future internet

Article

A Hybrid SWIM Data Naming Scheme Based on TLC Structure

Zhijun Wu * and Bohua Cui

School of Electronics & Information & Automation, Civil Aviation University of China, Tianjin 300300, China; 15732158160@163.com

* Correspondence: zjwu@cauc.edu.cn

Received: 5 July 2020; Accepted: 17 August 2020; Published: 25 August 2020

Abstract: Aiming at the problem of low interconnection efficiency caused by the wide variety of data in SWIM (System-Wide Information Management) and the inconsistent data naming methods, this paper proposes a new TLC (Type-Length-Content) structure hybrid data naming scheme combined with Bloom filters. This solution can meet the uniqueness and durability requirements of SWIM data names, solve the "suffix loopholes" encountered in prefix-based route aggregation in hierarchical naming, and realize scalable and effective route state aggregation. Simulation verification results show that the hybrid naming scheme is better than prefix-based aggregation in the probability of route identification errors. In terms of search time, this scheme has increased by 17.8% and 18.2%, respectively, compared with the commonly used hierarchical and flat naming methods. Compared with the other two naming methods, scalability has increased by 19.1% and 18.4%, respectively.

Keywords: index terms SWIM; naming method; hybrid; TLC structure; bloom filter

1. Introduction

The core goal of SWIM is to promote efficient and safe information sharing between different business systems or applications of civil aviation software and hardware [1]. However, with the increase of the civil aviation business, the amount of information has gradually increased, with various types of information and various data types. Data communication and exchange mechanisms are not timely and effective, which severely restricts the efficient sharing of civil aviation information [2]. Therefore, because of these problems, it is necessary to design an applicable naming method for the business system and data of SWIM to solve its data transmission, sending and receiving, and interaction problems.

Data-oriented naming schemes are roughly divided into two categories: mainly flat naming [3] and hierarchical naming [4]. However, both methods have some problems. The former defines object names as hashes of public keys and labels. It meets uniqueness and durability requirements and allows network elements to verify the integrity of the data without the need for any external mechanisms, thereby preventing denial of service attacks. However, binding user-friendly names to flat self-authenticated names require some external mechanism. This process may be attacked. Also, flat names are difficult to aggregate for routing scalability [5]. The hierarchical content naming scheme solves the issue of aggregations, however, it requires users to know the encoding rules used by the content owner or naming authority to map readable names to object identifiers (OID). Besides, an external mechanism is required to bind the OID to the key to verify what matches the requested OID. Not only end-users but also infrastructures such as content routers should know the public key of the requested content OID [6]. Otherwise, even if the end-user can authenticate the content, the network may continue to transmit incorrect data, resulting in a denial of service attack.

This paper proposes a new naming and aggregation scheme, in which the content identifier OID is composed of a set of variable-size information elements (IEs), and each IE is encoded as

TLC (Type-Length-Content). Information elements can flexibly form hierarchical or peer-to-peer relationships. In addition to the TLC structure, the SWIM network does not have any restrictions on the OID assignment, and it is not necessary to know the meaning of the type and value [7]. This TLC structured namespace can guarantee the uniqueness and durability of the name, and achieve better scalability and reliability. To solve the suffix vulnerability problem, this paper uses Bloom filter to aggregate elements and generate summary values, which can express more accurate information. Besides, the content router can flexibly control the degree of aggregation or the popularity of content objects according to the distance to achieve a balance between the required resources and the compression of routing information [8]. This hybrid naming scheme is superior to prefix-based aggregation in the probability of routing discrimination errors and achieves an improvement in search time and scalability.

The rest of this paper is arranged as follows: Section 2 mainly introduces related work. The third section introduces the hybrid TLC naming format. The fourth section mainly analyzes the performance of the proposed naming and aggregation scheme and gives the comparison results with other schemes. The fifth section is a summary of the full text.

2. Related Works

The existing application of SWIM has created huge economic value for society. With the deepening and development of SWIM research, SWIM's system model and other aspects are gradually meeting people's needs. In recent years, many scholars at home and abroad have conducted relevant research on how to name SWIM data resources, how to efficiently process and send resources, and so on. These issues are considered to be the basic issues and core demands of future SWIM construction. At present, research in this area is still in its infancy, and research in various countries is in the process of exploration, and there is no relatively mature solution. Zhang Hongke, Su Wei et al. [9] proposed the concept of adding "access identification" and "service identification" to the naming model, and the transmission of data can be completed through the mapping relationship between the two. T. Koponen et al. [10] use a flat naming mechanism to process network data resources, regardless of the association between information, and unique content by matching a pair of public and private keys. It is presented in P: L format, where P represents the hash value of the public key and L represents the attribute of the resource. V. Jacobson et al. [11] proposed a naming method for the hierarchical structure of a typical tree. A named name can be used to easily identify a content name. D. Trossen et al. [12] adopted the classification of resource data into one type of application, used different application identifiers to forward different data, and used the mapping method of application identifiers to aggregation identifiers for naming. Balakrishnan et al. [13] propose mapping human-readable canonical names to flat, not human-readable (universally unique identifier-like) identifiers. An additional specification [14] proposes a formal description of the interaction in the uniform resource name resolution process. In request for comments 2611 [15], the namespace is defined as the collection of unique identifiers that have already been assigned. However, it is also defined as a collection of all valid names in the literature [16]. A Extensible Open Router Platform Resource Locator [17] is a hybrid partitioned-descriptive name type. Other distributed name resolution architectures (such as those based on content addressable network [18]) decouple the naming scheme from the name resolution process, allowing a location-independent naming scheme (for both physical and logical meanings of "location").

Compared with the research situation abroad, China's research process is developing at a slower pace, and many directions are just beginning, with no substantial research results. Sun Qiong et al. [19] proposed a flow label-based technology, which uses the hash lowest cost first search tree algorithm to perform an in-depth analysis of rule set features and to define flow labels in detail. Qu Huaqiao et al. [20] proposed a single-label and multilabel theory. Given the interdependence between classification labels and attributes in naming and their contribution to multilabel performance, multilabel classification based on label attributes was proposed. By calculating the attribute density of attributes in the sample set of each tag, the common attribute set is used as the group tag feature attribute, and multilabel classification is performed on this basis. Li Yang et al. [21] proposed a content naming system similar to URL, mainly to meet the requirements of readability, uniqueness, etc. in naming rules, including the prefix, location, format, etc. of the content name. Mao Wei et al. [22] adopted different syntax and character sets for different namespaces, introduced different naming techniques such as URI [23], URN [24], and UDDI, and carried out their comparative analysis. Quan Wei et al. [25] named the data by dividing the data into multiattribute labels. Each label formed could describe one or more service resources from different angles, and then these labels were formed into a set form. Thus, to form a complete resource name. Based on this naming method, combined with the description of the changing and invariant attribute characteristics, the resource name can be further divided into two levels: entity domain and behavior domain, to analyze the attribute information.

3. Naming Scheme of SWIM Data

This chapter will mainly introduce the SWIM data naming scheme. First, a brief introduction to the overall structure of the naming scheme. The second section mainly introduces the TLC structure in the naming scheme, encoding each information element into a TLC structure. The third section mainly introduces the aggregation based on the TLC structure. It mainly uses the Bloom filter [10] on the aggregated TLC elements to generate the digest value and applies a router to its content object to inform the prefix and digest value. The fourth subsection mainly introduces the mixed naming method, which deals with the content names in a hierarchical, flat, and attributed manner. The fifth section mainly introduces the algorithm implementation process of the hybrid naming scheme, and the sixth section mainly introduces the routing of service information.

3.1. Overall Structure of the Naming Scheme

The service resources are classified by tags, such as aviation, flight, weather, time, organization, and other category tags. Each label represents an attribute of the resource, and all the labels identified are collected. The multidimensional label is convenient for users to identify and select the service information they need. Through descriptions from different angles, subsequent search and matching can be maximized.

The overall design of the resource naming scheme is shown in Figure 1.

