Accurate and Invertible Sketch for Super Spread Detection

Subjects: Statistics & Probability

Contributor: Zheng Zhang, Jie Lu, Quan Ren, Ziyong Li, Yuxiang Hu, Hongchang Chen

Super spread detection has been widely applied in network management, recommender systems, and cyberspace security. It is more complicated than heavy hitter owing to the requirement of duplicate removal. Accurately detecting a super spread in real-time with small memory demands remains a nontrivial yet challenging issue.

Keywords: super spread detection ; security ; sketch ; data stream

1. Introduction

Super spread detection in real-time is crucial for security monitoring, network measurement, and resource alignment [1][2][3]. In general, flow size and cardinality are two basic statistics of interest, from which a variety of traffic information can be extracted. The flow size is the number of elements (e.g., bytes, packets, and contents) in a flow, and flow cardinality is the number of distinct elements (e.g., distinct addresses and ports) in a flow. The most significant difference between flow size and flow cardinality is that estimating a flow spread demands the removal of duplicate elements; however, estimating the flow size does not. A flow refers to data traffic that has the same or similar characteristics. This research focuses on super spread identification (super spread is a flow with high cardinality), which can be widely used for search trend detection, recommender systems, anomaly detection, and DDoS detection [4][5][6][7].

Many studies have been proposed to find heavy flows with large sizes and have achieved significant progress [8][9][10][11]. However, finding super spreads is a more challenging problem because of the difficulty of deleting duplicates. Many useful per-flow estimators with different theoretical precisions, such as bitmap [12], Linear Counter (LC) [13], LogLog (LL) [14], Adapt Counter (AC) [15], and HyperLogLog (HLL) [16] have been designed for various situations and datasets. However, allocating an estimator for each flow is impractical because the required memory always exceeds the available memory.

It is challenging to find super spreads in data streams with limited memory. It is not possible to keep track of all flows accurately considering that a considerable amount of memory will be wasted by the recording of large-scale data streams. One strategy uses a sampling method to count a small part of the flow according to its actual cardinality. M2D is one typical algorithm [17]; however, sampling strategies lose accuracy. Another strategy used in the design of the above data structure is called estimator sharing [7][18][19][20][21][22], which uses one cardinality estimator for multiple flows. Sketch is a type of probabilistic data structure widely used in the field of network measurement to record the frequency or estimate the cardinality of elements in multiple sets or streams, and sketch is usually much smaller than the input size. Sketch-based measurement is a passive measurement, which usually does not send any detection packets and does not cause additional network overhead. Sketch uses profiles to effectively store and retrieve the information of interest, thereby achieving the recording of the presence and volume information of active flows. Then, the majority vote algorithm (MJRTY) [23] is used to find the maximum cardinality flow from the estimator and server as a possible super spread. However, MJRTY does not work well if the cardinality of a super spread is not significantly larger than the other flows.

2. An Accurate and Invertible Sketch for Super Spread Detection

In order to save memory, estimator sharing for multi-flow spread is widely adopted. Estimator sharing hashes each flow to d estimators, each of which produces a spread estimation for flow f independently. The smallest estimation carries the least error. There are two parts for one super spread: the value of cardinality and the flow label. Targeting the above two parts, existing approaches can be divided into two categories. Both of them have made certain progress in super spread detection; however, they also have deficiencies mainly in detection performance or resource expense (e.g., memory occupancy and detection accuracy). The distinct memory consumption of single estimator is presented in **Table 1**. The details are as follows.

Table 1. Memory costs for different estimators.

Estimator	Memory	Remark
Bitmap	**m**	**m = n, n is the real cardinality of flow**
Linear Counter	**m**	**mln (m) > n**
LogLog	**32 m**	**Recommended as m = 128**
Adaptive Counter	**32 m**	**Recommended as m = 128**
HyperLogLog	**5 m**	**Recommended as m = 128**

Some approaches encode the flow label information into a sketch and then enumerate the entire label space to recover the candidate super spread. FAST [24] proposes an efficient data structure, namely, the fast sketch, which maintains multiple arrays of HLL sketches. For each arriving item, it splits flow label f into two parts. One part has been hashed to a d HLL array, and in each array records the element in one HLL. In this way, it can not only polymerize packets into a small number of flows but also further enable ISPs to discriminate the anomalous keys.

CDS [25] proposes a new data structure for locating the hosts associated with high connection degrees or significant variations in connection degrees based on the reversible connection degree sketch. It constructs a compact summary of host connection degrees, realizing an efficient and accurate analysis, and reconstructs the host addresses associated with large fan-outs by a simple computation merely based on the characteristics of the Chinese Remainder Theorem.

Vector Bloom Filter [26], which is a variant of the standard bloom filter, improves the update efficiency significantly via bit-extraction hashing. It can extract bits directly from the source ID and obtain the information of super spreads by using the overlapping of hash bit strings.

The approaches given above can find and derive super spreads; however, the computational cost is too high to afford the recovery of the flow label because the enormous flow label space and inaccuracy as an estimator have to be shared by many flows.

