Research on Data Mining Technologies Appling Intrusion Detection

Zhang Qun
School of Computer Science
Hubei University of Technology
Wuhan, China
184034298@qq.com

Huang Wen-jie
Network Administration Center
Hubei University of Technology
Wuhan, China
184034298@qq.com

Abstract—Aim of the intrusion detection is to availably detect the intrusion from large stores of data. IDS(Intrusion Detection System) analyses sorts of data resource with data mining techniques to extract useful patterns and rules that could be used to guide the IDS to analyse intrusion. Allowing for characteristics of intrusion detection, a novel approach of mining frequent serial episodes from streams of time series, called TFSE, is proposed. TFSE(Time-table-joined Frequent Serial Episodes) is applied to analyse large stores of data related with time.

Keywords- intrusion detection; data mining; time serial pattern; TFSE

I. INTRODUCTION

The familiar methods of sequence pattern mining are: etc. The mining objects of these mining methods are many shorter sequences. Mining more frequently-appeared sequence pattern from these sequences, but which does not give a great consideration to the timeliness of data. Time-series data stream is a very long sequence, and the frequent episodes model in this sequence needs to be mined. The algorithms above do not directly aim at mining the episodes model in time-series data streams. Data stream mining has its own features. Incremental updating of model mining, maintenance of model and second mining are all new problems.

II. RELEVANT DEFINITIONS AND FORMULAS OF TFSE ALGORITHM

Wherever Episodes are the combination of several events types. For episodes composed of L events, its length is L. Using sign |A| to represent its length, this episodes is recorded as L-length episode.

The events in episode is sequential and in the moment for one certain event occurrence only happens this certain event, and such kind of episode is serial episodes. The events in serial episodes have definite time sequencing. For example, for serial episode CD, the occurrence time of C and D are t1 and t2 respectively, and if t1<t2, D occurs after C. Several or zero event may happen between t1 and t2. Event A occurs in time t is assigned as (A， t).

978-1-4244-6065-6/10/$26.00 ©2010 IEEE

Episodes time table has episodes <(A， t1)(B， t2)>. If adding the beginning occurrence time of event into table, episodes time table of episodes AB is showed as in table 1.

TABLE I. EPISODE TIME TABLE OF EPISODES AB

Time table(AB)

Start Time	*End Time*
t1	t2
t3	t5
t7	t9

The first row of episode time table is the occurrence time of first event while the second row is the occurrence time of final event.

If the length of episode is K, its corresponding episode time table is K-length time table.

Slip time windows: during the sequence analysis, the occurrence time of commonly-mined sequence events are close. Now one time window is used to define its closeness degree and the sequence events all are occurred within this time window. There is data stream <e1， t1> and time window breadth win and if tn-tm≤win， tm≤tn, then the events between tm and en (including tm and tn) are just the events of the time window (tm， tn). The next time window closely-following window (tm， tn) is (tk，tn+1). The value of tk should enable the events within the window to be as many as possible.

For example, the data streams fragments D A C B B A C D AA C B A A are showed as in table 2.

TABLE II. CASE DATA

event	D	A	C	B	B	A	C	D	A	A	B	A	A	D	…
Time (sec)	1	2	4	6	7	10	13	14	16	17	18	19	21	22	…

The breadth of time window is 4 seconds, first window(1 ， 1), second window(1 ， 2), windows as following: (1，4) etc.

The calculation formula of episode support degree: supposing the amount of events in time sequence is D, episodes is A, the support number in time sequence is C, then the support degree of episodes A is Support=C/D.

Deciding the min-support min_sup , if the support degree of serial episodes A is SupportA and SupportA≥ min_sup, then A is called frequent serial episodes.

The calculation formula of episodes confidence degree: episode B is sub-episode of episode A, and the support degree of A is Sa while the support of B is Sb, then the confidence degree of rules A->B is confidence=Sa/Sb.

III. TFSE ALGORITHM

A. S TFSE algorithm is indicated as following:

(1) Scan the time sequence, the episodes L1 with length being 1 and episodes L2 with length being 2 are produced and construct the 2-length time table.

(2) Calculate the support degree of L1 and eliminate the episodes of L1 that do not meet min-support. Make cutting and pasting performance to L1and L2 and make L2 as initial seed sets.

(3) According the seed sets Li with a length of i(i>1), the candidate episodes sets Ci+1 with its length of i+1 can be produced through connecting and cutting performances. Make database linking operation to the two suitable episodes time table, and create in the corresponding episodes time table of Ci+1. Make the candidate episodes meeting min-support to be new seed sets.

(4) Repeat the step three until there is not any new episodes pattern or new candidate episodes producing any more.

Producing candidate episodes by TFSE algorithm mainly has two steps.

(1) Connecting: if the obtained episode from eliminating the first event in episodes model s1 and eliminating the final event in episodes s2 is the same, then connection of s1 and s2 can be done. That is to say, add the final event of s2 to s1.

(2) Cutting: if the some sub-episodes of candidate episodes is not episode model, then it is impossible for this candidate episode to be episode model and it should be deleted from candidate episodes.

