International Journal of Innovations & Advancement in Computer Science
IJIACS
ISSN 2347 – 8616
Volume 6, Issue 9
September 2017

Detecting Super Spreders using A Novel Data Streaming Method and Compare with Existing Technique

Mamtakhatri
M.E. Scholar
Y.T.I.E.T., Bhivpuri road, Karjat, Maharashtra, India.

Vaishali Londhe
Assistant Professor
Y.T.I.E.T., Bhivpuri road, Karjat, Maharashtra, India.

Gayatri Naik
Associate Professor
Y.T.I.E.T., Bhivpuri road, Karjat, Maharashtra, India.

ABSTRACT----The Superpoints are those sources (or destination) that compromised host which connect to great deal of destination (or source) during a measurement time interval, hence it is important to detect superpoints in real time for network security and management. The available algorithms are insufficient to control the usage of the memory and deliver the desired amount of accuracy, so it is difficult to detect the superpoints on high speed link in real time. In this paper we propose a novel method, Vector Bloom Algorithm for detection of superpoints in real time distributed network. This algorithm makes use of six hash functions out of which, four take few consecutive from given input string as corresponding values respectively. The detection of superpoints is done by using overlapping hash bit strings of VBF and to improve the measurement accuracy of the superpoints. VBF is different data streaming method for detecting superpoints and guarantees its accuracy with low memory requirement. Theoretical analysis of the proposed system method can detect superpoints precisely and efficiently through comparison with Counting Bloom Filter Algorithm. Here we plan to compare the space of VBF algorithm with CBF as both are used to represent same dynamic set of data. The main goal of proposed system is restrict memory requirement high measurement accuracy and false positive probability as compared to CBF also to improve the complexity of system.

Keywords---IP flow, superpoint, bloom filter, cardinality estimation, random variable.

I.INTRODUCTION

Internet Service Provider (ISP) frequently collect traffic measurements for many purpose viz. customer counting measurements and traffic engineering, it is also used for traffic anomaly and cyber-attack attribution, network forensic analysis. Some networks internet are infected with various malicious activities such as DDos spam and so on. The important traffic feature which is considered for our interest are high cardinality. The number of distinct destination contacted by a host is called cardinality of host.[1].Therefore a superpoint(source) is a host that has a high cardinality as compared to cardinality of normal host.

Various wormthat have been persistent on internet may be used to perform attack like DDOs or exhaust all the system resource. A compromised host does fast scanning for worm propagation often it makes higher number connection in short span of time. These compromised host are called superpoints.

II. PROBLEM DEFINATION

It is important to study the problem of detection of superspreaders because it concentrates on some of objectives.

- The first goal is that we need to concentrate on is detection of these superpoints requires the network-wide view. The attack may intrude the network from multiple routes. If only cardinality of single router is monitored attacker may be issued at any of them, therefore it is important to merge the traffic measurement from multiple routers and get a network view of host cardinality [2].
- The next that should be considered is that packets coming from the same be removed for cardinality computation when the traffic measurements are merged, the cardinality of the host can't be simply added together to calculate the total cardinality since each connection may travel from source to destination through multiple routers from source to destination a design method should only count distinct connection
- Thirdly due to huge traffic measurement ISP collect only the summaries of traffics measurements from local routers, hence our goal is to design such a mergeable data structure to reduce the communication cost.

➢ Lastly it is possible to compute the cardinality for each and every host to identify the superpoints. Due to large the large size of IP addresses we wish to only estimate the cardinality of only those host with superpoint using limited space and time for execution.

The above mentioned challenge are partially achieved in previous work and there had been no algorithms which could solve all the above challenges in the best of our knowledge .In our proposed project we design a data streaming algorithm to compute mergeable and reversible sketch, which can identify superpoints in network-wide traffic measurements. Our data structure summarizing traffic measurements is designed in such a way that it is based on the noise group testing, which can be used to identify an high cardinality host in an efficient manner in distributed network monitoring system.

Our main idea is to consider identification of superpoint host as channel coding problem which also provides a new theoretical analysis for the problem.

III. EXSTING WORK FOR DETECTION OF SUPERPOINTS

There exist a lots of work on detecting the high cardinality host, some which are mentioned below.

- *Flow Sampling*

In 2005Venkatraman implemented two algorithms for detection of superspreaders, first was one level filtering algorithm for identifying high cardinality host over an interval of time. Second was two level filtering algorithm which was comparatively more efficient than prior, it kept proper digest to identify superpoints.In 2005 Qi. Proposed two joint data streaming and sampling algorithm for detection of super sources and super destination. The simple algorithm uses a standard hash based flow sampling algorithm a bitmap structure to maintain the flow record but it stores the measured IP records in the memory directly, and all the flows are sampled by the same sampling rate, as a result , the estimated accuracy of superpoints is decreased. The sampling algorithm comparatively reduced the amount of memory used and data that needs to be processed so it could be considered as efficient and scalable for collecting network-wide traffic measurements. However a drawback has that accuracy is limited by low sampling rate in very high speed robust networks. This is overcome by our algorithm VBF.

- *Bitmap Method*

Coa. Designed online sampling approach for identifying high cardinality consisting of two phase filtering threshold bitmap and bias correction. Kyu-Young Wang designed an algorithm for counting the number of unique values along with the presence of duplicates, an traditionally the counting was through soring whereas this uses linear counting, which is based on hashing. Cristian Eastan presented a bitmap algorithm that concentrates on the problem of distinct values (flows) seen on high speed link which may be used to detect DDOS attacks and port scan. Yoon et. al. designed a method for storage of data called virtual vectors, to compute cardinality of source. The time consumption for storage and look up for distinct values is a serious issue, since VBF does not store IDs explicitly, the VBF is exempt from the consumption.

