Iraqi Journal of Information and Communications Technology(IJICT)
Vol. 6, Issue 2, August 2023

www.ijict.edu.iq

ISSN:2222-758X
e-ISSN: 2789-7362

EVALUATION PERFORMANCE OF BLOOM FILTER IN BLOCKCHAIN NETWORK

Kilan M.Hussein [1], **M. F. Al-Gailani** [2]
[1,2] College of Information Engineering, Al-Nahrain University, Baghdad, Iraq
kilan.m.h@uodiyala.edu.iq [1], m.falih@nahrainuniv.edu.iq[2]
Corresponding Author: M. F. Al-Gailani
Received:03/01/2022; Revised: 07/06/2022; Accepted:02/09/2022
DOI:10.31987/ijict.6.2.206

***Abstract*- Abstract- A blockchain is a secret, scalable, and decentralized p2p network in which all nodes follow similar protocols, preventing any single node from controlling the basic structure. In the blockchain, Bloom filters are used to preserve the privacy of lightweight nodes. The Bloom filter is a memoryefficient randomized data structure used to represent a set-in order to support related queries. Bloom filters offer a trade-off between network element size and bandwidth and privacy metrics in untrusted environments. This paper proposes an analysis to evaluate the performance of Bloom filters. The evaluation results are based on the statistical distribution, standard deviation, entropy and y-deniability that are used by the attacker to analysis the leakage of the Bloom filter algorithm. It has shown that less than 50% of the filter size was used and the average of anonymity metric was less than 66%. Furthermore, the experimental results show that the hide metric measure (y-deniability) depends on the total number of elements in the network and the degree of privacy in the network needs a large number of elements and more bandwidth using the Bloom filter algorithm.**

keywords: **Blockchain, Bloom filter, privacy, lightweight nodes, false positive.**

I. INTRODUCTION

Blockchain technology has experienced tremendous growth in recent years, both in terms of research and application. Blockchain technology is a type of distributed ledger technology. It is transparent and cooperatively maintained [1] [2]. It has the potential to offer a novel security solution for data transmission and storage in untrustworthy environments [3] [4]. Blockchain has proven to be particularly effective in a variety of application areas, including smart contracts, insurance, finance, banking, and a number of other fields, according to researchers [5] [6] [7].

âBlockchainâ is responsible for Bitcoin's enormous success [8] [9]. As a result, blockchain has fostered creativity in recent years, and a number of new applications are already being developed using distributed and secure blockchain features [10] [11]. The majority of existing blockchains necessitate a significant amount of storage and computing power. Furthermore, downloading and indexing data into the blockchain takes time on a local level[12] [13]. Some devices are limited in terms of resources, with limited capabilities in computing and network access. Therefore, these devices support a lightweight node of operation [3].

Several libraries currently implement various filters to improve the privacy of lightweight nodes (e.g., Bitcoinâs BIP37-Bloom filter and Ethereumâs LES). In order to mask the addresses of lightweight nodes, it is possible to specify filters with a certain false positive probability, which plays an important role in the privacy of users in the Bitcoin network. Clients have to download these libraries manually and have no control over the types of filters they use. For example, the LES library in Ethereum does not use a filter and hence does not provide address privacy for lightweight nodes. The Bloom filter is supported by Bitcoin as shown in Table I [14] [15].

ISSN:2222-758X
e-ISSN: 2789-7362

TABLE I
OVERVIEW OF LIGHTWEIGHT NODES LIBRARIES.

platform	Library or protocol	Support
Bitcoin	Bitcoin improvement protocol 37	Using privacy by Bloom filter
	Everything should be downloaded	full anonymity
	Electrum	Get unconfirmed transactions by revealing addresses
	Filter guarantees	Filters are supplied to lightweight nodes
Litecoin [7]	BIP37	Using privacy by Bloom filter
Dogecoin [7]	BIP37	Using privacy by Bloom filter
Ethereum [18]	LES	No privacy

Full nodes currently support and provide security, forwarding, filtering, and processing of conformed transactions for each lightweight node without compensating it in existing deployments. This provides little motivation for full nodes to assist lightweight nodes, which is one of the reasons for the network's rapid decline in full node numbers, as shown in Fig. 1.

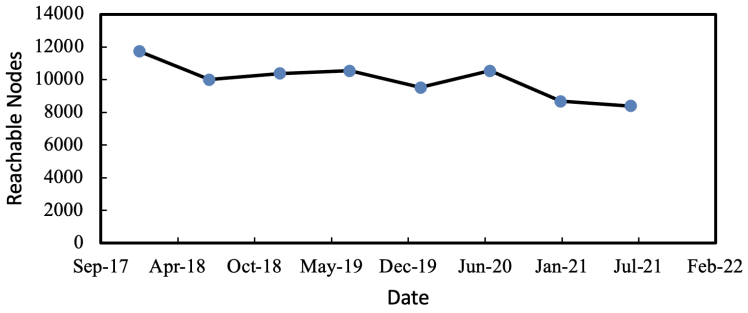

Figure 1: The Number Of Full Bitcoin Nodes Accessible Between 01/2018 and 01/2021.[16]

Since the interaction between full nodes and lightweight nodes is not logged in the blockchain, it is equally difficult to figure out the number of lightweight nodes in the network and assess the level of service they observe [14] [17].