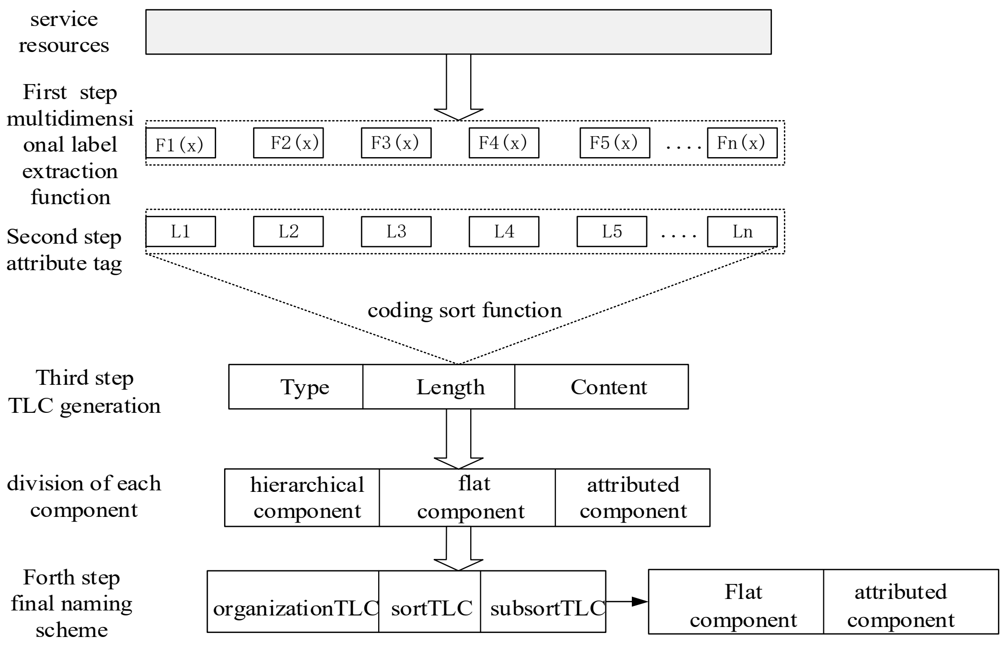

Figure 1. The overall design of the naming scheme.

The steps involved in the naming scheme design are as follows.

Step 1: First, set the attribute analysis dimension according to the content attributes of the resource and the needs of the user, and express it as a collection of multiple attribute tags: where x is the content and k is the k dimension tag extraction function.

Step 2: After the content is processed by the label extraction function, a property set of the corresponding relationship is generated for each content of the query: Each corresponding label represents an attribute of the content, and this attribute set is used to represent the characteristics of the desired content.

Step 3: The collection of attribute tags is used to generate the required TLC form through the processing of the encoding function. The generated form is processed according to the "hierarchical-flat-attribute" component to form the final naming scheme.

Service resources are divided into two levels: static entity domain tags and dynamic behavior domain tags. Static indicates that the content information does not change with time and scene changes and is a basic feature label of content information. Dynamic means that the information will change dynamically with the change of the network. It is a behavioral domain label. Resource attribute tags are not limited to these and can be revised as needed. Table 1 is an example of an attribute label classification.

Table 1. An example of attribute label classification.

Label Field	Attribute Value	Attribute Value			
Entity domain label	Provider Resource Category Area	Pro 1 Aviation Area A	Pro 2 Flight Area B	Pro 3 Meteorological Area C	Pro 4 Other Area D
Behavior domain label	Popularity Credibility	Level 1 A	Level 2 B	Level 3 C	Level 4 D

Based on the division of the attributes of "entity domain label" and "behavioral domain label", resources can be effectively unclassified, which is convenient for later maintenance and management to meet user needs.

3.2. *TLC Structure in Naming Scheme*

In the proposed naming scheme, each OID consists of a set of variable-size information elements (IE) [26], and each IE is coded as a type-length-content (TLC). The size of the type and length fields is fixed, for example, 1 octet, and the size of the content field is variable. The type field indicates the type of element, and the length field defines the size of the content field included in the TLC. TLC may contain multiple sub-TLCs in its content field [27].

As an example, a content owner or naming authority may assign an aviation flight plan with an OID, such as "institutionTLC-sortTLC-subsortTLC-titleTLC-formTLC-segdivisionTLC". The character "-" is used to separate elements for presentation purposes and does not belong to OID [28]. This content OID naming includes the content owner's organization ID, category ID, sub-category ID, title ID, where the ID is just the specified number. Another content owner can use different conventions to name their content objects.

TLC naming meets uniqueness and durability requirements. As long as the content owner has a globally unique organization ID, the local content OID can be guaranteed to be globally unique. Because OIDs have the feature of not referencing locations, OIDs can be stored anywhere, which ensures the persistence of naming. Even if the object changes its management domain, the same OID can be used.

3.3. TLC Structure Polymerization

The information of location needs to be aggregated. However, traditional prefix-based route aggregation is ineffective. Because content objects can be copied in multiple locations [29], and a node may not have location information for all content objects whose OID starts with a given prefix. The content starts with a given prefix, which leads to a suffix vulnerability. The TLC structure can adopt a better aggregation mechanism because the OID can be easily extended to carry summary information. In this paper, we apply Bloom filter [30] to the aggregated TLC elements to generate digest values and apply routers to its content objects to inform the prefix and digest values. Bloom filter is a computationally effective hash-based scheme that allows a certain probability of error [31,32].

Suppose that the router in SWIM has the location information of certain content objects. The OIDs of these content objects start with the same institutionTLC, sortTLC, and subsortTLC, and use aggregated OID (sOID) to represent them. The sOID consists of an abstract of the organization TLC, sortTLC, and sub-category TLC (prefix) of these content objects, as well as its headTLC, formTLC, and segdivisionTLC (suffix). The format is organization TLC-category TLC-subcategory TLC-digest TLC1-digest TLC2-digest TLC3, in which the values in digestTLC1, digestTLC2 and digestTLC3 are generated by Bloom filters from content headTLC, formTLC and segdivisionTLC [33].

To generate the value field of digest TLC1, the Bloom filter is used in the title TLC of content objects, and the routing interfaces of these content objects will be aggregated [34,35]. The router treats the headTLC element as a binary number in the process and does not need to know its meaning. To use a Bloom filter to generate a summary, you need to assign a bit array of bits, where all bits are initially set to 0, and you also need k independent hash functions, $h_1, h_2, \ldots, h_k$, to add elements to the summary, you need to provide it to k the hash function to get k array positions. Using any of the hashed elements in these functions gives a value between 1 and m, and the hash value indicates the position in the array. All corresponding bits are set to 1. The same process can be used to generate summary values for format and segmented TLC [36]. The generation of aggregate OID is shown in Figure 2 [33,36].

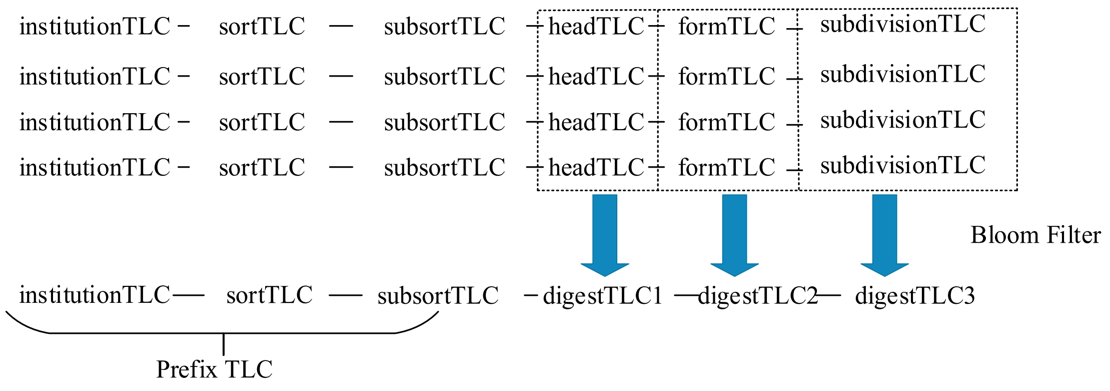
institutionTLC – sortTLC — subsortTLC – headTLC – formTLC – subdivisionTLC
institutionTLC – sortTLC — subsortTLC – headTLC – formTLC – subdivisionTLC
institutionTLC – sortTLC — subsortTLC – headTLC – formTLC – subdivisionTLC
institutionTLC – sortTLC — subsortTLC – headTLC – formTLC – subdivisionTLC
Bloom Filter
institutionTLC — sortTLC — subsortTLC – digestTLC1 — digestTLC2 — digestTLC3
Prefix TLC

Figure 2. Generation of aggregate object identifiers (OID).

