

A Closer Look at Content-Centric Internet Research Projects

H.S. JEON^{ab}, I.S. CHOI^b, B.J. LEE^b, and H.Y. SONG^b

^a Graduate School of Innovation & Technology Management, KAIST(Korea Advanced Institute of Science and Technology)

^b ETRI(Electronics and Telecommunications Research Institute), Korea

{jjeonhs, ischoi, bjlee, hsong}@etri.re.kr

Abstract— Today's Internet services mainly aim at providing content, rather than simply exchanging data end-to-end. Hence content itself is getting more important and a number of clean-slate approaches to content-centric Internet have been studied. This paper examines current content-centric Internet research projects. In particular, we study these research projects with respect to content naming issues, content resolution issues, and delivery and caching issues.

Keywords— content, Internet, Future Internet, naming, networking

I. INTRODUCTION

Today's Internet services mainly aim at providing content, rather than simply exchanging data end-to-end. Especially, content services like Internet TV, Video on Demand (VoD), and file sharing have become popular and their traffic is rapidly growing now. According to the recent IP traffic forecast [1], the content service traffic will continue to be approximately 90 percent of global consumer traffic by 2015.

With these new patterns, the content replication and distribution issues have received considerable attention and it resulted in overlay networks such as Peer-to-Peer (P2P) networks and Content Distribution Networks (CDN). P2P networks allow nodes to share their contents by direct exchange and CDN places a number of surrogate origin content server at the network edge in order to enhance content delivery quality.

As content itself is getting more important, however, there are studied a number of approaches to clean-slate architectures beyond the overlay network. The clean-slate approaches suggest radical changes to the current Internet. They consider content as a main object and provide a general platform for obtaining the contents, not end-to-end communication between hosts. Such a resulting general platform is also considered as one of candidate architectures for Future Internet [2].

These clean-slate approaches have been studied under several research projects: SAIL [3], PURSUIT [4] and COAST [5] in Europe, and DONA [6] and CCN [7] in USA. This paper aims at providing an overview of the research projects. In particular, we study these research projects with respect to content naming issues, content resolution issues, and delivery and caching issues.

This paper is structured as follows. We discuss the basic design issues for content-centric Internet in Section 2.

Sections 3 to 7 describe current content-centric Internet research projects (DONA, CCN, NetInf, PURSUIT and COAST), and we concluded in Section 8.

II. BASIC DESIGN ISSUES

Content-centric Internet has been designed considering many challenging issues. In this section, we will discuss basic challenging issues which research projects have in common, content naming, content resolution, and content delivery and caching.

A. Content Naming Issue

As today's content is typically disseminated through Web service, Uniform Resource Locators (URLs) are considered as names for content and thus clients request contents with the URLs. The URL includes information that is resolved with specific IP address of the host storing content named in the URL. Hence, the request from client is routed to the storage location with the resolved IP address. This shows that the current content naming scheme is tightly coupled with storage locations.

Content-centric Internet basically lets the content name be decoupled from the storage location so that the name should be used to route the content request to suitable copies of content as well as original location of the content. Such a new name paradigm is mostly implemented by two different types: a *human-readable/hierarchical* name, and a *self-certifying/flat* name. A. Ghodsi in [8] also mentioned hybrid of a human-readable, hierarchical names and a self-certifying names.

B. Content Resolution Issue

With respect to the content resolution, i.e., content finding, there are two approaches in the prior research projects: *lookup-by-name* and *route-by-name*.

Lookup-by-name approach literally is to look storage locations up in a database which maintains information on location of all contents. If non-hierarchical name is used, the database should handle enormous information. Distributed Hash Table (DHT) can be used for the solution to the scalability issue.

In route-by-name manner, finding contents is accomplished through direct routing content request messages from clients to servers, which are origin servers or caching network nodes, based on the name of the requested content. Hence,

intermediate network nodes, e.g., routers, should maintain routing information based on the content name.

C. Content Delivery and Caching Issue

The resolved contents can be delivered by the normal IP based connection which could be independent with traversal path of content request packets. Also, the resolved contents can be delivered along a determined delivery path between a server and client. The determined delivery path has been created by considering network status.

Caching is to temporarily store contents for future efficient delivery and it is crucial to most content-centric Internet researches. In addition to edge-network caching deployed in CDN, in-network caching becomes an inherent functionality on network nodes, e.g., routers.

III. DATA-ORIENTED NETWORK ARCHITECTURE

Data-Oriented Network Architecture (DONA) has proposed a clean-slate design of naming and name resolution for content-centric Internet. This section discusses DONA in terms of its basic design issues.

