A TLV-Structured Data Naming Scheme for Content-Oriented Networking

Hang Liu
InterDigital Communications, LLC
781 Third Avenue
King of Prussia, PA 19406

Dan Zhang
WINLAB, Rutgers University
671 Route 1 South
North Brunswick, NJ 08902

***Abstract*—Content-oriented networking (CON) addresses the inefficiency of IP networks in supporting content distribution, by decoupling location from identity at the network level, and retrieving a content object through its name or identifier, instead of its storage location or host IP address. A naming scheme is critical in CON designs because a name is used to identify the object and acts as the key for location resolution and routing. This paper proposes a new TLV-structured naming scheme with Bloom filter summarization. The proposed scheme satisfies the requirements of name uniqueness and persistence, and enables more secure content-oriented trust model and effective routing state aggregation for scalability. It also addresses the "suffix-hole" problem encountered in conventional prefix-based routing aggregation. We analyzed the performance of the proposed TLV-structured naming and Bloom filter aggregation scheme and our early evaluation results shows that the proposed scheme greatly outperforms the prefix-based aggregation in terms of routing resolution error probability.**

Keywords-content-oriented networking; information-centric networking; future Internet; naming

I. INTRODUCTION

The Internet is primarily used for content retrieval and information access. Multimedia content traffic, especially video traffic is growing at an exponential rate. Another trend is that more and more users are accessing information over networks using their smart mobile devices equipped with multiple network interfaces. However today's IP networks were built to interconnect fixed computing nodes (terminals, servers, etc.), in which communications are based on the node's IP addresses. Such discrepancy causes inefficiencies in networks as well as in application designs. For example, a video clip may be encoded in multiple formats and resolutions, and can be replicated at multiple hosting locations. A mobile user is interested in the content itself, rather than its host. It is desirable that the content is delivered from the best/closest host, and the delivery matches the user device's operating conditions in the proper format and at the best available quality. The TCP/IP architecture is not "intelligent" enough to meet the user's requirements because it relies on the host-to-host communication model. This leads to developing complex application-specific solutions such as content delivery networks (CDNs) and P2P overlays. But these application-specific solutions are costly and inefficient because they have limited information about underlying network status and require extra management mechanisms.

Content-oriented networking (also referred to as information-centric or name-oriented networking) has been considered as an innovative architecture for the future Internet to address the above mismatch, which decouples content from hosts at the network level and retrieves a content object by its name (identifier), instead of its storage location (host IP address). This new networking paradigm naturally enables mechanisms such as in-network caching and delivery of content from the best location(s) to optimize bandwidth and improve users' content access performance. It also allows network operators to interconnect their content service networks across domains to make more content available and share resources for cost reduction, as well as frees application developers from reinventing application-specific delivery mechanisms. Moreover, it solves or mitigates other Internet problems such as mobility and multi-homing.

There are many challenges in the content-oriented networking (CON) design, such as how to name content objects, locate a particular content object, and deliver the content in a scalable and efficient way. In this paper we focus on the naming issue. The naming scheme is critical in CON designs because a name or an object identifier (OID) is used to identify the content object and acts as the key for location resolution and routing mechanisms. Note that the terms, "name", "identifier", and "OID" are used interchangeably in this paper, unless otherwise stated.

For scalable, efficient, and secure content access, one would want that a naming scheme supports the following features.

- **Uniqueness:** An OID should be globally unique to identify an object.
- **Persistence:** Once an OID is assigned to a content object, users would like that the OID remain valid as long as the underlying object itself is available and not changed, that is, users would like to access the object with this name even if the location and administrative domain of the object is changed.
- **Trustworthiness:** CON designs secure the content rather than the communication path between two communication points. The content is signed by the original content owner or creator (e.g. Disney). End users and network elements such as CON routers can authenticate the content by verifying the signature. Naming plays a role in this content-oriented trust model from two aspects: binding between the user-friendly human-readable name and its corresponding CON OID, and binding between the OID and the public key

(to authenticate the data). The integrity needs to be ensured from the user-friendly name to the CON OID and from the OID to the content.

