Survey On: Denial-of-Service Attacks with Bloom-Filters

Akshay Vinayak Parte, Anant Sunil Sonsale, Mayur Jagannath Dhole, Sharmila A. Chopade
Computer Engineering,
DYPIET,
Ambi, Pune, India

***Abstract*- Now a days, World Wide Web (WWW) is most popular but, vulnerability is the major issue as DoS attacks are possible while the packet is being forwarded through the Internet. The standard packet forwarding system requires storage for the purpose of storing the information about routing table, making more overheads on memory. Bloom-filter is data structure which is stored in packet header. Bloom-filters contain information about packet forwarding and packet path so that forwarding can take place and forwarding node does not need to store any information. But using bloom-filters lead to possibility of DoS attacks in network, therefore the security mechanism of bloom-filter needs to be well tested against DoS attack. Thus implementation of additional security feature is required.**

Keywords - Bloom-filter, DoS, packet forwarding, routing table, network security.

I. INTRODUCTION

Today security is the major concern in the Internet as the information contained within the packets that are forwarded over the network are more likely to be attacked by the attackers. When user wants to forward the data to any multicast groups or unicast, the structure of routing table at each node is mandatory and each node has to cache the per flow, state information at every node [6]. Hence bigger storage and computation at each node is required which leads to the load on the nodes. The forwarding of the data to multicast groups or unicast is a common method which has been used in the network. To process packets athwart the network without storing any state or per flow information at each node, the interesting method has been proposed that's called Bloom-Filter based Forwarding. In-packet bloom-filter-based forwarding is comparatively easy. The packet accumulates the delivery tree as a set of forwarding-hop identifiers (FHIDs), which can be links, nodes, or in-out interface pairs on the delivery tree. The set of FHIDs existing in the delivery tree is concealed as a bloom filter [7] data structure, which facilitates adequate testing of set membership. Forwarding of packets on a network of the nodes is done by checking which possible FHID's, e.g., nodes, links and in-out interface pair is in the bloom filter. As bloom-filter does not store any state or per flow information, it can provide superior scalability as compared to the standard IP routing.

There are possibilities of Denial-of-Service attacks, when the data is being forwarded in the networks, which may lead to the negative result and that cause the data forwarding to select different route which has not concealed in the filter by the forwarder. Forwarding the data on the network causes the data to repeat in the network itself and finally results in denial of service attack. So in order to prevent the Denial-of-Service attacks, there are three security methods which has been used in the current bloom-filter structure: I. Limited number of items must be stored in the bloom-filters. II. Make forwarding-hop identifiers secret and centralize bloom-filter computation. III. Per-flow forwarding-hop identifiers must be evaluated cryptographically. The security of these propositions has not been calculated thoroughly until today. Before deploying bloom-filter-based forwarding on open networks bloom filters need further improvements on security. The security mechanisms in the current system do maximize the rate of Denial-of-Service attacks.

Bloom-filter is a possible data structure for storing sets. The main purposes of the bloom-filter are element addition, membership testing, but not the element elimination. Data elements are set in the filter and the filter is realized as a byte array with constant length of me and small k hash function is used in the data set and the corresponding k bits in the bit array are set to 1. The membership testing is also done by checking whether the complementary k bits in a bit array. If demanded bits are set, then it will give a positive result and if not sometimes it will lead to negative results. The bloom -filter is said to be possible data structure because sometimes it will lead to negative results when performing membership testing. The roots of faulty results grow if the number of n bits inside the filter is more. Bloom-filters are commonly used to cache large sets or when the memory usage needs to be minimized. To dodge negative results, restrict the number of bits to be set in the filter. The bloom filters must opt into network packet headers. Therefore, short filters of size 256-1024 bits, which can cache roughly 20-100 data items, are considered. For Bloom-filter-based forwarding following methods have been introduced: I. Make use of bloom filters in the multicast router to minimize the space required for caching the multicast forwarding table [6]. II. Source routing where the multicast delivery-tree is concealed as a bloom-filter in the packet headers. III. Storing the lists of receivers in

