Using Data Stream Management Systems for Computer Worms Monitoring

Shikhaliyev Ramiz Huseyn[a], and Seung-Woo Seo[b]

[a]*Institute Information Technology, Azerbaijan National Academy of Sciences, 9, F.Agayev str., Baku, Azerbaijan Republic, AZ1141, ramiz@science.az*
[b]*School of Electrical Engineering and Computer Sciences, Seoul National University, Seoul 151-744, Republic of Korea, sseo@snu.ac.kr*

Abstract

This paper discusses some of the questions on the data stream processing, that includes data stream models, continuous query processing, data stream management systems, and the possibility of using them in monitoring and detection of worms.

Keywords: computer worms, network traffic analysis, data stream, data stream model, a continuous query, database management systems, data stream management systems.

1. Introduction

In recent years, widespread computer worms have a very serious threat to the security of computer networks (CN), particularly for corporate networks. This is mainly due to low levels of redundancy, and the structure of corporate networks compared to the Internet.

A computer worm is a program that self-propagates across a network exploiting security or policy flaws in widely-used services [1]. At the same time, for spreading computer worms use different strategies such as scanning [2].

A computer worms life consists of the following phases: target finding, transferring, activation, and infection [3]. In the first two phases, computer worms exhibit high network activity, and therefore relatively easy to detect them. Therefore, early detection of computer worms is very important by continuously monitoring and analyzing the large volume of network traffic in real time. However, the traditional methods of data analysis that are used in monitoring tools, the task becomes extremely difficult task because of the continuity, consistency and speed revenue streams, as well as their change over time, etc. In addition, the rapid and continuous supply of large amount of data creates difficulties in storage, computation and communication via computer systems.

In recent years, there have been several new technologies introduced to address the analysis of data streams continuously moving in real time. At the same time, some existing software applications such as database management systems (DBMSs) and the generators of the rules [4] can be used for this purpose.

DBMSs can serve as a good platform for the development of tools for analyzing network traffic. However, with traditional DBMSs, there are some serious drawbacks that limit their functional ability to process data flows continuously moving in real time [5].

Based on the foregoing, for data collection and analysis of network traffic that comes continuously, consistently and in real time, it is proposed to use the data stream management systems (DSMSs) [6].

2. Data stream models

A data stream is a real-time, continuous, ordered (implicitly by arrival time or explicitly by timestamp) sequence of items [7]. Typically, incoming data stream is very fast and impossible to order them arrival, but due to the limited memory streams storage becomes difficult task to as a whole.

Since the elements can act in isolation, the data streams can be modeled as a sequence of lists of elements [8]. Individual elements of the streams may take the form of relational tuples or the specification of objects.

Data stream have multiple models and can be formally described as follows. The input $s_1, s_2, \ldots$ stream after the other, element by element, and describes the main signal, which is one-dimensional function - $S:[1..N] \rightarrow R$. In this case the input can consist of multiple streams or multidimensional signals. Depending on how the items describe the data stream models characterized by:

- Time series model. Each s_i-th element is equal to $S[i]$, and they come in order of increasing values of i. This model is suitable for time-series data, for example, by observing the amount of network traffic, the CN every 10 minutes, etc. In each such period of time we observe the following new data;

- Model cash register. Here, the s_i-th elements of an increase to $S[j]$-s. We assume that $s_i = (j, I), I_i \geq 0$, this means that $S_i[j] = S_{i-1}[j] + I_i$, where S_i is the signal state after the observation of i - th element in the stream. As the cash register, many s_i-th may eventually grow to a certain. This model is the most popular data stream model. It is suitable for monitoring the CN IP-addresses that access the Web server and send packets to the network, etc.;

- Model of the turnstile. Here, the s_i-th elements of an updated on $S[j]$-s. We assume that $s_i = (j, I_i)$ this

means that $S_i[j]=S_{i-1}[j]+I_i$, where S_i is the signal after the observation of i-th element in the stream and I_i can be positive or negative. This model is the most common data stream model, which allows us to study the dynamic situation, where the elements constantly coming in and leaving it.