- **Scalability:** There are a huge number of content objects on the Internet and their locations frequently change due to dynamic caching and replacement. It is desirable that certain aggregation can be used to reduce the number of routing states and associated routing update overhead for scalability.

There are two naming schemes proposed in the literature: flat self-certifying names [1, 2, 5, 6] and hierarchical identifiers [3, 4]. However, both of the approaches have their share of problems. The former defines an object name as a hash of a public key and a label. It meets uniqueness and persistence requirements, and allows network elements to verify the integrity of the data without need of any external mechanism, preventing denial-of-service attacks. But some external mechanism is required for binding the user-friendly name to the flat self-certifying name. This process can be subject to substitution attacks. In addition, flat names, consisting of a random looking series of bits with no semantics and structure, are difficult to aggregate for routing scalability.

To improve scalability, a hierarchical content naming scheme with a structure like binary-encoded URLs is introduced [3]. This scheme partially addresses the first aspect of the trustworthiness requirements: binding between the user-friendly name and its corresponding CON OID. However it requires that a user knows the encoding rules employed by the content owner or naming authority to map the human-readable name to the OID. Moreover an external mechanism is needed to bind the CON OID to the key in order to authenticate the content matching the requested OID. Not only the end users but also CON infrastructure such as content routers should know the public key for a requested content OID. Otherwise the network may keep delivering false data even if the end user can authenticate the content, leading to denial-of-service attacks.

Hierarchical naming structure can help scalability. It was proposed to aggregate routing entries for the hierarchical OIDs with a common prefix to reduce the number of routing states, just as network address prefix aggregation in IP routing [3]. For example, if all the content objects whose name starts with "example.com" are stored in a single node, a single entry is needed for these content objects in CON routers to route a content request to this node. However as content objects are cached or replicated at multiple places, prefix-based aggregation becomes less effective. A caching node may not have all the content objects for a given prefix. For example, there are a total of N (e.g. N=2000) content objects with the prefix "example.com," and a node only stores M (e.g. M=1000) of them. If the prefix-based aggregation is used to avoid the M routing states and associated routing update overhead, a lot of information will be lost. This is because a routing announcement with prefix-based aggregation can only express "some of the content objects with this prefix (e.g. example.com) can be reached via me." We refer to this as a suffix hole. Suffix holes introduce uncertainty in locating a particular content object and then reduce routing efficiency. In addition, URLs or DNS names have their traditional semantics, somehow related to the location. The components in the above hierarchical naming (e.g. example.com/video/WidgetA.mpg) needs to be given new meanings ("example.com" is the object owner or creator, not host, "video" is the object type, not directory, WidgetA.mpg is the object title and format, not the file name). Otherwise, the name becomes misleading if the administrative domain or location of the object is changed.

In this paper, we propose a new naming and aggregation scheme, in which a content object ID (OID) consists of a set of variable-size information elements (IEs), and each IE is encoded as type-length-value (TLV). The information elements can flexibly form hierarchical or peer relationships. The network imposes no restrictions to the OID assignment except the TLV structure, and does not have to know the meaning of types and values except certain "well-known" types. This TLV-structured namespace can provide name uniqueness and persistence, and enable better scalability and trustworthiness. To address the suffix-hole problem, we propose to use a Bloom filter [10] to summarize the aggregated elements and generate digest elements. Then a routing advertisement can express more accurate information such as "the content objects with this prefix and digest value can be reached via me." Furthermore, a content router or a distributed name resolution function can flexibly control the aggregation degree based on the distance or popularity of content objects in order to balance between needed resources and routing information compression.

The remainder of the paper is organized as follows. Section II presents a high-level overview of the system model. In Section III, we describe the new TLV naming and Bloom filter aggregation scheme. In Section IV, we analyze the performance of the proposed naming and aggregation scheme, and present the evaluation results. The conclusions and future work are discussed in Section V.

II. SYSTEM MODEL

Before we present the proposed naming scheme, we discuss the CON system model in order to have a more complete picture. In CON, a network element, e.g. a content router, has storage capability. Content objects are cached or replicated at multiple places. The CON employs a publish/subscribe model, in which a content provider publishes the availability of its objects and a user subscribes to (requests) the objects. The object name is used as the key in this publish/subscribe operation. Thus the providers and users do not have to know each other's locations and be online at the same time.