In Bitcoin, Xie et al.[19] proposed a revolutionary way of preserving the privacy of lightweight clients as an alternatives to the Bloom filter. The basic idea is to use private information retrieval (PIR) to ensure that lightweight clients can receive relevant transactions from untrusted full nodes without revealing whose transactions they are requesting.

In blockchain, Cai et al.[20] proposed a new approach to privacy protection that makes information queries utilize a mixed strategy (Nash equilibrium) mechanism.

Bitcoin has a proper anonymity set that can be used to hide the addresses of lightweight nodes. In any case, the information leakage associated with the use of Bloom filters in Bitcoin needs to be properly investigated [21] [22] [23].

ISSN:2222-758X
e-ISSN: 2789-7362

II. RELATED WORKS

The performance of the Bloom filter has been studied previously, and the question of its issue of privacy is always a topic for the study area. Much work has been done to address the issue with the privacy provisions of Bloom filters for lightweight nodes.

In 2010, Christensen et al. [24] suggested a method for calculating the false-positive of a Bloom filter and experimentally demonstrated that standard analysis of false positives is ineffective. Their analysis did not use any privacy metric.

In 2012, G. Bianchi et al. [25] quantified the privacy qualities of Bloom filters. However, when attackers gain access to multiple Bloom filters for the same client, their analysis cannot resolve the privacy issue.

In 2014, Kim et al.[26] presented the experimental analysis of the classical false positive rate, and then suggested a new false positive rate, which their model implemented only for the small Bloom filter size. A. Gervais et al. [21] investigated the privacy implications of using Bloom filters for current lightweight client applications and showed that the existing Bitcoin integration of Bloom filters results in the leakage of a significant amount of information about Bitcoin users' addresses. Their analysis was implemented only for one or two Bloom filters.

In 2017, Kota et al. [22] proposed a privacy-preserving Bloom filter for lightweight nodes based on y-deniability. Where y-deniability as a privacy metric reflects the number of true positive Bitcoin addresses that are obscured by false positives in Bloom filters, they do not secure the synchronization and transaction verification processes.

In 2018, Merve et al. [27] examined the anonymity and privacy of bitcoin, as well as bitcoin and alternatives that enhance anonymity and privacy. They only provide suggestions for creating an upgrade to anonymity and privacy.

In 2019, Qin et al. [28] designed a lightweight node protocol instead of a Bloom filter by utilizing Private Information Retrieval (PIR) to produce a totally private and high-performance query. They applied it to a small database.

In 2020, L. Y. Yeh et al. [29] proposed a dual-level Bloom filter technique to address privacy and performance efficiency concerns. Irrelevant IP lists are rejected at the first level, and the others are determined from the requested lists at the second level. They do not analyze privacy for different database sizes.

In 2021, Lin et al. [5] proposed a mechanism to distinguish which address information should be entered into the Bloom filter for querying based on the mixed Nash equilibrium strategy, along with the historical query records and privacy protection level for lightweight nodes. They analyzed on small database. To our knowledge, the information leakage related to the use of Bloom filters has not been thoroughly investigated. As a result of our findings, a careful examination of the current implementation of lightweight nodes is warranted. The following contributions are made to this study:

1. A detailed analysis is provided to evaluate the performance of Bloom filters.
2. It has been shown that less than 50% of the filter volume has been used. In addition, the average anonymity scale is less than 66%.
3. It was found from the results of the experiment that the hidden metric (y-deniability) depends on the total number of elements in the network.
4. It was also shown that the degree of privacy in the network needs a large number of elements and a larger bandwidth using the Bloom filter algorithm.

ISSN:2222-758X
e-ISSN: 2789-7362

III. BACKGROUND

A. Bloom filter

The Bloom filter (BF) was proposed in 1970 for lightweight nodes. The bloom filter (BF) in the lightweight node is assumed to be the largest number of components that can be fitted into the M elements and it is also important not to exceed the false-positive target (Pt). Table II summarizes the notation used in this paper. The size of the filter is calculated by:

$$m = -\frac{M \ln(P_t)}{(\ln(2))^2} \tag{1}$$

TABLE II
USED NOTATION

Notation	Meaning
BF(S)	The Bloom filter has a set S.
E	Elements added to the Bloom filter
m	Bloom filter size (in bits)
h	Hash functions Number
(E; h; n)	Bloom Filter constraints for False positive function
P_t	False positive target rate
M	Max element to be inserted
P_f	False positive

The filter mainly consists of an m-bit array and these bits can be accessed using the hash function h and a modest size m of the filter [25].

$$h = \ln(2)\frac{m}{M} \tag{2}$$

Given the intended false positive target Pt of BF, the total E of the added elements in the BF can be determined using the following formula [21] [22].

$$E \approx -m\left(\frac{\ln\left(1-\frac{x}{m}\right)}{h}\right) \tag{3}$$

Where x bits are set to 1 , and the total m bits of BF and the total h hash functions bits are constant. To add the component or element e, where $e \in [0,1]$,into the Bloom filter BF, then for every $j \in [1,2,\ldots,h]$, $[\mathrm{Hj(e)}] \leftarrow 1$. The Bloom filter can generate a false-positive number, not a false-negative number. A number of approaches have been used to approximate the false-positive number of BF.

