CONTENT NAME RESOLUTION SERVICE IMPLEMENTATION FOR CACHE AND FORWARD NETWORK ARCHITECTURE

BY PUNEET KATARIA

A thesis submitted to the

Graduate School - New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Master of Science

Graduate Program in Electrical and Computer Engineering

Written under the direction of

Professor D. Raychaudhuri

and approved by

_____

_____

_____

_____

New Brunswick, New Jersey

OCTOBER, 2010

ABSTRACT OF THE THESIS

Content Name Resolution Service Implementation For Cache And Forward Network Architecture

By PUNEET KATARIA

Thesis Director:

Prof. Dipankar Raychaudhuri

Cache aNd Forward (CNF) is a proposed architecture for content delivery services in the future Internet. The CNF architecture takes advantage of reductions in storage to design a network that directly addresses the mobile content delivery problem. The CNF architecture uses a content name resolution service protocol, along with a reliable hop-by-hop transport protocol, storage aware routing protocol in place of end-to-end TCP for reliable delivery of large files. This thesis presents the algorithms proposed for a distributed name resolution protocol and design and experimental evaluation of the protocol on ORBIT in context of a multi-hop wireless access network scenario. The protocol is designed using hashing technique such that when a host queries for a file, the name service will be triggered and will return the addresses of nodes that cache the file. Since our architecture is about caching and forwarding large content files, enabling hosts to retrieve files from the network and not necessarily from the origin server, we need to uniquely identify the files. To that

effect, we propose to identify a file using a unique content identifier (CID) where CID is obtained by a one way hashing (SHA1) on the content itself. The aim here is to optimize selection of cache location and serve the host with the file from the nearest location. If the selected cache location is determined to be temporarily degraded, either due to poor channel conditions or mobility, the protocol uses multiple hash technique to provide alternate cache locations and the decision is based on the ETT metric provided by the routing protocol. The CNRS protocol over multi-hop 802.11 access networks with CNF routers has been implemented as a real-time proof-of-concept prototype on the ORBIT testbed. Baseline results for CNRS with hop-by-hop transport show that content based CNF network architecture performs better than TCP/IP stack. Using different content distributions, we have shown that multiple hashing, popularity based and location based caching provide significant gains over the baseline algorithm.

Acknowledgements

I would like to thank my adviser Prof. Dipankar Raychaudhuri for his constant guidance and support throughout the entire duration of my MS research. He gave me enough freedom to work in the field of my interest while making sure that I was always on the right path. I would also like to thank Professor Yanyong Zhang for motivating me to find the basic answers first before moving on to more complex scenarios. I thank Dr Sanjoy Paul for his initial guidance to help me get started on my thesis.

This work would not have been possible without Shweta Jain, Shivesh Makharia and Gautam Bhanage. Their constant support and enormous patience to deal with many of my doubts and questions have made this work conceptually sound. It was a great experience working with them and because of them I can confidently say that working on CNF project has been a very good learning experience. My time in Winlab was made memorable by my friends and peers in the CNF team Snehapreethi Gopinath, Mohnish Kulkarni and Lijun Dong. With them there was never a dull moment while working in the lab.

Additionally, I would like to thank Prof. Yanyong Zhang and Prof. Wade Trappe for being a part of my thesis committee, and for the invaluable comments and insights.

I am thankful to the many friends and office mates I have known in WINLAB as well. Finally, I would like to thank my family and friends for being with me and supporting me throughout my thesis work.

Dedication

To My Parents

Abbreviations

CNF	Cache aNd Forward
CNRS	Content Name Resolution Service
ORBIT	Open Access Research Testbed for Next-Generation Wireless Networks
WINLAB	Wireless Information Networks Laboratory
SHA	Secure Hash
MD5	Message Digest 5
DTN	Delay Tolerant Network
TNA	Transient Network Architecture
CID	Content IDentification
ETT	Estimated Transmission Time
ETX	Estimated Transmission Count
FTP	File Transfer Protocol
GB	Giga Byte
IP	Internet Protocol
MAC	Medium Access Control
Mbps	Mega bits per second
LFU	Least Frequently Used
OLSR	Optimum Link State Routing
STAR	Storage Aware Routing
PLR	Packet Loss Ratio
RTT	Round Trip Time
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
URN	Uniform Resource Name

TABLE OF CONTENT

LIST OF FIGURES

1 INTRODUCTION

The overwhelming use of today's network is for a client to acquire a named content. The named content can be a web page, a song, a picture, or a movie. For example, Torrent, which contributes more than 30% of traffic in the Internet, is all about acquiring named contents [1]. In order to solve this problem of efficient content distribution, we need a new networking approach that supports dissemination of cached content in an efficient manner.

The TCP/IP based internet architecture has proved effective through a period of growth and technological change in the network, but now faces a new set of challenges. Assumptions of stability and end-to-end connection have traditionally guided the design of these protocols, and have led to efficient information transfer and effective recovery strategies during periods of congestion. Now, however, this end-to-end strategy is challenged the wireless access technology that alters the nature of internet traffic, and challenges the assumptions upon which its protocols were built. Mobility has led to instability of Internet connectivity and made the easy assumptions of end-to-end traffic flow increasingly unreliable.

Because the changes caused by wireless mobility are fundamental, their solution requires fundamental changes in the architecture and protocols of the future Internet. We outline a cache-and-forward architecture that exploits the decreasing cost and increasing capacity of storage devices to provide unified and efficient transport services to end hosts that may be wired or wireless, mobile, and

intermittently disconnected. Fundamental to this architecture is a content network that provides in-network caching of content. The goal of the CNRS project is to design, implement and evaluate content caching and content retrieval that incorporates the following elements:

Distributed Caching: Distributed caching of popular content throughout the network, thus making file sharing a first-class service and enabling efficient content distribution.

Enhanced Naming: Routing to and from mobile terminals that exploits location information provided by an enhanced name service.

Delay tolerant architecture: For mobile nodes, the architecture enables opportunistic push-pull delivery of files, both to and from the wired network.

In this chapter, the CNF architecture and its key features are briefly described. This is followed by a logical representation of the CNF protocol stack where the interaction between CNRS and other protocols is outlined. The problem statement tackled in this thesis is described followed by related work.

1.1 Overview of CNF Architecture

The key concepts of the CNF network are shown in Figure 1-1.

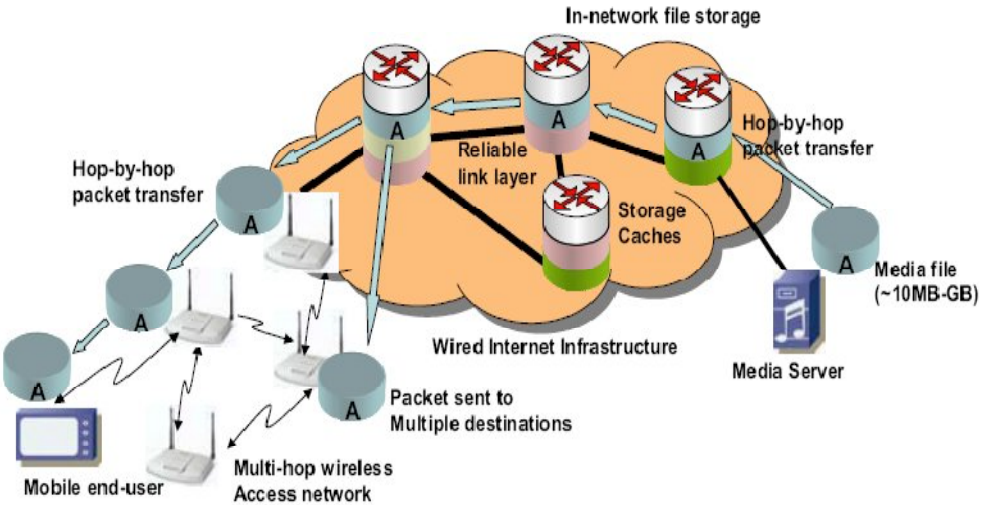

Figure 1-1 CNF Architecture

Source: Sanjoy Paul, Roy Yates, Dipankar Raychaudhuri, Jim Kurose. The Cache-and-Forward Network Architecture for Efficient Mobile Content Delivery Services in the Future Internet

The network is shown as consisting of wired as well as multi-hop wireless access networks. Consider a wireless user requesting for content from the media server. This internally triggers the content resolution service that suggests a possible cached location for the content. The wireless node sends the request to the suggested location and if there is a cache hit, the response is transported hop by hop towards the client. If there is a cache miss the request is forwarded to the origin server and in case an intermediate router has a cached copy of the requested file, the request propagation stops and the content response containing the file is returned following a hop by hop transport protocol. Any router along the route may decide to cache the file in its local storage space. It is possible that before the requested file reaches the client, the user moves away from the original request location. CNF introduces the concept of a Post Office (PO) to enable such

disconnected operation. The PO is responsible to maintain a pointer to temporary caches and the identifier of the mobile destination of the cached content. When a mobile user disconnects before receiving the content in transit, the file is stored at an intermediate router and the PO is informed. Once the mobile reconnects to the network, the PO sends it the pointer to cached locations of the previously requested content. CNF uses storage aware routing protocol that leverages the storage space on the routers to opportunistically store data in transit in case the route to the destination is temporarily suboptimal i.e., slower than usual. The routers maintain two routing cost metrics; the short term expected transmission time and the long term average of the expected transmission time. By simple comparison of the two values, a router is able to decide to temporarily store or to forward CNF content.

1.1.1 Addressing efficient content dissemination issues

Efficient use of bandwidth is the primary requirement for content distribution. This requirement is addressed by caching content at the CNF routers and delivering the requested content from the nearest CNF routers, thereby reducing congestion and saving the delivery bandwidth all the way from the distant server of the object. The advantage of delivering the content from a cache is low latency which is the most important metric from the end-user's perspective.

CNF architecture is driven by the philosophy that content is a unique entity irrespective of the location where the content is stored. Thus the requested content

can be delivered by anyone in the network as long as the content is fresh and authentic.

Freshness of the content is tracked by using version numbers such that each time content is modified the version number is changed. This concept is similar to published books where freshness is tracked with publication date. Authenticity of the content is verified by using a digital signature from the original content hosting server.