Some other approaches separate the cardinality and the flow label using existing frequency-based sketches to return high-frequency keys and use another data structure to store the flow label of the candidate super spread. *cSkt* [27] extended the *Count-Min* sketch [28] with an external heap for tracking super spreads and associated each bucket with a distinct cardinality estimator, which is simple and easy to implement.

OpenSketch [29], which offloads part of the measurement function to the data plane from the control plane, combined reversible sketch [30] with bitmap algorithms. In the data plane, it provides a simple three-stage pipeline involving hashing, filtering, and counting to implement measurement tasks of cardinality and flow labels. In the control plane, it provides a measurement library to realize automatic configurations of the pipeline and resource allocation for different measurement tasks.

And Liu [7] et al. combined Fast Sketch [24] with an optimal distinct counter for super spread detection. It designs a reversible and mergeable data structure for a distributed network monitoring system, which means implementing network traffic measurements at each local monitor and reporting high-cardinality hosts productively based on compressed information, thus avoiding querying every single host in the network.

The approaches above can find and derive the super spreads; however, existing invertible frequency-based sketches, as proposed above, have heavy processing overhead: either incurring high memory access overhead for heap updates or inducing an unaffordable update overhead that grows linearly with the key size.

Among them, *gmf* [25], and *SpreadSketch* [20] are two of the state-of-the-art implementations. The core technologies introduced by *gmf* are a generalized geometric counter, a generalized geometric hash function, and an innovative geometric minimum filter that can eliminate duplication and block the vast majority of mice or small streams. Therefore, after the original flow passes through the filter, only a small number of flows are tracked using the hash table. In this way, *gmf* separates a super spread from the vast majority of small flows. This method greatly reduces memory usage, but cannot accurately measure the cardinality of super spread.

SpreadSketch can simultaneously measure the diffusion of a lot of traffic and distinguish the super spreads among them. It extends the *Count-Min* [28] while replacing each counter with multi-resolution bitmaps, a label field, and a register [22]. The label field is used to record a flow label. However, if the cardinality of streams mapped to the same bucket is very close, especially when memory is small, its detection is not accurate enough.

In conclusion, although the above methods have made some progress in super spread detection, they cannot meet the requirements of accuracy and performance for super spread detection at the same time, especially when the cardinality of a super spread is not significantly larger than other spreads and the available memory is small (less than 100 KB). Besides, the cardinality they provide is not accurate enough.

References

1. Tan, L.-Z.; Su, W.; Zhang, W.; Lv, J.; Zhang, Z.; Miao, J.; Liu, X.; Li, N. In-band network telemetry: A survey. Comput. Netw. 2021, 186, 107763.
2. Li, S.; Luo, L.; Guo, D. Sketch for Traffic Measurement: Design Optimization Application and Implementation. arXiv 2020, arXiv:2012.07214. Available online: https://arxiv.org/abs/2012.07214 (accessed on 5 January 2021).
3. Pendleton, M.; Garcia-Lebron, R.; Cho, J.-H.; Xu, S. A survey on systems security metrics. ACM Comput. Surv. 2017, 49, 62.
4. Cao, J.; Jin, Y.; Chen, A.; Bu, T.; Zhang, Z.-L. Identifying high cardinality internet hosts. In Proceedings of the IEEE International Conference on Computer Communications, Rio de Janeiro, Brazil, 19–25 April 2009; pp. 810–818.
5. Durumeric, Z.; Bailey, M.; Halderman, J.A. An Internet-Wide View of Internet-Wide Scanning. In Proceedings of the 23rd USENIX Security Symposium, San Diego, CA, USA, 20–22 August 2014; pp. 65–78. Available online: https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/durumeric (accessed on 10 May 2021).
6. Fayaz, S.K.; Tobioka, Y.; Sekar, V.; Bailey, M. Bohatei: Flexible and Elastic Ddos Defense. In Proceedings of the 24th USENIX Security Symposium, Washington, DC, USA, 12–14 August 2015; pp. 817–832. Available online: https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/fayaz (accessed on 20 January 2021).
7. Liu, Y.; Chen, W.; Guan, Y. Identifying high-cardinality hosts from network-wide traffic measurements. IEEE Trans. Dependable Secur. Comput. 2016, 13, 547–558.
8. Qun, H.; Lee, P.P.C.; Bao, Y. Sketchlearn: Relieving user burdens in approximate measurement with automated statistical inference. In Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication, Budapest, Hungary, 20–25 August 2018; pp. 576–590.
9. Lu, J.; Chen, H.; Sun, P.; Hu, T.; Zhang, Z. OrderSketch: An unbiased and fast sketch for frequency estimation of data streams. Comput. Netw. 2021, 201, 108563.
10. Liu, Z.; Manousis, A.; Vorsanger, G.; Sekar, V.; Braverman, V. One sketch to rule them all: Rethinking network flow monitoring with univmon. In Proceedings of the Annual Conference of the ACM Special Interest Group, Florianopolis, Brazil, 22–26 August 2016; pp. 101–114.
11. Yang, T.; Jiang, J.; Liu, P.; Huang, Q.; Gong, J.; Zhou, Y.; Miao, R.; Li, X.; Uhlig, S. Elastic sketch: Adaptive and fast network-wide measurements. In Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication, Budapest, Hungary, 20–25 August 2018; pp. 561–575.
12. Wu, K.; Otoo, E.J.; Shoshani, A. Optimizing bitmap indices with efficient compression. ACM Trans. Database Syst. 2006, 31, 1–38.
13. Whang, K.-Y.; Vander-Zanden, B.T.; Taylor, H.M. A linear-time probabilistic counting algorithm for database applications. ACM Trans. Database Syst. 1990, 15, 208–229.
14. Durand, M.; Flajolet, P. Loglog counting of large cardinalities. In Proceedings of the Algorithms-ESA 2003: 11th Annual European Symposium, Budapest, Hungary, 16–19 September 2003; pp. 16–19.
15. Cai, M.; Pan, J.; Kwok, Y.-K.; Hwang, K. Fast and accurate traffic matrix measurement using adaptive cardinality counting. In Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication Workshop on Mining NETWORK Data, Philadelphia, PA, USA, 26 August 2005; pp. 205–206.
16. Flajolet, P.; Fusy, É.; Gandouet, O.; Meunier, F. Hyperloglog: The Analysis of a Near-Optimal Cardinality Estimation Algorithm. In Proceedings of the Discrete Mathematics and Theoretical Computer Science, Nancy, France, 1 January