To approximate the false-positive in BF (which is calculated over all possible inputs), the false-positive of a BF filter (M; Pt) containing E elements can be calculated using equation (4) [21] [24]:

ISSN:2222-758X
e-ISSN: 2789-7362

$$P_f = \left(1 - \left(1 - \frac{1}{m}\right)^{hE}\right)^h \tag{4}$$

B. Privacy Degree

1) Single Bloom filter: Amidst the many positive qualities that the adversary properly possesses, the likelihood of (e), $P_a(e)$, is the true positive value that match the BF, which is calculated based on the equations [5]:

$$\begin{aligned} P_a(e) =& \frac{E}{E+F} \cdot \frac{E-1}{(E+F)-1} \cdots \frac{E-e+1}{(E+F)-e+1} \\ =& \prod_{i=0}^{e-1} \frac{E-i}{(E+F)-i} \\ =& \prod_{i=0}^{e-1} \frac{E-i}{E+|B-E|P_f(E)-i} \end{aligned} \tag{5}$$

Where B denotes the total number of elements in the system, F (unknown to the adversary) indicates the number of all false-positive elements. Consequently, the adversary can correctly predict the likelihood of all elements added to the BF as the following:

$$\begin{aligned} P_a(E) =& \frac{E!F!}{(E+F)!} \\ =& \prod_{i=0}^{E-1} \frac{E-i}{(E+F)-i} \\ =& \prod_{i=0}^{E-1} \frac{E-i}{E+|B-E|P_f(E)-i} \end{aligned} \tag{6}$$

Then the probability of correctly guessing a single element as a true positive is:

$$P_a(1) \approx \frac{E}{E+|B|P_f(E)} \tag{7}$$

2) Multiple Bloom filters: The degree of privacy provided by BF is represented by $\mathrm{Pa}(e)$. The lower the $\mathrm{Pa(e)}$ value, the better the privacy performance; the higher the $\mathrm{Pa(e)}$ value, the worse the privacy performance.. When an opponent acquires more than two BFs from the same lightweight client, the intersection of each pair of BFs can be used to discover the common components of the multiple filters. Identifying b BFs from the same lightweight node, an attacker can use equation (3) to estimate the number of elements inserted in each filter. Assume that BFs: B1, B2 ..., and Bb are ordered by the number of insertable items in ascending order. Based on formula (8), the higher the number of BFs an attacker can access, the lower the error in categorizing the real element, and the higher the value of $\mathrm{Pa}(.)$, the worse the BF privacy effect [5] [21].

$$P_a(e) \approx \prod_{i=0}^{e-1} \frac{E-i}{E-i+|B|\prod_{\forall_e} P_f\left(E_e\right)} \tag{8}$$

ISSN:2222-758X
e-ISSN: 2789-7362

Then the probability of correctly guessing a single element is:

$$P_a(1) \approx \frac{E}{E + |B| \prod_{i=1}^{BFb} P_f(E_i)} \tag{9}$$

C. Measuring Anonymity

Privacy is often framed in terms of an "anonymity set". This refers to the number of other indistinguishable entities in the system. However, given the extent to which the above-mentioned analysis techniques are probabilistic in nature the 'degree of anonymity' in the system, d, is a more appropriate formal definition [30]:

$$d = \frac{H(m)}{H_M} \tag{10}$$

Where H(m) is the entropy of the system considering the observations made by the attacker:

$$H(m) = -\sum_{i=1}^{m} pi \log_2 pi \tag{11}$$

Where m represents the size of the Bloom filter, pi the probability of using each bit or location in the Bloom filter entity. H_m is the maximum entropy state in the Bloom filter, under which each bit is likely to be used equally [30].

$$H_m = \log_2 m \tag{12}$$

In case of simple entropy, maximal anonymity is achieved when $H_m = \log_2 m$ and with normalized entropy when $d = 1$.

D. y- deniability

False positives can hide true positives in the Bloom filter. This is called y-Deniability, and it shows how many true positives are covered up by false positives [22].

When the element e is deniable with a probability of y, the Bloom filter is said to be y-deniable. The y of an mbit array Bloom filter is computed as follows [25].

$$y = \left(1 - 4^{-Bs \times Pt}\right)^{-\log_2 Pt} \tag{13}$$

Where $Bs = (B - M)/M, M$ is the maximum number of elements added in the Bloom filter and Pt is the target false positive rate.

E. Bandwidth

The bandwidth used by the transaction dissemination process will grow significantly as connectivity improves. Bloom filters offer bandwidth in exchange for a privacy trade-off. But these privacy benefits are only achieved at the cost of some lost bandwidth. Therefore, the false positive rate controls the bandwidth/ privacy trade-off, which must be carefully tuned.