The format of the digest TLC is shown in Figure 3 [33,36]. Its type is "well-known" by all routers in SWIM. The content field of the digest TLC includes a fixed-size subfield that indicates the number of aggregate elements added to the digest, and the rest of the content field is used to store the digest value itself.

Generally, the OID of a content object is composed of TLCs, TLC1-TLC2-TLC3 . . . TLC. For content objects whose OID has the public first TLC $(j + 1)$, the router can use aggregate OIDs to represent them so that only a single routing state of these content objects is needed [22]. sOID is composed of the prefix (previous j public TLC) and $(J - j)$ digest TLC, TLC1- . . . TLCj -DigestTLC $(j + 1)$ . . . DigestTLCJ. The value in DigestTLCi $(i = j + 1, \ldots, J)$ is obtained by taking the first TLC from each OID and applying the Bloom filter on it.

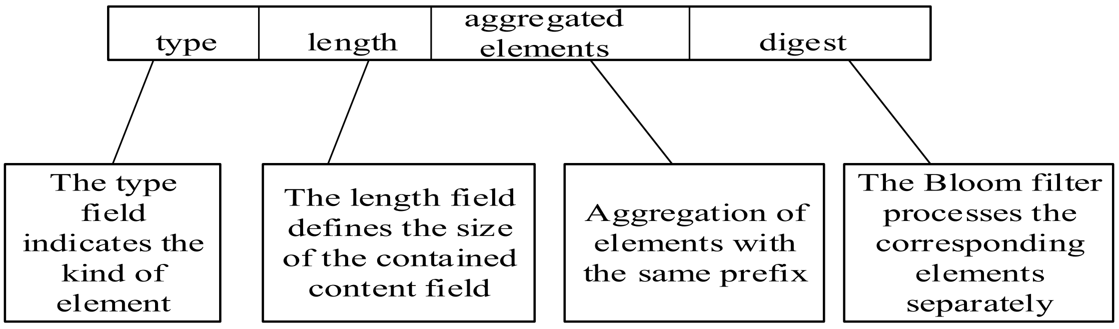
type
length
aggregated elements
digest
The type field indicates the kind of element
The length field defines the size of the contained content field
Aggregation of elements with the same prefix
The Bloom filter processes the corresponding elements separately

Figure 3. The format of digest type-length-content (TLC).

The router in SWIM can flexibly control the degree of aggregation based on the popularity of the content object or the distance to the location where the content object resides [37]. For example, aggregation is not performed on the route advertisement messages of content objects residing in the local domain, but the gateway router publishes the digest OID of its content objects to the external domain. For example, the size of the prefix can be adjusted, that is, the number of nonaggregated TLCs in the sOID to balance between maintaining the routing state and the network resources required for information loss. It can be known from the sOID in the received publication message [38] that requests for content objects with this prefix and summary can be provided by this domain.

For the match between the OID and sOID of the query, the corresponding j prefix of the uncollected TLC should be the same, and each digest TLC in the sOID should give a positive match to indicate that the corresponding element in the OID of the query may exist. Since sOID carries the summary value of the last $(J - j)$ TLC of the summary content OID, this helps to alleviate the problem of suffix vulnerabilities while achieving routing scalability.

3.4. Mixed Structure Naming Based on TLC

The TLC structure can be divided into multiple categories as needed, but when the number of TLC reaches a certain value, the named query rate will be affected [39]. In response to this problem, the TLC structure is further processed, and the "hierarchical -flat-attribute" mixed structure naming method is adopted to make up for the shortcomings of each naming method by taking advantage of each naming method [40]. Store each part separately with different information and arrange them in an orderly manner to form a broad naming format. For the convenience of the following text, the hybrid naming mechanism is divided into three parts: (1) hierarchical components; (2) flat components; (3) attribute components. Each formed component will carry different information, and the different combinations of the three parts have resulted in six alternative naming formats. The mixed naming structure is shown in Figure 4 [41,42].

These six naming methods contain almost all possible alternatives and can also maximize the compatibility of naming. Any naming method can be converted into a single naming method for processing.

Considering the scalability of the system, to reduce the size of the routing table, the naming method needs to have a certain degree of aggregation. Since the superiority of hierarchical naming in terms of aggregation is great, the hierarchical naming method is used in the first place. The naming style of "hierarchy-based" is shown in Figure 5 [41,42].

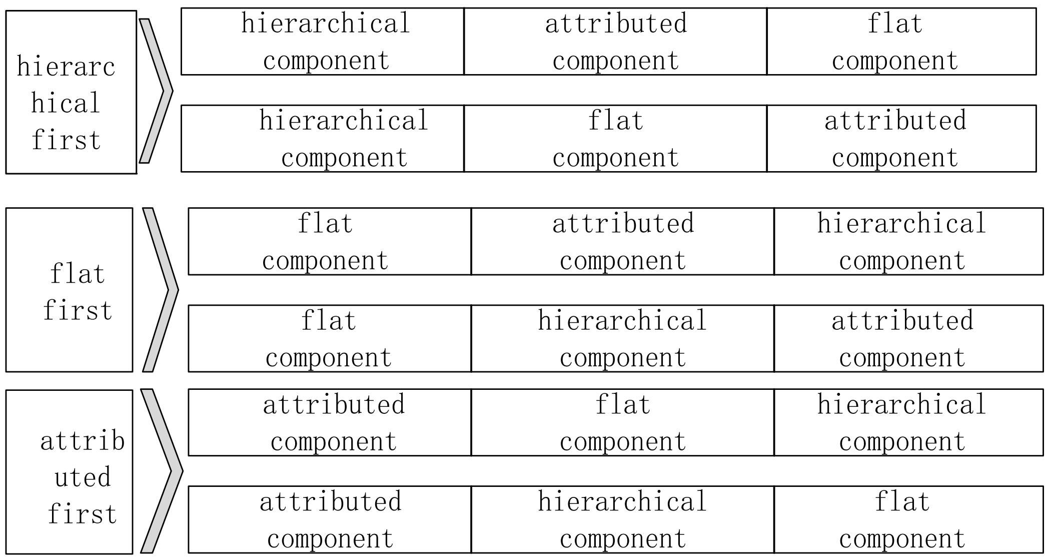

Figure 4. Mixed naming structure combination.

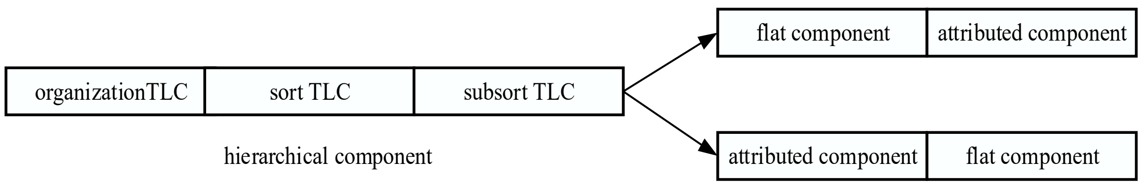

Figure 5. "Hierarchical component-based" hybrid naming structure.

Flat naming can realize the name self-authentication through the self-authentication hash value to uniquely represent the content name. The same hierarchical component can uniquely determine the content through the planarization component. After layering the components, a limited-length component is needed to determine the resource name in the smallest range. Therefore, the planarized component is placed behind the layered component. The flattening component is implemented through an encrypted hash algorithm, which can fix the name characters of infinite length to a limited string. Therefore, together with the hierarchical components, the content of the resource is uniquely identified, and the two processing forms are combined to maximize the advantages and value of our naming scheme.