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

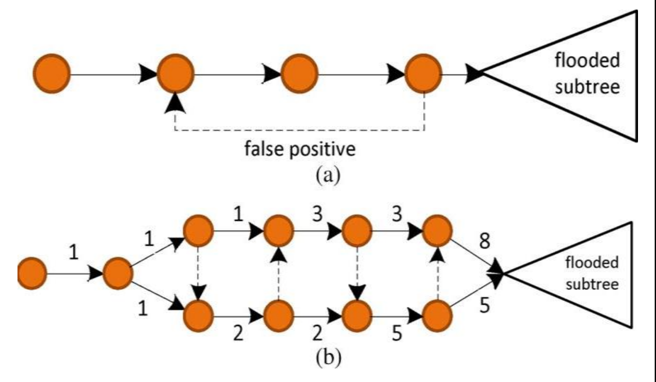

Fig. 1. Forwarding anomalies [4]. (a) Forwarding loop. (b) Repeated

the packet header as a bloom-filter [8], [9]. In the first method each outgoing interface of a multicast router has a bloom-filter that encodes the multicast-group addresses which are possible through outgoing interface. Faulty results are admissible because they only cause the multicast packets to be sent to some unnecessary paths. In the second method source tree or set of routes through which data has to be handled is encoded in-packet Bloom-Filter. By assigning each outgoing link with m bit link identifiers and k pseudo-randomly selected bits are set. But the disadvantage is that functions and inputs used to select the bits differ in various proposals. In the third approach the set of receiver through which data has to be handled is set in the filter and each node will open the filter and the filter will also verify whether it has any other node to forward. Based on the information contained in the filter forwarding decision will be taken. But the main disadvantage in this is security and the possibility of denial of service attack and it will also give the faulty results.

The potential security problems are created due to the following two factors: I. Traffic amplification and II. Potential for false positives Fig.1. Firstly, multicast protocols are vulnerable to injection attacks in which the attacker attempts to forward unauthorized packets into the multicast delivery tree. Secondly, false positives can allow packets to repeat or hop from one branch of the multicast tree to another, resulting in traffic to be distributed in the entire multicast tree.

1) *Injection Attack*: In this type of attacks, if the attacker has control over just one node that can send to a target, then a botnet will be able to send packets to the target. The injection attack is achievable under increasingly harsh security assumptions, following is the list of assumptions: a) Injection with permutations. b) Injection with unknown topology. c) Injection with Flow-Specific FHIDs.

2) *Resubscription Attack*: The issue with the distributed filter discovery is that nothing can force the subscriber to start the join packet-off with a zero value in filter. A malicious subscriber can set the starting filter in the join packet to any filter value that it knows. For example, when some honest subscriber leaves the multicast group, the malicious subscriber may observe this as a change in the bloom filter of the received multicast packets. It may then send a join packet with the previous multicast filter as the starting filter value and will not allow any nodes from leaving the multicast group. Resubscription attack must be taken into consideration the attack works well with filter permutation. The permutations carried out in the downstream direction for the filter in the packet header and in the upstream directions for the filter in the join packet cancels each other. Usual updating of the FHIDs, on the other hand, may reduce the outcome of the attack because the old filters will discontinue to function after the update. In order to prevent and avoid the resubscription attack completely, outlets from the multicast group must be timed together with global FHID updates.

II. RELATED WORK

A. Revisiting IP Multicast: Many years of research into multicast routing have led to a naturally depressed view that the complications of multicast routing and inter-domain multicast routing in particular can only be run-over by restricting the service model multicast [2]. This paper recommends a new method for implementing IP multicast that leads to a re-evaluation of commonly held views [2].

B. In-Packet Bloom-Filters: An in-packet bloom-filter-based source-routing architecture is introduced and examined, which is contrary to Distributed Denial-of-Service attacks Fig.2. An in-packet bloom-filters is based on forwarding identifiers that act concurrently as path designators, i.e. represent which route the packet should take, and as potentiality, i.e. adequately allowing the forwarding nodes along the route to reinforce a security policy where only clearly authorized packets are forwarded. The close representation is based on a little bloom filter whose candidate components (i.e. Link names) are computed at packet forwarding time dynamically using a loosely integrated time-based mutual secret and additional in-packet flow information (e.g., invariant packet content). The capabilities are thus usable and flow-dependent, but do not require any per-flow network state or memory lookup, preliminary security examination suggests that the self-routing efficiency can be an effective building block towards Distributed Denial-of-Service resistant network architectures [3].

C. Forwarding Anomalies in the Bloom-Filter Based Multicast: Multicast protocols make use of in-packet bloom filters to encode multicast trees, the system is highly scalable because no per-flow state is needed in the routers and because routing decisions can be done efficiently by simply verifying the presence of outbound links in the filters. Yet, the growth of previous approaches is restricted by the possibility of forwarding anomalies caused by negative inherent in bloom filters. Different anomalies examined are, namely I. packet storms, II. forwarding loops and III. flow duplication. A stateless solution that increases the robustness and scalability of bloom-filter-based multicast protocols are explained. In general, it shows that the parameters of the filter need to be different to guarantee the stability of the packet forwarding, and which represents a bit permutation technique that effectively avoids both accidental and maliciously created anomalies. The solution to prevent anomalies, the context of