- *Reversible Sketches*

Wellamproposed an algorithm to detect superspreaders based on combination of three techniques FM sketches, Bloom filter and hash table. The main idea that proposed scheme is to first filter out the source with low fan- out and count only fan-out of potential superspreader. Wang developed a new reverse sketch method for detecting host with sketches build a compact summary of connection degree of host efficiently and accuracy. Minlan Yu. designed a software that could defined traffic measurement architecture open sketch provide a three stage pipeline(hashing, filtering, counting) which is implemented with commodity switch component and support many measurement task. T. Wellam and Lia had put forth the sketch the sketch guided filtering scheme for detection of superspreaders in the measurement of high speed network traffic. Thismethod consist of linear counting sketches that rapidly eliminate records (flow) with potentially low fan out during a measurement interval.

There are various drawbacks of existing system.

- It occupies the large storage because each standard bloomer filter maintain look up table for storing the IP address.
- It does not maintain the main false positive i.e. it increases the false negative.
- It does not maintain the accuracy of the superpoint.
- It takes more time for merging conjunctive bit stream i.e. it requires more time for generating IP address.

IV. PROPOSED SYSTEM

The following is system architecture of the proposed system which describes VBF with its comparison with Counting Bloomer Filter.

International Journal of Innovations & Advancement in Computer Science
IJIACS
ISSN 2347 – 8616
Volume 6, Issue 9
September 2017

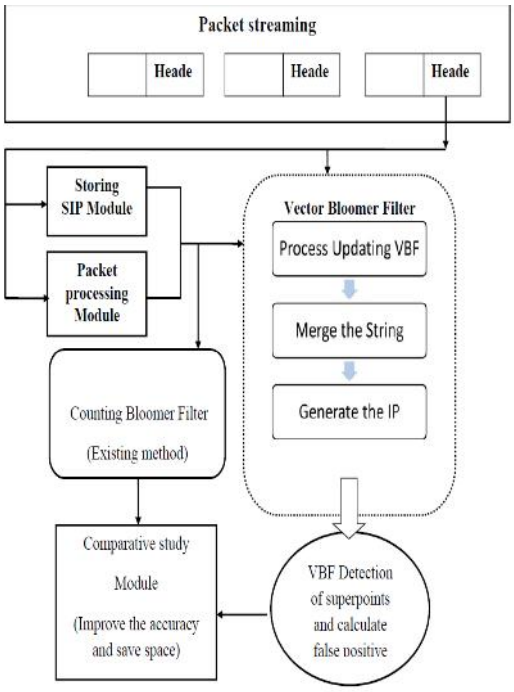

Fig 1: The Proposed System Architecture.

The modules in architecture are as explained below:

- *Packet Streaming Module:*

In this module deals with the packet is the information which consists header field. Every header node represents the details information of source IP or Destination IP.

- *Storing SIP Module:*

It maintains the records of IP address of each Host. It also maintain the information of the each packet header and it provide the updateunit of the VBF for reconstructing the IP address.

- *Packet Processing Module: (Existing Process Module)*

This module consists of three arrays and four hash functions. When a packet comes, we extract the sip and dip of the packet. Its sip is hashed by Hi(sip) to generate three row indexes x1, x2, x3, and the dip is hashed by D(dip) to generate a column index y. Then we set Bi[xi][y], i= 1, 2, 3 to be 1.

- *Vector Bloom Filter:*

This is a different variant of the existing bloom filter, it consist of (K+L+1) hash functions. In the mentioned hash function the first K hash functions take consecutive bits from the given input string as function values, which are used to reconstruct host addresses. The middle L hash function can hash source IPs into value interval uniformly are used to addresses filter some host candidates out. The last function is used to develop column index of the bits for storing the flow that is arriving.

The vector can be explained as that each entry in VBF is not single bit but it is in the form of bit vector. Rather than having one array of size n which would be shared by (K+L) hash function, each of the hash function has n consecutive vectors not overlapping with other. In our work K and L are set of 4and 1 respectively. The meaning of each hash function is explained in the table below

Function Name	Output
h1	First byte, high 4 bits of second byte of SIP.
h2	Second byte, high 4bits of third byte of SIP.
h3	Third byte, high 4 bits of fourth byte of SIP.
h4	Fourth byte, high 4bits of first byte of SIP
h5	Generate uniform output of 12 bits string
F	Generate uniform output over (0,..,m-1)

Table 1: Hash Function of VBF

International Journal of Innovations & Advancement in Computer Science
IJIACS
ISSN 2347 – 8616
Volume 6, Issue 9
September 2017

The functions *hi* $(1 \le i \le 5)$: {0, 1, · · ·, 232 − 1} →

{0, 1, · · ·,212 − 1} use SIPs as function input. The function f : {0, 1, · · ·, 264 −1} → {0, 1, · · ·,m −1} maps flow labels into bitmaps with size m. Since the functions $h1$, $h2$, $h3$ and $h4$ only select some consecutive bits as their output, they can be evaluated quickly. Ideally, $h5$ and f have the following desired properties:

(i) Speed: the hash function must be computed quickly because it has to be applied to each arrival packet.