When performing the longest prefix matching, it is impossible to match a unique structure from the given content, cause of different resources, the name prefix may contain different attribute information, such as sending time, publishing location, content format size, etc. This requires the participation of attributed components, matching content names with the same prefix to determine the desired information. In this process, the role of the attributed component is to further clarify the specific attribute mark of the content name, and further provide users with customized needs.

Figure 6 gives an example of the naming of "hierarchical-flat-attribute" based on the TLC structure [41,42]. For example: "Organization TLC-Sort TLC-Subsort TLC|fsfdhgk46u6gfjhgjhfk|Weather avi 2048p" is a hybrid naming of "hierarchical-flat-attribute-based" based on the TLC structure. Among them, "|" is used to separate the components, "Organizational TLC-sortTLC-subsort TLC" is the hierarchical component, "fsfdhgk46u6gfjhgjhfk" is the flat component, "Weather avi 2048p" is the attribute component.

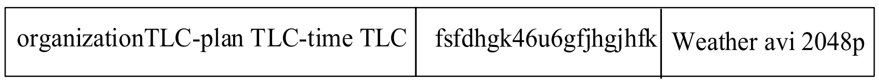

Figure 6. Example of the naming of "hierarchical-flat-attribute" based on TLC structure.

3.5. Algorithm Implementation of Hybrid Naming Scheme

In the process of SWIM providing or obtaining services, the realization process of the content name generated by SWIM service resource information is shown in Algorithm 1. The SWIM service resource first obtains the type, length, quantity, service quality, and other information of the service theme provided in the SWIM network according to the label attribute theme, and generates a summary value through the Bloom filter on the TLC element with the same name prefix. Determine the specific content suffix of the given information. Next, according to the TLC structure of the given format, the hierarchical components are divided according to the standard top-level string set. The flattened and attributed components are divided through the calculation of the hash algorithm and the identification of special attributes.

Algorithm 1. Flow of hybrid naming scheme.

```
Input: Service resource call information
Output: content name
1: Procedure SHash(Prefix)
2: $W_S \leftarrow$ strlen(prefix)
3: Hash [0 ... k] $\leftarrow 0$;
4: Word $\leftarrow 0$;
5: Hash $\leftarrow$ DJB(Prefix)
6: For i $\leftarrow 1$ to $k$ do
7: Location $\leftarrow$ hash$[i-1]$ % $w\_s$;
8: Hash$[i] \leftarrow$ (Prefix[location] + hash[i − 1]) % w;
9: Word $\leftarrow$ word$|$(i << hash$[i]$);
10: Prefix$\left(x^T, x^B\right) \leftarrow x$;
11: If $x < div$ & ! match$\left(x^T\right)$ $then$;
12: Update$\left(x^T\right)$;
13: Or else, if $x > div$ &! match$\left(x^T\right)$ $then$;
14: $Positioning\ in\ “BF” : j \leftarrow h_i\left(x^B\right)$;
15: $Increase\ j\ calculator\ value$ : $C_j \leftarrow C_j + 1$;
16: Or else x > div
17: $Positioning\ in\ “BF” : j \leftarrow h_i\left(x^B\right)$;
18: $Increase\ j\ calculator\ value$ : $C_j \leftarrow C_j + 1$;
19: End
```

3.6. Routing Based on Swim Content Name

Each piece of data in SWIM has a name to ensure the liquidity of the content by not revealing its address. Finally, addresses no longer need to be allocated and managed in the local network, which is especially helpful for sensor networks. The SWIM service node consists of three parts: a forwarding information table (FIB). FIB is used to determine the interface of a data packet, where a data packet comes from, and where it is going. Waiting for the interest table (PIT), PIT stores all requests that do not meet the FIB table to future content sources. Content storage table (CS), CS is a buffer memory, each node stores the searched name to respond to the request. The data forwarding process based on name forwarding in SWIM is shown in Figure 7.

When a node intends to retrieve content, it sends an interest packet indicating the name of the required content. During each hop of forwarding, the result of the forwarding decision-making lookup operation depends on the FIB, PIT, and CS tables of the requested name. If the searched content does not match successfully in CS, PIT will update the tracking interest to the next interface. Then, by looking up the FIB table to find the most suitable forwarding port, the interest packet will be forwarded to the next hop. The data packet will be returned to the requesting node to activate the line in reverse.

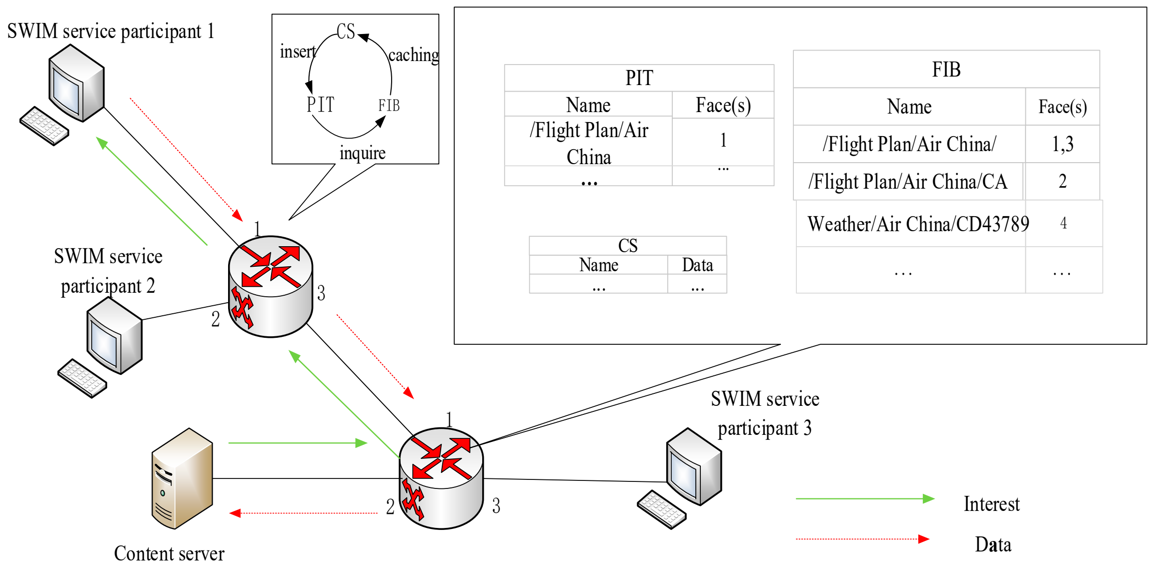

Figure 7. The name-based data forwarding process in SWIM.

The content storage table of the routing node is mainly used to cache data packets passing through the node so that subsequent interest packets can respond quickly when requesting the same data packet. When the cache space of the content storage table of the node is full, a corresponding replacement strategy is needed to support the new data packet cache and discard the old data packets. The cache of this new routing node is different from the node cache in the traditional IP network. The data packet cached in the traditional IP routing node is only for a specific connection. The data packet cache is mainly to facilitate retransmission after packet loss. It is not used to be requested by other nodes. The data packets cached in the router can be reused to better realize the sharing and distribution of data; while the data packets cached in the IP router cannot be reused, which leads to a large number of redundant transmissions in the IP network.

FIB stores a collection of prefixes with different names mapped to different interfaces. It is mainly used to forward interest packets to potential data packet nodes. FIB is similar in structure to PIT, mainly composed of name prefix and interface list. In the structure of FIB, since one prefix can map multiple interfaces, it can easily realize the multipath forwarding of interest packets. In the creation or update of FIB, it is very similar to the traditional routing table, which can be configured manually or automatically generated through related protocols.

4. Experiment and Result Analysis

In this chapter, the information interaction topology between SWIM access nodes, including experimental scenarios and content data sources, is simulated based on the routing interaction mechanism between the service content of multiple SWIM participants such as airports, airlines, and air traffic control bureaus. Data volume, etc. test and verify the experimental content indicators, mainly including TLC aggregation based route resolution error tolerance, naming scheme query time on different layers, and query time for different data volumes on specified layers, and finally the detection rate and false alarm rate are verified under different schemes.

