NIST), 2001.

Roesch M. Snort-ligh tweight intrusion detection for networks. In:Proceedings of the 13th USENIX Conference on System Administration. Seattle, Washington; 1999. pp. 229 e 238.

Vallentin M, Sommer R, Lee J, Leres C, Paxson V, Tierney B. The nids cluster: Scalable, stateful network intrusio n detection on commodity hardwa re. Lecture Notes in Computer Science 2007;4637:1 07e 26.

S. Zanero, and S. M. Savaresi, “Unsupervised learning techniques for an intrusion detection system,” in Proceedings of the 2004 ACM symposium on Applied computing, Nicosia, Cyprus, 2004, pp. 412-419.

J. Xu, and C. R. Shelton, “Intrusion detection using continuous time Bayesian networks,” *J. Artif. Int. Res.,* vol. 39, no. 1, pp. 745-774, 2010.

L. Hanguang, and N. Yu, “Intrusion Detection Technology Research Based on Apriori Algorithm,” *Physics Procedia,* vol. 24, Part C, no. 0, pp. 1615-1620, 2012.

R. Zhong, and G. Yue, "DDoS detection system based on data mining." Proceedings of the Second International Symposium on Networking and Network Security, pp. 062-065, 2010.

X. Wu, V. Kumar, J. Ross Quinlan *et al.*, “Top 10 algorithms in data mining,” *Knowledge and Information Systems,* vol. 14, no. 1, pp. 1-37, 2008/01/01, 2008.

Y.K.Alp , O.Arikan , "Erp Source Reconstruction By Using Particle Swarm Optimization ", IEEE ICASSP 2009.

Yin Fen Low, and Daniel J. Strauss "An Inverse Transform Technique for the EEG Phase Reset Analysis " Proceedings of the 4th International IEEE EMBS Conference on Neural Engineering Antalya ,Turkey, April 29 - May 2, 2009

Zhang.Ch , Li.Zh , Zheng.D “An Improved Algorithm for Apriori” IEEE, ETSC 2009 Wuhan, China ,March 7-8 2009 ,PP.995-998.

A. G. Delavar, B. N. Lahrood, M. Nejadkheirallah *et al.*, “ERDQ: A Real-time Framework for Increasing Quality of Customer Whit Data Mining Mechanisms.” *Information Sciences,* vol. 180, no. 12, pp. 2375-2389, 2010