(ii) Uniformity: the output must be sufficiently uniform on the hash range, A_1 ıe function behaves like a truly random function.

The properties described are also the properties of the hash function which we use for packet selection. In general the more complex are the hash function more are the uniform generated output, although it required computation due to its complexity. The specific complexity & ease of implementation.

Algorithm for updating

- ***Initialize***
- ***Array $A_i[j][k]$=0,1,…5 , j=0…2^{12}***
- ***K= 0,…m-1.***
- ***Update vector on arrival of packet(S_x,D_x)***
- ***Col=f(S_x,D_x)***
- ***for i from1 to 5***
- ***row=h_i(S_x)***
- ***A_i[row][col] is set to 1.***

Let us assume that IP address u be $S_i(1 \le i \le 8)$ be superpoint identifier is 4bit string and IP address is 32 bit string. Now $h_1(u)$ consist of bits $S_1S_2S_3$ $h_2(u)$ is$S_3S_4S_5$ $h_3(u)$ consist of bits $S_5S_6S_7$.Finally $h_4(u)$ consist of bits S_7SS_1 Here last bit of prioris overlapping with next. The data structure used in our system is denoted as A=(A_1,A_2,A_3,A_4,A_5). For A($1 \le i \le 5$) consist of 4096xm bit array. Initially all the bits are sets to 0, when a packet $p_x(S_xD_x)$ where S_x is source IP and D_x is destination IP arrives each A_i is updates by making its row $h_i(S_x)$ and col $f(S_x,D_x)$ as 1 as diagrammatically represented in fig

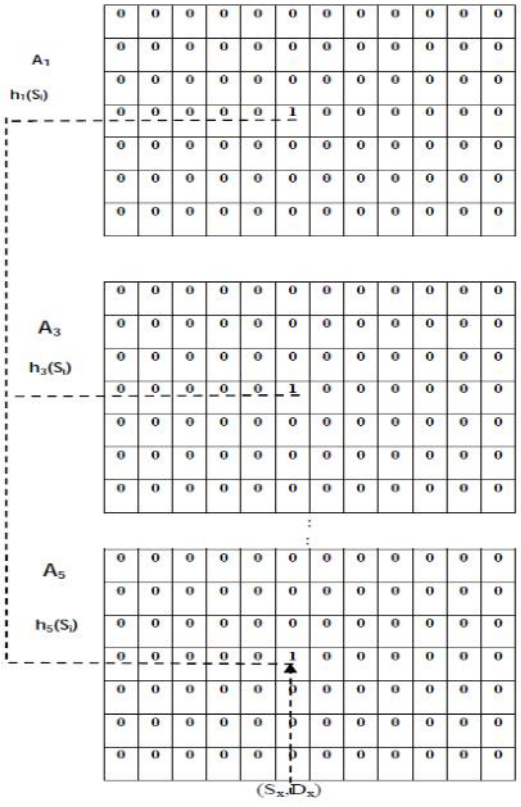

Fig2:Updating inVBF

International Journal of Innovations & Advancement in Computer Science
IJIACS
ISSN 2347 – 8616
Volume 6, Issue 9
September 2017

For every $p_x(S_x,D_x)$ define hash function such that $h_i(S_x,D_x)=(h^i(S_x),f_x(S_x,D_x))$ where $(1\leq i\leq 5)$.The VBF consist of hash function h_1^+S. For every packet arriving five bits $Ai[h^+(S_x,D_x)]$ $(1\leq i\leq 5)$ are set to high or 1. When a source IP considered as process unit, in each array. There is a vector corresponding to source IP. Consequently this is also called VBF.

Notation	*Meaning*
*m**	A give threshold of being superpoint
Z^*	Threshold for the number of zero bits
Hi	i-th row function of VBF
F	Column function of VBF
m	Size of row vector of VBF that is the size of column of two dimension array
A_i	4096xm bit array associated with h_i
H_i	Set of $(u,V[u])\leq Z^*$in $A_i[u]$
$C(A_i[j])$	Number of zero bits in row vector $A_i[j]$
C(V[w])	Number zero bits in vector V[w]
▪	The Noise
s	Number of distinct flows in measurement period
U_mV_m	Number and fraction of zero bits in vectorV[w]
U_*V_*	Number and fraction of zero bits in A_5, respectively
A	s/4096 load factor of A_5
r	$(1/\alpha)\ln(2-e^{-\alpha})$
B	$(k/m)+(1-r)\alpha$

Algorithm For Merging A String

MergeStings(Input1, Input2, Output)

- ***Output ← ϕ***
- ***For each (u,V[u]) belongs to Input 1 do***
- ***For each(w, V[w]) belong to Input2 do***
- ***If (tail[u] is equal to head[w]) then***
- ***v=merge1(u,w)***
- ***V[v]=V[u] and V[w]***
- ***If C(V[v])<= Z^* then***
- A_3***Output ← Output ∪ (v,V[v])***

According to cardinality threshold m^*, zero bit number can be given threshold Z^*.The point is how to obtain Z^*according to m^*, this will be described in forth coming section. The measurement proceeds in interval. Each interval may be a value between 25 sec to 350 sec. In practical application, the period may be selected as per the objective of measurement.