4.1. Experimental Environment

In the experiment, the information interaction between the three SWIM access node departments was simulated, and the service information involved all originated from the real civil aviation network. According to SWIM's architectural characteristics and business processes, and the experimental environment was built to test this scheme. Using the open-source enterprise service bus as the service infrastructure, the service registration node is designed to enable the service provider to publish service information, query the service called by the service requester and return the description of the service to help the service requester and the service provider Bind to complete the service call in SWIM. The network topology is shown in Figure 8.

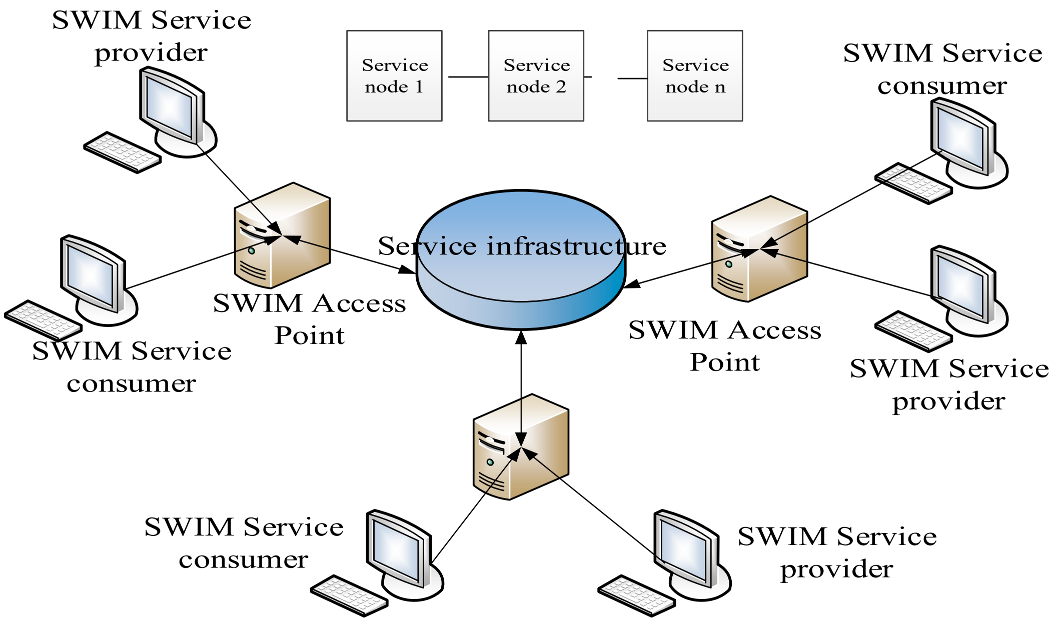

Figure 8. Experimental test environment.

The simulation platform is composed of PC, server, router, and switch. Among them, a PC as a service acquirer obtains SWIM services, and a PC as a service provider provides SWIM core services, including service registration, query, and management functions.

The basic configuration of each server in this experiment is shown in Table 2.

Table 2. Environment configuration.

Computer Setup	Remarks
processor	Intel(R) Core (TM) i5-3470
main frequency	3 GHz
RAM	4 GB
operating system	Linux
database systems	MySQL 5.6
software	MyEclipse 10
application server	Tomcat 6.0
browser	IE 9.0

The SWIM access node is mounted on the ActiveMQ server, MySQL is used as the database to store SWIM service data, and the flight information service and aviation information service are provided externally with the SWIM flight information exchange model and the SWIM aviation information exchange model as standards.

In the experiment, take aviation, flight, and weather data separately, take values of different lengths and numbers, and then process them in layers. The name prefix table is used as the data set, and the size is 3 m and 10 m respectively. The 3 m prefix table contains 4,532,167 name prefix entries, the data set size is 56 MB, and the 10 m prefix table is 896,262 entries. The name hierarchy and length of the two prefix tables are shown in Tables 3 and 4, respectively.

Table 3. Three-meter prefix table.

Layer	Name Length	Percentage	Number of Names
1	4.11	0.01	224
2	14.82	27.30	2,129,835
3	20.63	15.38	391,377
4	25.76	0.85	21,664
5–9	31.86	0.06	1694

Table 4. Ten-meter prefix table.

Layer	Name Length	Percentage	Number of Names
1	6.15	0.01	1248
2	16.34	75.34	7,150,441
3	20.56	22.06	2,106,879
4	27.65	2.56	264,978
5–9	35.67	0.05	1578

Table 5 shows all the data sets collected and used in the experiment, including a total of 10 data sets from test 1 to test 10. The data source is randomly selected from the blacklist data set. Test 1 is a data set with a data set size of 1 MB, and the subsequent data sets are sequentially increased by 1 MB data.

Table 5. Experimental collection data set.

Data Set	Test 1	Test 2	Test 3	Test 4	Test 5
Data size	1 m	2 m	3 m	4 m	5 m
Data set	Test 6	Test 7	Test 8	Test 9	Test 10
Data size	6 m	7 m	8 m	9 m	10 m

4.2. Analysis of Fault Tolerance of Route Resolution Based on TLC Aggregation

Suppose a SWIM node has a collection of content objects, their OID contains the same first j TLC, and an OID contains J TLC, TLC1, TLC2, TLC3 and so on. TLC J In the SWIM node, the $(J-j)$ suffixes in each Ω OID are aggregated based on Bloom filter to generate an sOID, TLC1, TLC2, … , TLCj, … , Digest TLC $(j+1)$, … , Digest TLCi, $i = j+1, \ldots, J$. When another node requests content to resolve the route, a false match may occur. Due to suffix aggregation, even if the first j TLCs in the published sOID match the requested content sOID, or each digest TLC in the sOID can show that the corresponding TLC in the subscription OID exists, but the publishing node may not the requested content. When the false positive rate during TLC aggregation occurs, this route resolution error occurs at least once.

Use Ω_i $(i = j+1, \ldots, J)$ to represent the collection of the first TLC in the content OID held by the publishing node:

$$\Omega_i = \{TLC_i \; of \; OID : \; OID \in \Omega\} \tag{1}$$

It should be noted that some OIDs may have the same TLC. To have a false positive rate, k ≥ 1 filters are required, and the routing error resolution probability as follows.

$$\prod_{h=1}^{k}\left(\left(1-\frac{n_{ih}}{u_{ih}}\right)p_{f}, i_h\right) \tag{2}$$

Among them, $n_i = |\Omega|$ represents the number of different TLCs in the set, and u_i represents the number of possible values obtained by the TLC i.

The exact matching probability of the remaining $(J - K)$. Bloom filters is $\prod_{h' = k = 1}^{J} \frac{n_{ih'}}{u_{ih'}}$. Therefore, the false positive rate of the Bloom filter is:

$$\begin{aligned} p_e^{bloom} &= \sum_{k = 1}^{J-j} \sum_{j+1 \leq i_1 < \ldots < i_k \leq J} \prod_{h = 1}^{k} ((1 - \tfrac{n_{ih}}{u_{ih}}) p_f, i_h) \prod_{h' = k = 1}^{J} (\tfrac{n_{ih'}}{u_{ih'}}) \\ &= \prod_{i = J+1}^{J} ((1 - \tfrac{n_i}{u_i}) p_f, i + \tfrac{n_i}{u_i}) - \prod_{i = j+1}^{J} \tfrac{n_i}{u_i} \end{aligned} \tag{3}$$

where P_f is the single false positive rate associated with the Bloom filter.

In contrast, the prefix aggregation discards the last $(J - j)$ TLC in OID. When the best match occurs when the prefix TLC matches, as long as one of the suffix $(J - j)$ TLC does not match, the error rate will occur. The occurrence rate is:

$$p_e^{prefix} = 1 - \prod_{i = j+1}^{J} \frac{n_i}{u_i} \tag{4}$$

In the array experiment, for convenience of description, set $\frac{n_i}{u_i}$ to r. Among them, $i = j + 1, \ldots, J$.

