Name Management Using IOTA in ICN

***Abstract*—Information-centric networking (ICN) has received a wide attention as a next-generation network. Unlike conventional IP networks, which forward packets based on IP addresses, ICNs packets are sent based on the name of the contents which are cached to the routers involved, then delivered to the consumer. Since any participating party can upload content to the ICN, the risk of content poisoning attack (CPA) is ever-present. In CPA, an attacker degrades the cache efficiency by uploading fake content under a real name posting as a legitimate publisher. As a countermeasure to CPA, many existing methods determine the legitimacy of content using digital signatures with public keys, and alerts routers of unjustified content upon detection. However, it is difficult for them to detect fake-CPA attacks which use public keys of fabricated content to generate digital signatures. Most methods also lack counter-measures to spoofed fake-CPA attacks, in which the certification authority (CA) that manages the public key or its staff member colludes with the attacker to rewrite the legitimate publisher's public key to the attacker's and inject fake contents that pretends to be authentic contents that are high in popularity into the cache. In this paper, we propose a method to prevent the spoofed fake-CPA by managing content names with IOTA, a distributed ledger technology that blocks tampering of contents registered on the system. We also numerically compare the search time and memory requirement of four search methods that search content names managed in the ledger in the proposed method. As a result, we confirm the trade-off between the search time and memory requirement.**

I. Introduction

Although anyone can upload content as a publisher on ICN (information-centric networking), a CPA (content poisoning attack) [6], in which an attacker pretends a legitimate publisher to upload fake content under a real content name, thereby degrading the cache functionality, has been pointed out. As a countermeasure against the CPA, a method has been proposed in which consumers determine the legitimacy of contents by digital signatures using public-key cryptography and notify routers of unauthorized contents [3].

Meanwhile, there are two types of the CPA: a fake type that matches the digital signature generated by the public key associated with the content, and a corrupted type that does not match [4], so it is difficult for the method described in [3] to detect the fake-CPA. Especially, it is also difficult to detect the spoofed fake-CPA, in which the staff of the certification authority (CA) that manages the public key colludes with the attacker to rewrite the legitimate publisher's public key to the attacker's and inject fake contents that pretends to be real and popular contents into the cache. It is assumed that the impact of this attack will be significant if popular content with many accesses is spoofed. In addition, the attacker's content is more legitimate because the public key of the legitimate publisher is taken over and rewritten.

A spoofed fake-CPA occurs when a public key is managed by only one authority, such as a CA. In this paper, we propose a method to prevent it by managing content names using IOTA [8], a DAG (directed acyclic graph)-based distributed ledger technology. Although blockchain issues with scalability, IOTA is highly scalable.

In the proposed method, it is necessary to search for the transaction in the ledger at the time of content registration by the publisher and at the time of delivery request by the consumer. We compare the search time and the amount of memory requirement among four search methods.

II. CPA

While it is easy to detect corrupted-CPA by checking signatures, it is difficult to detect the fake-CPA. The fake-CPA can be classified into the original fake-CPA and the spoofed fake-CPA. In the original fake-CPA, the attacker injects fictitious content created by the attacker into the router by requesting it from the colluder. In this case, an attacker publishes his own fake content on the network. In the original fake-CPA, there are no requests for fake content from legitimate users, but only from a few colluders, so the amount of fake content flowing through the network is small. Therefore, the impact of the attack is small because fake content is cached in the router only temporarily. On the other hand, the spoofed fake-CPA injects fake content into the router by requesting it from legitimate users. In this case, an attacker publishes fake content on the network that imitates real content, taking over the name of legitimate content and distributing the fake content. In the spoofed fake-CPA, since many legitimate users request fake content, it is highly likely that the fake content will remain in the cache for a long time, and the impact of the attack is estimated to be high. This paper focuses on preventing the spoofed fake-CPA, which has high impact of attacks.