1.2 CNF Protocol Stack

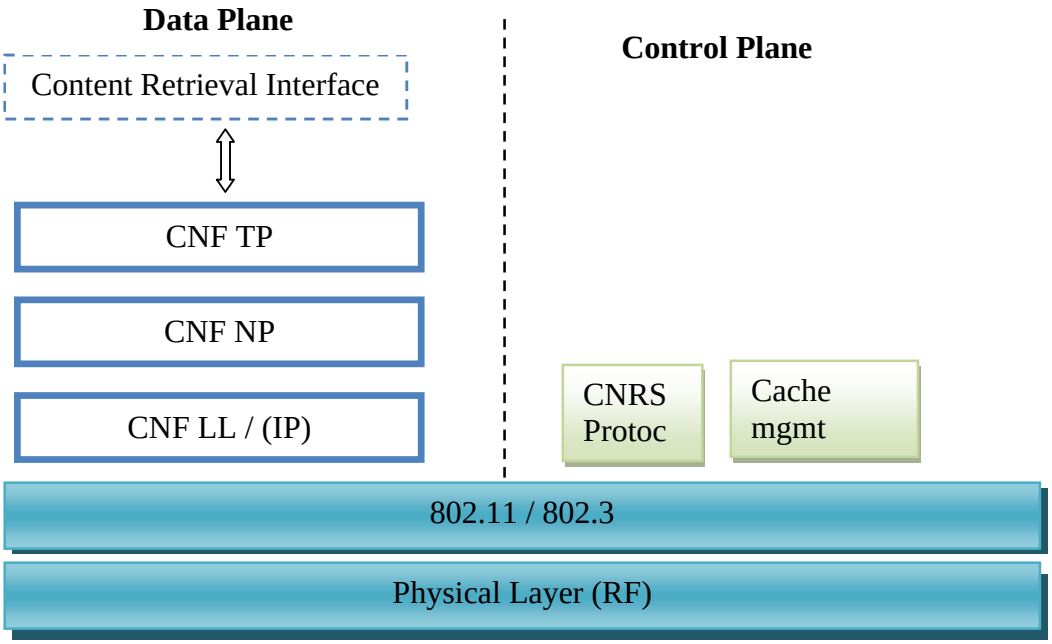

Figure 1-2 CNF Protocol Stack

The logical diagram of the CNF protocol stack is shown in Figure 1-2. There are several levels of storage in the router available for different layers of the protocol stack as explained in Section 3.2.

Data plane is broken down as the CNF Link Layer, CNF Network Protocol and CNF Transport Protocol. Applications send data comprising of large files to the transport layer which segments them into moderately sized chunks.

1. Link Layer: This layer comprises the logical link between two adjacent CNF nodes. The complexity of CNF Link Layer is reduced as compared to TCP/IP as the CNF upper layers provide all the features of the protocol.

2. Network Layer: This layer is responsible for the finding the next hop to the intended destination and runs the OLSR based storage aware routing protocol.

3. Transport Layer: Per hop reliability is achieved by CNF TP by exchanging acknowledgements as opposed to the end to end reliability in TCP.

Control plane provides the caching features of the protocol.

Content Name resolution Service (CNRS): This component maps the CID to the corresponding content. The CNRS uses a content hash to implement this matching as explained in Section 3.

Cache Management: This component runs the cache and cache replacement algorithms for the content. This algorithm generally uses popularity as a metric but other metrics such as distance from source may also be used.

The controls plane protocols leverage the embedded storage within CNF routers for caching content as explained in the next section.

1.2.1 Routers with storage

We propose that routers in the CNF architecture have sufficient storage.

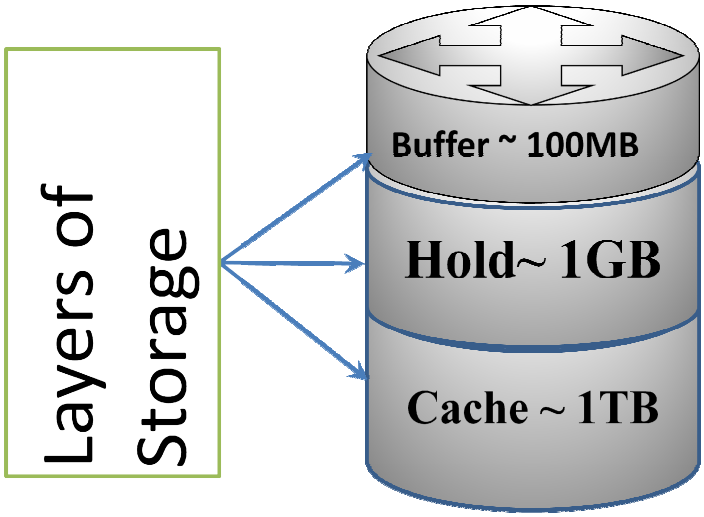

Figure 1-3 CNF Router Layers of Storage

Source: Dipankar Raychaudhuri, Shweta Jain. Emerging Wireless Technologies and the Future Internet

Specifically, we envision routers with three levels of storage as shown in the figure. The first level of storage called the buffer is used for storing the data in transition at the link layer. We propose a second level of storage to temporarily

store data when the router is in the store state. This level of storage is called the hold. The total space allocated to the hold should be large enough to store multiple files that are en route to a destination. The third level of storage in a router is known as cache. The cache space is used by the CNF application to cache files for longer time duration. The cache should be large enough (of the order of ~TB) to store several files. CNRS leverages this cache storage at routers to cache popular content and tries to reduce file retrieval time.

1.3 Problem Statement

Caches in the network create more complex scenarios. To receive a popular file, a host would query a file name service that would return the addresses of nodes that cache the file. To send a popular file, a host might request that a cache near the destination send the file. A number of networked applications have adopted a cache-and-forward architecture in the past. From the early UseNet news [2] to todays commercial Content Distribution Networks such as Akamai, several large scale networked applications have focused primarily on a one-to-many push of content from an origin server. Our focus here is on providing network services to provide many-to-many, user-driven cache and forward services. The challenge here is to provide cache location selection and cache location resolution services without adding overhead to the network and without making use of additional infrastructure (Handle systems [3] provides resolution services similar to CNRS but uses additional infrastructure like Global Handle registry and Local Handle Services)

1.4 Related Work

We split the related work into three different categories, wherein we

1) Compare handle systems as a name resolution service with CNRS
2) Comparing CID as a content identifier with URL's
3) Outline architectural differences between proposed future internet and CNF networks

1.4.1 Handle System Comparison

The Handle System [3] is a general purpose distributed information system that provides secure identifier and resolution services for use on networks such as the Internet. It includes an open set of protocols, a namespace and implementation of the protocols. The protocols enable a distributed system to handles/identifiers, of arbitrary resources and resolve those handles into the information necessary to locate, access or otherwise make use of the resources. The handle system requires set up of a DNS like architecture consisting of Global Handle Registry and Local Handle Services. CNRS is a distributed architecture that resolves the identifier locally using hashing. The handle system requires each digital object be registered under the DOI system to ensure a unique identifier. CNRS uses the CID concept where CID is obtained by hash of the content itself.

1.4.2 Comparing URL's with CID

CID's are resource identifiers with the specific requirements for enabling location independent identification of a resource, as well as longevity of a reference. A URL [4] is location bound and defines the mechanism as to how to retrieve the resource over the web. A CID is just a name and isn't bound to a network location. The resource identified by a CID may reside in one or more locations at any given time, may move, or may not be available at all. As such a URL is identifying a place where a resource may reside, or a container, as distinct from the resource itself identified by the CID.

1.4.3 Architectural Differences

Disruption/Delay Tolerant Networking (DTN) architecture [5] has significant similarities with CNF architecture. However, there are major differences as well. DTN network is an extension of the TCP/IP network for disconnected environment. As a result, applications interface with DTN network in a manner similar to how they interface with TCP/IP networks. In CNF network, applications interface with the network in a way very different from the way they interface with the TCP/IP networks. An application requests the network to retrieve a content specified by Content ID (CID) and this is very different from connecting to a specific machine for the purpose of retrieving information. DTN network is driven by disruption while

CNF is driven by a combination of mobility support, content delivery and intermittent connectivity.

1.5 Thesis Organization

The rest of the thesis is organized as follows. In chapter 2 we explain the data plane protocols adopted for the ORBIT implementation of the CNF protocol stack. We explain the design of our content name resolution service and different hashing algorithms in chapter 3. Chapter 4 details implementation of CNRS on the ORBIT testbed, experiment scenarios and the results obtained in comparison with the TCP stack. Finally Chapter 5 summarizes the conclusions and future work.

2 CNF PROTOCOL STACK

2.1 Transport Layer implementation

The hop by hop transport concept has been proposed in the HOP [6] protocol. Here files are divided into bulks of 1MB and sent over UDP to the next hop. Reliability is added to on every hop, over UDP. This concept is fundamentally similar to the CNF transport layer. Therefore, we take advantage of the available implementation and make modifications to adapt the HOP implementation as the CNF transport protocol. The details of modifications and implementation on ORBIT are explained in Section 4.

2.1.1 The Network Layer

This layer is responsible for finding the next hop to the destination. This layer uses a storage aware routing protocol which can choose to store rather than forward data. This protocol transmits files only when the end to end path quality is above a certain threshold. The aim here is to optimize the overall network throughput through opportunistic transmission during periods of good path quality. If the routed path is determined to be temporarily degraded, either due to poor channel quality or mobility, the protocol may decide to store the file.

The protocol uses a modified Optimum Link State Routing [7] (OLSR) protocol called OLSR-D protocol. This protocol keeps track of the network by sending HELLO and TC messages.

Every router calculates two moving averages of ETT from information received from each other. The first average is over a large time window. The size of the window may be a tunable parameter. The second average is over a fraction of this larger window size [8].

2.1.2 Link Layer and its implementation

The file is fragmented into batches of MAC protocol data units which are typically

1KB packets. We used a modified madwifi driver [6] at the link layer. Per hop reliability is achieved through block ACKs at the Transport Layer. Hence the ARQ ACKs is disabled for the link layer and burst mode is enabled. The 802.11e burst mode transport is used to send blocks of data over the wireless link.

3 CONTENT NAME RESOLUTION SERVICE PROTOCOL DESIGN

3.1 Content Naming and Resolution Service (CNRS)

The efficiency of content retrieval in CNF network depends on content caching, content discovery and content delivery [10]. Content caching algorithms decide which CNF router(s) should cache the content. Since copies of the same content are cached in multiple CNF routers in the network, discovering the CNF router with the desired content is the main task of Content Name Resolution Service. Once a CNF router that caches the content is discovered, the delivery process is governed by the underlying transport and routing protocol. Since multiple copies of the content are available in the network, it is imperative that the network considers all cached copies as same. To solve this issue we identify content by using globally unique content identifiers (CID). The concept of a CID is in contrast to identifiers in the Internet, where content is identified by a URL whose prefix consists of a string identifying the location of the content. We assume content requestors (i.e. end nodes) are equipped with the CIDs of the requested contents, but not their locations.

CNRS is the framework for mapping content names/CID in the network to content location. Any request for content will be forwarded to the cached router location within the network. If content is not cached then the request is forwarded to the CNRS server which would resolve the CID to its attributes. Attributes corresponding to a CID consist of information pertinent to the content, such as Content Location, content size, popularity ratio, information on cacheability, etc.

3.2 CNRS Hybrid Architecture

CNRS servers are scattered in each stub as well in the core network. As shown in Fig 1, in Stub A, content hosting/content publishing servers named A.1, A.2, A.3 are connected to CNF caching routers to construct a local CNRS network. Each stub also has a CNRS server. Requests for content from clients could be served by one of the wired or wireless caching routers based on the hashing algorithms explained below.