The OID contains $J = 6$ TLCs, $(J - j)$ contains the maximum of three suffix aggregations; u_i is set to a normal value of 500, and the size of the Bloom filter with $i = 4, 5, 6$ in TLCi is set to 128 bits.

To conduct comparative experiments, the probability of parsing errors in the case of traditional prefix aggregation was tested. The analytical error probability function changes with the number of TLCs in OID and rOID as shown in Figure 9.

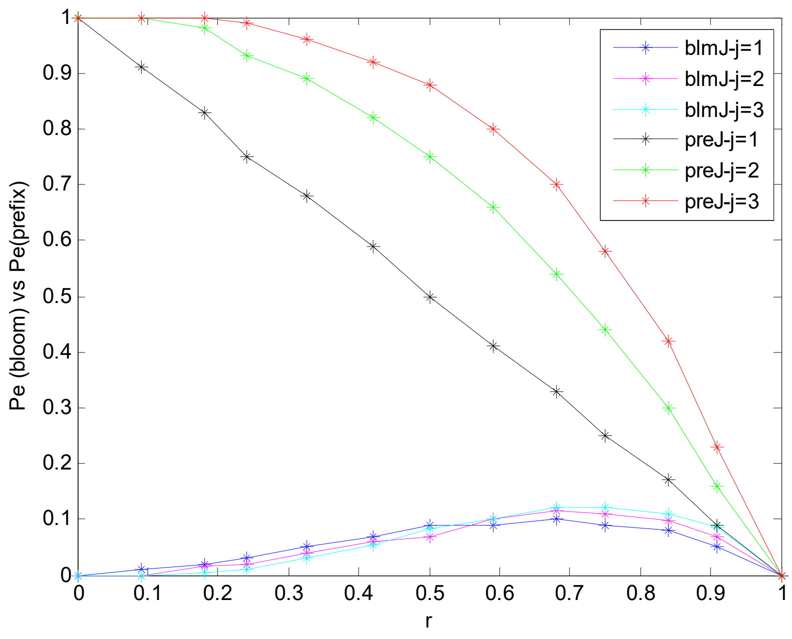

Figure 9. Comparison of error resolution rate between Bloom filter and prefix aggregation.

As shown in Figure 9, the Bloom filter-based aggregation scheme has greatly improved performance compared to traditional prefixes. When r tends to 0, the performance of the Bloom filter is more obvious, mainly because the digest of the sOID in the Bloom filter provides more reliable information to detect whether there is a content cached in the publishing node. Although the Bloom filter exceeds the bandwidth occupancy, computing performance, and storage space, this effect is negligible. The resolution error probability tends to 0 when r tends to 1, because of this case, the publishing node owns almost every content object. When r takes an intermediate value, the error resolution rate reaches the maximum. When r is small, the error rate decreases as the number of TLCs in the collection increases, because Bloom filters rarely exhibit false positives. However, when r starts to increase, the trend is just the opposite. When r is large, the reliability of the Bloom filter decreases. The Bloom filter aggregation scheme is almost the same as the prefix aggregation scheme. However, when the number of TLCs in the prefix starts to decrease, the performance of the prefix aggregation scheme is even worse.

4.3. Analysis of Search Efficiency and Scalability Based on Hybrid Naming

The naming scheme based on the TLC hybrid structure proposed in this paper improves the search efficiency mainly in the following two aspects:

i Compared with a single TLC naming method, the naming mechanism based on "layer-plane" can effectively control the length of resource naming.
ii After route aggregation, you can use Bloom Filter to quickly find the content of plane components.

To check the search efficiency of this naming method, different TLC layers were used to analyze the impact, and the query time of TLC structures with different layers was obtained, as shown in Figure 10.

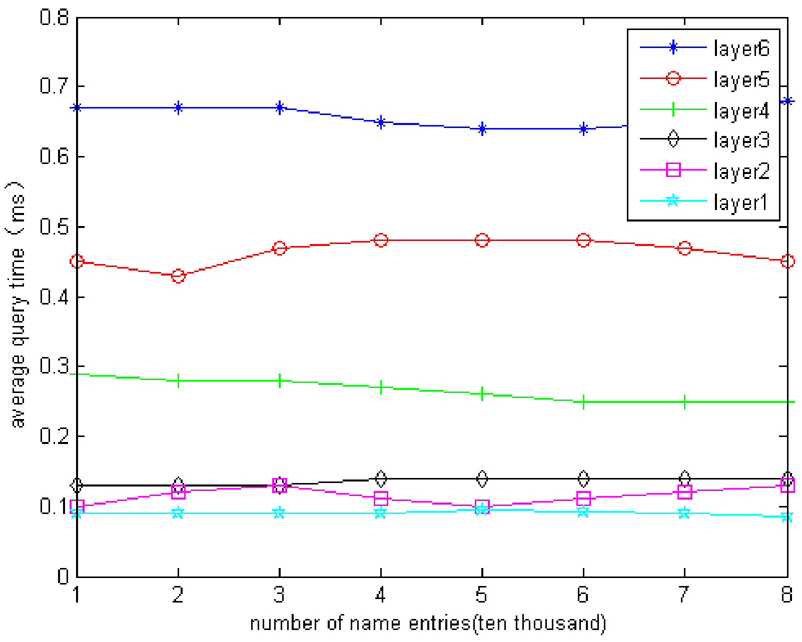

Figure 10. Comparison of query time of different layers in the mixed naming method.

It can be concluded from Figure 10 that the more layers of names, the longer the average query time, and the mixed naming scheme can control the number of names in a certain range. Therefore, compared with a single hierarchical structure, this solution can effectively reduce the query time of the name.

The comparison of query time can indirectly reflect the scalability of naming. The time comparison for a large-scale name search is shown in Figure 11.

It can be seen from Figure 11 that when the number of query levels of the name is greater than 5, and the number of queries is greater than 160,000, the time used increases sharply, and the search work cannot be carried out normally, which explains that the scalability is low. However, when the number of layers is limited, the search time increases slightly, but it is generally controllable. Through the analysis of this experiment, it can be concluded that the naming and searching method of the mixed method can control the overall length of the resource name and improve the scalability.

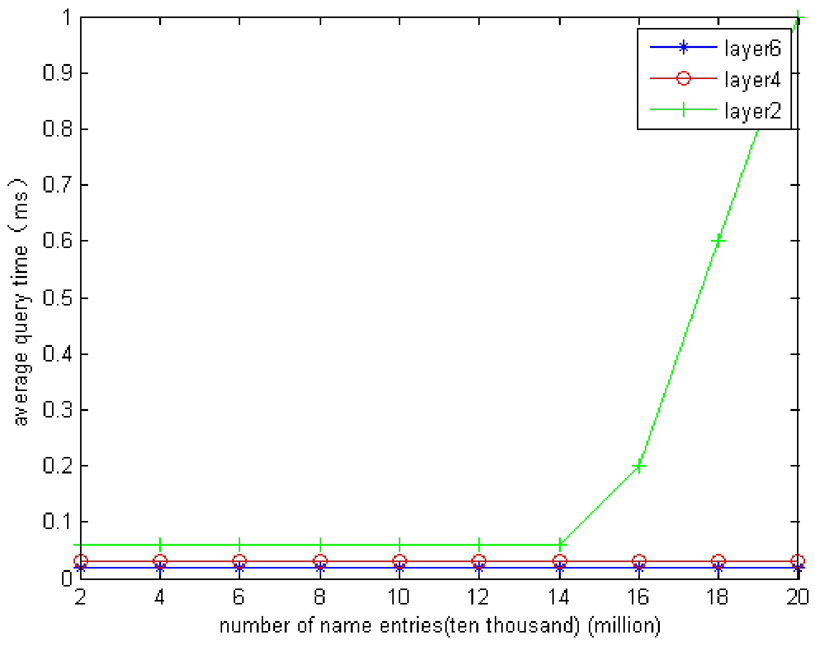

Figure 11. Comparison of query time for large-scale content names.