A. Flat and Self-certifying Naming Scheme

DONA uses a flat name binding a cryptographic key. The name is of the form $P:L$ where P is a cryptographic hash of the principal's public key and L is the label chosen by the principal. Note that the principal is one who creates or verify contents. With names of the form $P:L$, clients actually obtains content as a triplet $\langle \text{data}, \text{public key}, \text{signature} \rangle$. Hence, the clients can verify the integrity of the received content with enveloped principal's signature. Also, the provenance of the content can be verified by checking that the received public key hashes to P . However, such a self-certifying nature of DONA name poses a challenge to usability because human cannot understand it and also its flatness raise a scalability issue.

B. Anycast Name Resolution

For locating contents, DONA makes use of a route-by-name manner based on a hierarchical tree. The hierarchical tree consists of new network entities called *resolution handlers (RHs)* as shown in Figure 1. The RH maintains routing information mapping a content name to a next-hop RH and it is typically located in each routing domain.

DONA defines two new basic primitives, FIND and REGISTER, to perform a name resolution on the hierarchical tree. Each client and server has known its local RH. A server which provides content named $P:L$ sends REGISTER($P:L$) message to its local RH so that the $P:L$ content is registered to each RHs. A client trying to get content named $P:L$ sends FIND($P:L$) message to its local RH. Once receiving FIND($P:L$), the RH forwards it a next-hop RH to reach that message to a server or near RH caching the content $P:L$.

C. IP Transport and Caching

In DONA, content is delivered using a normal IP routing and forwarding. After a server which generated REGISTER or a caching RH has received a FIND, the server sends the

requested contents to a source IP address in FIND message using the existing IP infrastructure.

When receiving a FIND($P:L$), a RH determines if the content $P:L$ should be cached or not. Once it is determined to cache the content, the RH changes a source IP address of the received FIND($P:L$) to its own and forwards the FIND($P:L$) to next-hop RH. Hence, the RH can get the content named $P:L$ from a origin server and it caches that. If content is broken into a series of chunks, clients would request each chunk and RHs would install cache in unit of the chunk.

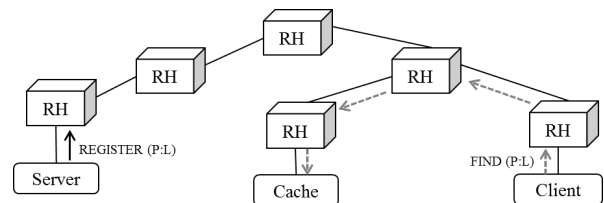


Figure 1. Hierarchical DONA architecture

IV. CONTENT-CENTRIC NETWORKING

The main concept of the Content-Centric Networking (CCN) is to change the network paradigm from named host to named data. For the name based networking, the CCN proposes naming rule, name resolution mechanism, content caching, and transport mechanism.

A. Hierarchical Naming Scheme

The name of content in the CCN is opaque binary objects composed of a number of components and follows hierarchically structured namespace. In addition, the name and content authentication is performed by digital signatures by entities.

There are two kinds of packets in CCN: Interest and Data. When a user or client requests a content using Interest packet, the exactly matched content is returned using Data packet. Both Interest and Data packet include above the CCN content name.

B. CCN Content Resolution

In the CCN architecture, content is registered into a CCN node with name prefix that is distributed using IGP flooding mechanism. The content resolution is carried out based on the registered name prefix as route-by-name manner. Figure 2 represents the CCN node architecture. The Content Store is similar to the buffer memory of an IP router and the FIB is almost identical to an IP FIB.

When an Interest packet arrives on some face (e.g., interface in IP), a longest prefix match lookup is performed in turn: Content Store, Pending Interest Table (PIT), and Forwarding Information Base (FIB). If there is matched Data packet in the Content Store, the data is sent out the face that the Interest arrived on and the Interest is discarded. Otherwise, if there is an exact-match PIT entry, the Interest arrival face is added to the PIT entry. Then the Interest is discarded that means same Interest is already forwarded toward the potential source. Otherwise, if there is a matching FIB entry, the arrival

face is removed from the matched entry the Interest is sent upstream toward data. Finally, the Interest is added to new PIT entry to wait Data packet. Data packet processing is simple since Data packet is not routed.