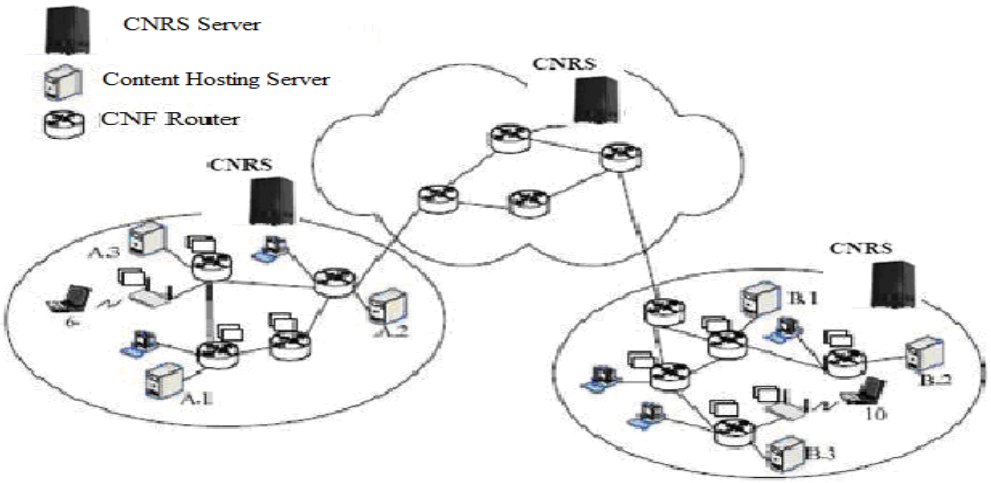

Figure 3-1 CNRS Architecture Diagram

3.3 CNRS Interface

Following description explains the basic CNRS interface with other network entities

Put / insert operation - A content originating server informs the corresponding CNRS server of its location. Once content is cached by a router, that information is also passed to the corresponding CNRS server by a put operation

Get / retrieve operation – A content requesting node sends a get request to the corresponding server to find content location and possible cached locations.

Delete operation – Once content is purged from a router, a delete operation removes the corresponding location information of that content.

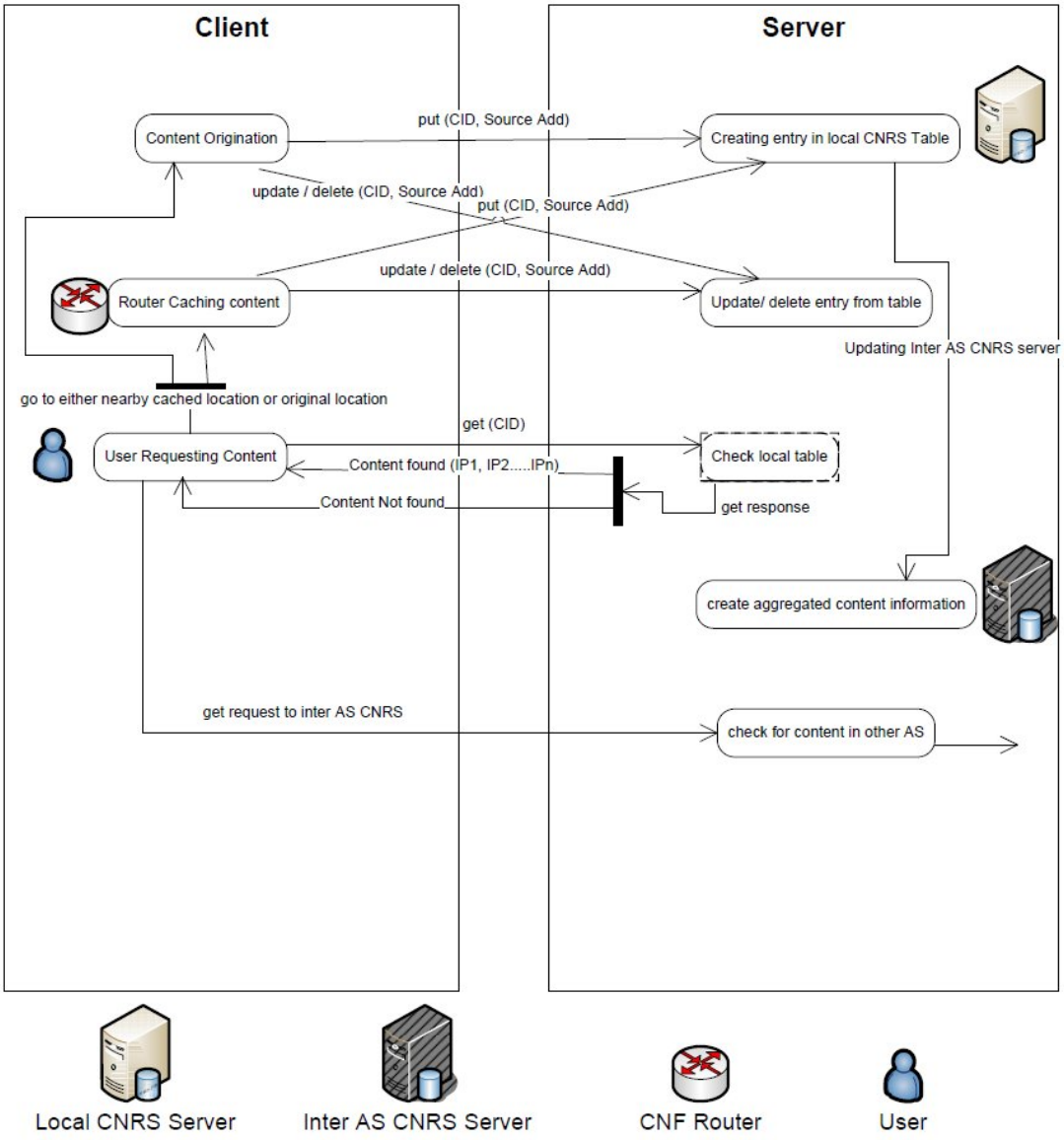

Figure 3-2 CNRS Client Server Interface

3.3.1 Put (100) / update (100001) / delete (1000001) Operation

[The bits in the bracket indicate the control bits assigned to differentiate between different packets. For more information on the control bits refer to the appendix A]

Once a content is created and published by a node, the node uses the well-known hash function on the CID to generate the KeyID, which in turn informs the node which CNRS server in the same AS it should send content name resolution information to, provided that IP addresses of all CNRS servers within that AS are known . Then the information is updated to the local CNRS server with its attributes, such as CID, size, popularity statistics, whether it is cacheable or not.

A put message packet is as shown:

<table>
<tr><td colspan="4">0 7 15 23 31</td></tr>
<tr><td colspan="4">Source Address</td></tr>
<tr><td colspan="4">Destination Address</td></tr>
<tr><td>Control Bits</td><td colspan="2">CID</td><td>Size</td></tr>
<tr><td>Popularity</td><td>Cacheable</td><td>TOS</td><td>Checksum</td></tr>
</table>

Figure 3-3 Put Control Packet

The Packet Fields are explained below:

Source Address: It is the address of the content originating server or the router that has cached the content.

Destination Address: It is the address of the corresponding local CNRS server that is responsible for the particular content ID

Control Bits: These 8 bits inform the nodes of the type of message that is included in the packet

If the first bit is 1, it indicates it is a content (CNRS) packet

If first 4 bits are 1000, it is a put request message. If the fifth bit is 0, it is a put request by a content originating server else the request is by a router caching the content within the network (Appendix A).

CID: It contains 16 bit Content ID which is stored in the CNRS table.

Size: It contains size of the content file in KB

Popularity: It is a ratio to indicate the content popularity to enable cache management.

Cacheable: It is set to a value as decided by the caching algorithm (popularity and cacheable fields are yet to be clearly defined)

TOS (Type of Service): Set to tell which type of packet it is. It indicates the content is (1) popular and needs caching with high TTL, (2) transient and needs caching with small TTL, (3) short message that needs quick forwarding, or (4) real-time and requires forwarding within bounded time.

Checksum: An 8 bit checksum to check for errors.

Table entry at CNRS server

Once a CNRS server receives a put request, it checks if the CID entry is already present in its table. If the entry is there, the new location is added to the entry. If not, then a new entry is created for that particular CID. A corresponding put-ACK message is sent to the node that initiated the put message.

An entry in CNRS server is as following. More fields can be added, such as routing history for enhancement.

Table 1 Entry in CNRS

0–16	17–33	33–41	41–49	49–57	57–65	65–72
CID	Origin bit	Origin address	No of cached locations	Size (KB)	Popularity	Cacheable	TOS

0–16	16–31
CID	Cached Location$_1$
CID	Cached Location$_2$
CID	
CID	
CID	Cached Location$_n$

Figure 3-4 CNRS Table Entry

Origin Bit: It indicates if the content originating server is in the same AS or not. If it is, then origin address contains the address of that content originating server.

No. of cached locations: It gives count of number of routers within AS which have cached that particular content and whose location information is available to the CNRS server.

Cached locations [1, 2…n]: Cached locations exist as a linked list to the main CNRS table with no. of cached locations value indicating the number of entries in the linked list.

Size, popularity, cacheable and TOS are same as in the put message packet.

Put- ACK message packet

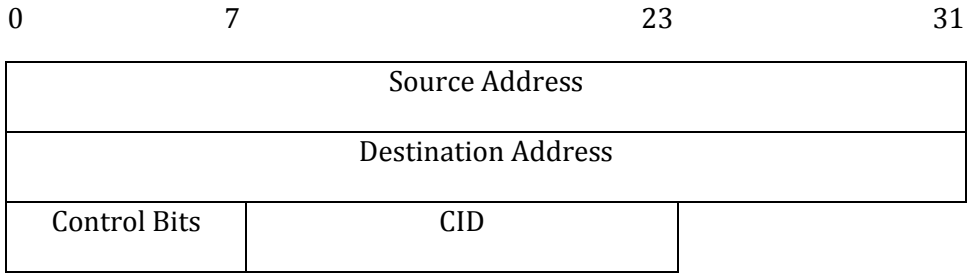

Figure 3-5 CNRS ACK Packet

Source Address: It is the address of the CNRS server that received the put request message

Destination Address: It is the address of the node that initiated the put request message

Control Bits: If the first 4 bits are 1001, then it is a put ACK packet.

Communication between local CNRS and Inter AS CNRS

Each local CNRS server sends an update periodically to the Inter AS CNRS server. This update contains range of CID's whose entry is available in its table.

Communication between two Inter AS CNRS

Once updates from local CNRS are received, the Inter AS CNRS creates an aggregated list of CID based on Run length coding. This aggregated list of CID's is then sent to the other Inter AS CNRS. An important factor to be noticed is the cache hit/miss, since the aggregated result does not guarantee of content retrieval.

Get / retrieve operation (1010)

A content requesting node generates a get message. Based on the content ID the message is forwarded to the corresponding CNRS server.

A get message packet is as follows:

<table>
<tr><td colspan="3">0 7 23 31</td></tr>
<tr><td colspan="3">Source Address</td></tr>
<tr><td colspan="3">Destination Address</td></tr>
<tr><td>Control Bits</td><td>CID</td><td>TOS</td></tr>
<tr><td>Checksum</td><td></td><td></td></tr>
</table>

Figure 3-6 Get operation control packet

Control Bits: If the first 4 bits are 1010, it is a get message.