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

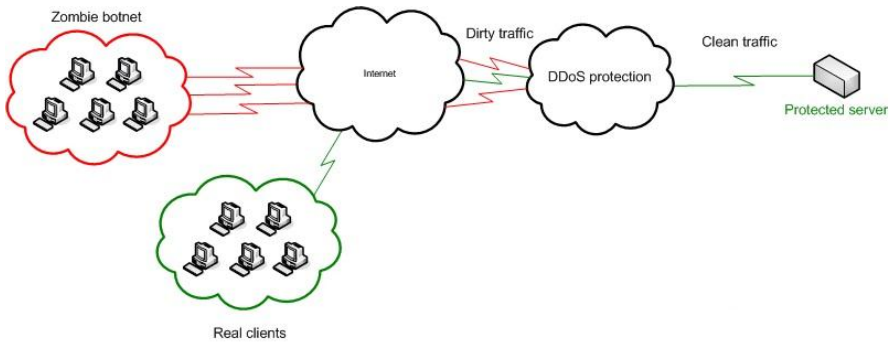

Fig. 2. Distributed Denial-of-Service resistant network architecture.

Bloom-Cast is described, which is a source-specific inter-domain multicast protocol, which makes use of analytical methods and simulations [10].

D. Data Center Networking Based on In-Packet Bloom-Filter with Distributed Open Flow Controllers: Novel data center architecture based on the load balanced forwarding with in-packet bloom filters implemented by two support services that disburse the directory and topology state of Open Flow controller applications. By setting up an army of Rack Managers acting as Open Flow controllers, the architecture promises scalability, performance and fault-tolerance. The guesswork that packet forwarding itself can become a cloud internal service realized by leveraging cloud application well known practices such as high availability, low-latency, key-value storage system. Besides, the critical argument that the centralized controller model of Open Flow networks is prone to a single point of failure and given that the direct network controllers may be distributed physically; yielding a "sweet spot" in between distributed and centralized networking paradigms. [11].

E. Secure Forwarding on the NetFPGA Using In-Packet Bloom-Filter: An in-packet bloom-filter allows one to carry forward source-routed packets with minimum forwarding tables; the Bloom-Filter is encoded with the identities of the links through which the packets need to be forwarded. If the link identities are made content dependent, e.g. by computing the next-hop candidate link identifiers by implementing a cryptographic function over some information carried in the packet header, the Bloom filters differentiates itself from packet-to-packet, making the forwarding fabric resistant towards unauthorized traffic. The early implementation and testing of in-packet bloom filters, forwarding node is depicted that employs cryptographically computed link identifiers. The test is carried on two different cryptographic techniques for the link identity computation and also for making the forwarding decisions. The cryptographic algorithms are implemented and tested on the Stanford NetFPGA. The efficiency and performance of the cryptographic algorithms are also briefly discussed [12].

F. Bloom-Filter-Based Forwarding: The growth in routing-table, multicast scalability problems, and Denial-of-Service (DoS) attacks by botnets are the major problems in the modern Internet and to fix such problems bloom-filter-based forwarding has been recommended. The protocols are source-routed and comprise the delivery tree encoded as a bloom-filter in each packet. Based on this in-packet information the packets are sent forward by the nodes without considering routing tables and without caching per-flow state. The protocols have demanding vulnerabilities and make several false positive security guesses. Denial-of-Service attack against broad classes of bloom-filter-based protocols are presented and concluded that the protocols are not yet ready for deployment on open networks [1].

G. Injection Attacks and Z-Formation: C. Rothenberg, P. Jokela, Nikander, M. Särelä, and J. Ylitalo [3], [1] identified possibility of injection attack in bloom-filter based forwarding and suggested that attackers can examine correlations among filters and therefore derived new filters allowed the injection attacks. Their planned solution, called Z-formation calculate the link identifiers vigorously on per-packet basis. The bloom-filter, hash function is implemented as a cryptographic hash that accepts its parameters from a flow identifier in the packet, a sporadically changing secret key local to each forwarding node, and the received and sent interface identifiers on the projected path of the packet. In this manner, the bloom-filters stores much more routing context as compared to the links on the routes. The need is to maximize the difficulty level of reverse-engineering the secret link identifiers or concatenating the filters of various flows (which either can be done by calculating their bitwise OR). Additionally, the path filters must be refreshed once in a while as the identifiers are time dependent, which will not allow the misbehaving senders to send to the path.

H. Forwarding Anomalies and Bit Permutations: Denial-of-Service attacks are also possible due to forwarding anomalies caused by the false positives. The observable case is a packet with all-ones filter: In which a packet is forwarded to all nodes and the packet loops around and cycles in the network.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

M. Särelä et al. [4], [10] figured out that false positives can also result in un-wanted loops if a packet is mistakenly sent back to an previous node in the multicast tree, and false positives among two paths of the multicast tree may result in flow duplication. To overcome these issues, M. Särelä et al. suggested per router permutation of bloom filters. The permutation is usually a shuffling of the bits in the filter. A fixed pseudorandom permutation is chosen for each node and applied to the filters in each packet that traverse through the node. This is sufficient to avoid flow duplications and loops because the bloom filter becomes a function of the route that the packet has traversed.