CON networks should be able to locate a requested content object and route the content request to the closest or best host(s) for serving the request based on the requested object name. Different routing algorithms can be used such as advertisement via simple flooding or distributed hash table (DHT) [11, 12, 13]. Once a CON router receives a request, it can either directly use the name for routing [2, 3] or maps the name to a location or address through a resolution mechanism [1, 4, 6]. The name of an object is independent of its location. Decoupling of naming and location as well as name-based routing enables CON to natively support in-network caching, mobility, and multihoming.

In CON, the requested content object may be delivered from a network element other than the origin server. It takes a content-oriented trust model by leveraging public key cryptography. The object is signed by the original content

owner using a private key that binds the CON OID to the content data. A user needs to know the CON OID of the desired content and the content owner's public key in order to retrieve and authenticate the content. CON network elements also should bind the requested content OID to the corresponding public key in order to prevent attackers from sending false content for denial-of service attacks. From the above discussions, we can see that naming plays an important role in CON routing and trust models. The proposed TLV naming scheme can be incorporated with a flooding-based or DHT-based routing mechanism.

III. CONTENT NAMING AND AGGREGATION

An OID in the proposed TLV naming scheme is composed of a set of variable-sized information elements (IEs), and each IE is encoded as type-length-value (TLV). The type and length fields are fixed in size, e.g. 1 octet, and the value field is of variable size. The type field indicates the kind of elements and the length field defines the size of the value field contained in the TLV. A TLV may contain multiple sub-TLVs in its value field.

The IEs in a content OID can have hierarchical or peer relationships, which offer flexibility and extensibility in content naming. They can be processed very fast using generalized parsing functions in binary format. It is also fairly easy to map text or XML-based human readable names to TLVs. During the content request routing and data transport process, CON routers (CRs) in the network do not need to know the meaning of type and value fields in a TLV, except certain "well-known" types, such as digest TLV, although the type and value may be meaningful for some higher layer applications. The network imposes no restrictions to the OID assignment except the TLV structure. A CON router just uses the length field to parse the TLV elements and treats the whole element as a binary number in object publishing, routing aggregation, and resolution process as described below.

As an example, a content owner or a naming authority may assign a movie clip with an OID like "*organizationTLV-categoryTLV-subcategoryTLV-titleTLV-formatTLV-segmentationTLV*." Note that characters "-" are used to separate the elements for notation purpose, and are not part of OIDs. This content OID naming reflects the application-level human-readable name, including content owner's organization ID, category ID, subcategory ID, title ID, format ID (a code to indicate audio/video coding format and resolution), and segmentation ID (a code to indicate the temporal starting point and duration of this movie clip as well as its version number). Note that an ID here is just an assigned number. Another content owner may name its content objects using a different convention. This TLV naming scheme can be used for naming anything, not just content.

The TLV naming meets uniqueness and persistence requirements. As long as a content owner has a globally unique organization ID (e.g. the hash of its public key or its binary-encoded DNS name), locally unique content OIDs assigned by this content owner can be guaranteed to be globally unique. Persistence follows from the fact that the OIDs don't refer to location, and thus the object can be hosted anywhere. The same OID can be used even if the object changes its administrative domain. In addition, for flexibility, a user can issue a request with an OID that contains wild card elements. A well-known wild card TLV or a wild card value in a TLV can be defined for this purpose.

To handle a large number of content objects on the Internet, aggregation of content location information is necessary. However, as mentioned before, the conventional prefix-based routing aggregation is not effective because a content object may be replicated at multiple locations and a node may not have the location information of all the content objects whose OIDs starts with a given prefix, leading to suffix holes. The TLV structure enables to employ better aggregation mechanisms because an OID can be easily extended to carry the digest information. We propose to apply Bloom filters [7] on the aggregated TLV elements to generate digest elements and a CON router to advertise both the prefix and digests for its content objects. Bloom filters are a computationally efficient hash-based scheme, allowing control of the error probability.