Two types of responses can be received from the local CNRS server.

Response to get – Content found (10110)

Packet structure is as follows:

<table>
<tr><td colspan="4">0 7 15 23 31</td></tr>
<tr><td colspan="4">Source Address</td></tr>
<tr><td colspan="4">Destination Address</td></tr>
<tr><td>Control Bits</td><td colspan="2">CID</td><td>Size</td></tr>
<tr><td colspan="4">Origin address</td></tr>
<tr><td>Packet length</td><td>Popularity</td><td colspan="2">Cacheable</td></tr>
<tr><td colspan="4">Cached location$_1$</td></tr>
<tr><td colspan="4">Cached location$_2$</td></tr>
<tr><td colspan="4"></td></tr>
</table>

Cached location$_n$		
TOS	checksum	

Figure 3-7 Get Response Packet

First five control bits as 10110.

Response to get – Content not found (10111)

Packet structure is as follows:

0	7	23	31

Source Address		
Destination Address		
Control Bits	CID	Checksum

Figure 3-8 Get Response Packet

First five control bits as 10111.

Once content not found message is received, the requesting node sends a get message to the Inter AS CNRS server, which directs it to the other AS which might have the content.

3.4 Algorithms for content mapping

Unique mapping based on IP and hash - CNF nodes within the AS can use the same hash function (SHA – 1) to find the router which might contain a cached copy of the content. Every CNF router has a unique RouterID, which can be obtained by using a hashing function such as SHA-1 on the IP address of that router. Every content will have a unique KeyID which may be obtained by employing the same hash function SHA-1 to the CID of the content. A content name resolution information is located on the first CNF router whose RouterID is equal to or follows the KeyID [min (RouterID - KeyID)].

3.4.1 Baseline Algorithm (SHA – 1 hash mapping)

SHA – 1 of Content ID (CID) for a particular content gives a 20 byte digest (hexadecimal numbers) This digest is compared with SHA-1 of IP address of all CNF routers within AS to give a distance metric approach

SHA1 (CID) - SHA1 (IP cnf router1)
SHA1 (CID) - SHA1 (IP cnf router2)
. .
. .

} The IP that gives the minimum distance metric is selected as the caching location for that content

The SHA hash mapping would work well only when wired nodes are present.

Including link quality (ETX) from the CNF routing protocol as a metric to choose caching location would provide an alternative to choose good nodes over low quality wireless nodes for caching. In the next algorithm, we use the OLSR long term ETX metric to limit the possibility of selecting low link quality nodes for caching

3.4.2 Multiple Hash algorithm using ETT metric

Here we perform the same hashing algorithm but on 3 different types of hashes namely SHA 1, SHA 2 and MD5. This gives us 3 content mapping locations. The CNRS control layer then looks at OLSR routing table to compare the best available long term ETT for the 3 locations. The advantage of using multiple hash over the baseline algorithm is that routers with bad link quality can be avoided for caching.

3.4.3 Popularity based algorithm

Here along with content name to CID mapping, we assume that knowledge of content popularity is also available. Based on content popularity, the hashing algorithm comes up with multiple caching locations within the AS. For example if popularity for some content is 2, the mapping algorithm provides 2 router locations. This indicates that popularity ratio of the content has increased and hence 2 locations will reduce the chances of a cache miss.

3.4.4 Location aware popularity based caching

The goal here is to select a set of nodes for caching popular content such that content is at minimum worst case distance for all end nodes/clients. Once such a set of nodes is known to us, then selecting one node from the set can be done based on the hash. The less popular content can then be cached at the edge nodes.

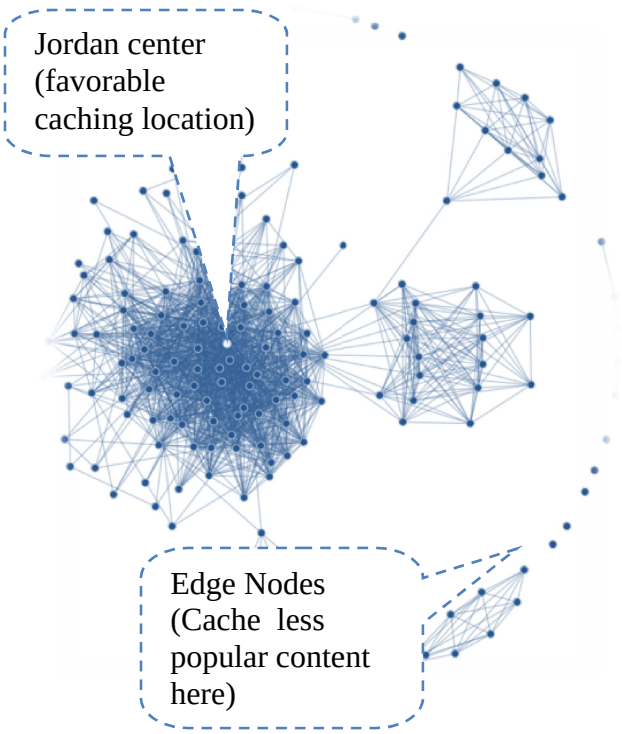

Figure 3-9 Network with central node

This algorithm requires implementation of a module which finds the center set (Jordan centers) which are the points of minimum eccentricity in a graph. This concept is used in social network analysis where they calculate the closeness centrality measure. Here in this diagram, the yellow dot indicates the central node which has the minimum eccentricity.

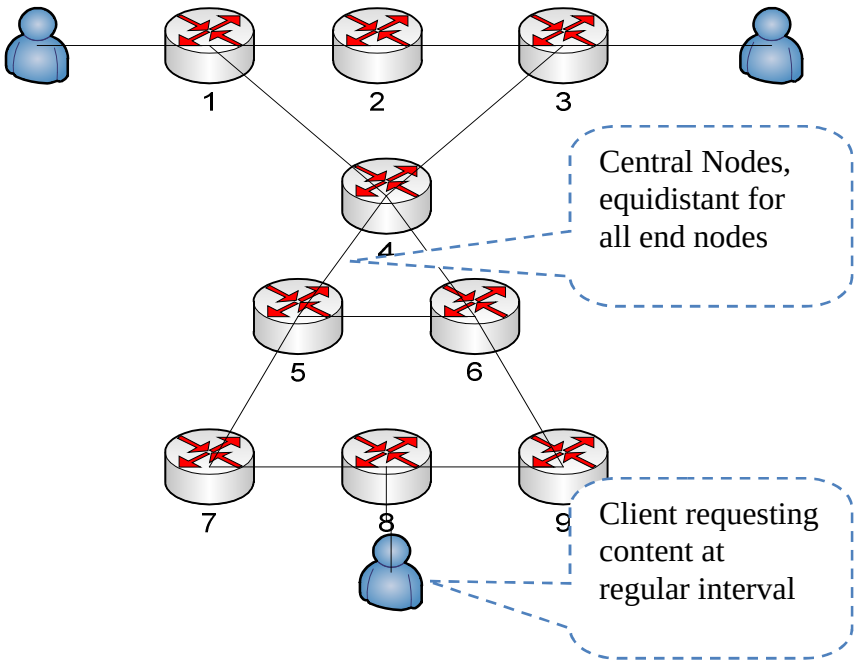

Figure 3-10 Network Topology with 3 central nodes

For example, for the given topology, we use the center set ideology to find {4, 5, 6} as favorable caching locations. While caching, we run hash algorithm on this set for popular content and use the remaining nodes for caching less popular content. In this way, the average file retrieval time for requests generated from extreme ends of the network can be reduced.

3.5 Flow Diagram indicating CNRS protocol working

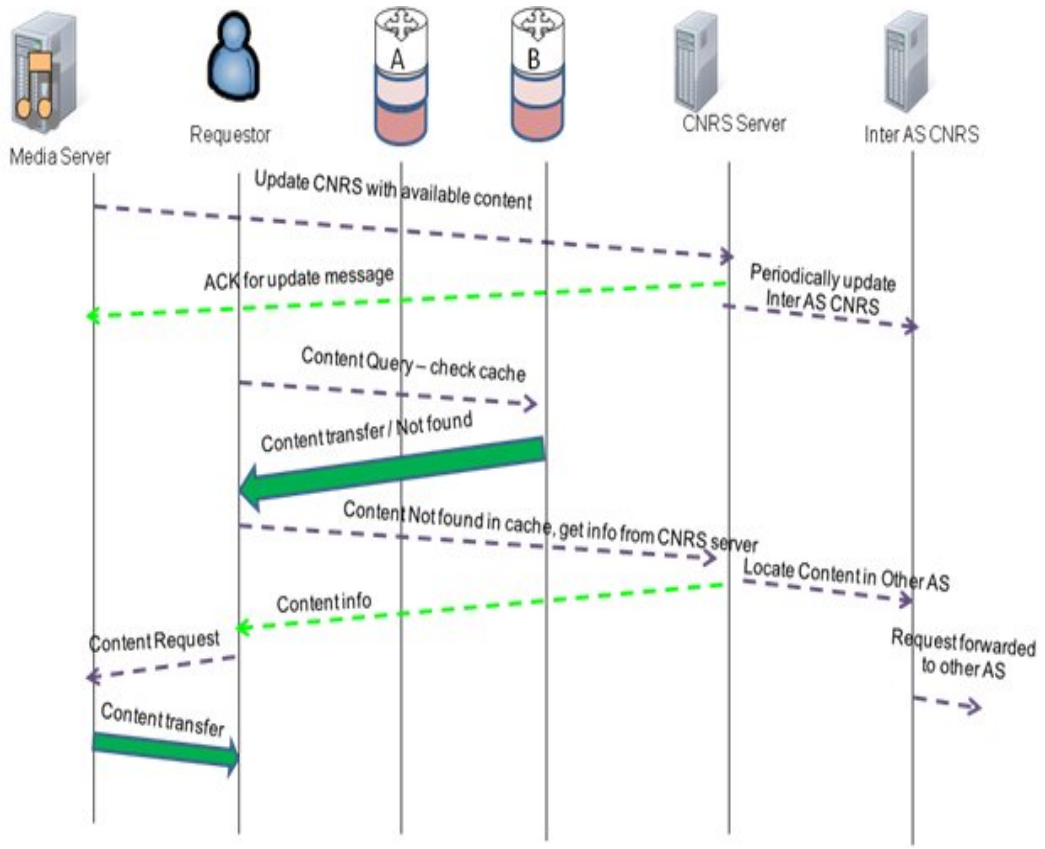

Figure 3-11 CNRS Flow Diagram

All servers publishing content send update messages to CNRS server with the content ID (CID). The CNRS server maintains a list of all these CID's and its attributes. When a client (EN) sends request for content, the CNRS protocol uses the hash algorithm to locate the cached router location within the AS. Request for the content is forwarded to the cached location. If the same content was requested earlier then a cached copy of that content will be available at the router location. If

there is a cache miss, the request is forwarded to CNRS server which sends back content location information to the client. The client can then retrieve the content from the server publishing the content. If the CNRS server does not find information for a particular CID, then request is forwarded to Inter AS CNRS server in the core network which maintains information of contents in other networks.