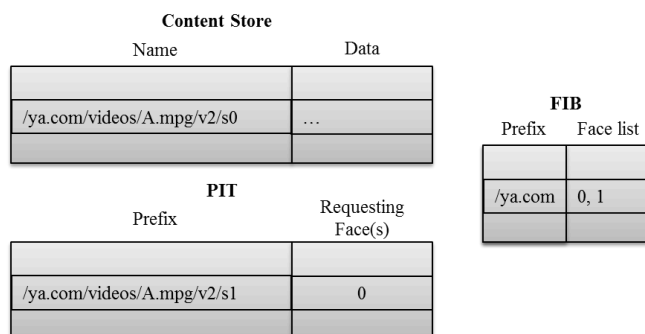


Figure 2. CCN packet forwarding engine model

C. CCN Transport and Caching

Since the structure of CCN forwarding and IP forwarding are almost identical and both use prefix-based longest match lookup, the CCN name prefix can be forwarded using existing IGP protocol, e.g., IS-IS or OSPF, TLV message without modifying the protocol architecture. Using this forwarding mechanism, all contents information registered each local CCN node is deployed on the whole network according to the physical network topology and finally every CCN nodes constructs complete FIB table automatically.

All CCN nodes can provide caching, subject to their independent resource availabilities and policies. However, it is to be noted that in CCN atomic objects are single packets, so it is possible that only a part of a bigger object is cached.

V. NETWORK OF INFORMATION

In this section, we take a look at the features and design choices of the Network of Information (NetInf) that is one of four work packages in Scalable & Adaptive Internet Solution (SAIL) EU FP7 project.

A. NetInf Flat Content Naming

The NetInf naming concept is closely related to the NetInf security concepts and follows flat name space. Basically, the NetInf names are location-independent and follow an identifier/locator split concept.

In the NetInf, content is represented as an Information Object (IO). The IO consists of three parts. Figure 3 depicts the NetInf information model. The ID is the name of IO itself and globally unique. It consists of three parts. The type field defines the type of the IDs to be treated by Name Resolution System (NRS). The authenticator 'A' field binds the ID to a public key PK and the hash function is a cryptographic hash function. The label field L contains a number of arbitrary identifier tags associated with an IO, with a fixed or variable structure specified by the Type field and can be related to data content. The Metadata consists of a set of attributes and provides semantic information about the IO. The Bit-level

Object (BO) is the basic data object itself or the digital representation of the object which could be real-world objects.

In the NetInf naming and security framework, trust is based on the IO itself. Together with additional metadata, the name can be used to verify the data integrity and other security properties, such as self-certification, name persistency, and owner authentication and identification.

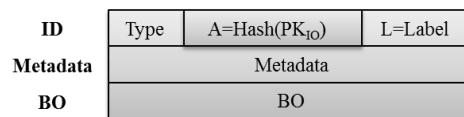


Figure 3. NetInf Information Object Model

B. Name Resolution System

To find content, NetInf introduces Name Resolution System (NRS) that resolve IDs or a set of NetInf attributes into a set of binding records for IOs that matches the ID. The location of the actual data (BO) is registered in the NRS with a binding record that contains the name structure together with the information to which the name is bound.

The NetInf architecture introduces an enhanced name resolution scheme that is Dictionary infrastructure. A Dictionary is a collection of binding records called D-RECs. Figure 4 shows the D-REC. A binding record is mapping ID to BT; $V_1 \dots V_N$; $Y_1 \dots Y_M$; TTL where the Binding Type (BT) identifies a particular resolution style and the V is a list of values associated to ID whose type depends from the BT and the Y is a list of metadata associated to the object identified by ID such as public key, signature, version, access list, accounting policy, and any other attribute. Finally, TTL is a field which contains a Time-To-Live value for refresh/expire policy for the binding record.

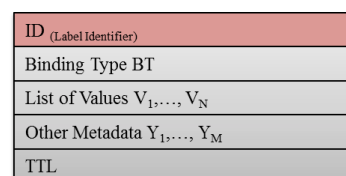


Figure 4. Dictionary record

C. NetInf Caching

The basic NetInf caching strategy is to cache all BOs that reach the node. BOs are erased when caches are full or their TTL expires. Cached objects are registered on the same NetInf node where they are, but they are not registered in the distributed Dictionary. Cached copies are used to serve just requests passing on the same NetInf node, at the Dictionary layer. Registering cached copies in the Dictionary could cause an overload of registration requests in the NetInf network.

VI. PUBLISH-SUBSCRIBE INTERNET TECHNOLOGY

Publish-Subscribe Internet Technology (PURSUIT) is a follow-up project on PSIRP [9] EU FP7 project and its basic concept is the publish-subscribed networking in which servers publish what they want to send and clients subscribe to the

publications that they want to receive. In this section, we will present the publish-subscribed networking methodology of PURSUIT.

