Finding Widespread Events with Simple Bitmaps

Syed Moeen Ali NAQVI\(^1\), Nonmember and MyungKeun YOON\(^{(a)}\), Member

SUMMARY  Finding widespread events in a distributed network is crucial when detecting cyber-attacks or network malfunctions. We propose a new detection scheme for widespread events based on bitmaps that can succinctly record and deliver event information between monitoring agents and a central coordinator. Our proposed scheme reduces communication overhead as well as total number of rounds, and achieves even higher accuracy, compared with the current state of the art.

key words: widespread event, network monitoring, bitmap, network management

1. Introduction

In distributed networks, monitoring agents are deployed at critical network segments or servers for network and security management. Their job is to monitor network lines and servers and report to a central coordinator when any special event is detected, so that coordinator may take necessary actions. In this paper, we assume two-tier architecture where multiple monitoring agents report collected information to one central coordinator as in [1], [2], which is practically implemented in industry.

Recently, Cai et al. defined a widespread event as an event that is commonly observed by all monitoring agents at the same time [2]. An event can be a simple IP address, a domain name, a flow id, or a log from any devices. Detecting widespread events in a distributed network is crucial as it can help detect cyber-attacks, network malfunctions and traffic abnormalities, etc.

Finding widespread events requires every agent to report all of its observed events to the coordinator at every monitoring period. This process may consume quite network bandwidth and computational resources. Some efficient methods to find widespread events have been proposed by using a special bloom filter family, called CFSP for Combinaible Filter Solution with Progressive filtering [2], [3]. The main idea is that each agent sends a variable-sized bloom filter to the coordinator instead of lengthy event ids. The coordinator combines all filters into one, and analyzes this combined filter to find potential widespread events.

In this paper, we present a new scheme to efficiently find widespread events by using bitmaps of the same size between agents and a coordinator. This scheme includes a pre-processing round to fix the appropriate size of bitmaps, and a postprocessing round to remove false-positives. Experimental results show that our proposed scheme not only saves communication overhead but also removes false-positives. In addition, the new scheme reduces the total number of communication rounds, compared with CFSP that is the current state of the art.

2. Problem Definition

We assume that two-tier monitoring architecture is used for event collection where \(k\)-monitoring agents and one central coordinator exist as shown in Fig. 1 and this assumption is also made in [1], [2]. Each agent is denoted as \(n_i\) (\(0 \leq i \leq k - 1\)) and reports what it has observed to the coordinator. This reporting process can be repeated periodically. We denote \(S_i\) as a set of events that agent \(n_i\) has observed, and therefore \(S_i\) should be sent to the coordinator. The coordinator finds any event that is included by all \(S_i\)'s. We denote the set of all widespread events as \(W\); therefore, \(W \subset \bigcap_{i=0}^{k-1} S_i\) and \(|W| \leq \min(|S_i|)\) by definition.

In this paper, we assume that from several dozens to thousands of monitoring agents are deployed for network monitoring, but our proposed scheme can work for even a larger scale of networks. In this highly distributed environment, reporting packets from agents to the coordinator may consume a significant amount of network bandwidth, especially around the coordinator, because monitored information should be reported to the coordinator periodically.

We define communication overhead as the total amount of bits transferred between all monitoring agents and the coordinator to detect widespread events as in [2]. The number of rounds is defined as the number of a pair of sending and receiving process between monitoring agents and the coordi-
ordinator. A false-positive ratio is defined as the total number of mistakenly-detected events divided by the total number of false-positives and true-negatives. These three metrics are used for performance evaluation in this paper, which is the same as in [2].

In this paper, two-tier monitoring architecture is considered that comprises a central coordinator and multiple monitoring agents. On the contrary, Chen et al. presented a fully distributed architecture where no central coordinator exists, but agents send and receive monitoring information each other [3]. This peer-to-peer model is beyond the scope of this paper. We believe that two-tier architecture is more practical and widely deployed for enterprise networks where critical security-related information is collected into an inside server. For example, enterprise networks generally have a centralized log collection system, called enterprise security management (ESM) or security information and event management (SIEM) [4]. As a central coordinator and monitoring agents can be supervised by the same authority, no monitoring information is open to outside. Another example is managed security service (MSS) or information sharing and analysis service (ISAC) where a center provides log collection and analysis service to its members. In this case, each member deploys monitoring agents and uploads collected information to the ISAC center with encrypted communication channels [5]. The center deploys a central coordinator.