4 EXPERIMENTATION AND RESULTS

We have already described the Transport, Network and Link layer implementation details in Chapter2. Here we look at the CNRS implementation.

Our implementation uses HOP for the transport layer. The implementation diagram for Hop is shown

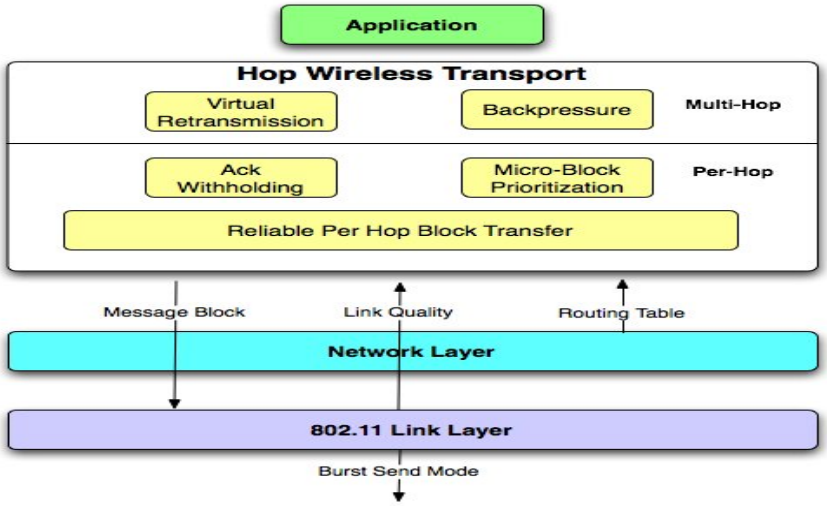

Figure 4-1 HOP architecture

Source: M. Li, D. Agrawal, D. Ganesan and A. Venkataramani. Block-switched networks

The HOP implementation is an open source code written in C++. Hop runs on ORBIT nodes on a LINUX 2.6x kernel and a Modified madwifi driver. We make the following changes in HOP. First we disable the backpressure flow control scheme in HOP since in CNF, the store and forward network layer provides the flow control mechanism. Hop connects to the dynamic routing protocol using a socket connection to obtain the next hop. The intermediate nodes store all the blocks in memory until they have been successfully transferred to the next hop. To enable storage, we disable the hop backpressure limit by setting it to a maximum limit (65536). If an entry for the next hop in missing in the kernel table, Hop considers the routing decision as store, and holds on to the packet. We implement a storage module to implement temporary storage of content in case the network layer decides to do store instead of forward. The basic unit of transfer, block, size is set to 1MB. All other HOP default settings are maintained. CNRS uses HOP as its transport layer and Modified OLSR to determine long term ETX.

4.1 CNRS + Hop-by-Hop Transport layer + Modified OLSR

CNRS is implemented in C++ as well and the CNRS modules use the BulkAPI sockets provided by HOP to interface CNRS and HOP as shown in the figure. To enable CNRS functionalities, we have modified the HOP sniffer filters to be able to accept CNRS packets. CNRS uses send (control bits, CID, data) and receive(control bits, CID, data) socket API's to communicate requests and responses. The control

packets indicate the type of packet being transferred. Figure explains the basic working of CNRS and its interface with HOP.

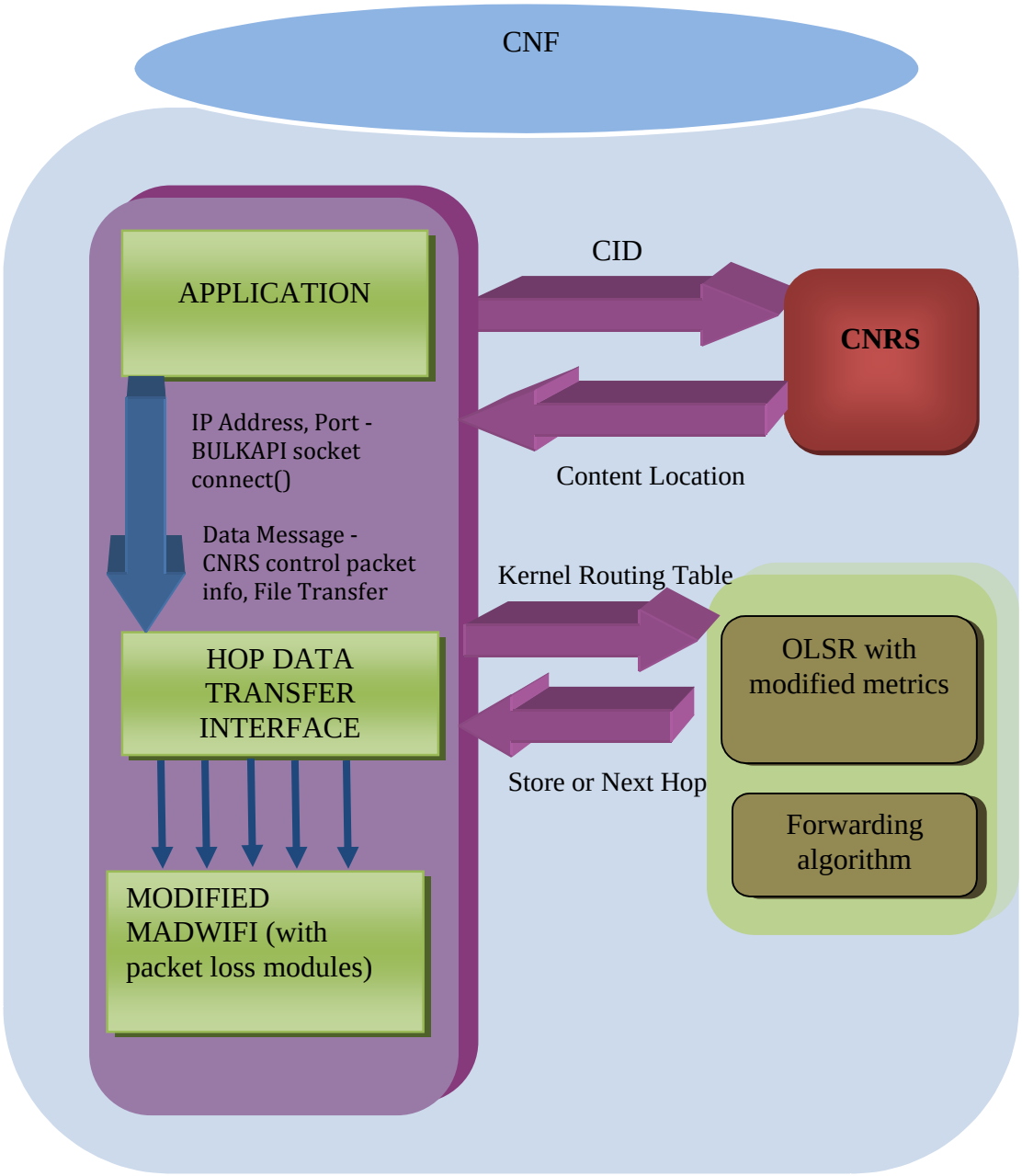

Figure 4-2 Software modules as implemented on ORBIT nodes

The driver maintains two queues – One for content and one for CNRS control packets. Control packets follow 802.11e best effort settings such that link layer provides reliability for these control packets. Per hop reliability is achieved through block ACKs at the transport layer. Hence, unlike the control packets, the ARQ ACKs is disabled for the data queue, and burst mode is enabled. The 802.11e burst mode transport is used to send blocks/bursts of data over the wireless link.

4.2 Experimental Setup

As mentioned earlier, all the experiments were set up on the ORBIT platform. As all the nodes in the grid are in the communication range of each other, we used selective packet dropping mechanisms to create different topologies. We program nodes using tools like mackill and iptables so that we can drop all packets at the medium access layer from nodes which are topologically more than a single hop away but otherwise in communication range of each other. Also to account for varying channel quality and loss rate, we modified the madwifi-driver to randomly drop packets and emulate lossy links.

The other experimental parameters are detailed in the table below:

Parameter	Value
Number of distinct content files in the system	30
Cache Space at each router	3 files (1.5MB)
Cache Replacement Policy	LFU
Number of content hosting servers	2
MAC Protocol	802.11g
Bit-Rate	Auto – rate
Mode	Ad-hoc
Channel	11

Figure 4-3 Experimental Parameters

Our aim is to study and compare the performance of wireless networks running CNRS under different network conditions.

4.3 Initial Results

In our first experiment, we create a multihop wireless scenario using the packet filtering techniques described in the previous section. We create a topology consisting of 2 clients, 2 origin servers and a CNRS server as shown in the figure. We study the effect of exponentially increasing offered load on throughput and average file latency.

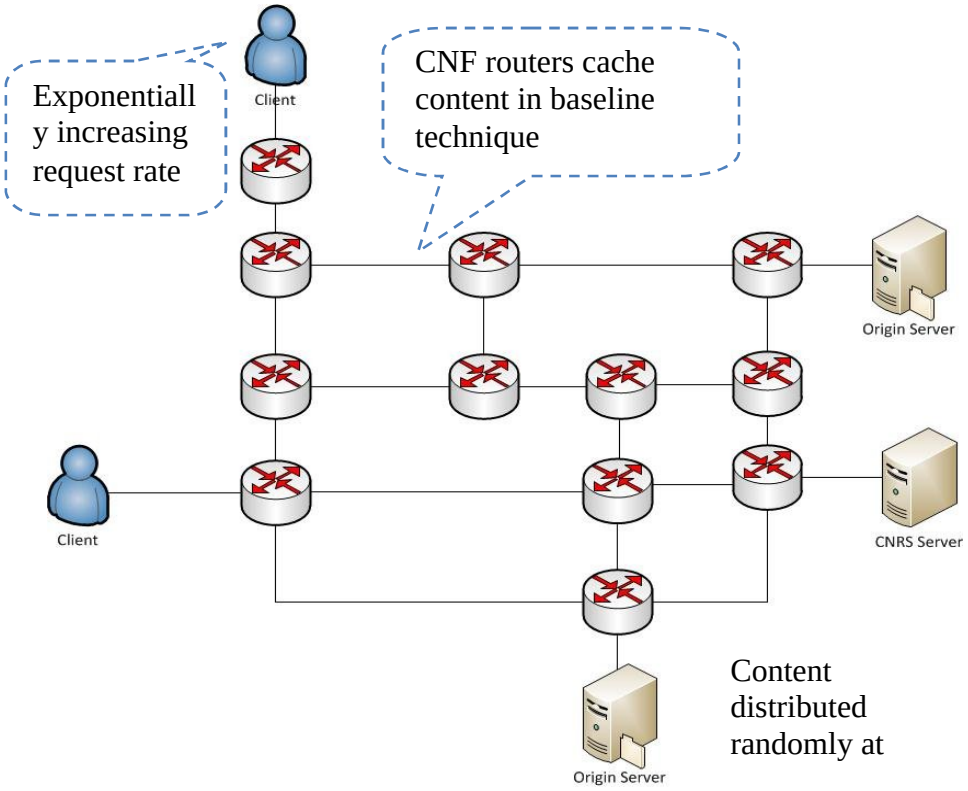

Figure 4-4 ORBIT MultiHop Topology

30 unique content files are distributed randomly at the 2 origin servers. Before the actual experiment, we send update control packets from the origin servers to the CNRS server. This way the CNRS server maintains a table that contains CID and location information for all 30 files. Each client requests for all 30 files three times each. The interval for file requests is reduced at each run of the experiment to increase the offered load

Each client requests for a file at the rate of λ per sec, where λ is a random variable with an exponential distribution. The size of the file to be transmitted is kept

constant for a given experiment.

