Volume 3, No. 12, December 2012
Journal of Global Research in Computer Science
ISSN-2229-371X
REVIEW ARTICAL
Available Online at www.jgrcs.info

Content Centric Networking and its Applications

Deepali Damodar Ahir[*1], Prof. Prashant. B. Kumbharkar[2]
*[1]Computer Science and Engineering, Pune/Siddhant Engineering College/Pune, Maharashtra, India
deepaliahir@gmail.com
[2] Computer Science and Engineering, Pune/Siddhant Engineering College/Pune, Maharashtra, India
Pbk.rscoe@gmail.com

Abstract: Growing use of digital devices and penetration of social networking has created lot of personal content which is being shared over internet. Content sharing is becoming popular and it is one of the most used applications of internet, yet internet design has little consideration for content sharing. It was mainly designed to connect different nodes in the network, so to keep up with latest trend and to efficiently use internet for content sharing a new concept is evolving called Content centric networking (CCN). CCN provides lower communication overhead and shorted download time, this paper will explain CCN, its applications, and possible enhancements to CCN.

Key words: Named data, Content Centric Networking (CCN), Mobile content sharing.

INTRODUCTION

Rapid developments in mobile technology have totally changed the way people used to perceive mobile phones. In earlier days, mobile phones were only used as devices to make phone calls. However, a variety of embedded features like messaging, web browsing, high quality digital and video camera, media player, third party applications, and Wireless Local Area Network (WLAN) capability have transformed mobile devices into multimedia computers. These mobile multimedia devices have become an essential part of everyone's daily life and the primary consumer device for creating digital content. Users are now capable of creating their own content, easily and fast, and likely to share their content with other people, as an extension of the human need for communication and experience sharing.

Today, only way to retrieve content is to establish an end-to-end connection with them. While the communication pattern in the Internet has been evolving since its early days, from client-server model to peer-to-peer networking, and to cloud networking, one significant point has been shown, i.e., the usage pattern of the Internet has become largely content-oriented. That is, content consumers do not care where and how to obtain a piece of content.

Yet the internet was designed as a *communications* network, not a *media distribution* network. These limitations impact every part of the ecosystem – from carriers to publishers, across wired and wireless communications – and operators need economical ways to solve these problems beyond marginal improvements on existing solutions and tools.

CCN is an alternative approach to the networking architecture based on the principle that a communication network should allow a user to focus on the data he or she needs, rather than having to reference a specific, physical location, from where that data is to be retrieved. CCN enables content caching to reduce congestion and improve delivery speed, a simpler configuration of network devices, and security built into the network at the data level. The initiative has continued to gain momentum with an open source code release, Android implementation release, and commercial engagements with prominent industrial partners by PARC [1]. When compared to the current TCP/IP communication model, CCN has the following different characteristics:

a. Receiver-centric communication model: Receivers pull information by sending an interest message. At most one data message is delivered in response to an interest.
b. Hierarchical content naming scheme: CCN does not address specific hosts, but content object itself. Content is given hierarchical names, which is similar to URLs. Interest packets are forwarded by doing longest-prefix matching at forwarding decision phase.
c. Cache and forward architecture: Every CCN devices can cache data and use them to serve future requests.

From the above characteristics, CCN is expected to resolve the problems in today's Internet such as mobility, security, multi-path support, and so on.

Rest of the paper is organized as follows, section 2 describes CCNx protocol, section 3 describe CCN in mobile environment, section 4 describe proxy assisted CCN, section 5 enlists possible applications of CCN to various sectors, section 6 catches the advantages and disadvantages with the conclusion.

CCNX PROTOCOL

Content sharing in CCN is done over CCNx protocol which is the transport protocol for CCN [2][3]. The CCNx protocol efficiently delivers named content rather than connecting

hosts to other hosts. Every packet of data may be cached at any CCNx router — combined with intrinsic support for multicast or broadcast delivery this leads to a very efficient use of the network when many people are interested in the same content.

CCNx protocol provides location-independent delivery services for named data packets. The services include multi hop forwarding for end-to-end delivery, flow control, transparent and automatic multicast delivery using buffer storage available in the network, loop-free multipath forwarding, verification of content integrity regardless of delivery path, and carriage of arbitrary application data. Applications run the CCNx protocol over some lower-layer communications service capable of transmitting packets. There are no restrictions on the nature of the lower-layer service: it may be a physical transport or another network or transport protocol. For example, applications will typically run the CCNx protocol on top of UDP to take advantage of existing IP connectivity. Since content is named independent of location in the CCNx protocol, it may also be preserved indefinitely in the network, providing effectively a form of distributed file-system service.

The CCNx protocol is general, supporting a wide range of network applications. It may be natural to think of stored content applications such as distribution of video or document files, but the CCNx model also supports real-time communication and discovery protocols and is general enough to carry conversations between hosts such as TCP connections. The protocol supports a broad range of applications by leaving the choice of naming conventions to the application. CCNx specifies the common functions that are independent of the contents of the names and data and the semantics of exchanges. In addition to CCNx, therefore, a complete specification of the use of the CCNx protocol for a particular application will require additional specification of the naming rules, data formats, and message semantics. Such specifications of application protocols on top of the CCNx protocol (plus possibly API specifications) are called *profiles*.