3. Bitmap-Based Widespread Event Detection

We present a new Bitmap-Based scheme for Widespread Event detection (BBWE) that consists of three rounds; fixing bitmap size, transferring bitmaps, and eliminating false-positives. We explain the three rounds of BBWE in detail.

During the first round of BBWE, each agent \(n_i\) records its observed events as a set of \(S_i\). Then, \(n_i\) reports \(|S_i|\) to the coordinator. The coordinator computes \(m = \frac{|S_i|}{2^{\log m}}\), that becomes the standard bitmap size, and notifies each agent of \(m\). We explain the average of \(|S_i|\) would be a good choice for the bitmap size in later.

Each agent reports its bitmap to the coordinator in the second round. First, \(n_i\) encodes its bitmap from \(S_i\); let \(B_i\) be the bitmap of \(n_i\) and \(B_i[j]\) is the \(j\)th bit of \(B_i\), \(0 \leq i \leq k - 1\), \(0 \leq j \leq m - 1\). An agent also keeps a hash table to record \(S_i\) and let \(H_i[j]\) be the \(j\)th slot of the hash table, \(0 \leq j \leq m - 1\). We use one hash function, \(h(e)\), to compute an index value for event \(e\) where \(0 \leq h(e) \leq m - 1\). Every agent has a bitmap of size \(m\) initialized to zero. For every \(e \in S_i\), \(B_i[h(e)]\) is set to ‘1’.

Then, all \(B_i\)’s are transferred to the coordinator that combines all \(k\) bitmaps into integer array \(A[m]\), in which \(A[j] = \sum_{i=0}^{k} B_i[j]\). The coordinator selects all \(A[j]’s\) equal to \(k\). Note that \(A[h(e)] = k\) if \(e\) is a widespread event. The reverse is not always true because hash collisions may accidently increase this value to \(k\). For each \(A[j]\) that is equal to \(k\), the coordinator notifies \(n_i\) of all \(j\)’s if \(B_i[j] = 1\).

During the third round of BBWE, agents and the co-ordinator filters out false-positives. The agent searches the bitmap and hash table for any event indexed from the notification of the coordinator in the second round. The agent generates a set of potential widespread events, denoted as \(R_i\), by gathering all these indexed events. Then, \(n_i\) reports \(R_i\) to the coordinator. For each event in \(R_i\), the coordinator counts the number of distinct agents that reported the event. Finally, the set of widespread events \(W\) is obtained by collecting events that are included in all \(R_i’s\), \(0 \leq i \leq k - 1\). The coordinator sends \(W\) to the agents. Finally, each agent knows \(W\).

Note that the communication overhead of transferred bits is mainly caused by round two and three. In round two, all agents send \(m \times k\) bits in total to the coordinator, which definitely increases with \(m\). In round three, each agent sends \(R_i\) to the coordinator, the size of which would decrease with \(m\). Therefore, we need to find an appropriate value for \(m\).

We assume that \(|W|\) is far less than \(\min(|S_i|)\), which is often the case in network security and management. In this case, setting \(m = \frac{\sum_{i=0}^{k} |S_i|}{k}\) at the first round would be a reasonable choice. To prove this, we first compute the false-positive probability that \(A[j]\) becomes \(k\) with no widespread event involved, which is denoted as \(P_f\). Then,

\[
P_f = \prod_{0 \leq j \leq k - 1} \left(1 - \left(1 - \frac{1}{m}\right)^{|S_i|}\right) = \prod_{0 \leq j \leq k - 1} \left(1 - e^{-\frac{|S_i|}{m}}\right) (1)
\]

which is maximized when \(|S_i| = |S_i| = \cdots = |S_{k-1}| = m\). Even in this worst case, \(P_f = (1 - e^{-\frac{1}{k}})^k\) that becomes around 0.01 and 0.0001 with \(k = 10\) and 20, respectively.