International Journal of Innovations & Advancement in Computer Science
IJIACS
ISSN 2347 – 8616
Volume 6, Issue 9
September 2017

Consider an example in order to detect a high cardinality host that is scanning at high speed, The interval can be set to small convenient value ,otherwise relatively high value is chosen. At end of each measuring interval, the hash strings whose corresponding vectors contain zero bit number less than or equal to Z^* are selected. Let the collected string be denoted by (u, V[u]), where u is a string of 12 bits and V[u] is bits $A_i[u]$ of row u. Thus set H_1, H_2, H_3, H_4, H_5 are formed which composed of (u, V[u]) from the five different hash spaces respectively.

Now, we introduce the string functions: tail, head, merge1and merge2. For a string w, tail(w) chooses low 4 bits of w and head(w) gets high 4 bits of w. For example,if w = s1s2s3s4, then tail(w) = s4, and head(w) = s1. Functions merge1 and merge2 may merge two overlapped strings into one. For example, for two strings abcand cde, a new string merge1(abc, cde) = abcdecan be obtained. Similarly, for two strings acdefgband bha, an IP address merge2(acdefgb, bha) = acdefgbhmay be generated.

Algorithm to generate IP

GenIP(Input1,Input2, Output)

- ***Output ← ϕ***
- ***For each (u,V[u]) belongs to Input 1 do***
- ***For each(w, V[w]) belong to Input2 do***
- ***If(tail(u)equals head(w)&& tail(w) head(u))then***
- ***v=merge2(u,w)***
- ***V[u]=V[u]&V[w]//bitwise and***
- ***ifC(V[v]≤ Z^*) then***
- ***Output ←OutputU (v,V[v])***

The process of merging strings is described in Algorithm 2.

A basic idea is that two overlapped strings are merged into one.

Algorithm To Detect Superpoints

DetectSuperpoints

- ***Mergence***
- ***Mergestrings(H_1,H_2,H_{12})***
- ***Mergestrings(H_{12},H_3,H_{123})***
- ***GenIP(H_{123},,H4,IP)***
- ***Filtering***
- ***Output ←ϕ***
- ***For each(v,V[v]belongs to IP)do***
- ***V[v]=V[v] & $A_5[h_5(v)]$***
- ***If C(V[v])≤Z^* then***
- ***Output ←OutputU (v,V[v])***
- ***Estimation***
- ***For each(u,V[u]) belongs to Output do***
- ***u is identified as superpoint with cardinality***

Cardinality(u)=$\ln \frac{m}{C(V[u])}$.

International Journal of Innovations & Advancement in Computer Science
IJIACS
ISSN 2347 – 8616
Volume 6, Issue 9
September 2017

Mergestrings algorithm shows that two strings may be merged into one if tail head overlapped and the counter of zero bits of the corresponding vector obtained from bit and operation is no more than Z^*.

The process of generating IP addresses is given by algorithm GenIP. According to GenIP, some IP addresses (superpoint ID's) can be produced. However cases in which superpoint ID's that do not exist in network may be merged by the given algorithms. These falsely merged superpoint are called phantoms. Finally Detectsuperpoint summarizes the detected process of detecting high cardinality host, the concept to compute▪ is also defined in Detectsuperpoint is also explained in next subsection.

In this paper, for the ease of description, the output of *h*5 is fixed at 12 bits long. In fact, the output may be longer or shorter. Certainly, δ needs to be recomputed in a similar way. Consequently, δ decreases as the output becomes longer and δ increases as the output becomes shorter. In general, the longer output requires more memory, but it also leads to higher accuracy.

It is important to determine the optimal value of m for function f. The choice of *m* is based on the following factors in the measurement period: the number of total flows, the number of superpoints, the cardinality range of superpoints, the requirement of measurement accuracy, and the available memory. The measurement accuracy of VBF is improved when *m* increases; nevertheless, the time and space complexity of VBF increases. We recommend to select a value for *m* such that α is in the interval [0.02, 0.2], where $\alpha = s/4096m$ is the load factor of *A*5. Naturally, the number *s* of flows needs to be estimated by rule of thumb in advance.

- *Counter Bloomer Filter:*

As mentioned with the treatment on standard Bloom filters, they do not support element deletions. A Bloom filter can easily be extended to support deletions by adding a counter for each element of the data structure. Probabilistic counting structures have been investigated in the context of database systems. A counting Bloom filter has *m* counters along with the *m* bits. first introduced the idea of a counting Bloom filter in conjunction with Web caches. A counting Bloom filter also has the ability to keep approximate counts of items. For example, inserting element *x* three times results in the *k* bit positions being set, and the associated counters incremented by one for each insert. Therefore, the *k* counters associated with element *x* are incremented at least three times, some of them more if there are overlaps with other inserted elements. The count estimate can be determined by finding the minimum of the counts in all locations where an item is hashed to.

The designed system is implemented to detect the path from source to destination. This means the system shows how the message packets travel from source to destination. The system accepts the destination IP similar to email services at the source along with the message packets that are to be transferred. The system demonstrates two ways of tracing the path that message traversed from source to destination viz. static approach and dynamic approach.

- *Static Approach*

In the staticmethod as soon as destination IP is entered to trace the path the suitable path to reach to destination and generates the signature which is to be verified at the destination for authenticity. If authenticated then message is received at destination.

- *Dynamic Approach*

In dynamic approach it detects all possible routes to send the message then tries to traverse ,here it discards the path where even if the single node busy. System finally selects a path where node are readily available and generates signature which is to be verified at the destination for authenticity. If authenticated then message is received at destination.

The system is analysed on various parameters. Let's consider each one of this in brief .