Assume a CON router has the location information for some content objects whose OIDs start with the same *organizationTLV, categoryTLV* and *subcategoryTLV*. We want to use a summary OID (sOID) to represent them. The sOID consists of the common *organizationTLV, categoryTLV* and *subcategoryTLV* (prefix) of these content objects and also the digests of their *titleTLVs*, *formatTLVs*, and *segmentationTLVs* (suffix). Its format is *organizationTLV-categoryTLV-subcategoryTLV-digestTLV*$_1$*-digestTLV*$_2$*-digestTLV*$_3$ where the values in *digestTLV*$_1$, *digestTLV*$_2$, and *digestTLV*$_3$ are generated by Bloom filters from the content *titleTLVs*, *formatTLVs*, and *segmentationTLVs*, respectively. Thus, the CON router can just use a single sOID to publish or announce the location information for these content objects, and other CON routers receiving the announcement only need to maintain one routing state for this sOID. By using the sOID in a routing announcement, it means "I have the location information for the content objects with this *organizationTLV, categoryTLV* and *subcategoryTLV,* and the digests of the title, format, and segmentation equal to the values in *digestTLV*$_1$, *digestTLV*$_2$, and *digestTLV*$_3$, respectively."

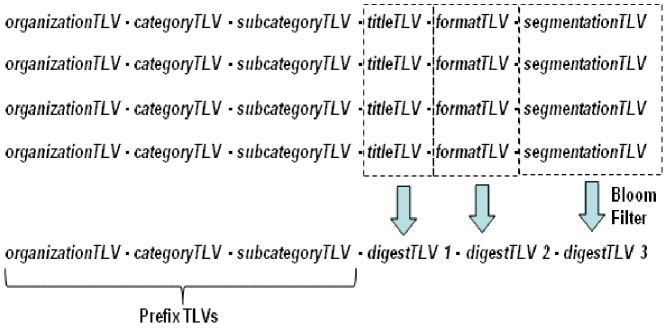

Figure 1. Summary OID generation

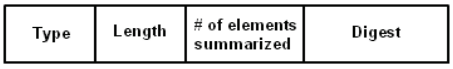

Figure 2. Digest TLV format

As shown in Fig. 1, to generate the value field of *digestTLV*$_1$, a Bloom filter is used on the *titleTLVs* of the content objects whose routing entries will be aggregated. The CON router treats a *titleTLV* element as a binary number in the

process and does not need to know its meaning. To build a digest with a Bloom filter, a bit array of m bits is allocated, with all bits initially set to *0*, and k independent hash functions, $h_1, h_2, ..., h_k$, are also needed, To add an element to the digest, feed it to each of the k hash functions to get k array positions, i.e. the element hashed with any of these functions gives a value between *1* and m, and the hashed value represents the position in the array. The bits at all the corresponding positions are set to *1*. The same process can be used to generate the digest values for format and segmentation TLVs. Fig. 2 shows the format of a digest TLV. Its type is "well-known" by all the CON routers. The value field of a digest TLV includes a fixed-size sub-field indicating the number of aggregated elements added to the digest; the rest of the value field is used to store the digest value itself.

In general, the OID for a content object consists of J TLVs, TLV_1-TLV_2-TLV_3 ... TLV_J. For the content objects whose OIDs have the common first j TLVs ($1 \le j \le J$), a CON router can use a summary OID to represent them so that only a single routing state for these content objects is needed. A sOID consists of a prefix (the first j common TLVs) and J-j digest TLVs, TLV_1-……TLV_j-$DigestTLV_{(j+1)}$ ... $DigestTLV_J$. The value in $DigestTLV_i$ ($i=j+1,...,J$) is obtained by taking the ith TLV from each of the OIDs and applying the Bloom filter on it.

A CON router can flexibly control the aggregation degree based on the popularity of the content objects or the distance to the location that the content objects reside. For example, no aggregation is performed in routing publishing messages for the content objects residing in the local domain, but a domain gateway router publishes the summary OIDs of its content objects to outside domains. The prefix size j, i.e. the number of non-aggregated TLVs in the sOID can be adjusted to balance between the network resources needed for maintaining routing states and the information loss. One learns from the sOID in the received publishing message that requests for the content objects with this prefix and digest may be served by this domain.