ISSN:2222-758X
e-ISSN: 2789-7362

The amount of transaction data that needs to be saved in the Merkle tree in contemporary blockchain systems is enormous. When a lightweight node validates the transaction data, all that is required for the full node is to deliver the hash values associated with the data in the specified verification path. This is the current strategy of the Bitcoin network to increase the efficiency of verification. When the information of multiple transactions has to be verified in several blocks, the number of layers in the Merkle tree may differ due to the different number of transactions saved in each block [3]. For a single lightweight node, which has $b(1, 2 \dots c)$ blocks of transactions and T number of hashes (transactions) in each block, the total D data bytes transmitted from the full node to the lightweight node is:

$$D = \sum_{b=1}^{c} 32 \left(\log_2 \left(T_b \right) + 1 \right) \tag{14}$$

IV. System and Threat Model

Assumption: if an adversary can spy on the network and get a single or several BFs for a lightweight client, the adversary can also obtain the factors used in the composition of the BF. Accordingly, the attacker can access all elements/transaction information that appears in the system, as well as their respective implementation orders, as shown in fig. 2.

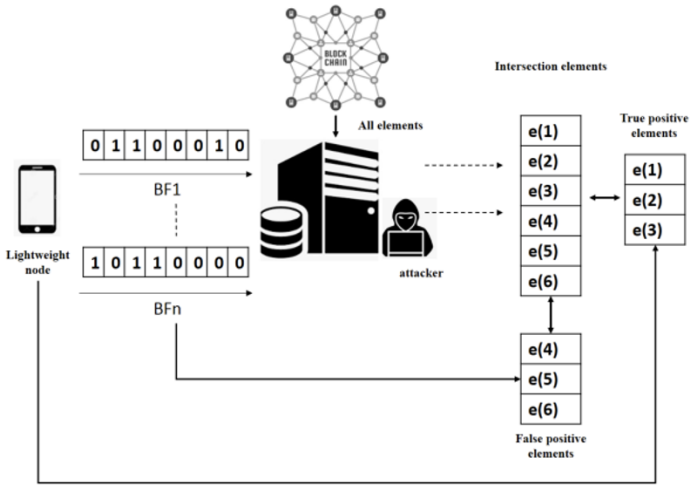

Figure 2: A Model Of a Lightweight Client Adversary Based on a BF

1) Generation of elements:

a. The h hashes element $\mathrm{e}(1)$ are determined, that is, $H1(\mathrm{e}(1)), H2(\mathrm{e}(1)) \dots Hh(\mathrm{e}(1))$

b. The modest h locations are adjusted to match the size of the Bloom filter, i.e., $B[H1(\mathrm{e}(1))], B[H2(\mathrm{e}(1))] \dots B[Hh(\mathrm{e}(1))]$, as shown in Fig. 3.

ISSN:2222-758X
e-ISSN: 2789-7362

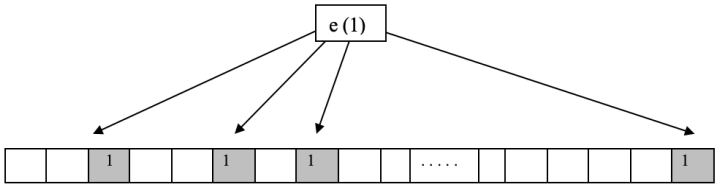

Figure 3: Insertion Of Element e(1)

2) Query of elements

a. The h hashes element corresponding to @e are calculated, that is H1(@e),H2(@e)...Hh(@e). The h corresponding positions of the Bloom filter vector are checked; i.e., $B[H1(@e)], B[H2(@e)]...\ B[Hh(@e)]$, whether set to 1 or not. If the value of anyone is 0 , then @e is not included in the set. However, if all of the associated bits are 1, @e may be in the query set, but it may not be the real query result. It is likely that a false positive result will arise at this point, as illustrated in Fig. 4.

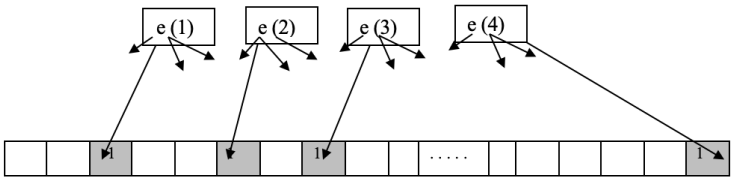

Figure 4: Query Of Elements e

The simulation was carried out using the $\mathrm{C++}$ language as mentioned in the Bloom filter algorithm. Fig. 2 depicts the implementation setup and the evaluation results as follows:

1- Frequency distribution:

The simulation was carried out several times with 102 elements and a false positive target at $\mathrm{Pt} = 0.005$, the filter parameters are: filter size $m = 1616$ bits and hashes $h = 11$ according to equations 1 and 2, respectively [21]. The results show as in Fig. 5 that the frequency distribution between (0 to 4) and only 796 out of 1616 locations was selected and 820 ($< 50\%$) unselected. Therefore, an attacker could neglect these 820 locations to analyses the Bloom filter.