The CCNx protocol is designed for end-to-end communication between applications and so it is intended to be integrated into application processing rather than being implemented as a separate layer.

CCNx definitions:

Given below are some definitions used in CCNx protocol:
Node

A CCNx network entity that implements forwarding and buffering.

Party:

Any entity in the network using the CCNx protocol to communicate. Note that parties are not just machines, but also applications using the protocol.

Message:

A CCNx packet. This term is used to avoid confusion with the lower-layer packet that may be carrying CCNx messages. A single lower-layer packet (for example a single UDP packet) MAY contain more than one CCNx message.

Message Format and Encodings:

Unlike many other protocols, the CCNx protocol does not have any fixed-length fields. Instead, CCNx data formats are defined by XML schemas and encoded with explicitly identified field boundaries. This design permits field values of arbitrary length, optional fields that consume no packet space when omitted, and nested structures. The use of XML structure does not imply that field values are text strings nor does it require that messages be encoded as human-readable text. Most fields are defined to contain arbitrary binary values, including those that identify content.

The wire format of CCNx messages is an efficient binary encoding of XML structures called ccnb, which defines such things as the byte order. There is also a text XML encoding which is useful for debugging, testing, presentation in documentation, etc. but MUST NOT be used on the wire.

Content Identification:

The CCNx protocol accomplishes transfers of content by *name*, irrespective of the identities or locations of machines involved. CCNx content names are hierarchically-structured, consisting of a number of *components*. The hierarchical structure is like that of IP addresses, but CCNx names and name components have arbitrary lengths and the component divisions are explicitly identified unlike in the classless model of IP addresses. The protocol does not depend on anything but the hierarchical structure of the names so they may contain arbitrary binary data such as encrypted data.

CCNx content names are not interpreted in the operation of the CCNx protocol itself, just matched. All assignment of meaning to names or their component parts comes from application, institution, and/or global conventions reflected in prefix forwarding rules.

A CCNx name sometimes identifies a specific chunk of data, but in other cases identifies a collection of data by naming a point in the name tree under which there may be multiple pieces of data. A name identifying a collection of data is analogous to the address of a network in the host addressing scheme of IP, where the network address can be seen to identify the collection of hosts attached to that network. In the case of naming a collection, the CCNx name is a prefix of the name of every piece of content in the collection, just as an IPv4 network address gives a prefix of the IP addresses of member hosts. For this reason, a CCNx name may be referred to as a *name prefix* or simply *prefix*. CCNx name can be represented using URI representation or using XML. A Name element represents a hierarchical name for CCNx content. It simply contains a sequence of **component** elements. Each Component element contains a

sequence of zero or more bytes. There are no restrictions on what byte sequences may be used.

URI Representation:

It is often convenient to use a URI representation for a CCNx Name. The scheme identifier for a CCNx scheme is "ccnx", and ccnx URI need not have the authority component (i.e. the part after initial // in familiar http URI).
Example:
An HTTP URI:
http://www.ccnx.org/releases/latest/doc//Name.html
Can possibly be represented in ccnx URI as: **ccnx://releases/latest/doc/Name.html**

XML Presentation:

There are no restrictions on what byte sequences may be used in a Component, so when displayed as XML, a base64Binary or hex Binary encoding may be needed. When the bytes happen to be printable UTF-8, a more human-friendly "text" alternative is available.

The CCNx name of a chunk of data always contains as its final, most specific component a value that is derived from the data, called the *digest component*. Since it is derived from the data itself, the digest component is redundant and so is not transmitted.

CCNx Message Types:

The CCN protocol is based on two packet types, namely Interest and Data packet(also known as content object) (Figure 1)

The Interest message is used to request data by name. The most important field from these is a content name. Figure 2 shows an example of content name. An Interest message can identify a chunk of content to retrieve very specifically. Alternatively, an Interest message can provide a name prefix and other qualifications to restrict what data is acceptable from the collection named by the prefix.

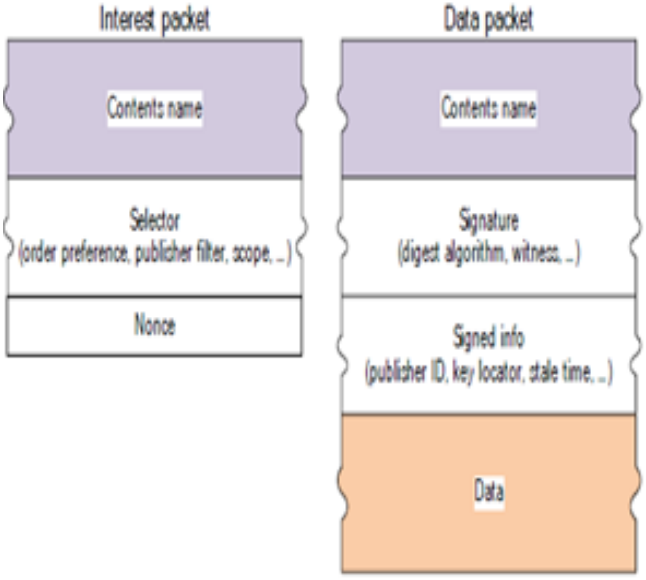

Figure 1.CCN Messages and their delivery

Figure 2. Example of Content Name

The data packet is used to supply data. A data packet not only contains a data payload but the identifying name (without the implicit digest component), a cryptographic signature, and identification of the signer (called the *publisher*) along with other information about the signing. Formally, a data packet is an immutable binding of a name, a publisher, and a chunk of data. Every data packet contains a valid signature. In this way, all data communicated with the CCNx protocol is attested.