III. Proposed Method

In the IOTA, new transactions select two of the unselected transactions, i.e., the tips, and transactions form a DAG. There are three types of tip-selection algorithms [7]: URS (uniform random selection) which selects two transactions randomly, URW (unweighted random walk) which starts from the genesis transaction, i.e., the first transaction, and selects tips with equal probability, and WRW (weighted random walk) which selects transactions considering the cumulative weights. The transition probability P_{xy} from transaction y to x in WRW is:

$$P_{xy} = \frac{e^{-\alpha(H_x - H_y)}}{\sum_{z:z\rightarrow x} e^{-\alpha(H_x - H_z)}} \tag{1}$$

where H_x and H_y is the cumulative weights of transaction x and y, and α (≥ 0) is the parameter of the cumulative weight. The WRW is effective in preventing splitting attacks [2] and parasite chain attacks [5] that cause double payments.

In the proposed method, each transaction contains pairs of prefix and corresponding content name which composes of the prefix, the public key, and the digital signature. Due to the nature of distributed ledgers, it is difficult to tamper with the registered data, thus ensuring its legitimacy. In [9], certificates attached to messages sent by vehicles in a cooperative intelligent transportation system are managed in IOTA, and this guarantees the transparency of certificate issuance.

Next, we describe the flow of the proposed method. When the publisher uploads the content, the content prefix as well as the content name are registered in the transaction. To prevent duplicate content names from being managed, transactions in the IOTA ledger are searched by content prefix to see if the same content name is already managed, and if it already exists, reject the registration; otherwise, register it. After the upload by the publisher completes, the consumer requests the prefix of the content. After that, the IOTA ledger is searched by the requested prefix and replies to the consumer with the name of the contents that have hit. Finally, consumer sends an interest with that content name and requests the content.

This guarantees that only the publisher who first registers the name on IOTA for a given prefix is a legitimate publisher, thus preventing the spoofed fake-CPA.

In the proposed method, the latency and amount of memory requirement for search are greatly affected by the search method of transaction. In this paper, we evaluate the following four search methods under the following assumptions.

hash-chain method (hash): The data is stored in the element of the hash table at the address corresponding to the hash value obtained by applying a hash function to the data search key. Although different data can have collisions that result in the same hash value, the hash-chain method allows multiple data to be managed in the same bucket by connecting them with a concatenated list. In the proposed method, the prefix converted into a numerical value is used as a search key, and the address of the DAG transaction that stores the name of the corresponding content is managed in a hash table.

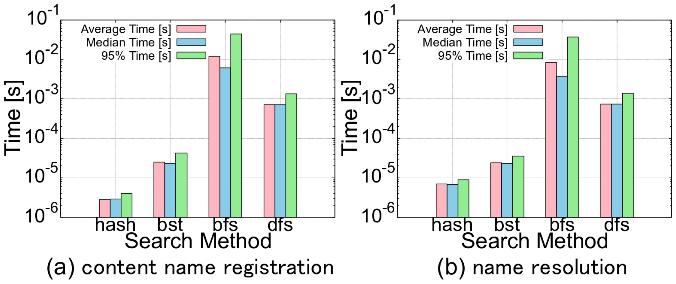
binary search tree (bst): It is a kind of tree structure in which each node key has the property *left node ¡ parent ¡ right node*. It begins its search at the root, the uppermost node, and moves to the left if the search key is less than the value of the node, and to the right if greater than that. This process is repeated until the content name is found. Like the hash-chain method, the content prefix is used as a search key in the bst, and each node maintains the address of a DAG transaction.

breadth-first search (bfs): It searches transactions on the DAG directly, breadth-first, starting with the transaction with the smallest number of hops from the genesis transaction on the DAG. If all the transactions with the same number of hops have been searched but none have been found, the search for a transaction with a higher number of hops is started and repeated until it is found, or all the transitions are searched.

depth-first search (dfs): It also directly searches for transactions on the DAG in depth-first order. As with bfs, the search begins at the genesis transaction, but searches deeply until it reaches a transaction without children, i.e., tip. If a tip is reached but not yet discovered, it returns to the last branch and searches for unexplored transactions.