➢ *Speed Of Transferring Packets*

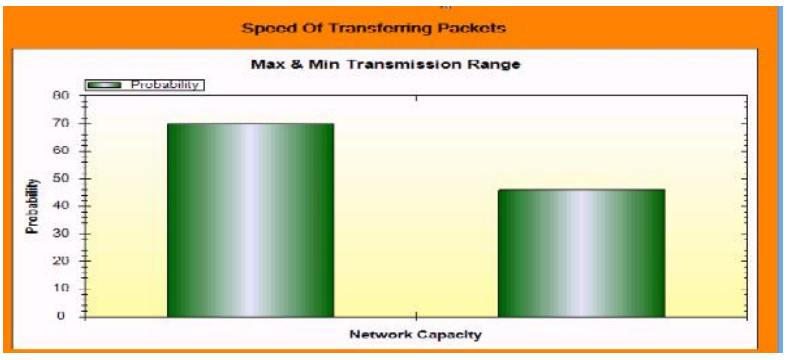

Fig:3*Speed Of Transferring Packets*

International Journal of Innovations & Advancement in Computer Science
IJIACS
ISSN 2347 – 8616
Volume 6, Issue 9
September 2017

The speed at which the data is transferred from on device to another is speed of transferring packet. In our system the above graph shows maximum and minimum transmission range of transferring packets which is taken with network capacity vs. probability. As shown in graph maximum probability is around 70 whereas minimum is between 40-45.

- *Pulse Width Of A Signal*

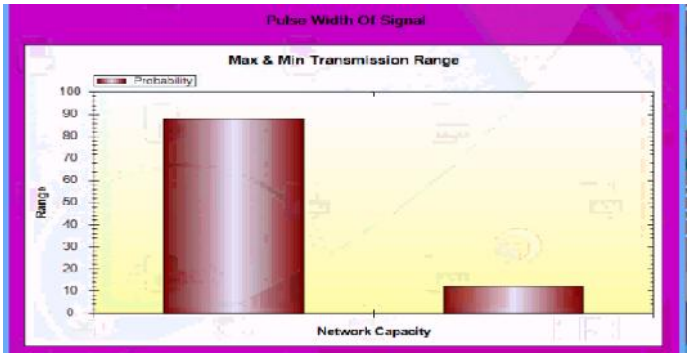

Fig4: Pulse Width Of Signal

Pulse Width (PW) is the elapsed time between the rising and falling edges of a single pulse .The graph shows pulse width in reference to maximum and minimum transmission range. The range of transmission of our system is represented as 10 ≤ range ≤ 90 in comparison to network capacity.

- *Network Capacity*

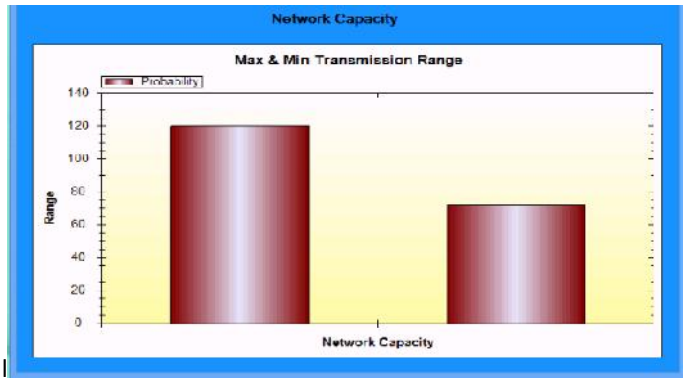

Fig5:Network Capacity

In the context of networks, capacity is the complex measurement of the maximum amount of data that may be transferred between network locations over a link ornetwork path. The range of data transfer over the location here is in the range of 60-120. This means the maximum probability is around 115-120.On the other hand lower end probability is around50-60.

- *Packet Delivery Ratio*

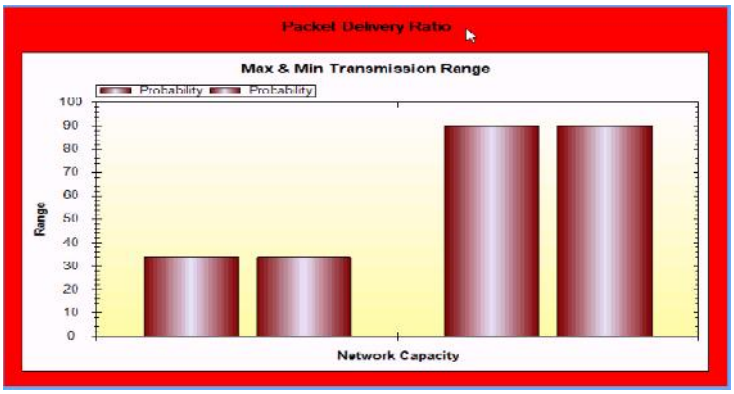

Fig6:Packet Delivery Ratio

International Journal of Innovations & Advancement in Computer Science
IJIACS
ISSN 2347 – 8616
Volume 6, Issue 9
September 2017

Packetdelivery ratio is defined as the ratio of data packets received by the destinations to those generated by the sources. The minimum packets that are send by source and received at destination are in range of 35-40 whereas maximum packets that are send by source and received at destination are around 90.

- *Signal Strength*

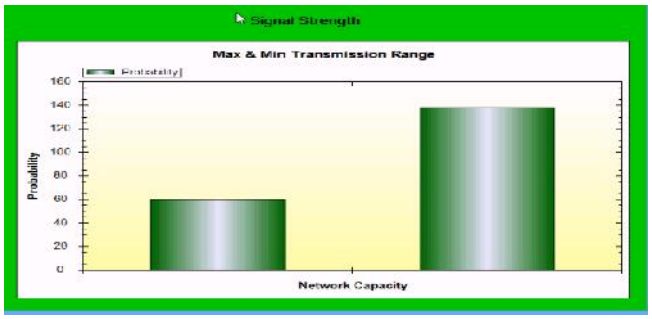

Fig7:Signal Strength

Signal strength (also referred to as field strength) refers to the transmitter power output as received by destination at a distance from the source. The signal strength of system is between 40 ≤ range ≤ 140.

- *Signal To Inference Ratio*

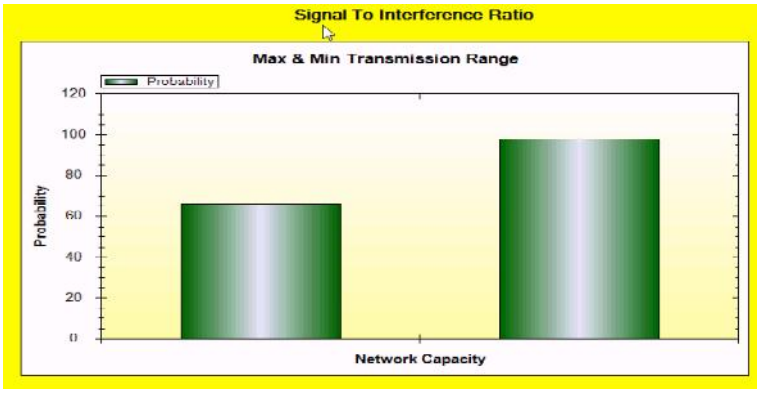

Fig8:Signal To Interference Ratio

The ratio of the magnitude of a given parameter of the desired signal, to that of the same parameter for any interference present. It is usually the ratio of their respective amplitudes, and is expressed in decibels. Its abbreviation is SIR. The SIR is in range of 60-100 units.

Comparison of Vector Bloom Filter to Reversible Counter Bloomer Filter

A reversible counting Bloom filter (RCBF) is a special counting Bloom filter which consists of *k* hash functions which protectively select some consecutive bits from original strings as function values. The reversible means that string hashed can be recovered whereas the VBF consists of six hash functions, four of which take some consecutive bits from the input string as the corresponding value, respectively. The information of superpoints is obtained by using the overlapping of hash bit strings of the VBF.

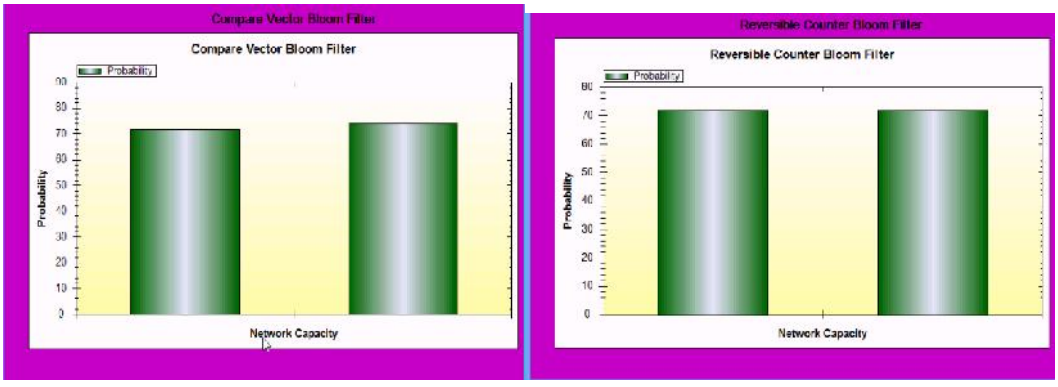

Fig9:Comparison OF VBF And RCBF

V. CONCLUSION

In this work, we have constructed a Vector Bloom

Filter (VBF) and designed the corresponding algorithm to

Identify superpoints. For each arriving packet, the VBF only needs to set several bits to 1. The computation complexity of the hash functions is low because they only take some consecutive bits as the output. The theoretical analysis shows that false positive ratio originated from phantoms is very low. The experimental results of the system shows the transfer of packets from source to destination also, compares the VBF with RBCF special type of CBF.

VI. REFERENCES

Paper Published By Me:

Khatri, Mamta R, et al. “Processing A Packet Data Streaming For Detecting Superspreders Using A Novel Data Streaming Method.” *INTERNATIONAL JOURNAL FOR RESEARCH & DEVELOPMENT IN TECHNOLOGY*, vol. 7, no. 3, 31 Mar. 2017, pp. 467–474.

1. Liu, Weijiang, WenyuQu, Jian Gong, and Keqiu Li. "Detection of Superpoints Using a Vector Bloom Filter." *IEEE Transactions on Information Forensics and Security* 11.3 (2016): 514-27.
2. Liu, Yang, Wenji Chen, and Yong Guan. "Identifying High-Cardinality Hosts from Network-Wide Traffic Measurements." *IEEE Transactions on Dependable and Secure Computing* 13.5 (2016): 547-58.
3. Guan, Xiaohong, Pinghui Wang, and Tao Qin. "A New Data Streaming Method for Locating Hosts with Large Connection Degree." *GLOBECOM 2009 - 2009 IEEE Global Telecommunications Conference* (2009): n
4. K. Xu, F. Wang, and L. Gu, “Behavior analysis of Internet traffic via bipartite graphs and one-mode projections,” *IEEE/ACM Trans. Netw.*, vol. 22, no. 3, pp. 931–942, Jun. 2014.
5. T. Li, S. Chen, and Y. Ling, “Per-flow traffic measurement through randomized counter sharing,” *IEEE/ACM Trans. Netw.*, vol. 20, no. 5, pp. 1622–1634, Oct. 2012.
6. J. Zhang, C. Chen, Y. Xiang, W. Zhou, and Y. Xiang, “Internet traffic classification by aggregating correlated naive Bayes predictions,” *IEEETrans. Inf. Forensics Security*, vol. 8, no. 1, pp. 5–15, Jan. 2013.
7. J. Zhang, X. Chen, Y. Xiang, W. Zhou, and J. Wu, “Robust network traffic classification,” *IEEE/ACM Trans. Netw.*, vol. 23, no. 3,pp. 1257–1270, Aug. 2015.
8. C. Hu, B. Liu, S. Wang, J. Tian, Y. Cheng, and Y. Chen, “ANLS: Adaptivenon-linear sampling method for accurate flow size measurement,”*IEEE Trans. Commun.*, vol. 60, no. 3, pp. 789–798, Mar. 2012.
9. F. Khan, N. Hosein, S. Ghiasi, C.-N. Chuah, and P. Sharma,“Streaming solutions for fine-grained network traffic measurementsand analysis,” *IEEE/ACM Trans. Netw.*, vol. 22, no. 2, pp. 377–390,Apr. 2014.
10. P. Wang, X. Guan, J. Zhao, J. Tao, and T. Qin, “A new sketch methodfor measuring host connection degree distribution,” *IEEE Trans. Inf.Forensics Security*, vol. 9, no. 6, pp. 948–960, Jun. 2014.
11. M. Rezvani, V. Sekulic, A. Ignjatovic, E. Bertino, and S. Jha, “Interdependen
12. tsecurity risk analysis of hosts and flows,” *IEEE Trans. Inf.Forensics Security*, vol. 10, no. 11, pp. 2325–2339, Nov. 2015.
13. W. Chen, Y. Liu, and Y. Guan, “Cardinality change-based early detectionof large-scale cyber-attacks,” in *Proc. IEEE INFOCOM*, Apr. 2013,pp. 1788–1796.
14. C. A. Shue, A. J. Kalafut, and M. Gupta, “Abnormally maliciousautonomous systems and their Internet connectivity,” *IEEE/ACM Trans.Netw.*, vol. 20, no. 1, pp. 220–230, Feb. 2012.
15. Y. Wang, S. Wen, Y. Xiang, and W. Zhou, “Modeling the propagation ofworms in networks: A survey,” *IEEE Commun. Surveys Tuts.*, vol. 16,no. 2, pp. 942–960, May 2014.
16. M. Egele, T. Scholte, E. Kirda, and C. Kruegel, “A survey on automateddynamic malware-analysis techniques and tools,” *ACM Comput. Surv.*,vol. 44, no. 2, Feb. 2012, Art. ID 6.
17. D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, andN. Weaver, “Inside the Slammer worm,” *IEEE Security Privacy*, vol. 1,no. 4, pp. 33–39, Jul./Aug. 2003.
18. D. Plonka, “FlowScan: A network traffic flow reporting and visualizationtool,” in *Proc. USENIX LISA*, Dec. 2000, pp. 305–318.
19. N. Kamiyama, T. Mori, and R. Kawahara, “Simple and adaptive identificationof superspreaders by flow sampling,” in *Proc. IEEE INFOCOM*,May 2007, pp. 2481–2485.
20. G. Cheng, J. Gong, W. Ding, H. Wu, and S. Qiang, “Adaptive samplingalgorithm for detection of superpoints,” *Sci. China F, Inf. Sci.*, vol. 51,no. 11, pp. 1804–1821, Nov. 2008.
21. M. Yoon, T. Li, S. Chen, and J.-K. Peir, “Fit a compact spread estimatorin small high-speed memory,” *IEEE/ACM Trans. Netw.*, vol. 19, no. 5,pp. 1253–1264, Oct. 2011.
22. P. Wang, X. Guan, W. Gong, and D. Towsley, “A new virtual indexingmethod for measuring host connection degrees,” in *Proc. IEEEINFOCOM*, Apr. 2011, pp. 156–160.
23. X. Guan, P. Wang, and T. Qin, “A new data streaming method for locatinghosts with large connection degree,” in *Proc. IEEE GLOBECOM*,Nov./Dec. 2009, pp. 1–6.