Any CCNx party can verify the signature on any Content Object message that it receives. Applications that are going to make use of the data must verify first. Verification may require public keys which are not themselves included in the Content Object message to be verified. A CCNx party discards a Content Object message that fails to verify.

The CCNx protocol does not provide a separate mechanism for key distribution, since keys can be distributed just like any other data using the general features of the protocol. Profiles for key distribution are separate specifications, not part of the CCNx protocol itself. Signature verification can confirm that a Content Object message has not been corrupted in transit (intentionally or otherwise) since it was originally signed, and that it was signed by the identified publisher, but these verifications are not the same as determining that the publisher in question is a trustworthy source of data for a particular application purpose. CCNx parties use trust management practices to decide whether to trust data from particular publishers for particular purposes, but such practices are not specified as part of the CCNx protocol itself. The CCNx protocol is designed to ensure attestation of data, without constraining decisions about key distribution and trust management.

Basic Exchange:

Communication using the CCNx protocol is receiver-controlled. A consumer of data transmits an Interest message over available connectivity and any party receiving the message and having data that matches, or *satisfies*, the request (according to the specifications in the Interest Message) may transmit a matching Content Object message. Data can only be transmitted in response to an Interest that matches the Data. An Interest message may be transmitted using broadcast or multicast facilities of the underlying

transport in order to reach many potential sources of data with minimal bandwidth cost.

A party can transmit at most one Content Object message in response to a single received Interest message, even if the party has many Content Objects that match. This one-for-one mapping between Interest and Data messages maintains a *flow balance* that allows the receiver to control the rate at which data is transmitted from a sender, and avoids consuming bandwidth to send data anywhere it is not wanted.

Nodes need to implement *suppression* mechanisms to minimize the potential for two different nodes to transmit two Content Object messages in response to a single Interest received by both nodes (for example via broadcast or multicast). The suppression mechanisms need to include randomizing response times and detecting the fact that another node has broadcast or multicast a response so the Interest may be discarded. Current version of the CCNx protocol does not specify suppression rules.

A receiver that desires to retrieve a large collection of data requiring multiple Content Object messages must transmit a sequence of Interest messages. For pipelining, a receiver may transmit multiple Interest messages without waiting for data in response to each one before sending the next. This is only possible if the receiver knows enough about the names of the Content Objects to construct different Interests in advance. Sending multiple Interest messages with identical requests will usually cause the same Content Object to be transmitted repeatedly, since senders must respond to Interests based only on what content they have available at the time, and not on any memory of what Content Objects they have previously transmitted.

The CCNx protocol does not assume that underlying transport of messages is reliable. To provide reliable delivery, Interest messages that are not satisfied in some reasonable period of time must be retransmitted. A receiver needs to maintain a timer on unsatisfied Interests for which it still wants the data, and retransmit them when the timer expires. Current version of the CCNx protocol does not specify timer values.

These rules govern the basic Interest/Data exchange for local communication that is communication between peers which directly receive message transmissions from each other. Since the CCNx protocol may be used on top of any packet transport, this definition of local is quite broad, including pairs of nodes physically far apart but using a long-distance connection such as a TCP connection as a tunnel to transmit CCNx messages directly. Multihop CCNx protocol communication requires forwarding of CCNx message which is specified in terms of the node model described in the next section.

CCNx Node Model:

A full CCNx node (as opposed to a limited CCNx party such as a single application) contains the following data structures to provide buffering/caching and loop-free forwarding.

Content Store (CS):

A buffer memory organized for retrieval by prefix match lookup on names. Since CCNx Content Object messages are self-identifying and self-authenticating, each one is potentially useful to many consumers. The CS needs to implement a replacement policy that maximizes the possibility of reuse such as Least Recently Used (LRU) or Least Frequently Used (LFU). The CS may retain Content Object messages indefinitely but is not required to take any special measures to preserve them; the CS is a cache, not a persistent store.

Face:

A *face* is a generalization of the concept of *interface*: a face may be a connection to a network or directly to an application party. A face may be configured to send and receive broadcast or multicast packets on a particular network interface, or to send and receive packets using point-to-point addressing in the underlying transport, or using a tunnel (for example a TCP tunnel). A face may also be the connection to a single application process running on the same machine, via an encapsulation like UDP or an OS-specific interprocess communication path. All messages arrive through a face and are sent out through a face.

Forwarding Information Base (FIB):

A table of outbound faces for Interests, organized for retrieval by longest prefix match lookup on names. Each prefix entry in the FIB may point to a list of faces rather than only one.