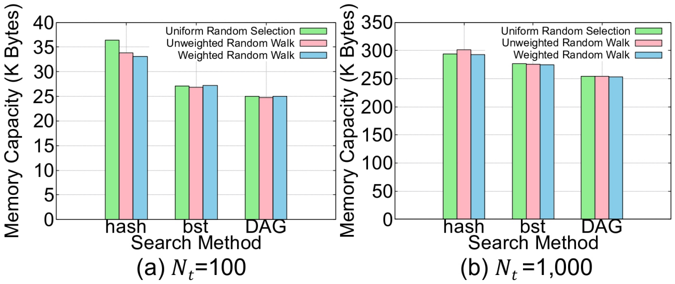
IV. PERFORMANCE EVALUATION

We compare the search time and memory requirement among the four search methods described in the previous section by computer simulation. It was performed using the IOTA simulator DAGsim [10] with modifications. Search time is evaluated by the mean, median, and the 95th percentile. In terms of memory requirement, bfs and dfs, which manage data directly on the DAG, are grouped together as a DAG and compared with hash and bst. In the experiments, the number of domains for which a web page was displayed without error when browsing the top 8,000 accessed web pages published on Alexa's webpages [1] in November 2017 was used as the set of the content names for evaluation. For WRW, the number of transactions N_t is set to 100 or 1,000 for WRW, and we set $\alpha = 0.1$ in the transition probability equation (1).



(a) content name registration (b) name resolution

Fig. 1. Search Time in $N_t = 1,000$



(a) N_t=100 (b) N_t =1,000

Fig. 2. Amount of memory requirement

A. Search Time

Figure 1 shows the measured search time in (a) content name registration and (b) name resolution for hash, bst, bfs, and dfs at $N_t = 1,000$. In both cases, the hash-chain method took the least, followed by bst, dfs, and bfs. In the hash-chain method, although the list in a bucket was searched based on the hash value, the amount of data in the bucket was small, so it was possible to search for the entire contents in a shorter time than others. In bst, the search time was also short because the necessary parts of the data were searched instead of all the data. On the other hand, dfs and bfs took long time because they searched all DAG in an exhaustive manner.

B. Amount of Memory Requirement

Figure 2 shows the amount of memory requirement for each method at (a)$N_t = 100$ and (b)$N_t = 1,000$. In all cases (a) and (b), the hash-chain method had the largest memory requirement, followed by bst and DAG. The hash-chain method required a large amount of memory because the bucket with the largest capacity in the hash table was allocated for all buckets, so memory efficiency was poor because some buckets had unused memory. On the other hand, bst required a node capacity equal to the number of transactions, so it required less memory than the hash-chain method because there was no extra memory. Whereas DAG required the least amount of memory because it did not need to be managed by other tables.

These results confirmed the trade-off between search time and memory requirement for each search method.

References

[1] Alexa webpage, https://www.alexa.com/siteinfo

[2] G. Bu, et al., G-IOTA: Fair and confidence aware tangle, IEEE INFOCOM WKSHPS 2019.

[3] W. Cui, et al., Feedback-Based Content Poisoning Mitigation in Named Data Networking, IEEE ISCC 2018.

[4] P. Gasti, et al., DoS and DDoS in Named Data Networking, IEEE ICCCN 2013.

[5] S. Ghaffaripour and A. Miri, Parasite Chain Attack Detection in the IOTA Network, IEEE IWCMC 2022.

[6] T. Nguyen, et al., Content Poisoning in Named Data Networking: Comprehensive Characterization of real Deployment, IFIP/IEEE IM 2017.

[7] J. Park, et al., A Block-Free Distributed Ledger for P2P Energy Trading: Case with IOTA?, CAiSE 2019, LNCS 11483, pp. 111-125, 2019.

[8] S. Popov, et al., Equilibria in the Tangle, Computers & Industrial Engineering, 136, pp.160-172, Oct. 2019.

[9] A. Tesei, et al., IOTA-VPKI: a DLT-based and Resource Efficient Vehicular Public Key Infrastructure, IEEE VTC 2018.

[10] M. Zander. 2018. Python IOTA Tangle simulation.
https://github.com/manuelzander/iota_simulation