ISSN:2222-758X
e-ISSN: 2789-7362

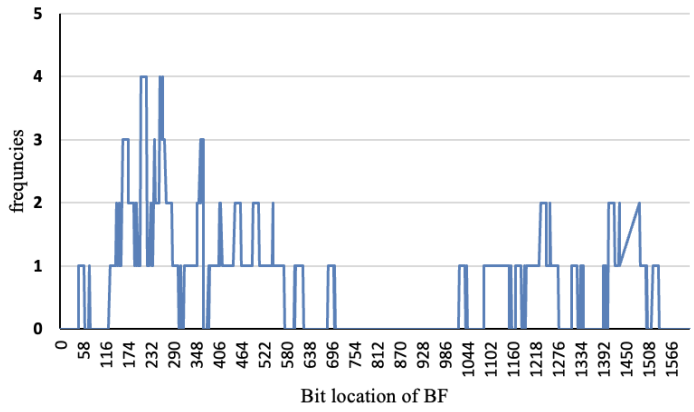

Figure 5: Frequency Distribution Of Bit Locations Bloom Filter

2- Statistical distribution measurements :

The simulation was performed several times using different numbers of insertion elements $(102, 204, 306, \ldots)$. The results in Table III show that the SD (standard deviation) changes with a mean (0.794366) and the entropy changes with an average (8.533086577). Fig. 6 and 7 show the results of SD and entropy, respectively. The results show that any increase in the element numbers and the size of the Bloom filter does not increase the effect on the standard deviation and entropy and, as a result, does not affect the anonymity metric.

In Table III , the values of bit location and frequencies are obtained using the simulation by applying the algorithm of Bloom filter, then apply the formula for the mean, variants, SD, Entropy, H_M, and d to obtain the other results that indicated in Table III.

TABLE III
Bloom Filter Statistical Distribution Measurements

Insert elements	Bloom size	mean	variance	SD	Entropy	H_M	d %
102	1616	0.694307	0.640859	0.800537	6.955199255	10.65821148	65.25672029
204	3232	0.694307	0.532954	0.730037	7.794250473	11.65821148	66.85631398
306	4848	0.694307	0.614187	0.783701	8.141971945	12.24317398	66.50213381
408	6456	0.695167	0.63642	0.797759	8.398028787	12.65642486	66.35387859
510	8072	0.694995	0.723216	0.850421	8.561435075	12.97871046	65.96522129
612	9688	0.69488	0.573353	0.7572	8.861784099	13.24198315	66.9218802
714	11296	0.69529	0.387179	0.622237	9.185775534	13.46352437	68.22712448
816	12912	0.695167	0.999162	0.999581	8.880151235	13.65642486	65.02544644
918	14528	0.695072	0.706351	0.840447	9.174177064	13.82654849	66.35189594
1020	16136	0.69534	0.580247	0.76174	9.378092304	13.97799537	67.09182581
All mean		0.6948832	0.6393928	0.794366	8.533086577	12.83612085	66.45524408

ISSN:2222-758X
e-ISSN: 2789-7362

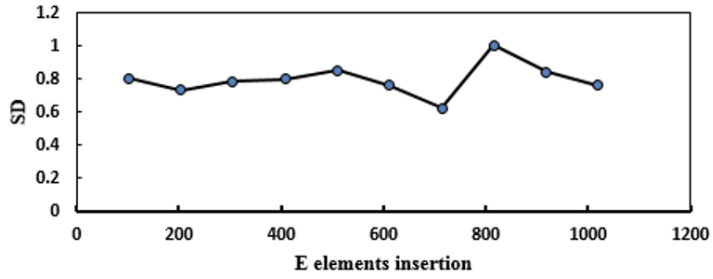

Figure 6: Statistical Distribution Of Standard Deviation in Bloom Filter

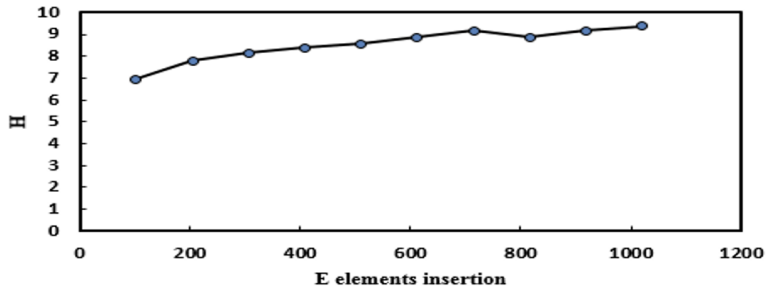

Figure 7: Statistical Distribution Of Entropy H in a Bloom Filter

3- Anonymity measurements:

The Bloom filters shown in Table III, can provide maximal anonymity with an average $\mathrm{H}_M \approx 13$ bit $(\mathrm{d} = 100\%)$ and normal anonymity with an average of $d = 66.4\%$ as shown in Fig. 8 . According to the definitions given in Bloom filters using these entropies, an attacker has an average of 33.6% chance of guessing the elements of the Bloom filter.