Given these parameters, the offered load is calculated as

Offered Load (Mbps) = λ * file size (in MB) * 8 * number of sources transmitting

By varying λ, we vary the offered load in the network.

For the following experiments, the packet size for each data transfer was set to 500KB. In the following experiments we measure throughput and average file latency for baseline hash algorithm vs no caching.

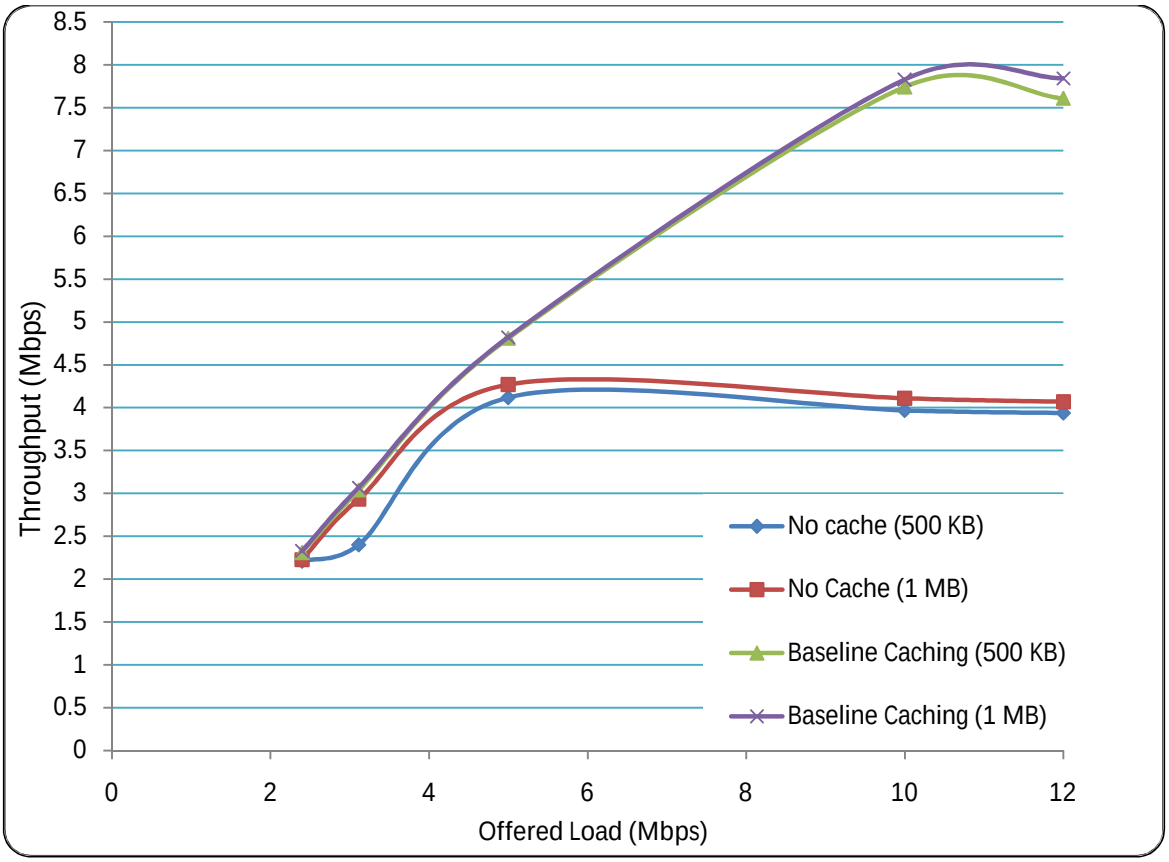

Figure 4-5 Throughput comparison under varying offered load

Figure shows that as the offered load increases, the network begins to saturate

after 5Mbps if no caching is available. The throughput rises steadily with the offered load in case of baseline caching since caching reduces load on the origin servers and also load on the network as the content can be fetched from nearby cached locations.

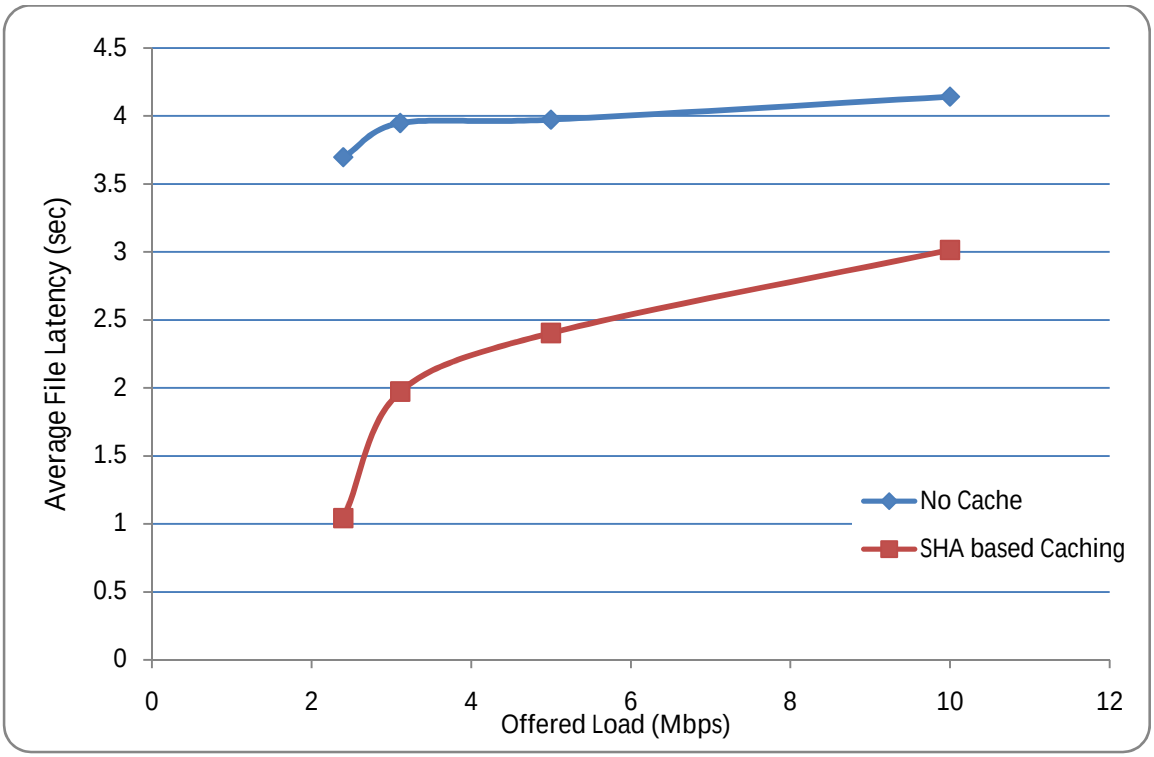

Figure 4-6 Average File Latency comparison under varying offered load

File delivery latency for each individual file is maintained throughout the experiment and average file latency is calculated. We can see that at an offered load of 3 Mbps, average file latency for no cache run of the experiment is almost double as that of caching based. Even at higher offered loads, the difference in latency is very high x.

4.4 Experiment on popularity based caching

For this experiment, we use 10 distinct files evenly distributed at the 2 origin servers. To define the popularity index, we use 2 different distributions for the experiment.

Equiprobable Distribution

Each client requests all the files 3 times. We have created a shuffle generator module that randomizes the requests. Thus the popularity index of each file will be the same.

Total file requests = Total files * Number of clients * 3

= 10*2*3

= 60

Gaussian Distribution

The total file requests are kept constant but since it's a Gaussian distribution some files will be requested more number times than the others and hence will have higher popularity index. Mean for the Gaussian distribution is kept as 4 and standard deviation as 2. The Gaussian distribution is created using box muller[] open source code. The file request distribution scenario is as shown in the figure