TFSE algorithm adopts producing candidate and test technology that is similar to PSP algorithm. Its distinctions with PSP algorithm is that how to test the candidates. PSP calculates the support number of every candidate sequence pattern through scanning the whole time sequence database. However, it is ok for TFSE algorithm operating the database linking to two suitable episodes time table, which greatly saves the expense of I/O and CPU.

B. The construction arithmetic of 2-length-time-table

Extract 2-length-time-table from time sequence and record the beginning and ending time of 2-length-episodes in the corresponding tables. When the data is collected, it also can use this algorithm to shift sequence into 2-length-time-table. Time sequence S={(Ai， ti)| A∈EventType，1≤i≤ N，ti-1 < ti < ti+1} is composed of a events types and the occurrence times of events.

Store the events in ti time window by using a double-linked circular list structure. The linked-list-node-data structure is defined as:

EventType; /*the types of events*/

Time; /*the occurrence time of event*/

Prev; /*the pointer that points to the previous node*/

Next; /*the pointer that points to the next node*/

Pointers WinHea and WinTail point to first event and final event of the window respectively.

When a new event(Ai，ti) enters the window, two thing must be done. The first one is to add the new event to the present window. The second one is to delete the event (Ak，tk) which does not belong to the present window, ti-tk≥win. The event(D，22) enters window and new event Ai and every event Bn of the window together construct episodes Bn Ai with a length of 2.

In order to conveniently calculate and save the EMS memory space, the only value should be used to sign each type of event. In construction algorithm, using array Count[] to store the support number of event. If the value sign of event F is 4, then the value of count [4] is support number of event F.

The construction arithmetic of a2-length-time-table is described as following.

```
1  InsertAfter(WinHead，A1，t1);
2 /*first event enter window*/
3 WinHead=WinTail;
4 for(i=2; i<=N; i++)
5 {Count[Ai]= Count[Ai]+1;
6  /*stored the number of events_Ai */
7  pt=WinTail;
8  if(WinTail == WinHead)
9    InsertAfter(WinTail，Ai，ti);
10  /*new event (Ai，ti) enter window*/
11  else
12  {WinTail = WinTail->next;
13   WinTail->EventType = Ai;
14   WinTail->Time = ti;
15  }
16 while(1)
17 {if((ti – pt->time ) > win) /*if event out window */
18   {Winhead = pt; break; }
19  else
20   {B = pt->EventType;
21    InsertTable(Table(AiB)，ti，pt->time);
22    }
23  if(pt == WinHead) break;
24  pt=pt->prev;
25 }
26 }
```

In the produced episode time tables will appear the records that are same with the beginning time or the ending time. In order to get the real and more support numbers of episode model in the time windows, the time span should be remained in its smallest for such records while the other records should be removed. Therefore, records should be chosen and added to episode time tables. Perform the function t InsertTable() hat has been selected and added into

recording functions. Because the treated time is ceaselessly increasing, if the beginning time of the record that is to be added is same with that of the record has been added, the record that is to be added should not be added in the episode time table. If the ending time of the record that is to be added is same with that of the previous record that has been added , the record that is to be added should cover the previous record that has been added. The 2-length-episode-time-table is shown as in figure 1.

C. *The linking performance of episode table and the calculation of support degree*

Calculate the support degree of episode s=<s1s2…sm…sk>, m=$\left\lfloor \frac{k+1}{2} \right\rfloor$. The corresponding episode time table of episode ss=<s1s2…sm> is table(ss) and the corresponding episode time table of episode se=<smsm+1…sk> is table(se). The corresponding episode time table(s) of episode s is produced through making database linking performance of table(ss) and table(se). The SQL sentence that is needed to be performed is as following.

INSERT INTO table(s)(startTime，endTime)
SELECT table(ss).startTime，MIN(table(se).endTime)
FROM table(se)，table(ss)
WHERE table(ss).endTime <= table(se). startTime
AND table(se).endTime – table(ss).startTime <= win
GROUP BY table(ss).startTime

The support number of Episode s can be obtained by SQL sentence select count(*) from table(s). The support degree can be calculated by using the calculation formula of support degree.

D. *Second mining*

- Period analysis. If some episode has periodicity, then it must be frequent in a certain range. Only one suitable support degree and window breadth (the window breadth should be no less than the period length) needed to be chosen. Thus, an period analysis of the episode time table produced during the mining of frequent episode can identify whether the corresponding episode has periodicity or not. if only period analysis is done, the time-spatial expense of this method is very large. However, in the circumstance that period analysis is required to be done while mining frequent episode is also demanded, the time-spatial expense of period analysis will be small.
- Regulations mining of non-frequent episodes. Usually, it is through mining frequent episode to mine episode regulations and the regulations mined in this way will not be complete. Although some episodes are not frequent(for they do not meet the minimum support degree.), it is possible that they have regularity. That is to say, the support degree is small while the confidence degree is big. Through a further mining of the episode that has big confidence degree and small support degree, the regulations of non-frequent episodes can be obtained. TFSE method could store episode time table and mine the regulations of non-frequent episodes, so only the performance of database linking for the episode time table that has big confidence degree and small support degree is needed until the mined episodes do not meet minimum support degree.
- Adding or changing mining parameters. The value to mining parameter directly influence the mining result. Besides the breadth of window, support degree and confidence degree, other commonly-used parameters are the duration time T, overlapping windows W of accidents and space of time INT in discovered models and so on.