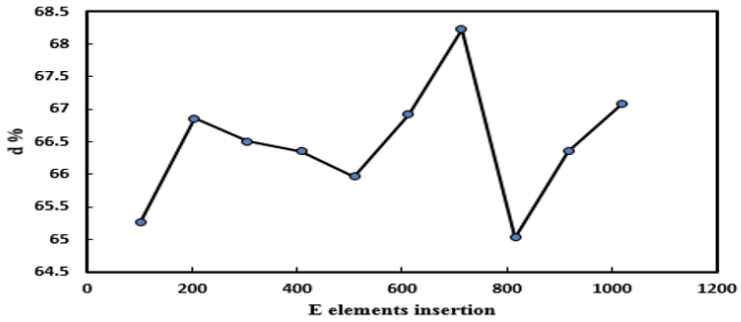

Figure 8: Normal Anonymity With an Average Of $d = 66.4\%$

4- Privacy measurements :

Fig.9 analytically shows the calculation of the privacy measurements $\mathrm{P_a}(1)$ according to Eq. (7). Obviously, $\mathrm{P_a}$ decreases,

ISSN:2222-758X
e-ISSN: 2789-7362

especially when the insertion elements E in the Bloom filter increase among all the $5,000,000$ elements in the full node, and at $P_t = 0.5\%$ there is no privacy from 1 to 36 elements insertion. As shown in Fig.9, the obtained $\mathrm{Pa}(1)$ is considerably larger in the case of two Bloom filters according to Eq. (9) when compared to the case where the adversary has access to only one Bloom filter.

Fig. 10 shows that the higher the number of inserted elements, the less effective addition is on $\mathrm{P_a}(1)$. This means that as the number of insertions and elements of a full node decreases, the adversary may also gain information about the privacy of lightweight nodes.

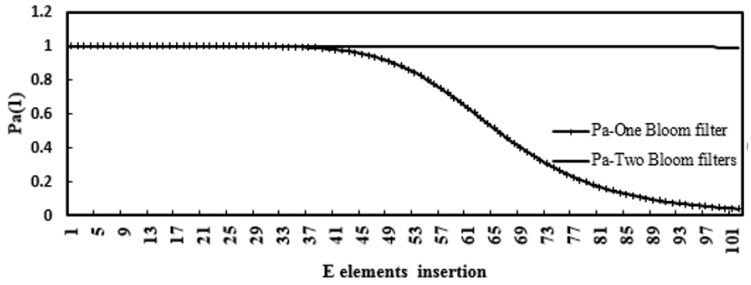

Figure 9: Analytical Analysis Of the Privacy Pa Vs E Elements Insertion for One and Two Bloom Filters

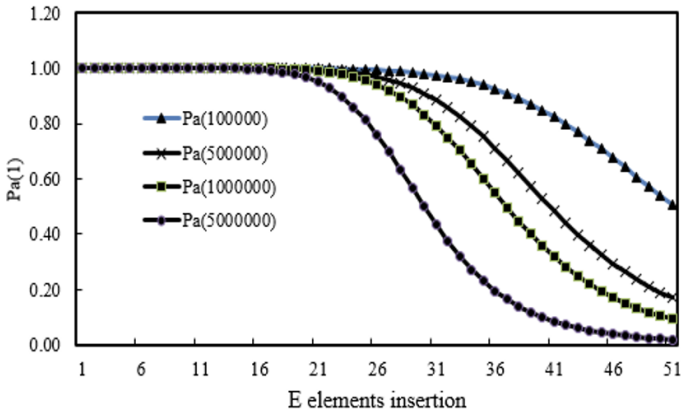

Figure 10: Privacy Pa Vs Elements Insertion E With Different Full Node Elements

5- y-deniability measurements :

The privacy properties of the Bloom filter are mainly characterized by the size of the masking set (Bs) and the Pt. Fig. 11 shows that when Pt becomes tighter, the same y-deniability objective necessitates larger B sets.

6- Bandwidth measurements:

Fig. 12 shows the behaviors of full node with and without a false positive Bloom filter. According to Eq. (13), the replay of full node needs to count the number of hash functions of a transaction whose content element is authenticated by the

ISSN:2222-758X
e-ISSN: 2789-7362

Merkle tree. For example, in a full node, if the Bloom filter generates 609 elements with a false positive and the transaction hash size is 32 bytes with a block contents 1000 transactions.

Then, Blocks $\times 32 \times (\log_2(\mathrm{N_b}) + 1) = 609 \times 32 \times (\log_2(1000) + 1) = 213$ Kbyte. The bandwidth increases logarithmically according to the number of transactions in each block and the false positive increases by the Bloom filter.