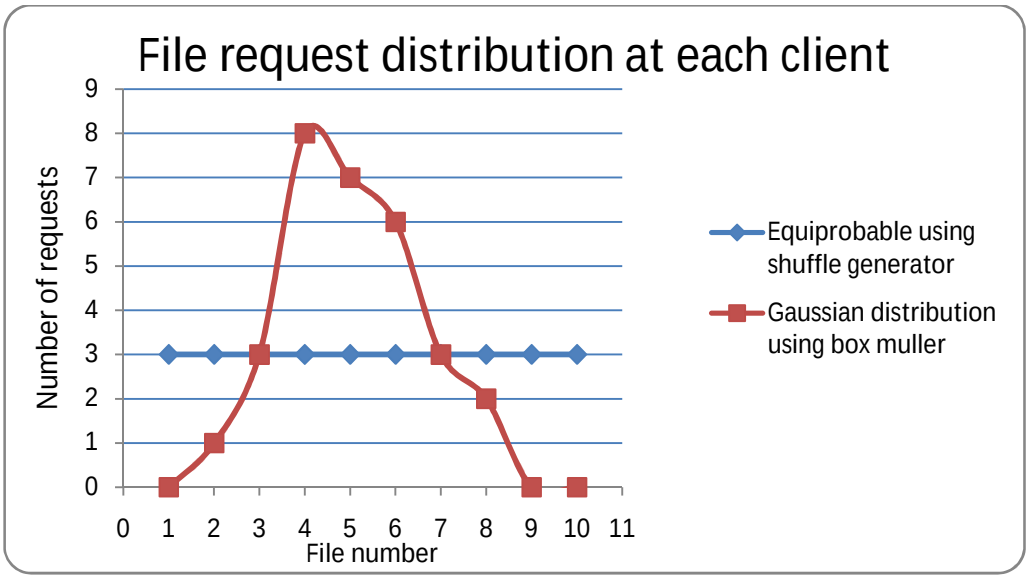

Figure 4-7 File request distribution

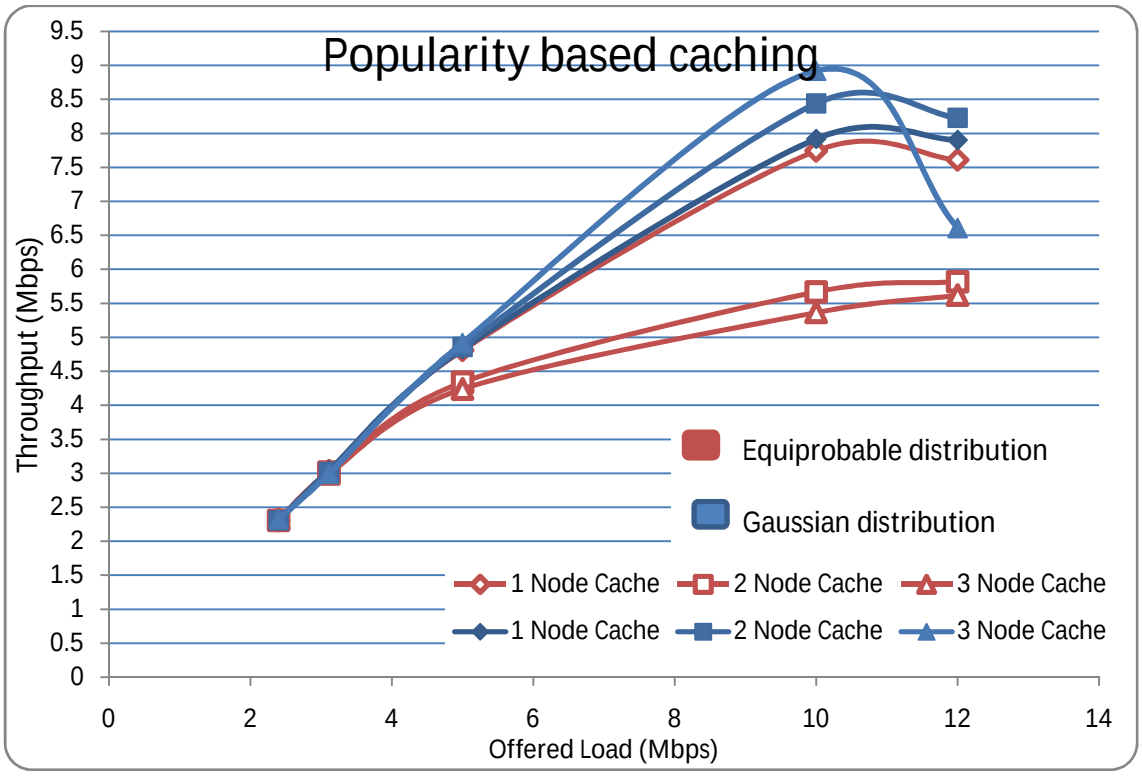

Figure 4-8 Throughput comparison in popularity based caching

Figure shows that if all files have equal probability of request (same popularity), then caching the same content at 2 or 3 routers, reduces the throughput. The overhead to send all the content from the origin server to more than 1 caching router creates network congestion and throttles the bandwidth. As opposed to this if files have different popularity, then only popular content will be cached at more than one location. The Gaussian distribution results show that caching popular content at more than one location improves overall network throughput. In real world scenario, some popular content will be requested by clients more number of times from a given network stub and this creates local popularity. Caching such content at multiple locations will increase the probability of cache hit and avoid cache replacement of popular content.

To compare CNRS integration with HOP and TCP, we kept the packet loss rate at each node at 1% and observed the throughput when client receives content from cached locations at varying hop count distances.

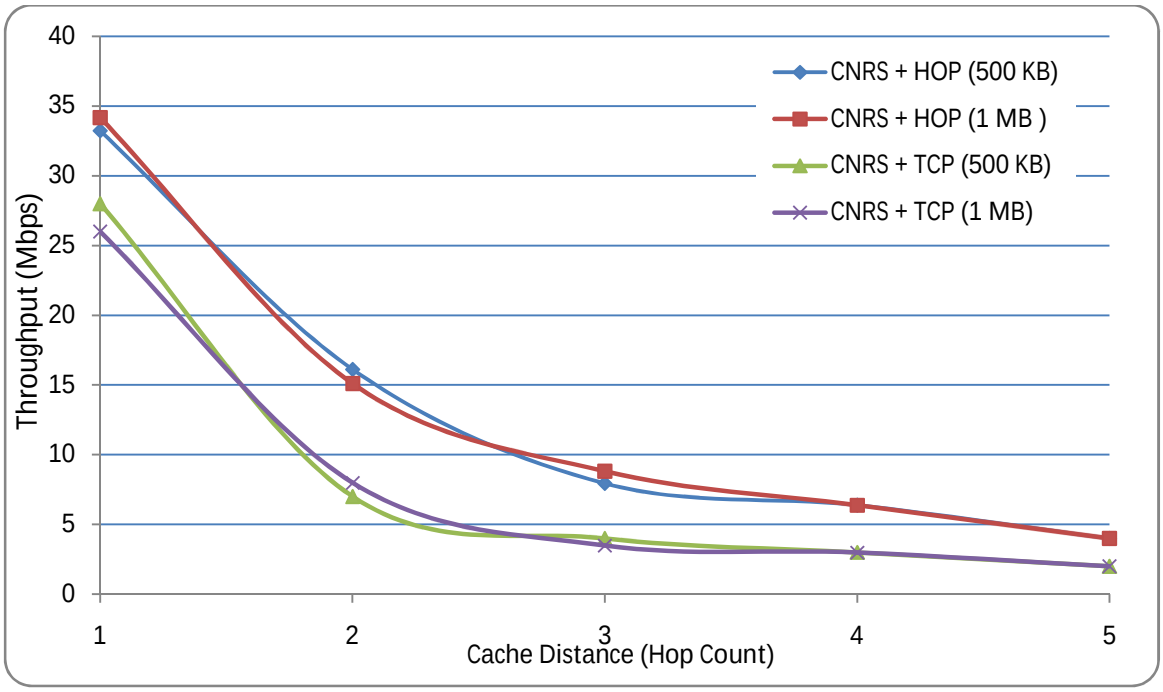

Figure 4-9 Transport Layer Comparison

Retransmissions take place from source when TCP loses packet. In CNF, hop by hop reliability is imposed by the HOP transport protocol and hence only one hop retransmissions take place. In the above figure, we can see the difference in throughput when CNRS is used with HOP as opposed to using CNRS with TCP.

4.5 Cache Hit Evaluation

We use the same distributions and request scenario to get a scatter plot for file

latency and cache hit percentage.

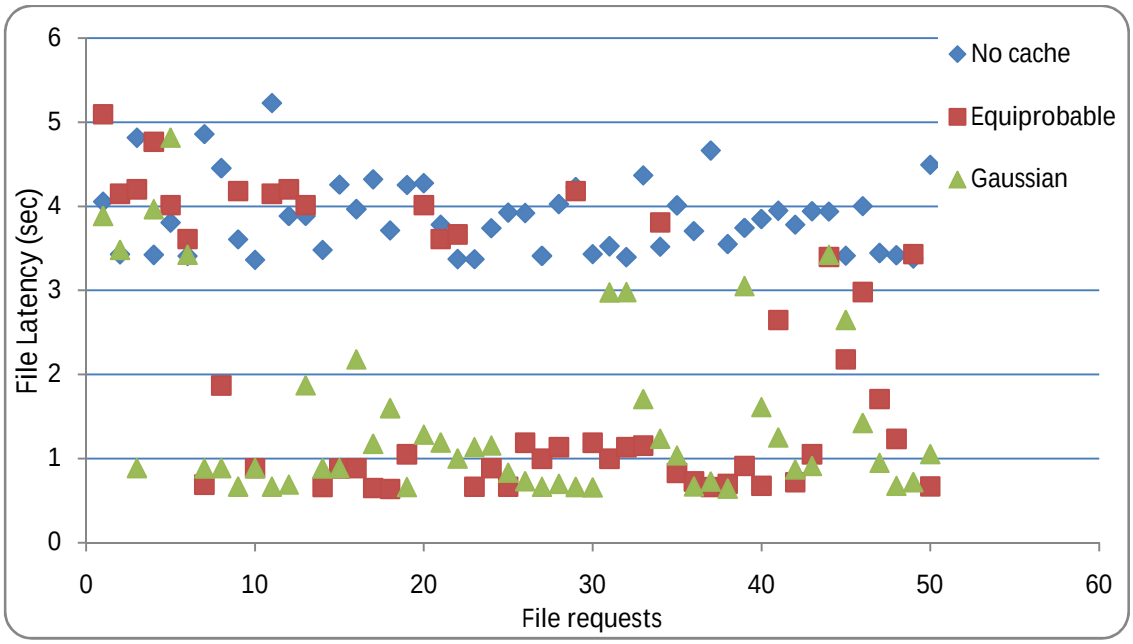

Figure 4-10 File Latency Scatter Plot

We can observe that for the first few file requests, the file latency for all three (no caching, equiprobable requests, Gaussian requests) is almost the same. Here since the files have not been cached yet, all requests are being served by the origin server. As files start getting cached, file retrieval latency for equiprobable requests and Gaussian requests reduces dramatically.

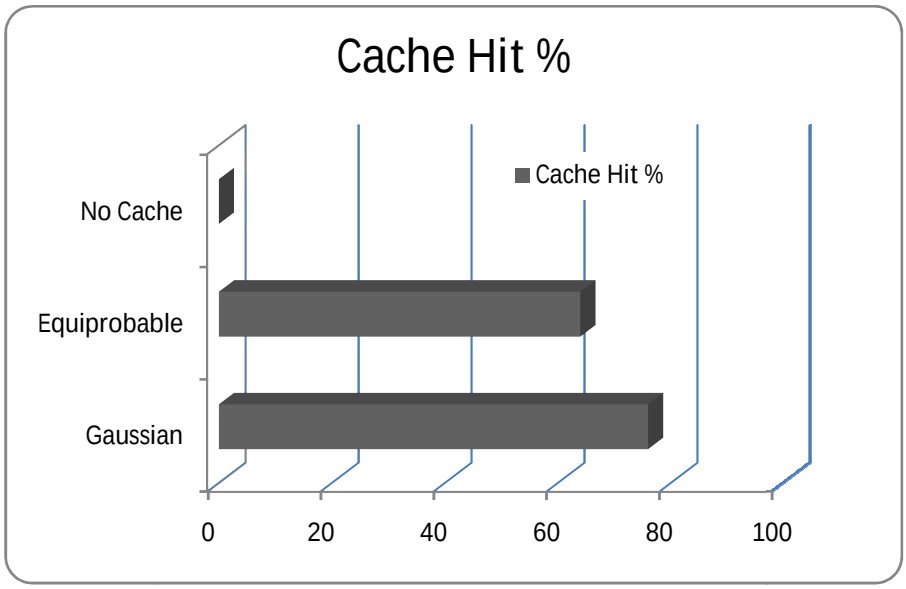

Figure 4-11 Cache Hit Evaluation

Some content files have a higher request rate in Gaussian distribution. The LFU cache replacement policy used for CNRS ensures that higher request rate files are saved in cache and least frequently used ones are replaced. The Percentage graph shows that LFU policy helps to achieve higher cache hit %.

4.6 Using multiple hash and ETX metric

In our next experiment we keep offered load constant and vary the packet loss at the madwifi driver for some routers. The black routers in the topology represent the ones that have lossy links.