If the parameter relevant to time should be added, time constraint only be needed to add during the construction of 2-episode length table and linking episode table, through which the mining result can be obtained. Other parameters are not added in TFSE method, which provides the conditions for the adding parameter constraint in second mining. The second mining progress of adding parameters, the treating target is episode time table rather than source time serial.

TFSE algorithm makes full use of intermediate-result-episode-time-table mined in the first time during the second mining, which enables the second mining to go quickly. If it does not destroy intermediate result, second mining can be done for many times for TFSE algorithm has the effect of one time of mining but many times of benefiting.

E. *The storage of time-episode –table*

If the source time serial is great and all the intermediate results have to be stored, the disk space spent is very large. If there is no necessary for second mining, the episode time tables that are no longer used in the mining progress can be removed and there is no need to store them. If the second mining is necessary, in order to reduce the spent space, through storing part of episode time tables, the remaining part that is not stored can be produced by database linking with the stored episode time table. Although it could reduce the spent space, it is at the cost of the episode time-table that has not stored.

IV. CONCLUSION

This paper, combing the features of intrusion detection, targeting at the mining problems of frequent serial episodes in the time-series data streams, proposes a method of mining frequent serialization in the time-series data stream TFSE(Time-table-joined Frequent Serial Episodes). This method introduce the conception of episodes time-table, with one episode corresponding to one episode time-table, and through performing the database linking operation among the episode time-table, corresponding new time table will be produced. The number of the record of episode time table is just the support number of this episode. This method not only uses the intermediate- result-episode-time-table and also stores it. Therefore, the method has the characters of smaller expanse in testing the candidate episode and of convenience of second mining, which, to a great extent, solve the problem

of difficulty in mining the long episodes. Through adding and changing the mining parameter, more information of greater value will be mined from the intermediate result.

ACKNOWLEDGMENT

Thanks for my parents take good care of me, and thanks for my friends care for me.

REFERENCES

[1] RealSecure http://www. iss. net/prod/rs. html

[2] NetSTAT http://www. cs. ucsb. edu/~kemm/netstat. Html

[3] AAFID www. securityfocus. Com/tools/57/scoreit

[4] http://www.20cn.net/ns/wz/otherwz/data/20030722032734.htm

[5] W. Lee and S. J. Stolfo. Data Mining Approaches for Intrusion Detection[A]. In: Proceedings of the 7th USENIX Security Symposium[C], San Antonio, TX, 1998

[6] H. Mannila , H.Toivonen , and A. I.Verkamo. Discovering Frequent Episodes in Sequences[A]. In: Proceedings of the 1st International Conference on Knowledge Discovery in Databases and Data Mining[C] Montreal, Canada, August 1995

[7] SU Pu-Rui , LI De-Quan , FENG Deng-Guo , A Host-Based Anomaly Intrusion Detection Model Based on Genetic Programming, Journal of Software, 2003, 14(6):1120－1126

[8] CAO Yuan-da , XUE Jing-feng , The Application of Weighted Association Rules in Host-Based Intrusion Detection System , Journal of Beijing Institute of Technology, 2002, 11(4):418－421

[9] Udo Payer, State-Driven Stack-Based Network Intrusion Detection System , 7th international Conference on Telecommunication ConTEL , pp.613-618, 2003

[10]

[11] Jitender Deogun, Liying Jiang, and Vijay Raghavan. Discovering maximal potentially useful association rules based on probability logics. In Proceedings of the Fourth International Conference on Rough Sets and Current Trends in Computing (RSCTC), 2004.

[12] Mannila H, Tovionen T H, Verkamo A I. Discovering frequent episodes in sequences[EB/OL]. http: ‖ www-courses.cs.uiuc.edu/～cs497jh/papers/97mannila.pdf, 2002-03-28/2002-05-12

TimeTable(BA)

startTim	endTime
7	10
18	19

TimeTable(AA)

startTim	endTime
16	17
17	19
19	21

TimeTable(AB)

startTim	endTime
17	18

TimeTable(AC)

startTim	endTime
2	4
10	13

TimeTable(BB)

startTim	endTime
6	7

TimeTable(DA)

startTim	endTime
1	2
14	16

TimeTable(AD)

startTim	endTime
21	22

TimeTable(CB)

startTim	endTime
4	6

TimeTable(CC)

startTim	endTime
11	22

TimeTable(CA)

startTim	endTime
13	16

TimeTable(CD)

startTim	endTime
13	14

TimeTable(DC)

startTim	endTime
1	4

Figure 1. 2-length-episode-time-table