To query whether an element exists based on the digest, the element is fed to each of the k hash functions to calculate k array positions. If any of the corresponding bits in the array are *0* then the element is not present. If all of the corresponding bits in the array are *1* then the element is likely to be present. Note that with Bloom filters, false positives are possible, but false negatives are not. It is possible that a collision in the digest occurs if the corresponding bits in the digest have been set to *1* during the insertion of other elements. Then the digest incorrectly indicates an element is present. However, the probability of a collision can be controlled by designing appropriate filters and limiting the number of summarized elements that are added to a digest. Given a filter, when the number of elements added to a digest exceeds the limit, we can divide the elements into groups. Each group generates a digest. We also define a new TLV, *concatenated digest TLV*, which contains multiple digest sub-TLVs, each is generated from a group of aggregated elements.

For a match between a queried OID and a sOID, the corresponding prefix j un-summarized TLVs should be exactly the same and every digest TLV in the sOID should give a positive match to indicate that the corresponding element in the queried OID is likely to be present. Since a sOID carries the digests of the last J-j TLVs of the summarized content OIDs, it helps mitigate the suffix-hole problem while achieving routing scalability.

To support the content-oriented trust model, we define two well-known TLV types, *name signature TLV* and *key signature TLV*. The value field of the name signature TLV is a signature generated by the content owner with its private key to bind the content OID to the human readable name of the content object. This private key can be the same key used to sign the content data for binding between the content data and the content OID (called the content key). The key signature TLV contains the signature signed by a higher-level authority or a higher-level key of the content owner. It binds the content owner's content public key to its human-readable organization name. We also define extended OIDs (xOIDs) that consist of the OID appended by a name signature TLV and a key signature TLV. In addition, a content owner uses the hash of its content public key as its own organization ID, i.e. the value field of the organization TLV in the OID. Then the OID contains the content public key information and is self-certifying as described below.

The name signature TLV and the key signature TLV are only used by end users and need not be sent in the content requests. An end user can obtain the higher-level public key through other means. The number of the higher-level keys is much smaller than the number of the content keys so that it is easy to manage. A user does not have to know the content owner's OID naming and encoding convention. One can employ external mechanisms (e.g. search engines and personal contacts) to obtain the xOID of the content object he wants as well as the public key used to sign the content by the owner. Before issuing the content request, the user can verify the correctness of the content public key by checking the key signature in the xOID, and verifies the correctness of the content OID by checking the name signature in the xOID. This prevents OID substitution attacks. The content public key information in the OID enables CON network elements to verify the authenticity of the content data and prevent denial-of-service attacks without requiring the infrastructure to obtain and verify the content public key through external secure mechanisms. When a user requests a content object with its OID, the data is delivered along with the content public key and signature. Once a network element receives the data, public key and signature triplet, it can immediately verify the correctness of the public key by checking that the public key hashes to the organization ID of the requested OID and the integrity of the content by checking the signature. The xOID addresses the security weaknesses in the prior works and satisfies the trustworthy need.

IV. Analysis and Preliminary Results

We present in this section a simplified quantitative analysis of the proposed Bloom filter-based aggregation method to furnish a few first-order insights. As part of our future work, we plan to conduct more thorough study on the impact of routing state aggregation (not only Bloom filter-based but also other aggregation techniques) to CON scalability.

For a Bloom filter with m bits in the array, k hash functions, and n aggregated elements, the false positive probability is approximately given as [10]

$$p_f = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^k. \quad (1)$$

Figure 3 shows the false positive probability p_f as a function of the number of aggregated elements n in the filter and the filter size m, assuming an optimal value of k is used. The probability of false positives decreases as m increases, and increases as n increases. Note that *sOIDs* are only used in the content object publishing, and are not carried in the data packets. Thus there is great flexibility in designing the filter length to meet the requirements of false positive probability and the maximum number of elements to be aggregated in a filter. For example, a false positive probability of 10^{-4} can be achieved with a 32-byte long Bloom filter when 27 elements are inserted in the filter (reducing the number of entries in the routing table by 27 times).