We also argue that setting \(m = \frac{\sum_{i=0}^{k} |S_i|}{k}\) would reduce \(|R_i|\) of the third round to \(\frac{|S_i|}{m} \approx 1\) if there is no widespread event. However, this analysis model does not reflect the number of widespread events that would make it complicated. In this paper, we verify the validity through experiments.

4. Experiments

The performance of BBWE is verified through simulation experiments. We compare BBWE with CFSP that is the current state of the art scheme. For fair comparison, we use almost the same setting as in [2]; the number of events observed by each monitor is randomly chosen between 100,000 and 1,000,000 by default. The set of widespread events is generated with the intersection ratio that is defined as \(R_{INTS} = \frac{|W|}{\min(|S_i|)}\), which is the ratio between the number of widespread events and the minimum \(|S_i|\). The default value of \(R_{INTS}\) is 0.5 and \(k = 100\). Note that default value of \(k\) was 10 in [2]. In this paper, \(k\) is increased to cover highly distributed environments. With these parameters, the average of \(|S_i|\) is around 550,000 \(\approx 0.5\) Mb.

Figure 2 compares the communication overheads of CFSP and BBWE as \(R_{INTS}\) changes. The communication overhead is the total number of bits transferred between agents and the coordinator. Although BBWE is designed for small \(R_{INTS}\) values, BBWE outperforms CFSP
with $R_{INTS} < 0.7$. Note that $R_{INTS}$ is likely to be small, less than 0.2, in practice. We also include RAW that represents a naive scheme where the bare values of $S_i$ and $W$ are transferred between agents and the coordinator. We limit the length of event id to 64 bits for reasonable comparison, which gives a favor to RAW.

We stress that BBWE not only saves the communication overhead but also enhances the accuracy. While CFSP causes false-positive errors, BBWE does not. Figure 3 shows this difference clearly. The gap of communication overhead increases when CFSP is tuned to be more accurate. Another advantage of BBWE is the reduced and fixed number of rounds; BBWE always requires three rounds, but CFSP needs $-\frac{\ln(f)}{2}$ rounds where $f$ is a false-positive rate. For example, CFSP needs around 7 rounds to satisfy $f = 0.01$.

A peer-to-peer scheme for widespread detection is recently presented where a monitoring agent exchanges its event list with each other and there is no central coordinator [3]. We compare BBWE with the peer-to-peer model in terms of communication overhead, which may interest some readers. The communication overhead of one peer node in [3] is smaller than BBWE; however, the aggregate overhead is larger than BBWE. For example, when $R_{INTS} = 0.5$, $k = 32$ and number of events randomly chosen between 100,000 and 1,000,000, BBWE consumes around 204 Mb between monitoring agents and a central coordinator. Under the same condition, the peer-to-peer scheme of [3] requires 1,280 Mb in total among monitoring agents.

5. Conclusion

This paper has presented a new scheme to identify widespread events in distributed networks. This scheme fits well for highly-distributed networks such as future internet of things as well as current enterprise-level distributed environments. The proposed scheme has an edge over the current state of the art in terms of both communication overhead and accuracy.

Acknowledgments

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (No. 2017-0-00513-001, Developing threat analysis and response technology based on Security Analytics for Heterogeneous security solution). This work is supported by Global Scholarship Program for Foreign Graduate Students at Kookmin University, Seoul, Korea.

References

[1] M. Marchetti, F. Pierazzi, M. Colajanni, and A. Guido, “Analysis of high volumes of network traffic for advanced persistent threat detection,” Computer Networks, vol.109, Part 2, pp.127–141, 2016.
[2] Z. Cai, M. Chen, S. Chen, and Y. Qiao, “Searching for widespread events in large networked systems by cooperative monitoring,” Proc. IEEE ICNP’15, pp.123–133, Nov. 2015.
[3] J. Chen, Z. Cai, and S. Chen, “Efficient distributed joint detection of widespread events in large networked systems,” Proc. IEEE Globecom’16, Dec. 2016.
[4] G. Suarez-Tangil, E. Palomer, A. Ribagorda, and I. Sanz, “Providing SIEM systems with self-adaptation,” Information Fusion, vol.21, pp.145–158, Jan. 2015.
[5] “http://www.nationalisacs.org,” National Council of ISACs, 2018.