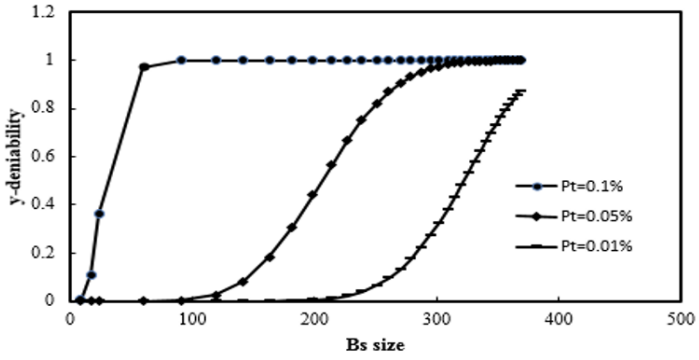

Figure 11: y-deniability versus hiding sets size B s , for P t=0.1% ; 0.01% ; 0.05%

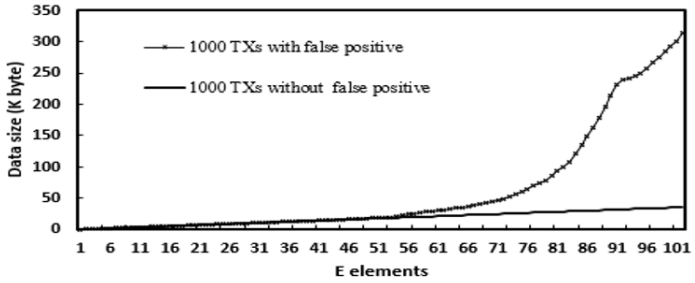

Figure 12: K byte bandwidth vs E elements with 1000 transactions/block

V. Privacy Preserving for Gervais et al. Study Comparison

As a study comparison, A. Gervais et al. [19] assumed that an adversary would be able to hack one or more full Bitcoin nodes and eavesdrop on communication channels in order to obtain one or more Bloom filters related to a lightweight node.

They used equations 6 and 7 to determine the degree of privacy Pa, and they investigated the privacy implications of using Bloom filters that lead to leaking information about Bitcoin users' addresses. In Fig. 13 , they run an experiment among all 900,000,000 Bitcoin addresses by setting the target false positive rate (Pt) to $0.1, 0.01, 0.001$, and 0.0001 , respectively. They computed Pa analytically with respect to the number of addresses $\mathrm{E} = 102$ that a lightweight node is equipped with. The results show that Pa increases when Pt decreases.

ISSN:2222-758X
e-ISSN: 2789-7362

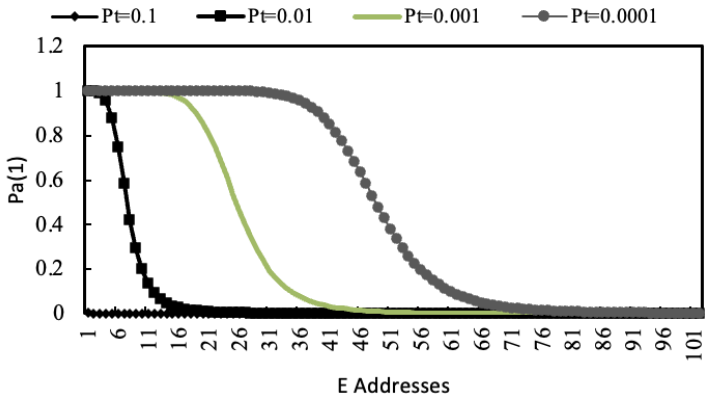

Figure 13: Pa Vs E Addresses for Different Pt

VI. Conclusion

Bloom filters are commonly used by lightweight nodes with low resources in blockchain applications to acquire useful information from full nodes. This kind of information acquisition inadvertently exposes the important information of the lightweight node according to the parameters set of the Bloom filter and the number of elements in the full node. This paper proposes an analysis of the Bloom filter algorithm regarding the measurements of statistical distribution, frequency, mean, standard deviation, privacy, y-deniability, and entropy that an attacker uses to analyze the Bloom filter algorithm leakage. Furthermore, the experimental results showed that the degree of privacy needs large elements in the network to use the Bloom filter algorithm. In addition, the analysis shows that there is almost no guaranteed privacy regarding the false positive rate, which needs more bandwidth to increase the privacy.

Funding

None

ACKNOLEDGEMENT

The author would like to thank the reviewers for their valuable contribution in the publication of this paper.

CONFLICTS OF INTEREST

The author declares no conflict of interest.

References

[1] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in 2017 IEEE International Congress on Big Data (BigData Congress), 2017, pp. 557-564: IEEE.
[2] J. Khamar and H. Patel, "An Extensive Survey on Consensus Mechanisms for Blockchain Technology," in Data Science and Intelligent Applications: Springer, 2021, pp. 363-374.
[3] J. Wang, B. Wei, J. Zhang, X. Yu, and P. K. Sharma, "An optimized transaction verification method for trustworthy blockchain-enabled IIoT," Ad Hoc Networks, vol. 119, p. 102526, 2021/08/01/ 2021.
[4] E. Androulaki, G. O. Karame, M. Roeschlin, T. Scherer, and S. Capkun, "Evaluating user privacy in bitcoin," in International conference on financial cryptography and data security, 2013, pp. 34-51: Springer.