4.4. Comparative Analysis

4.4.1. Query Time Analysis of Different Naming Schemes

To obtain the comparison results of the average query time based on TLC structure and hierarchical naming and flat naming under different naming levels. First, each information element in the collected data set is divided into the form of TLC, and different layers are divided into one layer, two layers, three layers, and up to six layers. Under the division of different naming layers, the average query time comparison results based on TLC structure and hierarchical naming and flat naming are shown in Figure 12.

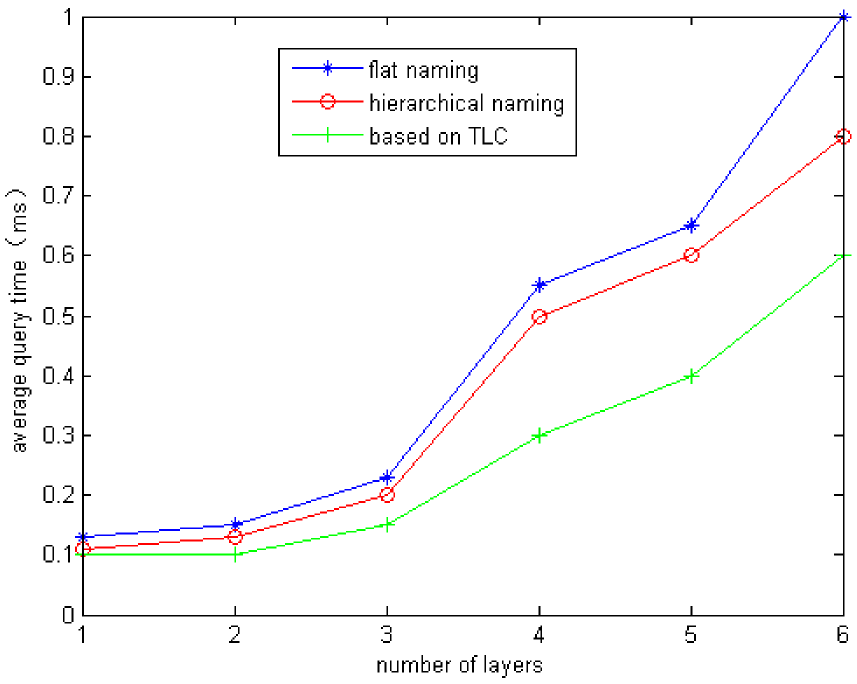

Figure 12. Comparison of the average query time under different numbers of layers.

As can be seen from Figure 12, when the number of layers is low (generally less than three layers), the query time of the three naming methods is almost the same. But when the number of layers is greater than two layers, the three naming methods begin to change. The query time of the naming method based on the TLC structure is significantly shorter than that of the flat naming method and hierarchical naming method. Through the above comparative experiments, it is proved that the naming method based on the TLC structure has a great advantage in query time compared with hierarchical naming and flat naming.

4.4.2. Scalability Analysis of Different Naming Schemes

Table 6 is a comparison chart of different naming methods in the case of six layers and different data amounts. The number unit is one million, and the query time is milliseconds. When the number of queries is greater than 160,000, the time for hierarchical naming and flat naming rises sharply, so the search work cannot be carried out normally, indicating that the scalability is low. Although the hybrid naming search time based on the TLC structure has increased slightly, it is generally controllable. Through the analysis of this experiment, it can be concluded that the naming and searching method of the mixed method can control the overall length of the resource name and improve the scalability.

Table 6. Query time of different methods under different amount.

Scheme	**Query Time of Different Methods Under Different Data Volume**									
	2	**4**	**6**	**8**	**10**	**12**	**14**	**16**	**18**	**20**
naming based on TLC structure	0.06	0.06	0.061	0.06	0.061	0.061	0.062	0.062	0.061	0.061
hierarchical naming	0.058	0.052	0.055	0.060	0.060	0.059	0.06	0.2	0.6	1
flat naming	0.063	0.061	0.064	0.061	0.060	0.061	0.061	0.22	0.65	0.99

4.4.3. Analysis of Detection Rate and False Alarm Rate of Different Schemes under Attack

This paper simulates two types of attacks based on the principle of the attack on named data Distributed Denial of Service (DDoS), namely attack 1 flood attack and random high-load attack named 2. It can be seen from Table 7 that the DDoS attack detection model based on TLC structure naming compares with the hierarchical naming and flat naming attacks in terms of detection rate and false positive is better. The detection rate for attack 1 is better, and the false alarm rate is higher, which is 99.96 and 0.003, respectively. The detection rate and false alarm rate of attack 2 are slightly worse, which are 96 and 0.85, respectively. However, in general, TLC-based naming is superior to hierarchical naming and flat naming, indicating that the former naming method is more secure and can effectively prevent DDoS attacks.

Table 7. Comparison of detection rate (DR) and false positive rate (FPR) results of different methods in attack 1 and attack 2.

Scheme	**Detection Rate (DR) and False Positive Rate (FPR) of Different Methods**					
	Attack 1		**Attack 2**		**Comprehensive Situation**	
	DR	**FPR**	**DR**	**FPR**	**DR**	**FPR**
Naming based on TLC structure	99.96	0.003	96	0.855	97.5	0.42
Hierarchical naming	99	5.76	91.89	9.91	95.9	7.84
Flat naming	98	3.46	90.56	9.45	95.2	6.25

5. Conclusions

SWIM is an infrastructure for information sharing and transmission of networked next-generation air traffic management promoted by ICAO. Its main function is to quickly and seamlessly communicate a wide variety of air traffic control business data. Therefore, the naming of SWIM data involves whether a large number of data can be accessed and used correctly, which is related to the normal operation of air traffic control services and ensuring flight safety. Based on the analysis of the various advantages and disadvantages of SWIM naming schemes, given their existing problems, this paper adopts the TLC structure and proposes a hybrid SWIM data naming scheme based on the TLC structure. It uses Bloom aggregation technology to provide routing select. Compared with the traditional prefix-based route aggregation, our proposed naming scheme can greatly reduce the resolution rate of routing errors, and at the same time improve the performance of the content name search rate, and meet the uniqueness, durability and reliability requirements.

Author Contributions: Aiming at the problems in SWIM (System-Wide Information Management), such as the variety of data and the inconsistency of data naming methods, etc, Z.W. proposes a new TLC (type length content) structure hybrid data naming scheme, and puts forward the algorithm and principle of the scheme. According to the proposed algorithm and principle, B.C. did experimental simulation to verify the results, and the simulation experiment proved that this scheme is better than prefix-based aggregation in terms of route recognition error probability. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Scientific Research Project of Tianjin Municipal Education Commission (2019KJ117), and the Fundamental Research Funds for the Central Universities of China (3122020076 and 3122019051).

Conflicts of Interest: The authors declare that there is no conflict of interests regarding the publication of this article.

References

1. Di Crescenzo, D.; Strano, A.; Trausmuth, G. SWIM: A next generation atm information bus-the swim-suit prototype. In Proceedings of the 14th IEEE International Enterprise Distributed Object Computing Conference Workshops, Vitoria, Spain, 25–29 October 2010; pp. 41–46.
2. Bob, S. Security architecture for system wide information management. In Proceedings of the 24th 2005 Digital Avionics Systems Conference, Washington, DC, USA, 30 October–3 November 2005.
3. Meserole, J.S.; Moore, J.W. What is system wide information management (SWIM)? *IEEE Aerosp. Electron. Syst. Mag.* **2017**, *22*, 13–19. [CrossRef]
4. Westmark, V.R. A definition for information system survivability. In Proceedings of the 37th Annual Hawaii International Conference on System Sciences, Big Island, HI, USA, 5–8 January 2004; p. 40.
5. Balakrishnan, K.; Leu, A.; Prabhu, V.; Veoni, J. A framework for performance modeling of SWIM. In Proceedings of the 2012 Integrated Communications, Navigation and Surveillance Conference, Herndon, VA, USA, 24–26 April 2012.
6. Lu, X.; Koga, T. Real-time oriented system wide information management for service assurance. In Proceedings of the 2015 IEEE Twelfth International Symposium on Autonomous Decentralized Systems (ISADS), Taichung, Taiwan, 25–27 March 2015; pp. 175–180.
7. Taylor, M.S. System-wide information management for aeronautical communications. In Proceedings of the 23rd Digital Avionics Systems Conference, Salt Lake City, UT, USA, 28 October 2004.
8. Moallemi, M.; Castro-Peña, C.A.; Towhidnejad, M.; Abraham, B. Information security in the aircraft access to system wide information management infrastructure. In Proceedings of the 2016 Integrated Communications Navigation and Surveillance (ICNS), Herndon, VA, USA, 19–21 April 2016.
9. Zhang, H.; Su, W. Fundamental research on the architecture of new network—Universal network and pervasive services. *Acta Electron. Sin.* **2007**, *4*, 593–598.
10. Koponen, T.; Chawla, M.; Chun, B.; Ermolinskiy, A.; Kim, K.H.; Shenker, S.; Stoica, I. A data-oriented (and beyond) network architecture. In Proceedings of the SIGCOMM Computer Communications, New York, NY, USA, 13 October 2007; pp. 181–192.
11. Jacobson, V. Networking named content. In Proceedings of the ACM CoNEXT, Rome, Italy, 26 March 2010.