A. PURSUIT Naming Scheme

Content in PURSUIT implies individual publications and it is denoted using a pair of identifiers: a *rendezvous identifier* (RId) and a *scope identifier* (SIId). The RId identifies publication itself and the SIId denotes the scope to which the publication belongs. Hence publication is placed in at least one scope and its RId should be unique within the scope. As shown in Figure 5, the scopes can be included within each other and publications can be associated to different scopes. Therefore, publication can be organized within scopes.

Considering security, PURSUIT utilizes the DONA naming scheme for the representation of both identifiers, RId and SIId. Hence both identifiers have a form of P:L where the P is the public key of the publisher and the L is a label indicating semantic information about the publication. Therefore, PURSUIT is based on a flat and self-certifying naming scheme.

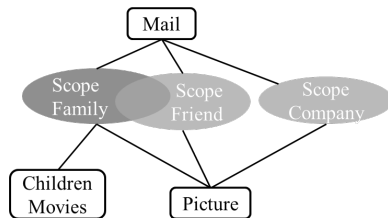


Figure 5. Concept of scope

B. Rendezvous System

PURSUIT follows a lookup-by-name approach with defining *rendezvous system* which functions as a matchmaker between publishers (i.e., content servers) and subscribers (i.e., content clients). Operations of rendezvous can be fallen into two scenarios: *domain-local rendezvous* and *global rendezvous*.

Domain-local rendezvous is based on *rendezvous nodes* that are special network nodes keeping track of available publications. Rendezvous node floods its presence announcement within its domain. As a result of the flooding, clients have known the presence of rendezvous nodes. Whenever a publisher (e.g., *Publisher A*) wants to announce some his content, he sends a publication message to the known rendezvous node. Hence the rendezvous node stores that the content is available at the *Publisher A*. Later on, if a subscriber (e.g., *Subscriber A*) wishes to get the content, previously published by *Publisher A*, she sends a subscription message to the rendezvous node and the rendezvous node decides that *Publisher A* should deliver the desired content to *Subscriber A*. Hence, the rendezvous node instructs *Publisher A* to transmit the content toward *Subscriber A* over the determined path.

Global rendezvous defines *rendezvous networks* (RNs) composed of local rendezvous. The RNs are globally connected using a hierarchical Chord DHT [10], which is called a *rendezvous interconnect* as shown in Figure 6.

Publishers authorized by a specific scope (e.g., *Company A*) advertise their content in the home RN of the scope. The home RN advertises the scope in a rendezvous interconnect. Subscribers that wish to receive the content belonging to the scope of *Company A* send rendezvous message to their local RNs. The local RNs pass the message to the rendezvous interconnect to find the rendezvous point to the scope. As a result, the subscribers are informed of a location of the publisher and send subscription messages to the publisher.

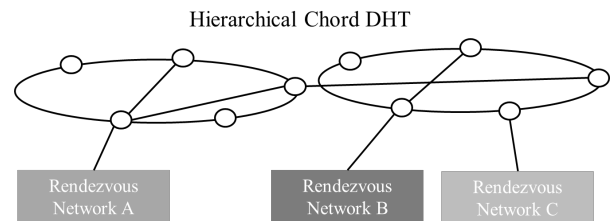


Figure 6. Rendezvous interconnect

C. Content Forwarding and Caching

PURSUIT defines *topology management* and *forwarding* functions to deliver contents over predetermined delivery path.

The role of the topology management is to collect and manage network topology information, as well as to provide the optimal content delivery paths from publishers to subscribers. Gathering topology knowledge can be done by using common topology discovery mechanisms, e.g., OSPF. Using network topology information (including policies and network conditions) and the locations of the publisher and subscribers, topology management finds a suitable delivery path that consists of a list of each point-to-point links. LIPSIN [11] encodes the deliver path into a constant-length *forwarding identifier* (FId) using Bloom Filters.

Forwarding function on transitive network nodes delivers contents along a determined path between publishers and subscribers by using the FId.

VII. CONTENT AWARE SEARCHING, RETRIEVAL AND STREAMING

Content Aware Searching, retrieval and Streaming (COAST) aims to build a content-aware overlay architecture on top of the existing network infrastructure. The content-aware overlay architecture could be composed of different overlay networks with different functionality. Current COAST defines three kinds of networks, Information overlay, content/service distributed overlay, and service/network infrastructure, as shown in Figure 7. This section presents an overview of COAST project.

A. COAST Content Naming

COAST defines two types of naming schemes, *COAST object identifier* (COI) and *COAST-aware URL* (CURL), to identify content according to its utilizers.