2007; pp. 127–146. Available online: https://algo.inria.fr/flajolet/Publications/FlFuGaMe07.pdf (accessed on 5 December 2020).

17. Zhang, Z.; Hsu, C.; Au, M.H.; Harn, L.; Cui, J.; Xia, Z.; Zhao, Z. PRLAP-IoD: A PUF-based Robust and Lightweight Authentication Protocol for Internet of Drones. Comput. Netw. 2024, 238, 110118.
18. Cormode, G.; Muthukrishnan, S. Space efficient mining of multigraph streams. In Proceedings of the Twenty-Fourth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, Baltimore, MD, USA, 13–15 June 2005; pp. 271–282.
19. Ma, C.; Chen, S.; Zhang, Y.; Xiao, Q.; Odegbile, O.O. Super spreader identification using geometric-min filter. IEEE/ACM Trans. Netw. 2022, 30, 299–312.
20. Tang, L.; Huang, Q.; Lee, P.P.C. SpreadSketch: Toward Invertible and Network-Wide Detection of Superspreaders. In Proceedings of the IEEE INFOCOM 2020-IEEE Conference on Computer Communications, Toronto, ON, Canada, 6–9 July 2020; pp. 1608–1617.
21. Venkataraman, S.; Song, D.X.; Gibbons, P.B.; Blum, A. New Streaming Algorithms for Fast Detection of Superspreaders. In Proceedings of the Network and Distributed System Security Symposium, San Diego, CA, USA, 2 February 2005; Available online: https://www.ndss-symposium.org/ndss2005/new-streaming-algorithms-fast-detection-superspreaders/ (accessed on 17 March 2021).
22. Estan, C.; Varghese, G.; Fisk, M. Bitmap algorithms for counting active flows on high speed links. IEEE/ACM Trans. Netw. 2003, 14, 925–937.
23. Boyer, R.S.; Moore, J.S. MJRTY: A Fast Majority Vote Algorithm. Autom. Reason. Essays Honor. Woody Bledsoe 1991, 1, 105–118.
24. Liu, Y.; Chen, W.; Guan, Y. A fast sketch for aggregate queries over high-speed network traffic. In Proceedings of the IEEE International Conference on Computer Communications, Orlando, FL, USA, 25–30 March 2012; pp. 2741–2745.
25. Wang, P.; Guan, X.; Qin, T.; Huang, Q. A data streaming method for monitoring host connection degrees of high-speed links. IEEE Trans. Inf. Forensics Secur. 2011, 6, 1086–1098.
26. Liu, W.; Qu, W.; Gong, J.; Li, K. Detection of superpoints using a vector bloom filter. IEEE Trans. Inf. Forensics Secur. 2016, 11, 514–527.
27. Zhou, Y.; Zhang, Y.; Ma, C.; Chen, S.; Odegbile, O. Generalized sketch families for network traffic measurement. ACM Meas. Anal. Comput. Syst. 2019, 3, 1–34.
28. Cormode, G.; Muthukrishnan, S. An improved data stream summary: The count-min sketch and its applications. J. Algorithms 2005, 55, 58–75.
29. Yu, M.; Jose, L.; Miao, R. Software Defined Traffic Measurement with OpenSketch. In Proceedings of the 10th USENIX Symposium on Networked Systems Design and Implementation, Lombard, IL, USA, 2–5 April 2013; pp. 29–42. Available online: https://www.usenix.org/conference/nsdi13/technical-sessions/presentation/yu (accessed on 15 January 2021).
30. Schweller, R.T.; Li, Z.; Chen, Y.; Gao, Y.; Gupta, A.; Zhang, Y.; Dinda, P.A.; Kao, M.-Y.; Memik, G. Reversible sketches: Enabling monitoring and analysis over high-speed data streams. IEEE/ACM Trans. Netw. 2007, 15, 1059–1072.