Pending Interest Table (PIT):

A table of sources for unsatisfied Interests, organized for retrieval by longest prefix match lookup on names. Each entry in the PIT may point to a list of sources. Entries in the PIT must timeout rather than being held indefinitely.

Note that the tables listed above may be interconnected through a single index to minimize the cost of lookup operations at the core of CCNx message processing and forwarding. Such an index must be ordered or prioritized to achieve the effect of the lookup order specified below.

Message processing in CCN:

Processing Interest Messages:

An Interest Message is processed according to the following sequence:

a. Lookup is performed on CS. If a matching ContentObject is found, it will be transmitted out the arrival face as a response to the Interest Message. Note that to match a ContentObject must satisfy all of the specifications given in the Interest Message. Note also that it may be that multiple ContentObjects

simultaneously match in which case the specification in the Interest Message will be used to determine which object to return. When a match is found in the CS, processing stops and the Interest Message is discarded since it has been satisfied.

b. Lookup is performed on PIT. If a matching Interest Message is found in the PIT it means that an equivalent Interest Message has already been forwarded and is pending. The arrival face of the new Interest Message is added to the list of sources of unsatisfied interests in the PIT entry and the Interest Message is discarded.

c. Lookup is performed on FIB. If a matching prefix is found in the FIB, an entry is created in the PIT identifying the arrival face of the Interest Message and the message is transmitted according to the *strategy rules* to one or more of the outbound faces registered for the prefix in the FIB. Note that one of the outbound faces may actually be connected to a local agent that uses the semantics of the name to dynamically configure new faces.

d. If no match is found in the previous steps then the node has no way to satisfy the Interest Message at present. It may be held for a short time before being discarded, since the creation of a new FIB entry may provide way to satisfy it.

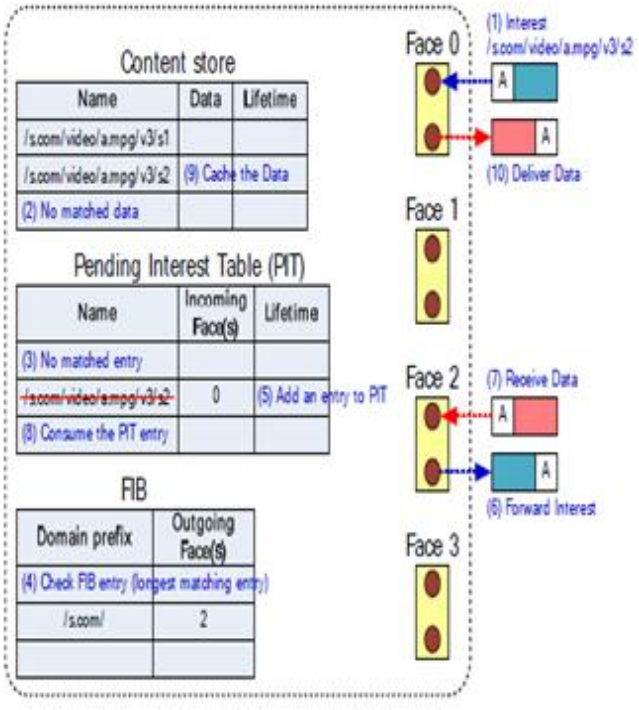

Figure 3. Processing of CCN messages.

Processing Content Messages:

A ContentObject is processed according to the following sequence:

Lookup is performed on CS. If a matching ContentObject is found it means that the newly arrived ContentObject is a duplicate which can safely be discarded, because any Interest Messages have already been satisfied and new ones will be satisfied out of the CS.

a. Lookup is performed on PIT. If there is a match in the PIT, the ContentObject is transmitted on all of the source faces for the Interests represented in the PIT. A node may perform verification of the ContentObject before forwarding it, and may apply various policy restrictions.

b. If no match is found in the previous steps, then the content is unsolicited. A node does not forward unsolicited data and discards unsolicited data but may store it in the CS in case it is subsequently requested.

Figure 3. Shows processing of CCN message in a sample scenario.

Strategy Rules:

Unlike in IP, CCNx FIB entries may point to multiple next-hop destinations simultaneously. The self-identifying nature of Interest Messages and ContentObjects means that loops can be avoided without establishing a spanning tree allowing only one destination for any particular prefix at any particular node. The possibility of multiple outbound faces means that different strategies may be employed to select among them when forwarding messages. A node needs to implement some strategy rule, even if it is only to transmit an Interest Message on all listed outbound faces in sequence. A node may implement many different strategies. Strategy rules should be specified by FIB entries: in this case the FIB entry may contain effectively a constrained program for handling the forwarding of Interest Messages addressing the particular prefix.

A node needs to discard PIT entries that have not been refreshed by the arrival of new Interest Messages within some timeout. The new Interests confirm that the potential recipients are still interested in receiving content. A node has to retransmit Interest Messages periodically for pending PIT entries. A node may use different timeouts for refreshing the liveness of received Interest Messages (*downstream interests*) from those used for retransmitting pending Interest Messages (*upstream interests*), and may use different timeouts associated for different faces. In this way, each node should match to both to downstream and upstream parties

CCN IN MOBILE ENVIRONMENT

Even though CCN can provide localized content delivery, there may happen cases in which such inherent benefits are tarnished if content requester is a mobile device. As shown in Figure 4(a), a mobile device X sends interest packets towards the original source Y. As the interest packets are forwarded via the network, they traverse through several CCN devices. Data packets from content source are routed back along the path that interest packets are traversed. Each CCN device along the path stores the forwarded data packets from the content source to content requester.