ISSN:2222-758X
e-ISSN: 2789-7362

[5] L. Ge, T. J. S. Jiang, and C. Networks, "A Privacy Protection Method of Lightweight Nodes in Blockchain," vol. 2021, 2021.
[6] D. Dasgupta, J. M. Shrein, K. D. J. J. o. B. Gupta, and F. Technology, "A survey of blockchain from security perspective," vol. 3, no. 1, pp. 1-17, 2019.
[7] Q. Feng, D. He, S. Zeadally, M. K. Khan, N. J. J. o. N. Kumar, and C. Applications, "A survey on privacy protection in blockchain system," vol. 126, pp. 45-58, 2019.
[8] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.
[9] F. Tschorsch, B. J. I. C. S. Scheuermann, and Tutorials, "Bitcoin and beyond: A technical survey on decentralized digital currencies," vol. 18, no. 3, pp. 2084-2123, 2016.
[10] G. O. Karame and E. Androulaki, Bitcoin and blockchain security. Artech House, 2016.
[11] M. Conti, E. S. Kumar, C. Lal, S. J. I. C. S. Ruj, and Tutorials, "A survey on security and privacy issues of bitcoin," vol. 20, no. 4, pp. 3416-3452, 2018.
[12] J. Xie, F. R. Yu, T. Huang, R. Xie, J. Liu, and Y. J. I. N. Liu, "A Survey on the Scalability of Blockchain Systems," vol. 33, no. 5, pp. 166-173, 2019.
[13] Y. Xiao, N. Zhang, W. Lou, Y. T. J. I. C. S. Hou, and Tutorials, "A survey of distributed consensus protocols for blockchain networks," vol. 22, no. 2, pp. 1432-1465, 2020.
[14] D. Gruber, W. Li, and G. Karame, "Unifying lightweight blockchain client implementations," in Proc. NDSS Workshop Decentralized IoT Security Stand., 2018, pp. 1-7.
[15] S. Park, S. Im, Y. Seol, and J. J. I. A. Paek, "Nodes in the Bitcoin Network: Comparative Measurement Study and Survey," vol. 7, pp. 57009-57022, 2019.
[16] Bitnodes. Available: https://bitnodes.io/dashboard/
[17] J. Lu et al., "One-hashing bloom filter," in 2015 IEEE 23rd International Symposium on Quality of Service (IWQoS), 2015, pp. 289-298: IEEE.
[18] S. K. Kim, Z. Ma, S. Murali, J. Mason, A. Miller, and M. Bailey, "Measuring ethereum network peers," in Proceedings of the Internet Measurement Conference 2018, 2018, pp. 91-104.
[19] Y. Xie, C. Zhang, L. Wei, Y. Niu, and F. Wang, "Private Transaction Retrieval for Lightweight Bitcoin Client," in 2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), 2019, pp. 440-446: IEEE.
[20] X. Cai, Y. Ren, and X. J. I. A. Zhang, "Privacy-protected deletable blockchain," vol. 8, pp. 6060-6070, 2019.
[21] A. Gervais, S. Capkun, G. O. Karame, and D. Gruber, "On the privacy provisions of bloom filters in lightweight bitcoin clients," in Proceedings of the 30th Annual Computer Security Applications Conference, 2014, pp. 326-335.
[22] K. Kanemura, K. Toyoda, and T. Ohtsuki, "Design of privacy-preserving mobile Bitcoin client based on y-deniability enabled bloom filter," in 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), 2017, pp. 1-6: IEEE.
[23] D. Kleyko, A. Rahimi, R. W. Gayler, E. J. N. C. Osipov, and Applications, "Autoscaling bloom filter: controlling trade-off between true and false positives," vol. 32, no. 8, pp. 3675-3684, 2020.
[24] K. Christensen, A. Roginsky, and M. Jimeno, "A new analysis of the false positive rate of a Bloom filter," Information Processing Letters, vol. 110, no. 21, pp. 944-949, 2010/10/15/ 2010.
[25] G. Bianchi, L. Bracciale, and P. Loreti, "Better Than Nothing" Privacy with Bloom Filters: To What Extent? 2012, pp. 348-363.
[26] J. Kim, "On the False Positive Rate of the Bloom Filter in Case of Using Multiple Hash Functions," in 2014 Ninth Asia Joint Conference on Information Security, 2014, pp. 26-30: IEEE.
[27] M. C. K. Khalilov, A. J. I. C. S. Levi, and Tutorials, "A survey on anonymity and privacy in bitcoinlike digital cash systems," vol. 20, no. 3, pp. 2543-2585, 2018.
[28] K. Qin, H. Hadass, A. Gervais, and J. Reardon, "Applying private information retrieval to lightweight bitcoin clients," in 2019 Crypto Valley Conference on Blockchain Technology (CVCBT), 2019, pp. 60-72: IEEE.
[29] L. Y. Yeh, P. J. Lu, S. H. Huang, and J. L. Huang, "SOChain: A Privacy-Preserving DDoS Data Exchange Service Over SOC Consortium Blockchain," IEEE Transactions on Engineering Management, vol. 67, no. 4, pp. 1487-1500, 2020.
[30] L. Tennant, "Improving the Anonymity of the IOTA Cryptocurrency," ed: White Paper, 2017.