12. Trossen, D.; Sarela, M.; Sollins, K. Arguments for an information-centric internet working architecture. In Proceedings of the ACM SIGCOMM Computer Communication Review, New York, NY, USA, 15 April 2010; Volume 40, pp. 26–33.
13. Balakrishnan, H.; Lakshminarayanan, K.; Ratnasamy, S.; Shenker, S.; Stoica, I.; Walfish, M. A layered naming architecture for the internet. In Proceedings of the ACM SIGCOMM Computer Communication Review, Portland, OR, USA, 30 August–3 September 2004.
14. Mealling, M.; Daniel, J.R. RFC 2483—URI Resolution Services Necessary for URN Resolution. 1999. Available online: ftp://ftp.internic.net/rfc/rfc2483.txt (accessed on 1 March 2020).
15. Daigle, L.; Van Gulik, D. URN Namespace Definition Mechanisms. Bcp Rfc. 2002. Available online: https://www.researchgate.net/publication/242370308_URN_Namespace_Definition_Mechanisms (accessed on 17 February 2020).
16. Coulouris, G.; Dollimore, J.; Kindberg, T. *Distributed Systems: Concepts and Design*, 3rd ed.; Addison-Wesley: New York, NY, USA, 2001.
17. XORP Inter-Process Communication Library Overview. 2003. Available online: http://www.xorp.org/releases/current/docs/libxipc/libxipcoverview.pdf (accessed on 23 February 2020).
18. Ratnasamy, S.; Francis, P.; Handley, M.; Karp, R.; Shenker, S. A Scalable Content Addressable Network. Proc. ACM SIGCOMM. 2001. Available online: http://citeseer.nj.nec.com/ratnasamy01scalable.html (accessed on 29 January 2020).
19. Sun, Q. *Research on the Packet Labeling and Lookup Technology in the Next Generation Internet*; BUPT: Beijing, China, 2020.
20. Qu, H.Q. *Multi-Label Classification Algorithms Based on Label-Specific Features and Mutual Neighbor*; Zhejiangshifandaxue: Jinhua, China, 2012.
21. LI, Y. *A Hierarchical Namespace and Resolution System Based on Content and Service*; BUPT: Beijing, China, 2014.
22. Mao, W. *Research on Internet Resource Identification and Addressing Technology*; Graduate School of Chinese Academy of Sciences: Beijing, China, 2006.
23. Berners-Lee, T. RFC1630: Universal Resource Identifiers in WWW. Available online: http://www.ietf.org/rfc/1630.1994 (accessed on 21 February 2020).
24. Moats, R. RFC2141: URN Syntax. 1997. Available online: http://www.ietf.org/rfc/ (accessed on 17 February 2020).
25. Quan, W. *Research on Resource Naming and Distribution for Future Networks*; BUPT: Beijing, China, 2014.
26. Caesar, M.; Condie, T.; Kannan, J.; Lakshminarayanan, K.; Stoica, I.; Shenker, S. ROFL: Routing on flat labels. In Proceedings of the SIGCOMM Computer Communications, New York, NY, USA, 13 May 2006.
27. Jacobson, V.; Smetters, D.K.; Thornton, J.D.; Plass, M.F.; Briggs, N.H.; Braynard, R.L. *Networking Named Content*; CoNEXT'09: New York, NY, USA, 2009; pp. 1–12.
28. Gritter, M.; Cheriton, D.R. *An Architecture for Content Routing Support in the Internet*, 3rd ed.; USENIX Symposium on Internet Technologies and Systems: New York, NY, USA, 2001; pp. 37–48.
29. Visala, K.; Lagutin, D.; Tarkoma, S. *An Inter-Domain Data-Oriented Routing Architecture*; Wksp on Rearchitecting the Internet: New York, NY, USA, 2009; pp. 55–60.
30. Raychaudhuri, D. *MobilityFirst Vision & Technical Approach Summary*; MobilityFirst External Advisory Board Meeting: New York, NY, USA, 2011.
31. Fan, L.; Cao, P.; Almeida, J.; Broder, A.Z. Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol. In Proceedings of the ACM SIGCOMM'98 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, Madison, MW, USA, 21 October 1998.
32. Choi, J.; Han, J.; Cho, E.; Kwon, T.; Choi, Y. A survey on content-oriented networking for efficient content delivery. In Proceedings of the IEEE Communications Magazine, New York, NY, USA, 7 March 2011.
33. Goel, A.; Gupta, P. *Small Subset Queries and Bloom Filters Using Ternary Associative Memories, with Applications*; ACM: New York, NY, USA, 2010.
34. Tourani, R.; Mick, T.; Misra, S.; Panwar, G. Security, privacy, and access control in information-centric networking: A survey. *IEEE Commun. Surv. Tutor.* **2016**, *20*, 566–600. [CrossRef]
35. Ganesan, P.; Gummadi, K.; Garcia-Molina, H. Canon in G major: Designing DHTs with hierarchical structure. In Proceedings of the International Conference on Distributed Computing Systems, New York, NY, USA, 22–26 March 2003.

36. Gupta, A.; Liskov, B.; Rodrigues, R. Efficient Routing for Peer-to-Peer Overlays. In Proceedings of the Conference on Symposium on Networked Systems Design & Implementation, New York, NY, USA, 29 March 2004.
37. ICAO. Manual on System Wide Information Management (SWIM) Concept. Available online: https://www.icao.int/airnavigation/IMP/Documents/SWIM%20Concept%20V2%20Draft%20with%20DISCLAIMER.pdf (accessed on 3 February 2020).
38. Ahmed, R.; Boutaba, R.; Cuervo, F.; Iraqi, Y.; Li, T.; Limam, N.; Xiao, J.; Ziembicki, J. Service naming in large-scale and multi-domain networks. *IEEE Commun. Surv. Tutor.* **2016**, *7*, 38–54. [CrossRef]
39. Ahlgren, B.; Dannewitz, C.; Imbrenda, C.; Kutscher, D.; Ohlman, B. A survey of information-centric networking. *IEEE Commun. Mag.* **2012**, *50*, 26–36. [CrossRef]
40. Liu, H.; Zhang, D. A TLV-structured data naming scheme for content-oriented networking. In Proceedings of the 2012 IEEE International Conference on Communications (ICC), Ottawa, ON, Canada, 10–15 June 2012.
41. SWIM Flight Data Publication Service (SFDPS). SFDPS Release 1.3.1. 2018. Available online: http://aixm.aero/sites/aixm.aero/files/imce/library/ATIEC_2016/13_day2_swim_flight_data_publication_service_integrating_swim_apps_to_visualize_aviation_data.pdf (accessed on 5 February 2020).
42. Lu, J.; Yang, T.; Wang, Y.; Dai, H.; Chen, X.; Jin, L.; Song, H.; Liu, B. Low computational cost bloom filters. *IEEE/ACM Trans. Netw.* **2018**, *26*, 2254–2267. [CrossRef]