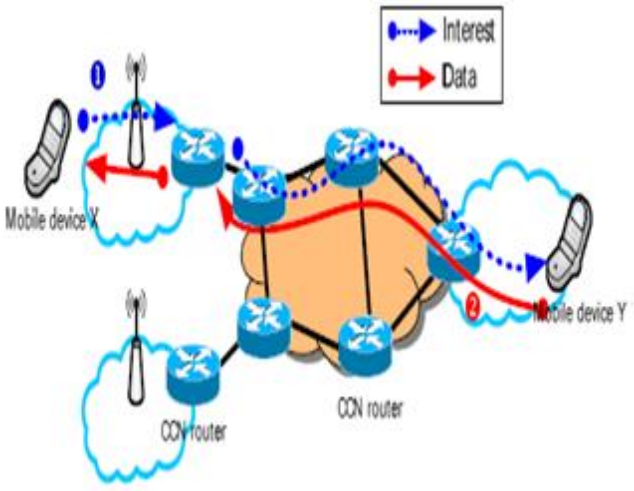

Figure 4 (a). Device X sends an interest to the original source

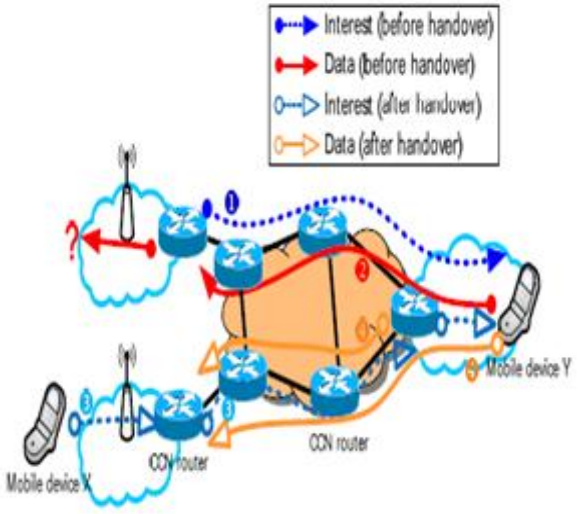

Figure 4(b). After handover, device X sends again interest packets; it may not utilize the already delivered and stored data packets

As described in Figure 4(b), the CCN networking device, which is located near the old location of mobile device X, does not know the movement of the device X. So, data packets are unnecessarily transmitted around the old location of mobile device X. Moreover, interest packets are repeatedly transmitted after handover to request data packets that were already requested at the old location. That is, after handoff event, a number of redundant interest packets have to be sent again to retrieve the already-requested data packets. However, as data packets stored near the old location are not available until routing table update, the re-requested data packets after handover are delivered from the remote-located original content holder, not from nearby CCN device with cached content. So, it results in long latency and too much control overhead during device movement

Reducing control overhead of CCN in mobile environment:

To reduce the control overhead and wastage of bandwidth of using CCN in mobile environment, J. Lee and D. Kim has proposed a proxy assisted scheme [4]. In this scheme user proxies with CCN functionality are configured in overlay architecture over IP networks. The CCN proxy can be either a mobile device or a PC that must be continuously active.

The CCN proxy behaves as the common point of routing path for all content sharing. In addition, the proxy serves as a point of indirection (providing relaying services) as well as providing additional services such as message filtering, secure association, and so forth.

The basic idea is that user devices ask the proxy to download the requested content on their behalf (as shown in Figure 5). The CCN proxy participates in the conventional CCN overlay, and takes care of all downloads of the devices. So, if a user device wants to get specific content data, it just sends to its proxy node the content query packet, which is similar with interest packet in original CCN architecture. That is, user devices do not need to resolve and make connections with other content holder devices by themselves.

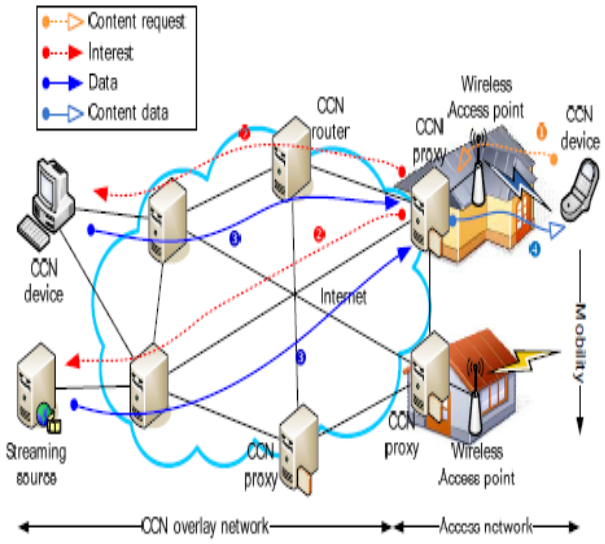

Figure 5. Proxy-based CCN content sharing

The CCN proxy receiving the interest packet may try to discover the content using normal CCN interest/data exchanges on behalf of the content requester. The requested content data are transferred from the proxy to the requester device. From this proxy architecture, the mobile device can reduce the overhead for both CCN routing information configuration and content sharing. The proposed proxy-based mobile CCN is explained in detail in next chapter.