These models are described in more detail in [9].

It is known that the application for processing data streams need to support continuous queries. The semantics of continuous queries is as follows. For simplicity, we assume that the time is represented as a sequence of integers. Let $A(Q,t)$ be a set of Q responses to the ongoing inquiry that takes place over time t, which τ is the current time, and start time is 0. If the request is a continuous monotone, it suffices to consider requests for new arrivals and the results of the elements added to the training samples. Thus, many answers monotone continuous query Q, occurring in time τ, is expressed by the formula [10]:

$$A(Q,\tau)=\bigcup_{t=1}^{\tau}(A(Q,t)-A(Q,t-1))\cup A(Q,0)$$

In contrast to the monotone continuous queries, non-monotonic continuous queries must be recalculated at each re-evaluation that leads to the following semantics [10]:

$$A(Q,\tau)=\bigcup_{t=0}^{\tau}A(Q,t)$$

3. Data stream management systems

DSMSs need to manage and process the data stream elements, before the data elements are replaced by subsequent incoming data elements. However, this task is very difficult in such intense and unpredictable environments, as the CN, particularly the Internet, in which the order of receipt of the data elements cannot be controlled, and the size of the data stream at the same time is unlimited.

Unlike traditional DBMSs, it can be carried out in DSMSs with continuous queries in relation to a continuous data streams in real time, come in and leave it at this time on processing the data is stored only in memory and the system can do a variety of data, for example, the data of the stock exchange or network traffic. However, like any DBMSs, DSMSs requires a scheme that describes, type and structure of the data being managed. Therefore, reusing and changing the DSMS applications for analyzing network traffic is not that difficult. In this case, because of the unique features of data streams and continuous queries DSMSs should have following requirements:

- A data model and query semantics should support the operation that are performed in ordered and controlled manner in time;
- Approximate structure must be used due to the inability to maintain full stream at a time [11];
- Plans for streaming queries cannot use the blocking operators, which must take the entire entry, pending the outcome;
- Because of the limitations in performance and memory the algorithms for processing data streams in real time can not be repeated on the same data more than one time;
- Applications that monitor streams in real time, should quickly respond to unusual data values;
- During the execution of continuous queries the conditions in which the system is located, may change (for example, changing the flow rate);
- The sharing of multiple queries is necessary in order to provide scalability.

The most widely used domain DSMSs is to analyze network traffic to the CN. DSMSs can be used to analyze the network traffic of the CN, they must process the data with performance proportional to the load of the CN. However, to reduce the flow of data on the current network load factors can be 4.1-9.1 [12].

Application for monitoring the CN DSMSs will create reusable components analysis of network traffic that can be applied for analysis of data in real time, as well as for the analysis of data collected in advance. With typical problems of analysis of network traffic are as follows:

- Analysis of system load, for example, how often certain ports used protocols such as FTP or HTTP, to connect to the server, it uses the bandwidth of different applications, as individual users or user groups are using the bandwidth;
- Analysis of the characteristics of network traffic, such as the average lifetime and packet size, the ratio between the number of lost packets and the lifetime of packets, etc.;
- Analysis of the characteristics of the sessions, such as the duration of the interaction of customers with Web servers, the response time of Web servers to client requests, the duration of the users of Internet services, etc.

Today there are many DSMSs, for example, STREAM [13], GigaScope [14], TelegraphCQ [15], StatStream [16], Tribeca [17], etc., which can be used for solving various problems of monitoring the CN. More extensive information on each DSMSs available in refs [13-17].

4. Conclusion

For monitoring and detection of computer worms is very important to track and analyze large volume of network traffic in real time. The DBMSs can serve as a good platform for the development of tools for analyzing network traffic. However, traditional DBMSs, there are some serious drawbacks that limit their functional ability to process data streams

In this paper for the collection and analysis of network traffic coming continuously, consistently and in real time proposed using the DSMSs techniques. DSMSs will cope with large data network traffic in real time and provide an opportunity to monitor and detect computer worms.