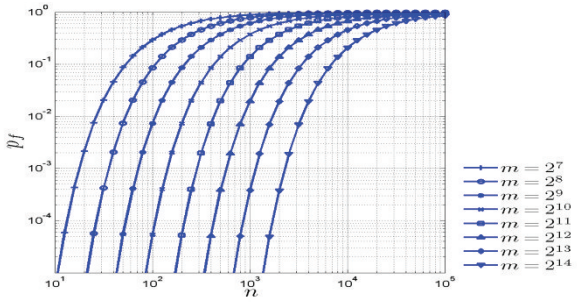

Figure 3. The false positive probability of Bloom Filters

Consider a CON node that hosts a set Ω of content objects whose OIDs have the common first j TLVs. An OID consists of J TLVs: TLV_1-TLV_2-TLV_3 ... TLV_J. The CON node performs Bloom filter-based aggregation for the |Ω| OIDs on their *(J-j)* suffix TLVs to generate a sOID TLV_1-……TLV_j-$DigestTLV_{(j+1)}$ ... $DigestTLV_J$ and publishes the sOID, rather than |Ω| OIDs, for scalability purpose. Bloom filter i is used to generate $DigestTLV_i$, $i=j+1, \ldots, J$. When another CON router resolves the location for a content request, the false positive match may occur due to aggregation. Even if the prefix j TLVs of the published sOID match those of the requested OID and every digest TLV in the sOID indicates that the corresponding TLV elements of the requested OID is likely to be present, the publishing CON node may actually not possess the requested content object. This routing resolution error occurs if at least one false positive occurs on an aggregated TLV.

Let $\Omega_i (i = j+1, \ldots, J)$ denote the set of the ith TLV in the OIDs of the objects hosted by the publishing node,

$$\Omega_i = \{TLV_i of\ OID: \quad OID \in \Omega\} \ . \quad (2)$$

$n_i = |\Omega_i|$ denotes the number of distinct TLV_i values in the set. Note that some OIDs may have the same TLV_i value. Let u_i denote the number of potential values that TLV_i can take. The routing resolution error probability can be calculated as follows (we omit the detail proof due to the space limit, and let interested readers refer to [9]). To encounter a false positive, we need $k \geq 1$ filters $(j+1 \leq i_1 \cdots < i_k \leq J)$ each return a false positive, which happens with probability

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

and an exact match on the other $J-k$ tags, with probability

$$\prod_{h'=k+1}^{J}\left(\frac{n_{i_{h'}}}{u_{i_{h'}}}\right).$$

Therefore the false positive probability is given as

$$\begin{aligned} p_e^{\text{bloom}} &= \sum_{k=1}^{J-j} \sum_{j+1 \leq i_1 < \cdots < i_k \leq J} \prod_{h=1}^{k}\left(\left(1 - \frac{n_{i_h}}{u_{i_h}}\right)p_{f,i_h}\right) \prod_{h'=k+1}^{J}\left(\frac{n_{i_{h'}}}{u_{i_{h'}}}\right) \\ &= \prod_{i=j+1}^{J}\left(\left(1 - \frac{n_i}{u_i}\right)p_{f,i} + \frac{n_i}{u_i}\right) - \prod_{i=j+1}^{J}\frac{n_i}{u_i} \end{aligned} \quad (3)$$

where $p_{f,i}$ is the individual false positive probability associated with Bloom filter i as calculated in (1). As a comparison, the prefix aggregation scheme simply discards the last *(J-j)* TLVs in an OID for aggregation. When a perfect match occurs on the prefix TLVs, an error occurs if at least one of the (J-j) suffix TLVs does not match. This probability is given as

$$p_e^{\text{prefix}} = 1 - \prod_{i=j+1}^{J}\frac{n_i}{u_i}. \quad (4)$$

We carry out a numerical experiment. For ease of exposition, we set n_i/u_i to r uniformly for $i = j+1, \ldots, J$. Figure 4 shows the routing resolution error probability as a function of the ratio r and the number of aggregated TLVs in the OID (i.e., $J-j$). The OID consist of *J=6* TLVs. Up to *J-j=3* suffix TLVs are aggregated and u_i are set uniformly to 500 for TLV_i, *i=4, 5, 6*. The Bloom filter size is set to be 128 bytes. For comparison, we also show the routing resolution error probability if the conventional prefix aggregation is used.