PROXY ASSISTED CCN

Proxy assisted CCN is a proposed variance of CCN to avoid control overheads in mobile environments. Each operation of proxy assisted CCN is described below in detail.

CCN overlay configuration:

First of all, a mobile CCN device does a secure association with a CCN proxy node for the content prefix announcement and content sharing. It is assumed that the proxy nodes

configure overlay network in advance and therefore recognize the identity information of others. Each individual device including proxy nodes carries a unique cryptographic identity in the form of a public key pair. There is a tight coupling between the key pair used by the device and its identity so that all devices can be easily identified. Through the identity information, proxy nodes can establish face configuration to construct routing tables. After the secure association, the mobile CCN device and the proxy node exchange the identity information as well as IP address with each other. Storing such information makes it possible to deliver content data without exchanging additional interest packets when either IP address of mobile consumer device or the serving proxy node is changed. That is, once there is a secure association between a mobile consumer device and one of proxy nodes, there is no further processing at other proxy nodes, except for network association (i.e., to create the face configuration between them).

Proxy-assisted CCN content sharing:

The mobile device that wants to receive a specific content data sends a content request message containing the requested content name to its proxy node as shown in Figure 6.

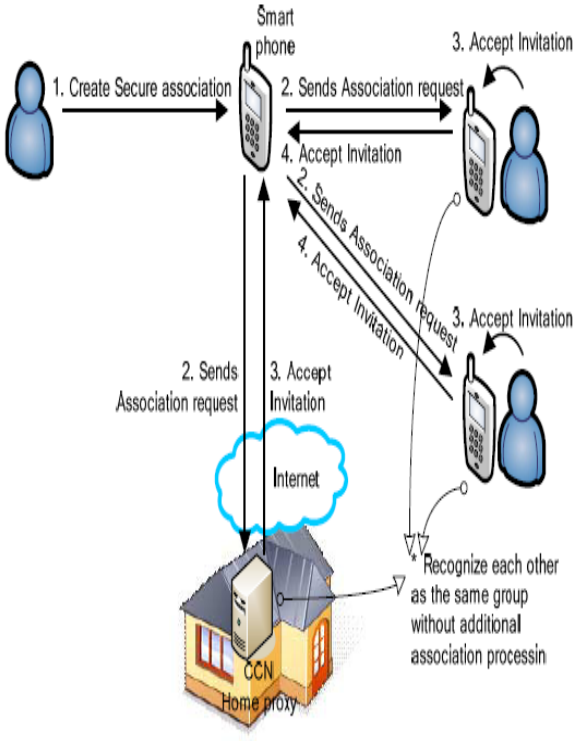

Figure 6. Secure association in mobile CCN environments

The proxy node receiving the content request message initiates a normal CCN content sharing procedure. That is, the proxy node assumes the content request as one request from its own application layer. After receiving the metadata information of the requested content from content holders, the proxy node delivers the metadata information for the requested content data to the mobile device. It is assumed that the metadata information can be acquired when CCN content is generated and it is piggybacked in the first segment data of the content. So, the mobile device can configure the fake PIT entries for the requesting content data without issuing further interest packets in segment unit of the content data.

Movement indication:

A mobile device detects when to change network status by using physical link information or subnet address. In the proposed architecture, both subnet change and proxy change are considered to be a handoff event. As the proposed scheme assumes CCN over IP overlay architecture, IP address change of content requester is directly related with the new face configuration of CCN architecture to the proxy node. In the same manner, the proxy change event should be notified to the previous proxy node to prevent unnecessary packet loss and resource consumption. When detecting that the movement event is imminent, the mobile device sends 'Hold request' message to a current proxy node.

After receiving the 'Hold request' message, the current proxy node stops delivering content data packets toward the mobile device and only stores the content data in its local repository for subsequent retransmissions. For that, the HO field of the relevant entry in PIT is set to 1. As shown in Figure 7, the CCN proxy node receiving interest packet that indicates specific content data configures content based routing entry for future content data delivery. That is, the routing table architecture of the proposed scheme has only difference in the 'HO' field of PIT configuration.