24. W. Liu, W. Qu, G. Jian, and L. Keqiu, "A novel data streaming methoddetecting superpoints," in *Proc. IEEE INFOCOM WKSHPS*, Apr. 2011,pp. 1042–1047.
25. S. Venkataraman, D. Song, P. B. Gibbons, and A. Blum, "New streamingalgorithms for fast detection of superspreaders," in *Proc. NDSS*,Feb. 2005, pp. 149–166.
26. J. Cao, Y. Jin, A. Chen, T. Bu, and Z.-L. Zhang, "Identifying highcardinality Internet hosts," in *Proc. IEEE INFOCOM*, Apr. 2009,pp. 810–818.
27. Q. Zhao, A. Kumar, and J. Xu, "Joint data streaming and samplingtechniques for detection of super sources and destinations," in *Proc.ACM SIGCOMM IMC*, Oct. 2005, p. 7.
28. T. Li, S. Chen, W. Luo, M. Zhang, and Y. Qiao, "Spreader classificationbased on optimal dynamic bit sharing," *IEEE/ACM Trans. Netw.*, vol. 21,no. 3, pp. 817–830, Jun. 2013.
29. K.-Y. Whang, B. T. Vander-Zanden, and H. M. Taylor, "A linear-timeprobabilistic counting algorithm for database applications," *ACM Trans.Database Syst.*, vol. 15, no. 2, pp. 208–229, Jun. 1990.LIU *et al.*: DETECTION OF SUPERPOINTS USING A VBF 527
30. C. Estan, G. Varghese, and M. Fisk, "Bitmap algorithms for countingactive flows on high-speed links," *IEEE/ACM Trans. Netw.*, vol. 14,no. 5, pp. 925–937, Oct. 2006.
31. M. Yoon and S. Chen, "Detecting stealthy spreaders by randomaging streaming filters," *IEICE Trans. Commun.*, vols. E94-B, no. 8,pp. 2274–2281, Aug. 2011.
32. G. Cheng and Y. Tang, "Line speed accurate superspreader identificationusing dynamic error compensation," *Comput. Commun.*, vol. 36, no. 13,pp. 1460–1470, Jul. 2013.
33. Q. Xiao, Y. Qiao, M. Zhen, and S. Chen, "Estimating the persistentspreads in high-speed networks," in *Proc. IEEE 22nd ICNP*, Oct. 2014,pp. 131–142.
34. T. Wellem, G.-W. Li, and Y.-K. Lai, "Superspreader detection systemon NetFPGA platform," in *Proc. ANCS*, Oct. 2014, pp. 247–248.
35. R. Schweller*et al.*, "Reversible sketches: Enabling monitoring andanalysis over high-speed data streams," *IEEE/ACM Trans. Netw.*, vol. 15,no. 5, pp. 1059–1072, Oct. 2007.
36. P. Wang, X. Guan, T. Qin, and Q. Huang, "A data streaming method formonitoring host connection degrees of high-speed links," *IEEE Trans.Inf. Forensics Security*, vol. 6, no. 3, pp. 1086–1098, Sep. 2011.
37. P. Wang, X. Guan, D. Towsley, and J. Tao, "Virtual indexing based methodsfor estimating node connection degrees," *Comput. Netw.*, vol. 56,no. 12, pp. 2773–2787, Aug. 2012.
38. T. Qin, X. Guan, W. Li, P. Wang, and M. Zhu, "A new connection degreecalculation and measurement method for large scale network monitoring,"*J. Netw. Comput. Appl.*, vol. 41, no. 1, pp. 15–26, May 2014.
39. Y. Liu, W. Chen, and Y. Guan, "Identifying high-cardinality hosts fromnetwork-wide traffic measurements," in *Proc. IEEE Conf. Commun.Netw. Secur.*, Oct. 2013, pp. 287–295.
40. W. Liu, W. Qu, X. He, and Z. Liu, "Detecting superpoints through a reversible counting bloom filter," *J. Supercomput.*, vol. 63, no. 1,pp. 218–234, Jan. 2013.
41. T. Zseby, M. Molina, N. Duffild, S. Niccolini, and F. Raspall, *Samplingand Filtering Techniques for IP Packet Selection*, document RFC 5475,Mar. 2009.
42. A. Pagh and R. Pagh, "Uniform hashing in constant time and optimalspace," *SIAM J. Comput.*, vol. 38, no. 1, pp. 85–96, 2008.
43. K.-M. Chung, M. Mitzenmacher, and S. Vadhan, "Why simple hashfunctions work: Exploiting the entropy in a data stream," *TheoryComput.*, vol. 9, no. 30, pp. 897–945, Dec. 2013.
44. M. Molina, S. Niccolini, and N. Duffield, "A comparative experimentalstudy of hash functions applied to packet sampling," in *Proc. ITC*,Beijing, China, Aug. 2005, pp. 1–10.
45. G. Cheng, W. Zhao, and J. Gong, "XOR hashing algorithms to measuredflows at the high-speed link," in *Proc. Int. Conf. Future Generat.Commun. Netw.*, Dec. 2008, pp. 153–155.
46. C. J. Martinez, D. K. Pandya, and W.-M. Lin, "On designing fast nonuniformlydistributed IP address lookup hashing algorithms," *IEEE/ACMTrans. Netw.*, vol. 17, no. 6, pp. 1916–1925, Dec. 2009.
47. Cypress Semiconductor Corporation. *Memory Products*. [Online]. Available:http://www.cypress.com/, accessed Nov. 2015.
48. WIDE. *MAWI Working Group Traffic Archive*. [Online]Available:http://tracer.csl.sony.co.jp/mawi/, accessed Nov. 2015.
49. JSLAB. *IP Trace Distribution System*. [Online]. Available:http://iptas.edu.cn/src/system.php, accessed Nov. 2015.
50. NLANR. *Passive Measurement and Analysis*. ftp://wits.cs.waikato.ac.nz/pma/long/ipls/3/, acces
51. Liu, W., Qu, W., He, X., & Liu, Z. (2010). Detecting superpoints through a reversible counting Bloom filter. The Journal of Supercomputing, 63(1),218-234. doi:10.1007/s11227-010-0511-2