Reference

[1]. N. Weaver, V. Paxson, S. Staniford, R. Cunningham, A taxonomy of computer worms, Proceedings of ACM CCS Workshop on Rapid Malcode, October 27, 2003, pp. 11-18.

[2]. C. Smith, A. Matrawy, S. Chow and B. Abdelaziz, Computer Worms: Architecture, Evasion Strategies, and Detection Mechanisms, Journal of Information Assurance and Security, 4 (2009), pp. 69-83.

[3]. Pele Li, Mehdi Salour, Xiao Su: A survey of internet worm detection and containment, IEEE Communications Surveys and Tutorials, Communications Surveys & Tutorials, Vol. 10, No. 1, 2008, pp. 20-35

[4]. L. Brownston, R. Farrell, E. Kant, and N. Martin, Programming expert systems in OPS5: an introduction to rule-based programming, Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1985, 471 p.

[5]. Carney, D., Cetintemel, U., Cherniack, M., Convey, C., Lee, S.,Seidman, G., Stonebraker, M., Tatbul, N., Zdonik, S.,:Monitiring Streams- A New Class of Data Management Applications, Proceedings of the 28th international conference on Very Large Data Bases (2002) (VLDB '02) Conference, Hong Kong, China, 2002, pp. 215-226.

[6]. Babcock, B., Babu, S., Datar, M., Motwani, R., Widom, J.: Models and Issues in Data Stream Systems, Proceedings of the Twenty-first ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 3-5, Madison, Wisconsin, USA. ACM 2002, pp. 1-16.

[7]. Lukasz Golab and M. Tamer Ozsu, Issues in Data Stream Management, SIGMOD Record, Vol. 32, No. 2, June 2003

[8]. P. Tucker, D. Maier, T. Sheard, L. Fegaras. Enhancing relational operators for querying over punctuated data streams. 2002.
www.cse.ogi.edu/dot/niagara/pstream/punctuating.pdf

[9]. A. Gilbert, Y. Kotidis, S. Muthukrishnan and M. Strauss. Surfing wavelets on streams: One pass summaries for approximate aggregate queries. VLDB Journal, 2001, pp.79–88.

[10]. A. Arasu, S. Babu, J. Widom. An Abstract Semantics and Concrete Language for Continuous Queries over Streams and Relations. Technical Report, Nov. 2002. http://ilpubs.stanford.edu:8090/563/.

[11]. Y. Zhu, D. Shasha. StatStream: Statistical Monitoring of Thousands of Data Streams in Real Time. In Proc. Int. Conf. on Very Large Data Bases, 2002, pp. 358–369.

[12]. Micheel, J., Braun, H.-W., Graham, I.: Storage and Bandwidth Requirements for Passive Internet Header Traces, Workshop on Network-Related Data Management, in conjunction with ACM SIGMOD/PODS 2001, Santa Barbara, California, USA, May 21-23, 2001

[13]. The STREAM Group. STREAM: The Stanford Stream Data Manager (short overview paper), IEEE Data Engineering Bulletin, Vol. 26 No. 1, March 2003, pp. 19-26.

[14]. Cranor, C., Johnson, T. Spatcheck, O., Shkapenyuk, V.: Gigascope: A Stream Database for Network Applications, ACM SIGMOD 2003, San Diego, California, USA, June 9-12, 2003, pp. 647-651

[15].Sailesh Krishnamurthy, Sirish Chandrasekaran, Owen Cooper, Amol Deshpande, Michael J. Franklin, Joseph M. Hellerstein, Wei Hong, Samuel R. Madden, Vijayshankar Raman, Fred Reiss, and Mehul A. Shah. TelegraphCQ: An Architectural Status Report. IEEE Data Engineering Bulletin, Vol 26(1), March 2003, pp. 11-18.

[16]. Yunyue Zhu, Dennis Shasha "StatStream: Statistical Monitoring of Thousands of Data Streams in Real Time" VLDB, August, 2002. pp. 358-369

[17]. Mark Sullivan: Tribeca: A System for Managing Large Databases of Network Traffic, USENIX Annual Technical Conference (NO 98), June 15-19, 1998, pp. 13–24.