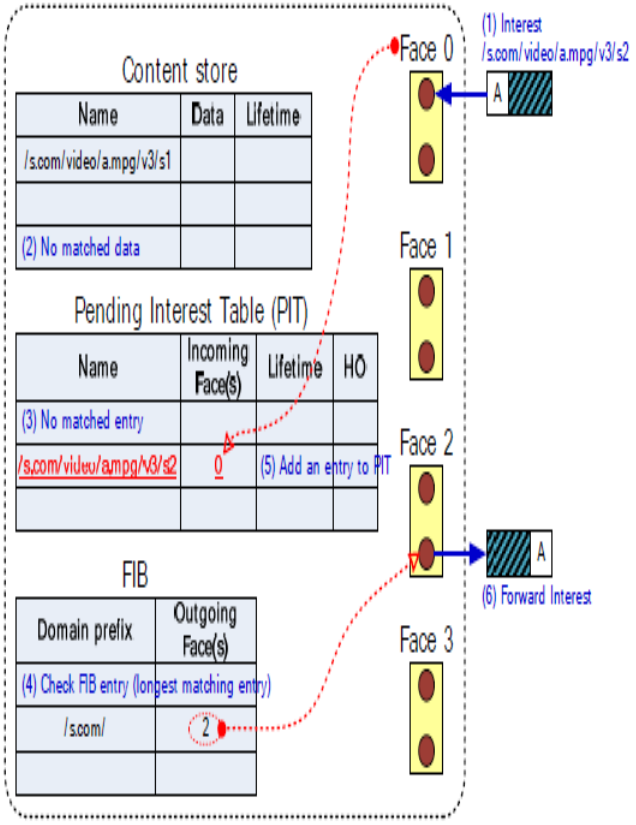

Figure 7. Interest packet processing at proxy CCN device

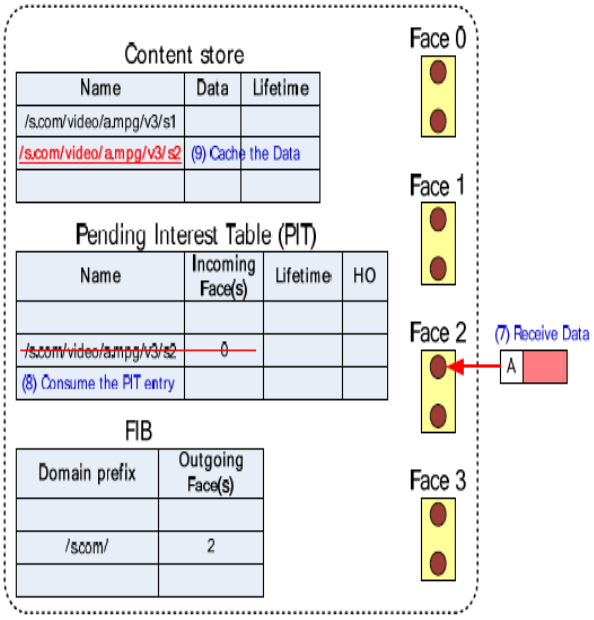

Figure 8. Modified content data processing at proxy CCN device during handover

After that, if a content data packet is received, the proxy node checks whether a specific PIT entry exists and the relevant entry has a HO indication bit. If HO bit is '1', the received content data packet is not forwarded and just stored at its repository (shown in Figure 8). Therefore, the proposed scheme can prevent unnecessary packet losses and network resource consumptions toward the path to old location of the mobile device.

Content migration processing:

Subnet change:

When acquiring new IP address, the mobile device notifies the new IP address information of its proxy node by using 'Handover notification' message piggybacking the content sequence numbers that it finally received at the old location. The CCN proxy node transmits the stored content data packets toward the new location of mobile device. That is, the proposed scheme does not require the mobile device to transmit again interest packets for the stored content data packets. After receiving the last content data segment that was requested at the old location, the mobile device just waits subsequent content data segments without using additional CCN interest packets.

Proxy change:

Through the proxy advertisement message, the mobile device can detect the existence of new proxy node. While doing the association processing with the new proxy node, the mobile device delivers the identity information regarding the previous proxy node to the new proxy node. So, the new proxy node can ask for delivering cached content data for the mobile device. To reduce the control overhead during the delivery of the cached content data, the new proxy utilizes the modified CCN interest, which indicates the requested content name and the content sequence number that was last received from the previous content proxy. Moreover, it is possible to temporarily have two simultaneous paths to receive the requested content data, one for the path from the indirect path via the previous proxy node and the one for the direct path to the new proxy (as shown in Figure 9).

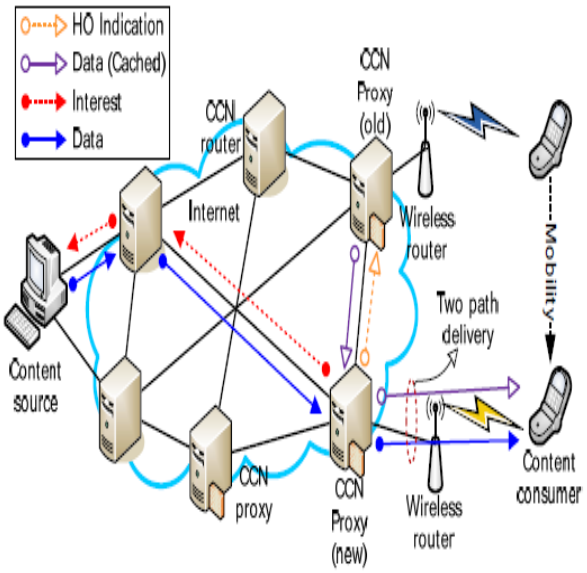

Figure 9. Two-path delivery in proxy change case

APPLICATIONS

CCN has applications in various fields apart from content dissemination of regular web server traffic. These possible applications are discussed below:

Medical devices and healthcare management systems:

Personal wellness, remote health monitoring, and assisted living are growing healthcare markets that are expanding the use of mobile devices and services that can securely record, store, and share personal healthcare data. Smart phones and tablets are increasing in both consumer and healthcare sectors and there are thousands of medical and healthcare applications available.