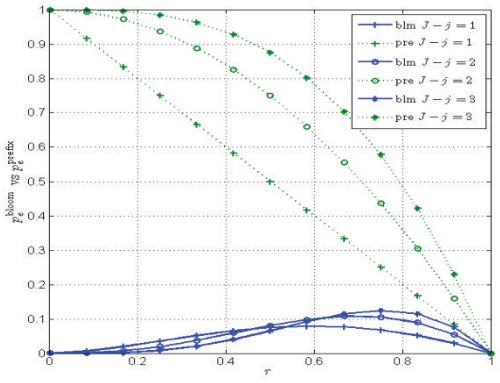

Figure 4. Routing resolution error probabilities of Bloom-filter and prefix aggregations

As shown in Fig. 4, the proposed Bloom filter aggregation greatly outperforms the conventional prefix aggregation. The benefits of Bloom-filter aggregation are more conspicuous as r approaches 0. This is because the Bloom filter digests in the sOID provide reliable extra information to test whether a requested OID is in the set of content objects hosted by the

publishing node. Although use of a Bloom filter has overhead in bandwidth, computing and memory usage, this overhead is minimal. The routing resolution error probability also decreases to 0 when r approaches 1 because in this case the publishing node has every possible object in possession. The routing resolution error peaks for moderate r when Bloom filters return more false positives. When r is small, the error probability decreases as the number of aggregated TLVs increases (i.e., increasing $(J-j)$) because the Bloom filters rarely return a false positive. However, this trend is reversed when r becomes large. With a large r, the Bloom filters become less reliable, and the Bloom-filter aggregation scheme degrades to the prefix aggregation scheme, whose performance worsens as the number of TLVs employed in the prefix shrinks.

V. Discussions and Future Work

This paper proposes a new content naming scheme using the structured TLVs and Bloom filter summarization to improve routing scalability and enable more secure content-oriented trust model. The early results show that the proposed scheme can greatly reduce routing resolution error probability compared to the conventional prefix-based routing aggregation.

For future work, we plan to implement the proposed scheme and integrate it in our CON system testbed. We'll also conduct extensive experiments to evaluate its performance and compare it with other naming schemes. In particular, we would like to get thorough understanding on the impact of routing state aggregation to the scalability.

References

[1] T. Koponen et al., "A Data-Oriented (and Beyond) Network Architecture," SIGCOMM '07, 2007, pp. 181–92.

[2] M. Caesar, T. Condie, J. Kannan, K. Lakshminarayanan, I. Stoica, and S. Shenker, "ROFL: Routing on Flat Labels," SIGCOMM, 2006.

[3] V. Jacobson et al., "Networking Named Content," CoNEXT '09, New York, NY, 2009, pp. 1–12.

[4] M. Gritter and D. R. Cheriton, "An Architecture for Content Routing Support in the Internet," 3rd Usenix Symp. Internet Technologies and Sys., 2001, pp. 37–48.

[5] K. Visala et al., "An Inter-Domain Data-Oriented Routing Architecture," Wksp on Rearchitecting the Internet, New York, NY, 2009, pp.55–60.

[6] D. Raychaudhuri, "MobilityFirst Vision & Technical Approach Summary," MobilityFirst External Advisory Board Meeting, Feb 2011.

[7] L. Fan, P. Cao, J. Almeida, A. Z. Broder, "Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol," ACM SIGCOMM '98.

[8] J. Choi, et al, "A Survey on Content-Oriented Networking for Efficient Content Delivery," IEEE Communications Magazine, March 2011.

[9] H. Liu, et. al. "An Information-Centric Networking Architecure," Tech. Rep., 2011.

[10] Goel, Ashish; Gupta, Pankaj, "Small subset queries and bloom filters using ternary associative memories, with applications", ACM Sigmetrics 2010.

[11] I. Stoica, et al, "Chord: a scalable peer-to-peer lookup service for Internet applications," SIGCOMM 2001.

[12] P. Ganesan, K. Gummadi, H. Garcia-Molina, "Canon in G major: designing DHTs with hierarchical structure," ICDCS, March 2004.

[13] A. Gupta, B. Liskov, and R. Rodrigues, "Efficient routing for peer-to-peer overlays," NSDI, March 2004.