III. CONCLUSION

In this paper, a survey is made on different bloom-filter based protocols to avoid denial-of-service attacks. Bloom-filter-based multicast has been recommended as a scalable and DoS-resistant architecture. Our survey highlights the security analysis of the bloom-filter-based forwarding and their DoS vulnerabilities. The outcome indicates that the abstract arguments: revisiting IP multicast, in-packet bloom-filters, forwarding anomalies in bloom-filter based multicast, data center networking based on in-packet bloom-filter with distributed Open Flow controllers and secure forwarding on the NetFPGA using in-packet bloom-filter, about the security of protocols are inexact. According to M. Särelä, M. Antikainen and Tuomas Aura [1], reverse engineering the secret link identifiers can be used for the purpose of creation of anomalies and routing loops. Also a form of packet injection attack can be made on bloom filters that may lead to distributed flooding attacks by botnets.

In particular, we conclude that Denial-of-Service attacks against bloom filter protocols is possible [1], and that the protocols are not yet ready for deploying in open networks.

ACKNOWLEDGEMENT

The authors would like to thank the publishers, researchers for making their resources available and teachers for their guidance. We also thank the college authority for providing the required infrastructure support. Finally, we would like to extend heartfelt gratitude to friends and family members.

REFERENCES

[1] Markku Antikainen, Mikko Särelä , and Tuomas Aura, “Denial-of-service attacks in bloom-filter-based forwarding,” IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 22, NO. 5, OCTOBER 2014.
[2] Ermolinskiy, S. Ratnasamy, and S. Shenker, “Revisiting IP multicast,” Comput. Commun. Rev., vol. 36, no. 4, pp. 15–26, 2006.
[3] P. Nikander, P. Jokela, C. Rothenberg, M. Särelä, and J. Ylitalo, “Self-routing denial-of-service resistant capabilities using in-packet bloom-filters,” in Proc. Eur. Conf. Comput. Netw. Defense, 2009, pp. 46–51.
[4] M. Särelä, C. E. Rothenberg, A. Zahemszky, P. Nikander, and J. Ott, “BloomCasting: Security in Bloom filter based multicast,” in Nordsec2010 Conference, 2011.
[5] S. Silva, R. Silva, R. Pinto, and R. Salles, “Botnets: A survey,” Comput. Netw., vol. 57, no. 2, pp. 378–403, 2012.
[6] B. Grönvall, “Scalable multicast forwarding,” Comput. Commun. Rev.,vol. 32, pp. 68–68, January 2002.
[7] B.H. Bloom, “Space/time trade-offs in hash coding with allowable errors,” Commun. ACM, vol. 13, pp. 422–426, Jul. 1970.
[8] X. Tian, Y. Cheng, and B. Liu, “Design of a scalable multicast scheme with an application-network cross-layer approach,” IEEE Trans. Multimedia, vol. 11, no. 6, pp. 1160–1169, Oct. 2009.
[9] X.Tian, Y. Cheng, and X. Shen, “DOM: A scalable multicast protocol for next-generation Internet,” IEEE Netw, vol. 24, no. 4, pp. 45–51, Jul.–Aug. 2010.
[10] M. Särelä, C. E. Rothenberg, T. Aura, A. Zahemszky, P. Nikander, and J. Ott, “Forwarding anomalies in Bloom-Filter based multicast,” in Proc.30th IEEE INFOCOM, 2011, pp. 2399–2407.
[11] C. Macapuna, C. Rothenberg, and M. Magalh aes, “In-packet Bloom-Filter based data center networking with distributed Open Flow controllers,” in Proc. IEEE GLOBECOM Workshops, Dec. 2010, pp. 584–588.
[12] A. Ghani and P. Nikander, “Secure in-packet Bloom –Filter forwarding on the NetFPGA”. in Proc. 1st Eur. NetFPGA Dev. Workshop, 2010, pp. 1–7.

BIBLOGRAPHY

Akshay Vinayak Parte pursuing BE in Computer Engineering from University of Pune at Dr. D.Y. Patil Institute of Enginnering and Technology, Ambi, Pune.

Anant Sunil Sonsale pursuing BE in Computer Engineering from University of Pune at Dr. D.Y. Patil Institute of Enginnering and Technology, Ambi, Pune.

Mayur Jagannath Dhole pursuing BE in Computer Engineering from University of Pune at Dr. D.Y. Patil Institute of Enginnering and Technology, Ambi, Pune.

Sharmila A. Chopade received Master Engineering Degree from Pune University and having 7 years of teaching experience. She have published 11 international papers. She have published one book on data structure.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)