Consumers, healthcare providers, medical equipment vendors, and physicians are recognizing the benefits of mobile healthcare devices and services to monitor and share patient vitals with physicians, to provide personal access to aggregated health information, and to use in emergency response situations where there is limited access to infrastructure.

However, these devices, systems, and applications also present challenges around ensuring privacy and security, connecting multiple heterogeneous devices, and aggregating and sharing information. Such obstacles hinder the usability and deployment of solutions using existing networking technologies. Instead of piecing together network components and security services or segmenting infrastructure and information, there is a clear need to

develop a secure unified communication platform for access, storage, and transmission of health and wellness data.

CCN-based technologies allow medical devices to wirelessly enroll, share, and access sensitive content securely without the need for deployed infrastructure or backend systems support. Using CCN's data-centric cryptographic methods, health and wellness data can be shared securely with privacy levels defined and set by the user or provider. CCN's highly usable enrollment procedures can be applied to multiple medical device types. The protocol-agnostic communication model of CCN enables the building of a low-cost communication platform for a set of heterogeneous devices.

Secure and cost effective lighting control systems:

Lighting systems for industrial, municipal, and residential buildings are increasingly transitioning to energy-efficient alternatives such as light-emitting diodes (LEDs) to reduce costs and environmental impact, as well as comply with energy-efficiency legislation.

Many new lighting control systems are beginning to use an IP-based architecture and functionality but this approach is costly and inefficient. It requires managing and allocating IP addresses, constructing firewalls for security, and managing complex IT configurations. There is also a high risk for security breach to building safety when systems are wirelessly controlled. There is an opportunity for a new approach to designing and deploying control systems that features enhanced security, intuitive naming, and easy manageability. Content-Centric Networking (CCN) technology provides a new communication and architectural model that provides secure, self configuring, cost-effective, and robust lighting control.

Specific elements of the work may include:

a. Design and implement bootstrapping mechanisms that use CCN protocol details for device discovery, maintenance, and association.
b. Outline naming strategies that associate fixtures with intuitive naming mechanisms.
c. Implement using Linux server for control logic implementation and for communication and exercise of policies with lighting fixtures. Commands can be entered from a console that can be a panel or a mobile device depending on industrial or residential settings.
d. Implement a mobile interface on an Android device to receive user command inputs that get communicated to the controller, which in turn communicate with lighting fixtures.

ADVANTAGES AND DISADVANTAGES

The Benefits of Content-Centric Networking:

a. Provides a seamless, ubiquitous experience - allows people to easily send and receive digital content from multiple locations, mobile devices, and diverse networks.
b. Simplifies the network use, reduce set-up time and doesn't require manual configuration through firewalls, VPNs and adhoc synchronization protocols.
c. Reduces congestion and latency - doesn't send irrelevant or redundant information through network pipelines.
d. Improves network performance while reducing operating costs - increases efficiency by at least three orders of magnitude.
e. Increases network reliability - robustly delivers information using any available medium.
f. Eliminates many security problems - secures information so integrity and trust are properties of the content, not of the channel.
g. Supports new and emerging applications - facilitates mobile and wireless access (which are currently relegated to the fringe of the network), and enables broadcast, voice over IP (VoIP), autonomous sensor networks, ubiquitous applications, and context-aware computing. Empowers the user - allows people to express intent to their networks (e.g., to prioritize specific content over others) and prioritizes their needs from inside the network.

Drawbacks of CCN:

a. CCN is still evolving concept and hence there is not much support available in terms of hardware or software.
b. CCN addresses problem of content dissemination but it may result in wastage of bandwidth when it is used over mobile devices. We have already discussed this problem and possible solution in earlier chapter.

CONCLUSION

Current internet is efficient for communication but not for distribution, CCN recognizes that great deal of information is produced once, then copied many times. This paper clears the idea about CCN, its applications, and the effects of device mobility over CCN based content delivery environments. The frequent network change of mobile consumer devices causes the performance degradation of content sharing, which leads to a bad user experience. So, the proxy-based CCN scheme is proposed to compensate for the effects of user movement.

The proxy node performs content sharing on behalf of mobile consumer devices and acts as the common point of old path and new path. From the proposed proxy architecture, it can prevent unnecessary packet transmissions at the previous location that the mobile device already moved out. Moreover, it can save energy consumption by reducing repeated transmissions of interest packets for data packets not received during network change. This feature is important in content centric networks since the device movement inherently causes dynamic topology change and high resource consumption.

REFERENCES

[1] http://www.parc.com/work/focus-area/content-centric-networking/

[2] http://www.ccnx.org/

[3] http://www.ccnx.org/releases/latest/doc/technical/CCNxProtocol.html

[4] "Proxy-assisted Content Sharing Using Content Centric Networking" Jihoon Lee, *Member*, IEEE and Daeyoub Kim, *Member*, IEEE

[5] http://www.ccnx.org/395/1/van-jacobsen-at-google/