COAST aware nodes (e.g., Search Engine, Cache Locator, and content-aware nodes (CNs)) use the COI to refer content. The COI consists of the COAST content identifier (CCI) and COAST metadata identifier (CMI). The CCI and CMI are

obtained by calculating a hash over the data and metadata bits, respectively. COAST recommends to utilizing MD5 or SHA-1 hash function for creating CCI and CMI.

COAST lets content providers use COAST-aware URLs (CURLs) to locate their contents whenever creating a new content. CURL includes a URL and a CCI, and it has the following format.

*http://<domain>:port/<path prefix>/< magic word>
<CCI or CMI><possible extension>*

The possible extension string allows the content to be easily identified by humans. As you understand the format of CURL, CURL returns both the original URL and the CCI (or CMI). Hence, non-COAST nodes just make use of the legacy URL information.

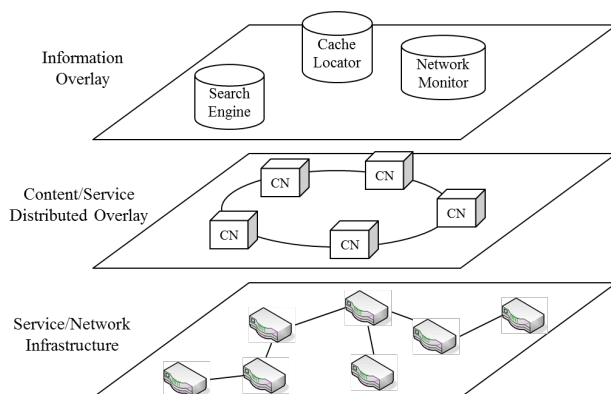


Figure 7. COAST high level architecture

B. Content Finding through Search Engine

As a lookup-by-name approach, COAST let clients find content by querying a Search Engine functional module in the information overlay. The result of the query will be returned as URL or CURL. Both non-COAST clients and COAST aware clients that receive URL just use the query result to reach the original location of the content (i.e., a content server). However, COAST aware clients use the CCI in CURL to fetch the content from COAST aware network nodes caching the content as well as the content server.

C. Content Forwarding and Caching

Network Monitor functional module in information overlay of COAST exploits the information about the distributed content storage and network nodes in order to optimize the

actual content delivery process. Also, before delivering, content adaptation according to user preferences and terminal capabilities can be performed.

COAST caches query results as well as content. Cache Locator functional module in information overlay maintains information about cached copies and points clients to the closest cached copy.

VIII. CONCLUSIONS

In this paper we have discussed three basic design issues of the research field content-centric Internet: content naming, content resolution, and content delivery and caching. Also, five research projects, DONA, CCN, NetInf, PURSUIT and COAST, have been briefly examined in terms of the three basic design issues. Our future work is to raise implementation issues of approaches of the research projects with respect to the three basic design factors and to compare their performance to a certain extent.

ACKNOWLEDGMENT

This research was supported by the Korea Communications Commission (KCC), Korea, under the 'R&D on Smart Node Technology for Cloud Networking and Contents Centric Networking' support program supervised by the Korea Communications Agency (KCA) (KCA-2011-11913-05003).

REFERENCES

- [1] Cisco Visual Networking Index (VNI), http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360_ns827_Networking_Solutions_White_Paper.html
- [2] Named Data Networking, <http://www.named-data.net/>
- [3] Scalable and Adaptive Internet Solutions, <http://www.sail-project.eu/>
- [4] Publish-Subscribe Internet Technology, <http://www.fp7-pursuit.eu/PursuitWeb/>
- [5] Content Aware Searching Retrieval and Streaming, <http://www.coast-fp7.eu/>
- [6] Koponen, t. et al., *A data-oriented (and beyond) network architecture*, ACM SIGCOM Computer Communication Review, 2007.
- [7] V. Jacobson et al., *Networking Named Content*, CoNEXT '09, 2009.
- [8] A. Ghodsi et al., *Naming in Content-Oriented Architectures*, SIGCOMM ICN'11, 2011.
- [9] Publish-Subscribe Internet Routing Paradigm, <http://www.psirp.org/>
- [10] G. Ganesan, K. Gummadi, and H. Garcia-Molina, *Canon in G Major: Designing DHTs with Hierarchical Structure*, IEEE Computer Society ICDCS'04, 2004.
- [11] P. Jokela et al., *LIPSIN: Line speed Publish/Subscribe Inter-Networking*, ACM SIGCOMM 2009, 2009.