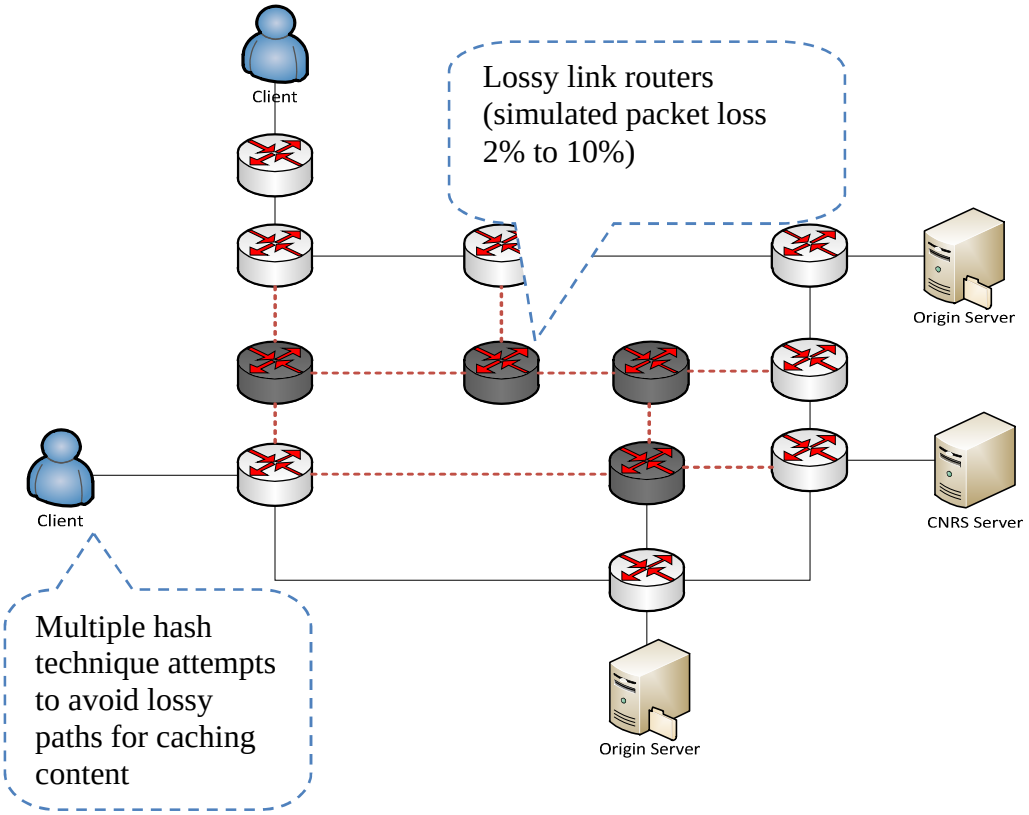

Figure 4-12 MultiHop Topology with lossy links

Packet loss at the 4 highlighted routers is varied from 2% to 10% and throughput is measured for both baseline and multiple hash algorithms.

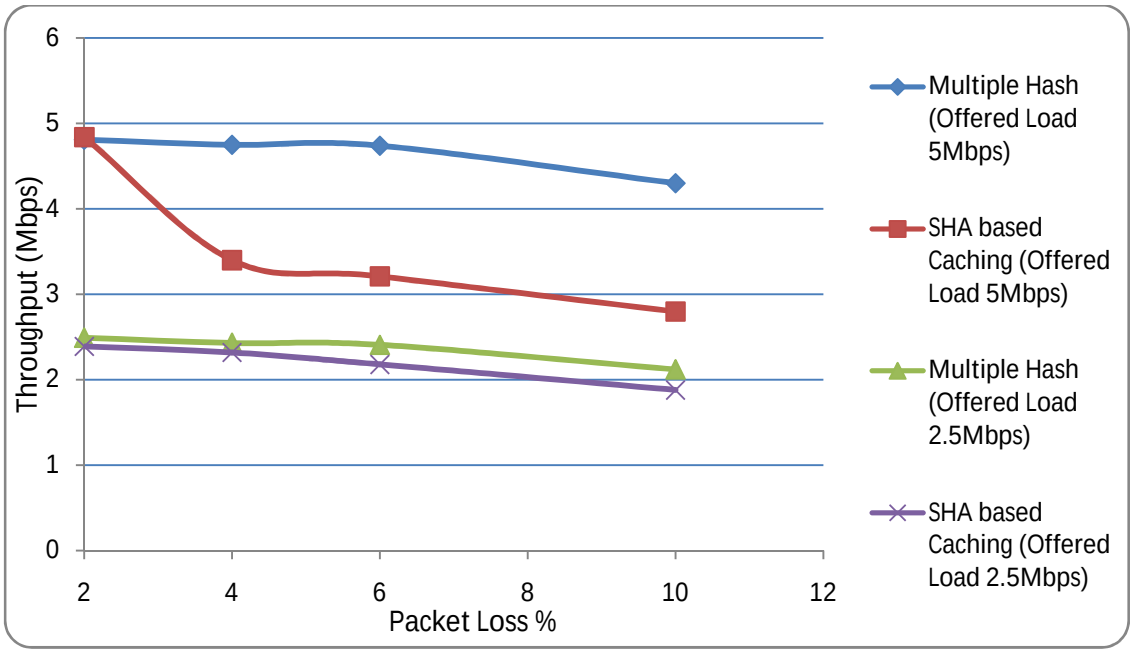

Figure 4-13 Throughput comparison under multiple hash technique

We can see that at higher offered load throughput drops comparably while using baseline hash technique. Multiple hash technique using SHA1, SHA2, MD5 provide 3 different options for selecting a cache location. The decision is then made based on ETX for the three locations. Using multiple hash technique, we get significant gains when the network contains lossy links.

5 CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

In this work, we proposed and evaluated the content name resolution service (CNRS) protocol. We implemented CNRS protocol integrated with HOP and OLSR on the ORBIT testbed. Locally resolving the content id to its attributes like location, popularity using hashing provides a better alternative that will scale well as compared to having a central look up service. It was observed via experimentation that for a variety of wireless mesh scenarios, our protocol gives significant gains in terms of system throughput. Our experiments also showed that if knowledge of content popularity or content request rate is used for caching content, then we get reduced file retrieval latency. This was cleary observable while using gaussian distribution for client requests as opposed to using equiprobable distribution. Experimentation results show that as the cache location distance increases from end user, the average file retrieval latency increases and hence we suggest that using Jordan center set idealogy for caching conent in larger networks will improve performance. Under lossy link conditions, with different traffic patterns, throughput provided by multiple hash technique were significantly higher than the baseline algorithm as it helps us avoid lossy link paths while caching and retrieving content.

5.2 Future Work

A feature complete CNF stack needs to be compared with the TCP/IP stack for a complete performance evaluation. CNRS scalability should be tested in larger networks and even the possiblity of having multiple CNRS servers within each AS. Control messages for inter AS CNRS communication has been discussed in this thesis. An implementation of that should be conducted to test exchange of summary cache information between each CNRS. CNRS should also be tested for networks having mobile nodes. This can be done on ORBIT using virtual mobility through spatial switching [(Kishore Ramachandran et al)]. Different content request distributions should be used to test the performance of the protocols. Experiments should be done on PlanetLab to test CNF in real life scenarios rather than emulating lossy links and disconnections. CNF should also be tested with WiMAX such that content transport switches through a WiMAX node when a node is not able to connect to an access point.

6 References

1. The cache-and-forward network architecture for efficient mobile content delivery services in the Future Internet. S. Paul, R. Yates, D. Raychaudhuri and J. Kurose. May 2008. First ITU-T Kaleidoscope Academic Conference on Innovations in NGN: Future Network and Services.

2. Design of Link and Routing Protocols for Cache-and-Forward. Shweta Jain, Ayesha Saleem, Hongbo Liu, Yanyong Zhang, Dipankar Raychaudhuri. 2009. Sarnoff Symposium.

3. Saleem, A. Master Thesis- PERFORMANCE EVALUATION OF THE CACHE AND FORWARD LINK LAYER PROTOCOL IN MULTIHOP WIRELESS SUBNETWORKS. Aug 2008.

4. T. Clausen, P. Jacquet. Optimized Link State Routing Protocol (OLSR). Oct 2003. RFC 3626.

5. OLSR. http://www.olsr.org/docs/README-Link-Quality.html. [Online]

6. Block-switched Networks: A New Paradigm for Wireless Transport. Ming Li, Devesh Agrawal, Deepak Ganesan, and Arun Venkataramani. Boston : s.n., Apr 2009. 6th ACM/USENIX Symposium on Networked Systems Design and Implementation (NSDI 2009).

7. The Cache and Forward Network Architecture:Implementation on the ORBIT testbed. Shweta Jain, Sneha Gopinath, Shivesh Makharia , Dipankar Raychaudhuri. 2009. Under Submission.

8. Performance Evaluation of In-network Integrated Caching. Lijun Dong, Yanyong Zhang, Dipankar Raychaudhuri and Sanjoy Paul. under submission.

9. “On the Cache-and-Forward Network Architecture” . Lijun Dong, Hongbo Liu, Yanyong Zhang, Sanjoy Paul and Dipankar Raychaudhuri. 2009, ICC.

10. Routing in CNF. Jain, Shweta. Under Submission.

11. Performance Evaluation of the ”Cache-and-Forward (CNF)” Network for Mobile Content Delivery Services. Hongbo Liu, Yanyong Zhang, and Dipankar Raychaudhuri. 2009. ICC.

12. A delay tolerant network architecture for challenged internets. Fall, Kevin. 2003. Proceedings of SIGCOMM.

13. Routing Metrics and Protocols for Wireless Mesh Networks. Campista, Miguel Elias M. et al. Brazil

14. Spectrum MRI: Towards Diagnosis of multi radio interference in the unlicensed bad. Akash Baid, Suhas Mathur, Ivan Seskar, Tripti Singh, Dipankar Raychaudhuri, Sanjoy Paul, Amitabha Das. Under submission. Dyspan.

15. A.Vahdat, D.Becker. Epidemic Routing in Partially-Connected Ad Hoc Networks. s.l. : Duke University Technical Report CS-200006, April 2000.

16. A. Lindgren, A.Doria, O. Schelen. Probabilistic Routing in Intermittently Connected Networks. 2004. Lecture Notes in Computer Science, Vol.3126/2004.

17. A. Anand, A. Gupta, A. Akella, S. Seshan and S. Shenker. 2008. Proceedings of the ACM SIGCOMM conference on Data communication.

18. Orbit Testbed. [Online] www.orbit-lab.org.

19. Olsrd, An adhoc wireless mesh routing daemon. [Online] www.olsr.org.

20. A.Wright. Mobile Phones Could Soon Rival the PC As Worlds Dominant Internet. April 2006.

21. Spectrum MRI: Towards Diagnosis of multi radio interference in the unlicensed bad. Akash Baid, Suhas Mathur, Ivan Seskar, Tripti Singh, Dipankar Raychaudhuri, Sanjoy Paul, Amitabha Das. Under submission

22 A delay tolerant network architecture for challenged internets. Fall, Kevin. 2003. Proceedings of SIGCOMM.

23. T. Clausen, P. Jacquet. Optimized Link State Routing Protocol (OLSR). Oct 2003. RFC 3626.

7 Appendix A - CNRS Controls bits

Control bits format for CNRS packets: