# Content Centric Mechanisms for Efficient Data Dissemination in Delay Tolerant Networks

Bilal Shaukat Gill

**School of Electrical Engineering**

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo 10.11.2015

**Thesis supervisor:**

Prof. Jukka Manner

**Thesis advisor:**

Dr. Jan Seedorf

Aalto University
School of Electrical
Engineering

AALTO UNIVERSITY
SCHOOL OF ELECTRICAL ENGINEERING

ABSTRACT OF THE
MASTER'S THESIS

Author: Bilal Shaukat Gill

Title: Content Centric Mechanisms for Efficient Data Dissemination in Delay Tolerant Networks

Date: 10.11.2015 Language: English Number of pages: 10+61

Department of Communications and Networking
Professorship: Networking Technology Code: S-92

Supervisor: Prof. Jukka Manner

Advisor: Dr. Jan Seedorf

Today's internet was founded as a host centric abstraction for connecting machines over a geographically distributed data base. Since then it has exploded into a trillion dollar industry for providing services and content to the world. For meeting these ever growing consumer demands, internet service providers have used bolt-on approaches to patch the internet. On the other hand, the last decade has witnessed the worst natural disasters on earth which resulted in total or partial destruction of communication infrastructure.

Understanding these challenges, researchers are committed to re-architect the internet with clean slate information centric approaches. These future internet architectures have shifted the dynamics from predominately location oriented models to data oriented models. These models provide location independence which eases the network configuration and implementation of network services in mobile environments.

In this perspective, this thesis aims to hack content centric abstraction to provide optimized solutions for delay tolerant network scenarios. We provide information aware mechanisms which help to take adequate forwarding and caching decisions in these dynamic and challenged environments.

This thesis proposes a unique popularity estimation algorithm and a name based prioritization algorithm for disseminating data more productively in intermittently connected networks. For evaluation it analyses the performance for both mechanisms and compare them with the latest solutions.

Furthermore the thesis discusses potential research areas in the field of information centric networking and future directions for this thesis.

Keywords: information centric networking,content centric networking, delay tolerant networking

# Preface

I would like to thank my advisor Jan Seedorf, for the opportunity to work on this thesis and for his valuable insight during the thesis. His guidance, suggestions, broad experience, depth over the subject and profound knowledge was crucial for realization of this thesis.

I would also like to appreciate the efforts of Dirk Kutscher and Mayutan Arumaithurai, who had a great role in identifying the thesis topic and defining the scope of this thesis. Their experience and authority over the subject was commendable and inspiring.

I would like to express my gratitude to my supervisor Professor Jukka Manner for his valuable input and guidance during my work with this Master's thesis.

Finally, I would like to thank my parents for their endless support and encouragement during all these years. Their love, support, prayers and kindness has always been a great source of strength.

The work was supported by the European Commission's Seventh Framework Program (FP7/2007-2013) under grant agreement number 608518 and NICT under Contract number 167.

Otaniemi, 10.11.2015

Bilal Gill

# Contents

**Abstract** ii

**Preface** iii

**Symbols and Operators** ix

**Acronyms** x

**1 Introduction** 1
- 1.1 Problem Statement . . . 1
- 1.2 Scope & Contribution . . . 2
- 1.3 Structure . . . 2

**2 Background** 3
- 2.1 Information Centric Networking . . . 3
  - 2.1.1 Naming in ICN . . . 3
  - 2.1.2 Name Resolution and Data Routing in ICN . . . 5
  - 2.1.3 Caching in ICN . . . 5
  - 2.1.4 Mobility in ICN . . . 6
  - 2.1.5 Security in ICN . . . 7
- 2.2 Content Centric Networking . . . 8
  - 2.2.1 CCN Node Model . . . 9
- 2.3 Delay/Disruption Tolerant Networking . . . 12
  - 2.3.1 DTN Architecture . . . 13
- 2.4 Bundle vs CCN Protocol for Challenged Networks . . . 17
- 2.5 Conclusion . . . 18

**3 Solution** 20
- 3.1 Motivation . . . 20
  - 3.1.1 Scenario & Assumptions . . . 20
  - 3.1.2 Gateway/Shelter Routers . . . 22
  - 3.1.3 Policies for Data Mules . . . 22
- 3.2 Name Based Prioritization Protocol(NBP) . . . 23
  - 3.2.1 Sailent Features . . . 23
  - 3.2.2 Protocol Overview . . . 23
  - 3.2.3 NBP Protocol API . . . 25
- 3.3 COPA: Content Optimal Popularity Estimation Algorithm . . . 26
  - 3.3.1 Goals . . . 27
  - 3.3.2 Filtering Effect . . . 27
  - 3.3.3 Reference Algorithm . . . 28
  - 3.3.4 Core Algorithm . . . 31
  - 3.3.5 COPA-CCN Integration . . . 36
  - 3.3.6 Analytical Model for Memory Estimation of Nonces . . . 38
- 3.4 Conclusion . . . 39

**4 Evaluation** **41**
4.1 Evaluation Scenario . . . 41
4.2 Implementation . . . 42
4.2.1 Generating Interests wrt Zipf Distribution . . . 42
4.3 Testing COPA & Reference Algorithm . . . 42
4.3.1 Reference Algorithm . . . 43
4.3.2 COPA . . . 43
4.3.3 Analysis . . . 44
4.4 Testing NBP Protocol . . . 46
4.4.1 Initial coordination in TCP . . . 46
4.4.2 Utilities in CCNx . . . 47
4.4.3 Modification of CCNx . . . 48
4.4.4 Test Setup . . . 48
4.5 Conclusion . . . 49

**5 Conclusion & Future Work** **51**
5.1 Conclusion . . . 51
5.2 Future Work . . . 52
5.2.1 Permanent Storage in CCN . . . 52
5.2.2 Acknowledgements and PIT Size . . . 52
5.2.3 Data Discovery in CCN . . . 52
5.2.4 Services over CCN . . . 53
5.2.5 Data Advertisements by Intermediary Routers in CCN . . . 53
5.2.6 Exploiting Multiple Interfaces in CCN . . . 53
5.2.7 Information Centric Mechanisms for Traditional Internet . . . 54

**A Appendix** **59**

# List of Figures


1 Communication using self-certifying names in a content oriented architecture . . . 4
2 Intrinsic and extrinsic binding in self-certifying and human readable names. Both naming schemes provide built in binding (solid lines) and need an external authority to provide the additional bindings (dotted lines) [12] . . . 4
3 An example of hierarchical names. . . . 5
4 Mobile IP [15] . . . 7
5 Protocol Stack CCN vs TCP/IP [7] . . . 8
6 Interest Packet (Left) and Data Packet (Right) [2] . . . 8
7 Interest packet processing in a CCN node [7] . . . 9
8 Request forwarding at a CCN node [7] . . . 10
9 Prefix Announcements in a CCN network . . . 11
10 Interest and Data forwarding in a CCN network . . . 12
11 Different Scenarios in DTN [17] . . . 13
12 DTNRG Architecture . . . 14
13 Node interaction in epidemic routing protocol, when two nodes come inside each other's radio range .Nodes exchange data which is not present in their own buffer memory [25] . . . 15
14 Spraying Phase (Source Spray) & Waiting Phase . . . 16
15 Performance vs Knowledge Trade-off [22] . . . 17
16 Disrupted Telecommunication Network after the Disaster. . . . 21
17 Name based priority exchange protocol from DMy to DMx. . . . 24
18 Interaction between end user and the data mule.. . . . 25
19 System Model/ Prioritize exchange Algorithm API. . . . 25
20 Aggregation in pending interest table . . . 28
21 Reference Algorithm. . . . 29
22 Popularity Estimation using Reference Algorithm . . . 29
23 Loop detection in data mule network . . . 30
24 Reference Algorithm saves every distinct nonce . . . 30
25 Overestimation of Popularity (/prefix nonce:popularitycount) . . . 33
26 Two end users request the same name from DMx . . . 34
27 End users are assigned a new nonce with count 2 for the same prefix 35
28 DMa receives duplicate query(loop) from the DMz, so drops the message (loop detection). . . . 35
29 Two new users request the same prefix from DMz . . . 36
30 End users and node DMz updates the popularity count for prefix /parc.com/art . . . 36
31 Flow Diagram of COPA for an incoming interest in a CCN data mule 38
32 Test bed scenario for COPA and Reference Algorithm, where red circles represent end-users . . . 41
33 Zipf curve generated for different alpha parameters . . . 42
34 Accuracy of the Reference Algorithm vs Initially Poured Interests . . 43

35 Show the accuracy of the COPA vs Initially Poured Interests . . . . 44
36 Memory Utilization Comparison between COPA and Reference Algorithm . . . . . 45
37 Encounter List of DMx with ID: 2500, during the previous experiment 45
38 COPA (Encounter list full) vs Reference Algorithm . . . . . 46
39 Problems with CCN protocol for initial coordination in NBP . . . . 47
40 NBP protocol vs Modified CCN protocol . . . . . 49
A1 Interest Packet Wireshark . . . . . 59
A2 Periodic Updates about the Neighbours . . . . . 59
A3 Snapshot from Wireshark, with CCN dissector . . . . . 60
A4 CCNDSTATUS utility . . . . . 60

# List of Tables

1 Comparison between CCN & DTN protocol [28] . . . . . 18
2 Prefixes with global priorities and corresponding information [29] . . 22
3 DET for saving acknowledgements. . . . . 52

# Symbols and Operators

| | |
|---|---|
| M | Total memory occupied by the nonces for each prefix in PIT |
| K | Total number of data mules in the network |
| m[Nji] | Memory occupied by a particular nonce of a distinct prefix in a data centric node |
| Pi | A distinct prefix in PIT |
| pi | popularity count of ith prefix |

# Acronyms

| | |
|---|---|
| ADU | Application Data Unit |
| BPQ | Bundle Protocol Query |
| CCN | Content Centric Networking |
| COPA | Content Optimal Popularity Estimation |
| CS | Content Store |
| DNS | Domain Name Server |
| DONA | A Data Oriented ( Beyond) Network Architecture |
| DPI | Deep Packet Inspection |
| DOS | Denial of Service |
| DTN | Delay Tolerant Networking |
| FIB | Forwarding Information Base |
| FTP | File Transport Protocol |
| FIFO | First In First Out |
| HTTP | Hypertext Transfer Protocol |
| IP | Internet Protocol |
| ICN | Information Centric Networking |
| LFU | Least Frequently Used |
| LRU | Least Recently Used |
| Mule | Mobile Ubiquitous LAN Extension |
| NDO | Name Data Object |
| NDN | Name Data Networking |
| NETINF | Network of Information |
| PIT | Pending Interest Table |
| PGP | Pretty Good Privacy |
| PKI | Public Key Infrastructure |
| RWI | Real World Identity |
| RTT | Round Trip Time |
| RFC | Request for Comments |
| SDN | Software Defined Networking |
| TCP | Transmission Control Protocol |
| TTL | Time to Live |
| URI | Uniform Resource Identifier |
| WOT | Web of Trust |
| XIA | Expressive Internet Architecture |
| DM-DM | Data Mule to Data Mule |

# 1 Introduction

Internet architecture was designed in 70's to provide an ability to connect hosts with resources that were located on remote hosts. This socket abstraction used IP addresses for identification which enabled different applications like FTP, telnet etc. These IP addresses were interfaces on different hosts normally on a fixed location. As a result we have a host centric abstraction, which focuses on forwarding packets among different stationary end host machines. Since then internet has evolved over the years from a communication network between different academic hosts to content distribution network with over 1 billion connected machines. This evolution has changed the dynamics of the current internet which need a lot of patches to meet the demand of ever growing data.
Current internet face some of the toughest challenges in the vigil of strong technical advancements and worst natural disasters in the history of mankind. Some of the new solutions proposed by the research community use infrastructure-based end to end mechanisms where information is stored in a centralized server. However relying on infrastructure based architectures and end to end transport protocols in the case of delay tolerant network scenario has proven to be an ultimate failure. To overcome these challenges, researchers have proposed a number of new *future proof* architectures and protocols for providing better support specifically for mobility and security.
Information Centric Networking(ICN) has emerged as the most promising data centric approach to provide the best results in both well connected and delay tolerant networks. This approach decouples the location from the source and provides efficient support for data dissemination in stressful environments where disruptions are frequent.

## 1.1 Problem Statement

Recently data centric approaches have been considered to provide solutions for the Delay Tolerant Networing(DTN) scenarios [1] [2]. Several information centric functionalities like name based routing and in network caching provide essential help to tackle the problems in challenged networks.
We particularly consider how content centric properties can be utilized to efficiently disseminate data in intermittently connected networks[1]. In this perspective we have designed and implemented two separate algorithms to provide support in DTN scenarios.
First algorithm tackles the problem of prioritizing data exchange between nodes in a DTN scenario using data centric functionalities of Content Centric Networking(CCN) abstraction. Second algorithm called Content Optimal Estimation Algorithm (COPA), estimates the popularity of a content in disruption tolerant networks. This popularity metric helps to make efficient forwarding and storing decisions in the intermittently

[1]Delay tolerant networks, intermittently connected networks, challenged networks and opportunistic networks are interchangeably used in this thesis.There is a very small difference between them, which is explained in Section 3.3.

connected network. To best of our knowledge, COPA is the first unique algorithm which estimates the popularity of a content in challenged networks with the help of data mules.

## 1.2 Scope & Contribution

This Master thesis is an extension to the research carried out in the *GREEN ICN* [3] project which is a consortium of European and Japanese partners. This project particularly considers to exploit the information centric protocols to provide support for scalable efficient data delivery in the after-math of a disaster scenario.
This thesis is aimed at improvizing the communication protocols to the next level with the help of data centric functionalities in delay tolerant networks. It studies the CCN abstraction intensively to find out the potential hacks for best effort delivery in challenged networks.
This research work contributes two unique popularity estimation algorithms for delay tolerant wireless networks. Results show that, the mechanisms aggregate the popularity quite accurately with decent number of encounters between all the data mules and end users in the network. Both of these popularity mechanisms provide different results in terms of accuracy, memory efficiency and network congestion.
CCN abstraction does not exploits the information centricity to its maximum extent in intermittently connected networks. This thesis also provides a content based prioritization protocol built on top of CCN abstraction. This protocol manipulates the three data structures of CCN layer to withstand long delays and forward interest packets to every encountering node. These content centric mechanisms mentioned above help to disseminate priority data to more nodes with in a delay tolerant environment.

## 1.3 Structure

The report is divided into following chapters.
Chapter 2 gives a detailed introduction about different aspects of ICN and DTN. It explains the CCN protocol in depth for the understanding of forthcoming chapters. Furthermore it analyses the differences and similarities of Bundle and CCN protocol.
Chapter 3 describes the scenario and the solution for efficient dissemination of data in delay tolerant networks. It explains the design of Name Based Prioritization(NBP) Protocol and COPA which helps to make appropriate caching and forwarding decisions during the content distribution in challenged networks.
Chapter 4 evaluates the above mentioned algorithms and demonstrates their effectiveness in challenged networks. Chapter 5 concludes the whole research work. It also provides future directions to this thesis and ICN in general.

# 2 Background

This chapter will preview the background knowledge on Information Centric Networks, Delay Tolerant Networks and CCN protocol. Considerable amount of research has been done on DTN and ICN in the last decade. The chapter also briefly goes through the basics of each of these technologies to understand the upcoming sections of this thesis.

## 2.1 Information Centric Networking

The short comings of current internet has pushed the researchers to find new clean slate solutions to their respective problems. There have been no substantial changes in the layer 3 and layer 4 of the current internet protocol stack in the last decade to accommodate the evolution of new services and applications. There are various future internet architectures proposed and studied previously, but ICN has emerged as the most promising contender.
The key features of ICN architectures are name based routing, built in multicast support, in network caching and location independent naming which provide scalability and robustness to the network. We will discuss, how some of the features in the following subsections help to fulfil the new demands of emerging data-intensive applications in the world.
There are lot future internet architectures [5] [2] [6] [7] [8] [9] [10] [11] that define information centricity as the core principal of their architecture. Most of these approaches differ mainly in routing and naming mechanisms.

### 2.1.1 Naming in ICN

Current traditional internet uses a lot of unoptimized solutions to provide steady connectivity to the end users. These patchy solutions are built around a thin IP layer, whose sole task originally was to transfer packets from the source to the destination. This location based protocol comes with a lot of implications, like DOS, flooding attacks. To overcome these security problems and to adapt with the needs of new applications, researchers have devised a new naming abstraction.
In ICN, naming the data decouples the resources from the location and helps to make inexpensive optimizations to the network as compared to deep packet inspection in the current internet. This location independence eases the network configuration and implementation of network services.
The key principal of an ICN architecture is to name each piece of data. The named data is matched, processed and forwarded through each of the routers in the network. Names can be understood as transport layer endpoints which fetch content from the most suitable providers in the network to reduce latency.
The naming in ICN can be categorized into flat and hierarchical names. Hierarchical names are human friendly names, which are easy to read and aggregate. On the other hand flat names or self-certifying names provide more security and authenticity [12].

**Self-Certifying Names** Self-certifying names are also called flat names. The benefit of using this naming structure is that any entity in the network can verify the binding between the public key and self-certifying name without the help of any outside authority [12]. This useful property proves the provenance of the data item, that whoever has the possession of the private key indeed published the data.
In Figure 1, we describe a simple communication between an end user and the network using self-certifying names. The name consists of a principal which is cryptographic hash of the public key and a globally unique label provided by a certain organisation (P: L). Once this name is queried, the network replies with a data item which includes the data, public key, label and signature etc. The origin of the data can be verified by creating the hash of received public key in the data item and matching it with P. The signature can also be checked that it was indeed signed by the corresponding private key.

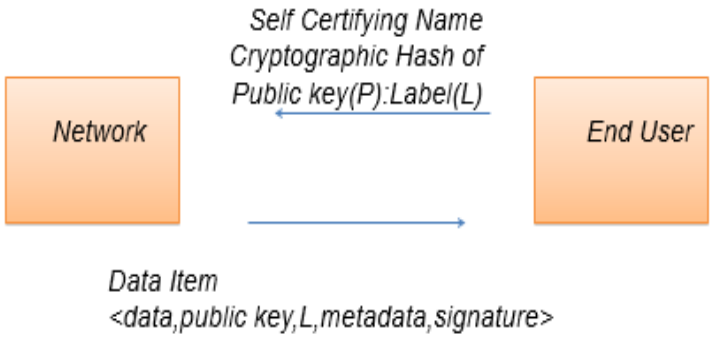


Figure 1: Communication using self-certifying names in a content oriented architecture

As shown in Figure 2, self-certifying names provide a binding between the Real World Identity (RWI) which is the principal of the data and the public key. It needs an external authority to provide the additional binding between the RWI and the names. This additional binding can be provided by some central authority like PKI or distributed network like Web of Trust(WOT).

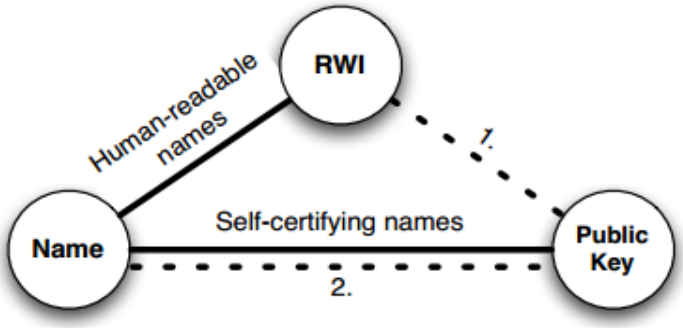


Figure 2: Intrinsic and extrinsic binding in self-certifying and human readable names. Both naming schemes provide built in binding (solid lines) and need an external authority to provide the additional bindings (dotted lines) [12]

**Hierarchical Names** Hierarchical names are human readable names and are more scalable as compared to flat names. They can be easily aggregated, as shown in CCN [2] in routing tables. They follow the same kind of hierarchy for scalability as DNS does for name resolution and the IP layer for prefix aggregation in the current internet architecture. Figure 3 shows a hierarchical name which is almost same as a URL.


Figure 3: An example of hierarchical names.

An essential part of NDN[2] project [7] which is further developing CCN, is that name resolution and name based routing for this naming structure can be aggregated across similar names. Human readable names provide a weak intrinsic binding between the name and the RWI [12]. The binding between the public key and the name is provided by the external authority like Public Key Infrastructure(PKI).

### 2.1.2 Name Resolution and Data Routing in ICN

The process of resolving a route to the publisher from where one can retrieve the data is called name resolution. While data routing involves building a path towards the end host to transfer the content. Both of these functions either are integrated [2][7] or independent [10][9] of each other in different ICN architectures. Some of these architectures provide both coupled and decoupled approaches [3].
Coupled or Integrated Approach: In this approach name resolution and data routing are coupled together. The data follows the same reverse path over which the request was sent. NDN [7] and CCN[2] architectures use the coupled approach as seen in Figure 10.
Decoupled or Independent Approach: There is a separate routing module or a different mechanism, which provides a path to send data to the end user. For example DONA [5] uses direct IP forwarding and routing from the publisher to the subscriber to route the data. However in PSIRP project [9], name resolution processing nodes contact the topology manager(TM) to create a route from the requester to the publisher. Finally TM send this route to the publisher which uses it for data delivery.

### 2.1.3 Caching in ICN

In traditional internet, specific mechanisms and protocols are deployed to provide address lookup for domain names. There could be multiple queries for finding a server for a piece of data, which increases latency. These multiple requests could also congest the network, in order to find the location of the content.

[2]Basic data structures and operation of NDN and CCN abstractions are same, except some fields.

On the other hand content distribution networks (CDNs) work on top of the transport layer providing services to the top content owners. These overlay networks employ network unaware mechanisms which underutilize the resources of the network.
The above mentioned challenges can be solved by using the data centric approach which is to place a cache in each of the routers . For each host query the content is fetched from the network and fed to the host. The intermediate routers which come in the path of data exchange make content replicas in their storage. So each request does not have to make a long round trip time, but retrieves the data from the closest possible place. Different kind of caching mechanisms are used to store the most popular items being queried from downstream routers. The cache replacement policies (e.g LRU, LFU) can be selected depending upon the behaviour of the network to overcome the problems like flash crowds and denial of service attacks.
The smart replication of data into different nodes and information awareness helps to manage and engineer the traffic in a much better way. According to [13], in network caching provides two major benefits: low response time and simplified traffic engineering. There are two kinds of approaches in caching used in data centric networks: off-path caching and on-path caching. In off-path caching data is replicated among different servers without knowing the forwarding path of different requesters. This kind of caching need to solve the optimization problem of cache locations. Off path caching must be supported by a name resolution system to provide the data from the nearest server as mentioned in Section 2.1.3.
On the other hand on-path caching mechanisms, the name resolution request exploits the intermediate routers looking up for data. If data is found, the data is routed on the same reverse path to the end host machine.

### 2.1.4 Mobility in ICN

According to the Cisco Visual Networking Index (VNI) report [14], global internet mobile traffic grew 69 percent and half a billion of mobile devices were added into the network in 2014.The emergence of smartphones and interesting applications have certainly encouraged users to buy cheap internet packages for their mobile devices. This certain boom towards mobile traffic has pushed the mobile operators to make patchy solutions to provide steady connectivity. Today the commonly used solution for mobile communication in current internet is Mobile IP.
Mobile IP is designed to maintain the TCP connection between the mobile end users and the static server as shown in Figure 4. This maintenance comes with an overhead in shape of a proxy agent which exchanges the packets between the mobile host and the server.

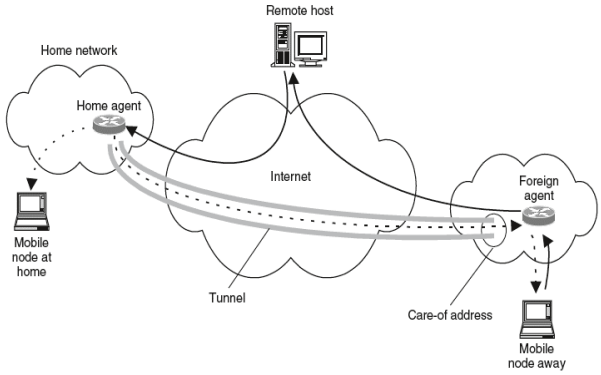


Figure 4: Mobile IP [15]

The information centric approach to handle mobility is to have publish /subscribe communication model. In this approach, end users send their queries into the network and also advertise the data items they withhold in their storage. Particularly in ICN, publishers do not have to answer requests directly from the users, since the intermediate nodes can answer the queries depending upon the content items in their buffer.
These features provide flexibility to the end users to move anywhere in the network and reissue the requests to the network. In response network replies with an answer from the nearby caches of routers.

### 2.1.5 Security in ICN

Internet was designed in 70's and 80's when security was not considered as a first class citizen. During that time the main concern for internet was to be robust and fault tolerant. In recent decades internet has been bullied by attackers, spammers and hackers in various ways due to its lack of security. The inability of current nodes to look inside the packet cost the security dearly.
All ICN architectures propose confidentiality and integrity for each piece of data item rather than wasting resources on securing the channel. These receiver driven architectures also prevent undesirable data transfers from the publishers. The decoupling of the resource from the location provide significant benefits to achieve privacy, as publishers and subscribers may not be aware of each other identities.
Security in data centric architecture is directly integrated with naming. Hierarchical names have intrinsic binding between the name and the real world identity, but need a third world party such as PKI to provide extrinsic binding between the key and name. They can be easily aggregated in the hierarchy of routers to provide scalability.
On the other hand, self-certifying names provide intrinsic binding between key and the name by combining the cryptographic hash of public key P and a label [12].Once name is fetched by the receiver, it can always check that the public key hashed to P. But this self-certification comes on the expense of scalability, aggregation and translation between flat names to human readable names.

## 2.2 Content Centric Networking

The word CCN was first coined in a google talk [16] in 2006 by Jacobson, where he argued that named data networking is better fit to the current needs of internet consumers. The request and response protocol used in the traditional internet architecture is too much complex since it was designed for a different applications and services based on host to host communication.
Name data networking replaces the universal IP packets with content chunks in the network stack as shown in Figure 5. CCN protocol has two different layers security and strategy incorporated in its protocol stack as compared to IP. It can benefit from multiple interfaces depending upon the connectivity (3G, WIFI, and Bluetooth) due to its relationship with strategy layer. A receiver strategy layer is responsible for selecting the best interface to forward the interest packets and deciding the priority of packets as well.

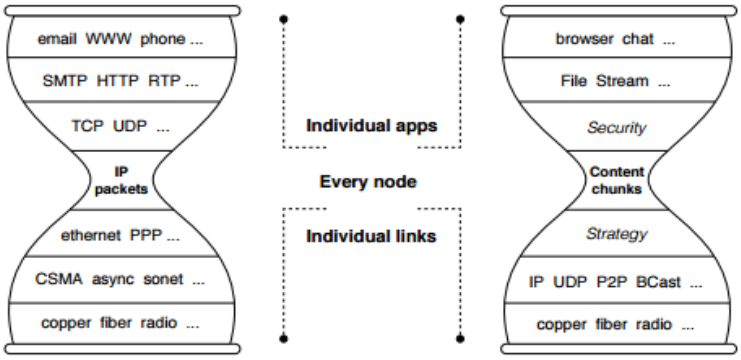


Figure 5: Protocol Stack CCN vs TCP/IP [7]

CCN names are hierarchical and can be aggregated at various intermediate routers. Name resolution and data routing are integrated in CCN, meaning that the data routes from the same reverse path where the request is routed to an information provider. CCN benefits from on path caching, where the network exploits the data inside the buffers of each intermediate router.

| Name |
| --- |
| Selectors(filters) |
| Nonce |
| Guiders(scope, Interest Lifetime) |

| Name |
| --- |
| Meta-Info(Content type, freshness period) |
| Content |
| Signature (signature type, key locator) |

Figure 6: Interest Packet (Left) and Data Packet (Right) [2]

There are two type of packets used for communication inside a CCN network, an interest packet and a data packet as shown in Figure 6. Both packets have names

which are matched in each of the intermediary routers to find the next hop. An end user sends an interest packet into the network towards the data producer, in response a data packet is sent on the reverse path to the end user. CCN data objects are immutable, which are verifiable against the cryptographic hash.
Each interest packet has a field called *interest lifetime* field which is used to time out the interest packet. This field should be chosen wisely depending upon the nature of the network and topology.
The interest packet also contains a *nonce* field, which is used to detect loops inside a CCN network. It is a randomly generated byte string which helps to discard duplicate packets which come through different paths to a node inside the network.

### 2.2.1 CCN Node Model

There are three basic data structures inside a CCN node to perform all the interest and data forwarding. These data structures add the capability of multicasting and caching inside a CCN router.
Content Store (CS) is a buffer memory same as in IP buffer, except that it contains data corresponding to names rather than flows. The content store follows Least Recently Used(LRU) or Least Frequently Used(LFU) techniques to cope with the memory restrictions. Any data which passes through the CCN router gets saved in the buffer memory.
Pending Interest table (PIT) store all the pending queries from the downstream routers to be served. It does not maintain flows, rather save interfaces from the downstream router to communicate in a hop by hop manner. So these tables are trails for each request which are then used to route data packets from the provider to the consumer. Once the query is satisfied, the corresponding interest table entry is also deleted.
Forwarding Information Base (FIB) is used to save the next hops towards the data providers. Data providers float prefix announcements into the network about the resources they save in their servers. Each router hearing the announcement adds an entry (face) towards the data provider in this data structure.

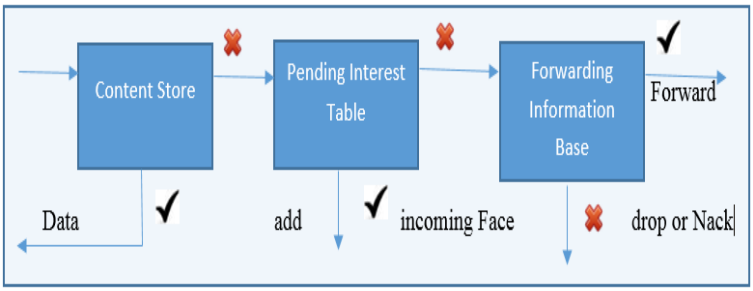


Figure 7: Interest packet processing in a CCN node [7]

When an interest packet arrives at a CCN node, a longest match lookup is done on the name. If there is data against the query in the CS, it is send to the requesting face towards the consumer. If data is not found in the CS, the prefix is matched

with all the entries in the PIT. If a match is found, it means a request is already sent to the upstream router. The requesting face is added in the PIT against the corresponding prefix.
If there is no entry in the PIT, the data is forwarded according to the entries in the FIB to the next hop. If there is no entry in FIB, either the packet is discarded or broadcasted depending upon the rules defined in the strategy layer.The processing of interest packets in a ccn router is shown in Figure 7.
If the router gets to know about the upstream link is not working due to network congestion, it sends a negative acknowledgement(NACK) to a downstream router. Once NACK is received by the downstream router, it can then send the interest towards some other interface from where data can be retrieved.

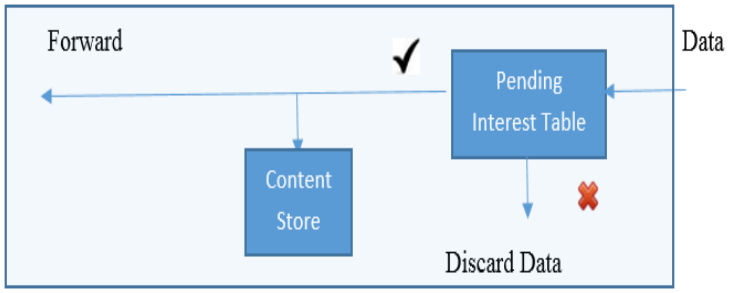


Figure 8: Request forwarding at a CCN node [7]

The data packet moves on the same reverse path, as the interest packet has traversed through. Once the data packet arrives at a CCN node, if a matching PIT entry is found then data is forwarded to the downstream router accordingly. After that the PIT entry is deleted and the data is stored in the CS(buffer memory). If there is no matching PIT entry found, it means the data is unsolicited and discarded. There is only one data packet for each interest packet which is similar as one ack for each data packet in TCP which provides the flow balance. The internal processing of data packet can be seen in Figure 8.
Conventional routing algorithms such as distance vector and link state can be used to announce the prefixes to the neighbour routers. These announcements help in construction of the FIB of each router in the network. Figure 9 shows an IGP network where *publisher 1 & publisher 2* announce the prefix /aalto.fi/news to their adjacent neighbours. These announcements are saved as FIB entries in each of the router with their corresponding faces as can be seen in Fig 10.

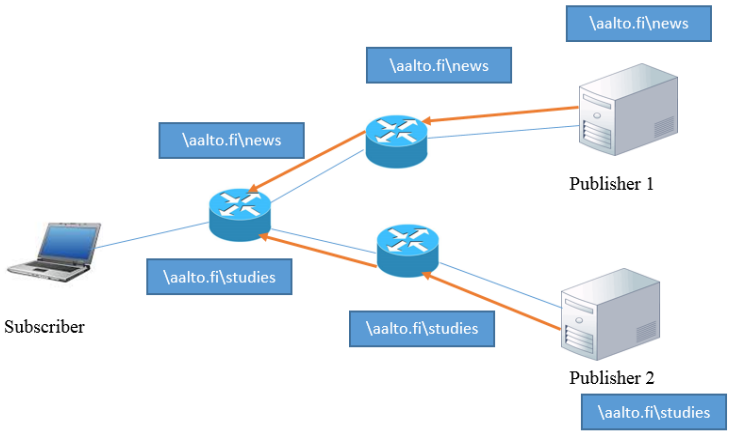


Figure 9: Prefix Announcements in a CCN network

In CCN abstraction packets could be requested and retrieved from an application or other network hardware interface. So the connection between the CCNx [4] core daemon and the application or any hardware interface is termed as face.
Figure 10 shows that, once the subscriber request the prefix /aalto.fi/news, the interest packet traverses from node A to node B and at the end to the publisher 1. This interest packets creates breadcrumbs (PIT entries) in each of the intermediate routers. Publisher 1 responds with a data packet, which follow the same path as the interest packet in the reverse direction. Once the data packet named /aalto.fi/news traverses through the intermediary node, the PIT entry is deleted and data is saved in the CS memory. The interesting point to understand over here is , the next time node $A$ or node $B$ is queried about the same name , they can answer the requesting end using their own buffer memory.

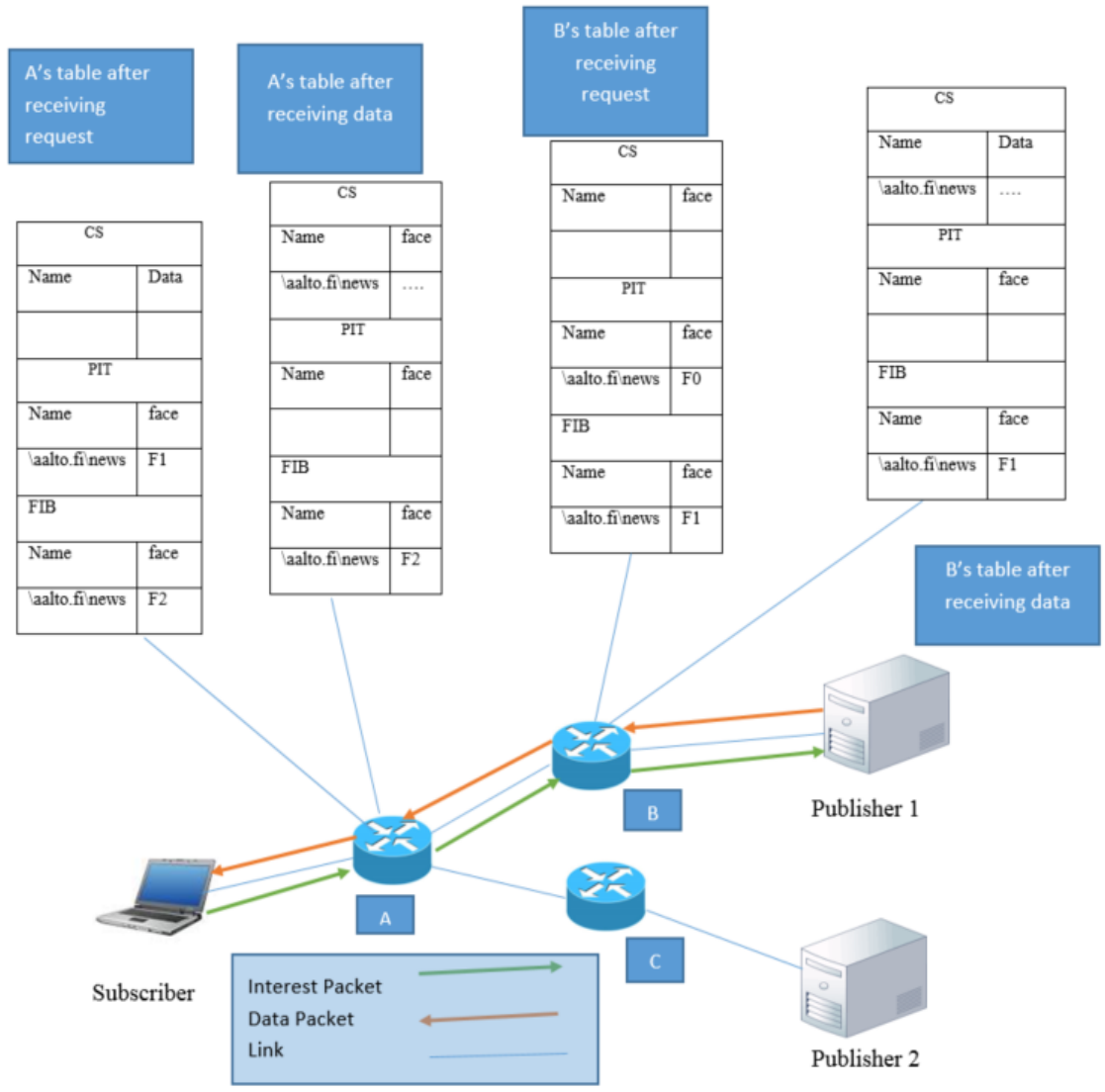


Figure 10: Interest and Data forwarding in a CCN network

## 2.3 Delay/Disruption Tolerant Networking

Disruption tolerant networks [17] are built to withhold long delays and to provide infrastructure for intermittently connected networks. Communicating in these kind of challenged networks is difficult and time consuming due to very long RTT's, since there is no end to end connectivity.
DTN can be a network of data mules in a disaster area which facilitate communication between end users and working base stations. Since there is no permanent link, these networks have to deal with long delays. Due to these implications standard TCP/IP protocol, which assumes an end to end path from the publisher to the subscriber cannot work. So researchers have devised various store and forward protocols[17] which work well in these kind of scenarios.
We studied that intermittently connected networks[18] , delay/disruption tolerant net-

works[17] , challenged networks and opportunistic networks[19] are interchangeably used in research community to defined almost the same kind of the network.There is a very slim difference between all of these specified networks.Delay tolerant networks might have knowledge about timing schedule for connectivity in different applications(e.g space communications), but opportunistic networks have no prior knowledge of future contacts and timing of encounters. Intermittently connected networks are infrastructure-less wireless networks which operate in stress full environments where contemporaneous end to end paths exist for communication, in result of high link disruptions.
There are several applications of DTN in different scenarios as shown in Figure 11 e.g. disaster recovery, environmental monitoring, space communications, underwater acoustic network, wild life tracking sensor networks, military networks and vehicular networks.

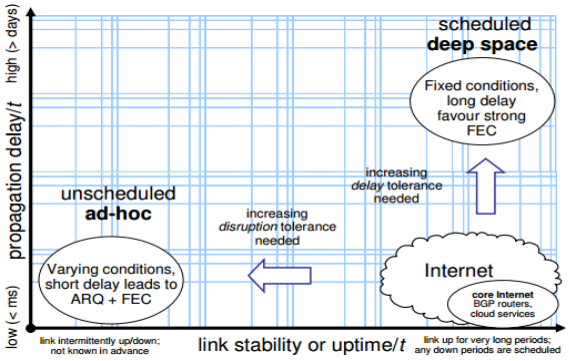


Figure 11: Different Scenarios in DTN [17]

### 2.3.1 DTN Architecture

DTN architecture was proposed by Kevin [20] in 2003, who got some concepts from a NASA project called interplanetary internet [21], which studied communication across the solar system. However author in this paper concentrated on providing an architecture to support communication in similar challenged networks rather than only communication between different planets.

**Bundle Layer** This architecture defines an abstraction layer called Bundle layer which can work on top of multiple convergence layers like TCP, UDP and SCTP as shown in Figure 12. This layer interacts with the convergence layer (underlying layers) to provide enough support to the application layer to run services. Any application built specifically for the Bundle abstraction, sends random length of messages which are called Application Data Units (ADU). Bundle layer can withstand long delays and can carry multiple ADU's.

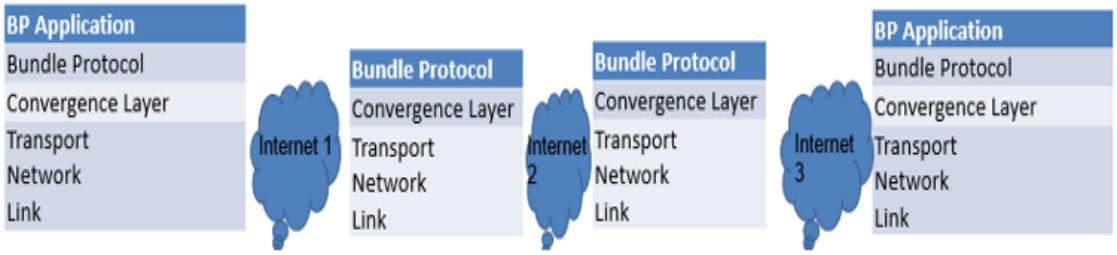


Figure 12: DTNRG Architecture

**End Point Identifier (EID)** Every source and destination node has an end point identifier, so that the middle nodes can transfer the bundles. These are names in DTN, which are similar as uniform resource identifiers (URI) in the traditional network architecture. They are normally used to send multiple ADU's in one bundle from source to destination. EID's could be used for name based routing, to move data towards its destination. The syntax of the EID based URI schemes is given below:
<Scheme name> : <scheme specific part>
An example of the EID could be: dtn://none, dtn://<some opaque string>
dtn://host.domain/some-further-id,http://www.aalto.fi/

**Late Binding** DTN architecture does not compels you to bind the packet with the address as it happens in the domain name system in the current internet architecture. However it supposes that name to address binding can be changed in the source, transit node or the destination node. This late binding specifically helps in challenged network where subscribers and servers are mobile. The decision to change the name to address binding at any transit point decouple the receiver from sender.

**Asynchronous Communication** Challenged networks cannot provide multiple handshakes due to intermittent connectivity, to start communication as done in TCP/IP. So DTN uses asynchronous communication to minimize the conversation between the sender and receiver. For example in today's internet to provide restful services, there are multiple requests and responses from the server and the end user while using HTTP protocol.

**Routing in DTN** Routing in DTN could be very challenging, due a number of issues [22].
(i) End users who request information, do not have any knowledge about the topology of the network. Question arises over here is to whether forward the request to each and every other node (data mule) to maximize delivery chances or selectively forward that request to minimize the congestion.
(ii) Resources in challenged networks are quite scarce. Forwarding each request to everyone can make multiple nodes to save the same data in their buffer memory. Duplication of data in multiple nodes provide redundancy and increase the chances

of quick message delivery. However it also reduce the resources for other messages to be saved in the memory.
(iii) Performance of a routing protocol can be judged based on various metrics like energy efficiency, average latency and communication bandwidth. Energy efficiency is very critical to mobile hosts, since storing and transmitting a message consumes energy.
(iv) Providing security in a DTN environment could be very complex, due to the absence of any central authority to provide security certificates. In this context authenticity of messages cannot be trusted, since it could be routed along some malicious nodes. However there are some cryptographic techniques [23][24], which can provide some guarantees to verify the provenance of the data.
(v) Acknowledgements received for the data by the end host can provide reliability. This could help to free up the buffer space in the intermediate routers upon learning about the successful message delivery.

**Routing Protocols for DTN** There have been different type of routing protocols proposed in the past for DTN like networks, most of them work to maximize the chance of delivery and reduce latency. DTN protocols can be classified into three categories as mentioned below.
1. Flooding (Replication) based protocols
2. Store & Forward (Knowledge based) protocols
3. Coding based protocols

We will discuss some of them to understand the semantics.

**Epidemic Routing Protocol** A flooding protocol proposed by Vahdat[25] for intermittently connected networks in 2000, where a node replicates each message to every other node coming inside the wireless range. Each node maintain a vector list for the data stored in their buffer memory. When two nodes interact with each other, they exchange only the data which is not in common with each other as shown in Figure 13. This protocol concentrates to deliver the message in shortest time by forwarding it to everybody. However it considers that every node has infinite memory and energy.

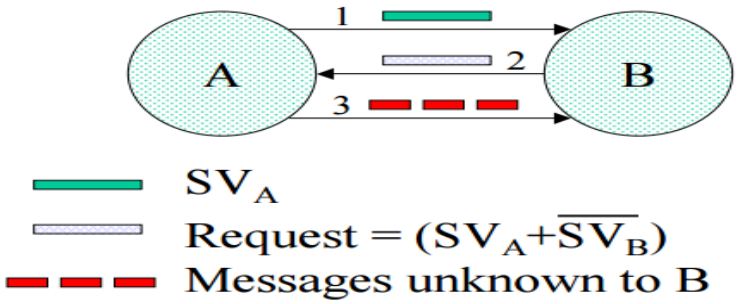


Figure 13: Node interaction in epidemic routing protocol, when two nodes come inside each other's radio range .Nodes exchange data which is not present in their own buffer memory [25]

**SPRAY & WAIT Protocol** Spray & Wait protocol [26] is more improvised version of epidemic routing protocol. It restricts the number of copies replicated inside the network rather than flooding it to every node to save energy and avoid network congestion. This protocol comprises of two phases, spray phase and wait phase.
In spray phase, L number of copies are spread by the source node to L distinct relays as seen in Figure 14. These relays then forward the copies to other nodes depending upon the different spraying methods. (Source and Binary spray)
If the message did not reached the destination in the spraying phase, every node carrying the copy will directly transmit the message to the destination node in this waiting phase.

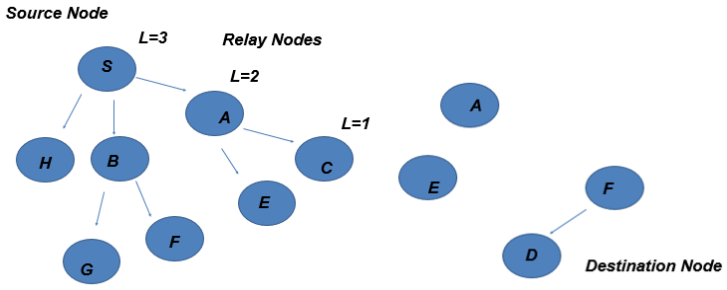


Figure 14: Spraying Phase (Source Spray) & Waiting Phase

**RAPID Protocol** Resource Allocation Protocol for Intentional DTN (RAPID) [27] replicates packets in the network depending upon the utility metric chosen .It considers different routing metrics: minimizing average delay, maximizing maximum delay and minimizing the number of packets replicated to deliver the message. RAPID is indifferent to Epidemic and Spray & Wait protocols, since it considers limited bandwidth and storage in each node.

It has three major components which help to make the decisions for routing in DTN.
(i) Selection algorithm selects which packets to replicate during an encounter with another node.
(ii) Inference algorithm calculates the utility of a message based on the routing metric.
(iii) Control channel sends the metadata to the encountering node.
When node $A$ and $B$ meet each other, both exchange their metadata with each other. If any message is destined toward the encountered node, these are delivered in decreasing order of utility. All other messages if not found in other node, are replicated based on marginal utility calculated. The communication ends when all the messages are replicated or when the nodes move out each other's radio range.

**Knowledge Oracles vs Performance** We discussed different routing protocol in the previous sub headings for DTN like environments. Figure 15 shows an interesting trade-off between performances of routing protocol versus number of knowledge oracles

present. Complete knowledge of these oracles such as contact times, mobility pattern, traffic demand etc , can produce optimal routes and less delay in intermittently connected networks. Limited knowledge of these variables can have a degrading effect on the efficiency of the routing protocol.

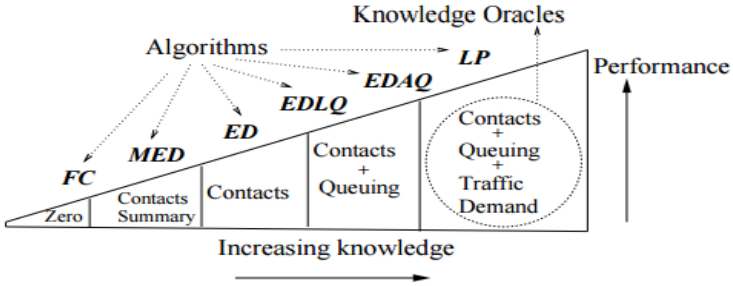


Figure 15: Performance vs Knowledge Trade-off [22]

## 2.4 Bundle vs CCN Protocol for Challenged Networks

In this chapter we described and studied CCN and Bundle protocol in detail. Both of these overlay protocols come from a different back ground and have been researched for almost a decade. CCN protocol was built to prove that data centric architectures can perform best with regards to the application and services usage in today's internet. However Bundle protocol objective is to provide communication support to different heterogeneous networks which have intermittent connectivity, long or variable delay, low bit rates, high error rates and frequent network partitions. Table 1 shows similarities and differences between both protocols, to summarize the concepts studied in this chapter.

DTN architecture uses an asynchronous push based model, where messages are forwarded towards the recipient in the store-carry-forward mechanism. Messages move all the way from source to destination to get the information. However CCN has a request reply mechanism placed into its architecture between each node to support the communication. Recently researchers have proposed an extension for DTN architecture called Bundle Protocol Query (BPQ) into its architecture so that the intermediate nodes can also answer for a query bundle, rather than following an end to end path between source and the destination.

Both of these protocols have an in network storage for different reasons. CCN uses buffer memory to reduce the response time to the queries in a completely connected network. Bundle protocol uses buffer memory to withstand long delays and disruptions to move the message from source to destination in intermittently connected networks.

Messages are forwarded in a hop by hop manner in both of these overlay protocols. Each intermediate node decides where to forward each message depending upon the knowledge oracles, rather than a pre-calculated path. DTN might not be a true hop by hop mechanism since it might not forward message to next node due to network congestion or low power.

Naming the content is the main principal of different ICN architecture. However

DTN architecture names the nodes and the end points. Most of the approaches in both domains support name base routing.

Table 1: Comparison between CCN & DTN protocol [28]

| **Feautre** | **DTN** | **CCN** |
|---|---|---|
| Communication Model | Push Model | Pull Model |
| Query Messages | No but BPQ | Yes(Interest) |
| Storage | Yes | Yes,but dynamic |
| Content | Optional | Yes |
| Asynchronous Communication | Yes(1 Way Links) | No |
| Node Identity | Yes | No |
| Names | URI | Hierarchical Names |
| Fragments | In Transit | Source Node |
| Hop by Hop Routing | Yes (depend upon condition) | Yes |
| Life times | Yes | Yes(interest only) |
| Overlay Protocol(Implementation) | Works on top of TCP,UDP | On top of TCP,UDP |

Both ICN and DTN architecture provide somewhat similar approach to two different scenarios. These abstractions are evolving and becoming more similar to provide communication in wireless networks.

## 2.5 Conclusion

This chapter describes the key properties of an Information Centric Architecture which makes it the most relevant future approach to address the communication problems in wireless networks. Its decision making capability at each hop and ability to respond from cache in a mobile node, helps to make efficient forwarding and caching decisions in a dynamic environment.
DTN environments could be very challenging , due to no information of network topology, resources and mobility of nodes. Because of its unpredictability , standard internet protocols cannot work in these kind of scenarios. Hence disruption tolerant protocols like Bundle ,are being tested to provide support in these infra-structureless environments.
One of the most famous data centric architecture CCN is developed by PARC which is the next generation architecture to solve challenges in content distribution , mobility and security. This receiver driven abstraction works in a request-reply model to provide data from the most closest caches. It does not need any help from a central authority as compared to other ICN approaches[9][5] to provide communication, which makes it ideal for DTN scenarios. Although it needs some manipulation in its data structures to provide support for challenged environments , which is discussed in the next chapter.
DTN, ICN and CCN technologies were described in detail in this chapter, to understand the basics of the problem solved in this thesis. This chapter at the end also provides comparison between ICN and DTN protocols, so the reader can under-

stand the decisions taken to design the popularity and priority estimation algorithm described in the next chapter.

# 3 Solution

In this chapter we will describe different algorithms we designed using information centric functionalities for efficient data dissemination in Disruption Tolerant Networks.
First we discuss Name Based Prioritization(NBP) protocol which shows how information exchange can be prioritized and optimized using data centric functionalities in a disaster scenario. We also give a system model API, which manipulates the CCNx [4] abstraction to work efficiently in DTN scenario.
Second we describe Content Optimal Popularity Estimation Algorithm (COPA) which is a concrete mechanism to aggregate the popularity of interests for data (as issued by end-users) in a DTN scenario using the key data centric functionalities . Due to limited network resources in this network such as buffer space and bandwidth, popularity of content can help to make better caching and forwarding decisions during data mule to data mule (DM-DM) communication. The lack of continuous connectivity makes it difficult to detect duplicate end-user requests, which can increase the network congestion, memory usage and lookup time. COPA also helps to detect duplicate updates about the popularity in a very scalable manner. In addition we also discuss a unique reference algorithm to aggregate the popularity which is not so scalable and memory efficient but more accurate with respect to popularity estimation and loop detection.

## 3.1 Motivation

In the after math of the disaster scenario, critical and important information should have preferential access to the network resources. This preference based on names can be easily provided in data centric networks in comparison to the current internet architecture. The current internet architecture has to dig through different encapsulations to look at the metadata, to find what exactly is inside the packet.
Several previous studies have shown that [29][1] ,name based routing has proven to be a better approach in infrastructure less networks than the current end to end tcp routing. There are various proposals for information centric architectures which use name based routing mechanisms[2][7]. One of them is CCN proposed by Jacobson[2] which also provides built in support for DTN. However the paper lacks technical details to show how CCN can work efficiently in the DTN scenario where topology of network changes quite frequently.
We provide a name based prioritization mechanism which is an enhancement to an earlier work [29] in DTN scenario for efficient data dissemination by exploiting the three main data structures of CCN abstraction. The requests are forwarded until it finds the data against them in the data mule network.

### 3.1.1 Scenario & Assumptions

For understanding the significance of the scheme, consider the enormous earthquake which jolted eastern Japan in March 2011 causing serious damages in various fields of

life. Telecommunication infrastructure was divided into different islands, as reported by different operators in Japan.
We assume that disaster has divided the network into different fragmented communities. In this kind of scenario, the easiest and cheapest way of communication between the people of the different fragmented communities is possible through data mules as shown in Figure 16. Most of the base stations are not working in the area, but there are some base stations still working, where different data mules can fetch and deliver the information.

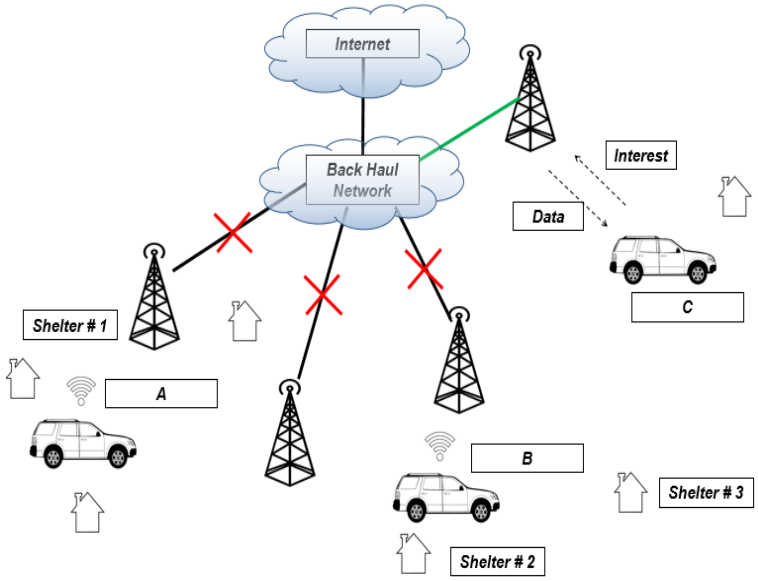


Figure 16: Disrupted Telecommunication Network after the Disaster.

Any moving vehicle equipped with linux based Wi-Fi routers on top of it can be used as data mules which could be cars, bicycles, helicopters etc. These data mules interact with end users, base transceiver stations and other data mules to disseminate information among the disaster stricken people and fragmented communities who have no internet access.
The data mules having CCNx abstraction on top of them make a data oriented architecture working in a disaster scenario which works as publish-subscribe model. This publish subscribe model provides a platform to publishers to send their data into the network and receivers can retrieve the data upon request.
Every end user runs a CCNx daemon at the backend, so that it can issue interest packets for the required data. We also assume that every end user can also become a publisher of the data, since any information in the disaster area is critical and can save lives. The end users can publish warnings about the roadblocks, health hazards, electricity short cuts etc. The end user FIB is manipulated accordingly, so that it can send requests to the data mules. The life time of an interest packet is set infinite or maximum, so that it could sit in the PIT until it finds the data against it.

### 3.1.2 Gateway/Shelter Routers

Due to disaster scenario, we assume that the area has been divided in different fragmented communities. In some cases most of the disaster stricken people have gathered in the fragmented islands in some shelter. These shelters contain gateway routers, which gather the requests from the all the end users inside the shelter.
In this manner, the data mules only interact with the gateway routers of the shelters to fetch the requests and deliver the data. These gateway routers save the time and energy of the data mules, since they don't have to interact with each and every end user in the fragmented community. Data mules take on the responsibility to transport messages between communities: when a mule encounters a fragmented community gateway, it delivers messages believed to be destined to the gateway's community.

### 3.1.3 Policies for Data Mules

The role of the router is in fact the policies set on the forwarding and caching mechanisms. In a normal instance with CCN, the interest to data exchange will work in First in First out (FIFO) manner, if you have piled up a lot of interests in the PIT. We believe that there should be a prioritized pending interest table which should fetch the critical data first from the corresponding data mule. So even if we loss connectivity we will get best out of the situation.
The traditional caching mechanisms like LRU and LFU used in CCN architecture might not efficiently [1] work in a disaster scenario. It's quite possible that it might flush the most critical data according to the situation.
We assume that governmental departments assign global priorities to prefixes according the scenario. These names are globally recognizable among all the data mules and base stations. So for example if there is an earthquake, the names like /Government and / emergency are given preference during scheduling as compared /chat, /YouTube between different data mules as shown in Table 2.

Table 2: Prefixes with global priorities and corresponding information [29]

| Name-Prefix | Priority | Life time of a Packet | Sender Authorization | Recipient |
|---|---|---|---|---|
| Government | High | Indefinite | Officials | All |
| Warning | Medium | Indefinite | All | All |
| First Responders | High | Indefinite | All | All |
| Police | Medium | depends | Police | Police |
| Chat | Low | Low | All | Public |

Furthermore each data mule has an identity. This identity is exchanged during DM-DM interaction, so that the loops created by the data mules during the disruption of network can be prevented. This identity also helps in accurate popularity estimation which is explained in Section 3.3.4 in detail. It must be clarified over here that no identity of the end user is passed on to the data mules to conserve the anonymity of the end users which is also a principal in an information centric network to preserve privacy and security.

## 3.2 Name Based Prioritization Protocol(NBP)

NBP protocol is designed to efficiently disseminate the information among the people after the disaster has disrupted the communication. It facilitates the communication between disaster stricken people of different areas through the assistance of a data mule network. We mold the three data structures of CCN accordingly in the mobile router to make it work in the DTN scenario.
The main target of NBP is to prioritize the exchange of the information among the data mule network based on naming. Data mules would communicate with end users, other data mules and the working base stations. For each of the entity, the protocol specification is different and is mentioned below.

### 3.2.1 Sailent Features

This protocol is specifically designed to prove that CCN abstraction can be used to disseminate data efficiently in a DTN scenario. Some of the key points are mentioned below.
• NBP protocol enhances CCN which enables the communication between different mobile routers, end users and base stations. It molds the CCN abstraction, so that interest and data packets can traverse through totally different paths and can manage large delays.
• It identifies the need of manipulation of FIB entries and interest packets, so that CCN protocol can work in disaster scenarios without any hurdle.
• To effectively use the intermeeting times between different data mules, it prioritizes the exchange of data with respect to the scenario, role of the node and popularity of the data.

### 3.2.2 Protocol Overview

NBP protocol on the top of CCN abstraction work as follows, during the DM- DM interaction.
1. *Whenever a node (data mule) X comes in the transmission range of node Y, both exchange the metadata about the data in the buffer.*

The meta data contains the list of names of all the data stored in node $Y$. All of the CS entries are matched with the local PIT entries. At this point of time, node $X$ knows what data can be extracted from node $Y$ based on the metadata received.
2. *Node X generates the interest packets towards the node Y, to fetch the data in a prioritized manner where priority gets higher for names from left to right as shown below and Figure 17.*
*Scenario > Role > Popular Content>Normal Interest*

The names regarding to the scenario are defined by the government and treated as first class citizens while exchanging and storing the data. Furthermore we consider each node might have different responsibilities depending upon their role. An ambulance node will have different important contents to send and receive as compare to

a police node. The popularity of the content is difficult to estimate in intermittently connected networks. We propose a mechanism in Section 3.3 which calculates the popularity of the content in a scalable manner.
The inter-meeting times of the data mules could be very short depending upon the situation. NBP makes sure the messages with high precedence are retrieved in prioritized manner during the exchange of information.
As mentioned in previous chapter, interests are forwarded to upstream routers according to the FIB entries. If there are no FIB entries, the interest packet is dropped [2] or a negative acknowledgement is sent to the downstream router[7].
Hence FIB cannot provide the functionality to forward interest packets towards potential sources in an intermittently connected network. So we add the FIB entries in the CCN router for all the data which can be retrieved from the encountering node. The manipulation of FIB divert all the interests towards the encountering node which in response fetch all the data.

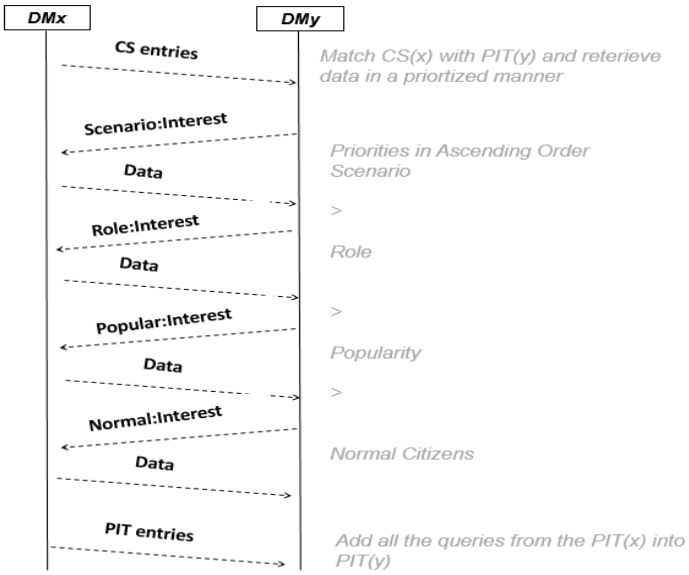


Figure 17: Name based priority exchange protocol from DMy to DMx.

3.*Add all the queries from the encountering node to the local PIT.*

Since we assume the movement of nodes is totally unpredictable, we believe the queries need to be forwarded in every direction. In this way the interests are forwarded into the data mule network unless the data is found against them. The CCN protocol makes sure there are no duplicate entries in the PIT.

NBP protocol resembles slightly with epidemic routing protocol [25], since it replicates all the queries to the encountering node coming in the wireless range. However NBP prioritizes the exchange during the encounter to give high precedence to the popular and important names.

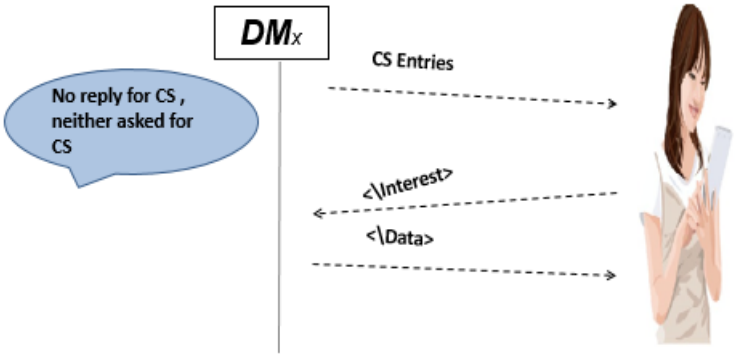


Figure 18: Interaction between end user and the data mule..

It is very important to create a difference between the end users and the data mules, since both are running CCNx software[4] . NBP application works on top of each of CCN abstraction with in the data mules. Since the data mules always exchange CS entries with each other , both enduser and data mule within split of a second can understand to whom they are interacting with. End users share all the interests with the data mule to extract the data out of them, as shown in Figure 18.

### 3.2.3 NBP Protocol API

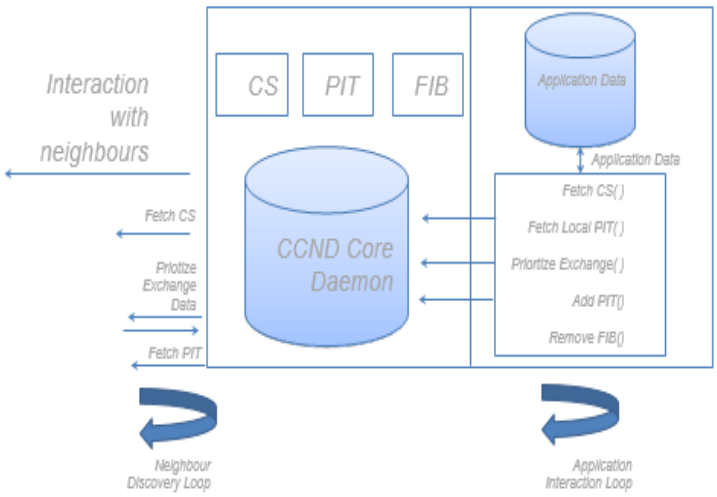


Figure 19: System Model/ Prioritize exchange Algorithm API.

The system model can be seen in Figure 19. The NBP algorithm API has two major loops which tackle the interaction with the neighbours and the application on top of CCNx.

**Neighbour Discovery Loop** This loop manages the interaction with the neighbours, once they come inside the radio range. It is also responsible for sharing the local CS and PIT list items with the other data mule.

**Application Interaction Loop** This loop is responsible for all the communication between application and the core CCND daemon. The functionality includes fetching names of locally stored content items and pending interest table entries, creating forwarding information base entries in a prioritized way and adding disjoint PIT entries from the encountering node.

*FetchCS()* retrieves the list of all the content items store in the buffer of the encountering node.

*FetchLocalPIT()* retrieves the list of all the pit entries from the local CCND core daemon.

*Priortize()* function is responsible for creating a list of requests which can be fetched from the encountering node depending upon the content items in the encountering node buffer. This list is then prioritized according to the important names with respect to the role ,scenario and popularity. Finally for the names in the list, FIB entries are created by the application, which in response fetch the data from the other node.

*AddPIT()* retrieves the list of pending interest table entries from the encountering node and add it to the local PIT. If CCND already has the same PIT entry it only adds the face of the request otherwise it also adds the entry to the PIT.

*RemoveFIB()* removes the FIB entries from the core CCND daemon , once both the data mules move out of the radio range of each other.

## 3.3 COPA: Content Optimal Popularity Estimation Algorithm

With per-hop routing [22] and limited network resources in intermittently connected networks and mobile adhoc networks, calculating popularity of content in a scalable manner can be a major challenge. Due to limited information of the topology of the network, this kind of routing might lead to loops in the network. This algorithm at hand targets a solution for both of the above mentioned problems in a scalable manner, which can work on top of CCN protocol.
In CCN, two different types of messages exist: a) interests for content (expressed via a name), and b) the actual data items to match a given interest. For optimizing the exchange of such messages at an ICN data mule encounter in order to maximize receipt of desired messages at end-users, it is very useful to estimate the popularity of a given ICN interest message: If data mules can estimate which interests are "more important" than others, they can surely optimize the exchange of messages at encounters with other data mules.

Concrete exchange algorithms for data mule encounters (e.g. traffic engineering/scheduling/prioritizing according to the popularity of interests in an opportunistic network) are outside of the design of COPA; it solely tackles the problem of estimating the popularity of interest messages in a completely decentralized manner among data mules in a DTN scenario. In other words, the algorithm give details about how to achieve scalable, distributed counting/aggregating of interests (in the sense of a popularity count) in an opportunistic network / DTN scenario where ICN interest messages and corresponding data items are being used. In particular, the algorithm tackles the problem of having a scalable solution with loop detection that works in a fully decentralized scenario with random, unpredictable movements of ICN data mules.
This popularity estimated by COPA , can be used by the NBP protocol for prioritizing messages which are more popular than others during the DM-DM interaction and DM-Shelter router interaction.

### 3.3.1 Goals

The goal of this concrete mechanism to aggregate the popularity of interests for data (as issued by end-users) in a DTN scenario using the key functionalities of ICN (name based forwarding). Due to limited network resources in this network such as buffer space and bandwidth, popularity of content can help to make better caching and forwarding decisions during Data Mule to Date Mule (DM-DM) communication. The lack of continuous connectivity makes it difficult to detect duplicate end-user requests, which can increase the network congestion , memory usage and lookup time. The technique also helps to detect loops in a decentralized scenario, in a very scalable manner.
The popularity estimation algorithm works by adding up all the interest messages from the end users or downstream routers. There are two ways in which popularity can be estimated in CCN abstraction. First way is to increment the popularity count when some prefix is *requested* from downstream router and the second way is to increase the counter once some data is *delivered* to an end user or downstream router. The ultimate goal of this mechanism is to speed up the process of fetching data against the most popular requests. This goal would be achieved by maintaining a popularity based on queries requested rather than data delivered. So we give preference to the requests made by the end users and decided to aggregate the popularity by incrementing the counter for each distinct request.
In past there has been some analytical research [30][31] about caching the data inside nodes in DTN, based on the popularity of the content. To best of our knowledge there is no prior art to calculate the popularity of content in these challenged networks.

### 3.3.2 Filtering Effect

Estimation of popularity in challenged environments is a huge challenge due to the limited knowledge of the topology. Nodes move around random in the area to fetch request and deliver data. CCN protocol normally uses LFU or LRU mechanisms to count the frequency of incoming requests for providing efficient storage. These

mechanisms cannot provide accurate estimation of popularity due to the aggregation of the requests from the downstream routers. This aggregation is done in PIT, to only send one request upwards to reduce network congestion. Therefore, popularity of content in intermediate routers do not reflect actual global popularity even in well-connected networks.

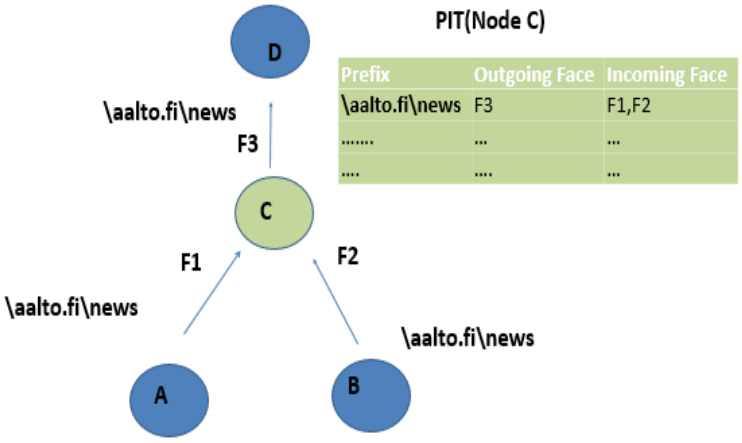


Figure 20: Aggregation in pending interest table

Figure 20 shows the filtering effect in node *C*. Node *C* get requests from two downstream routers *A* and *B* for a prefix /aalto.fi/news from faces F1 and F2. Node *C*, being the upstream router aggregates the request and sends only one request to it upstream router *D*. This aggregation limits the upstream routers to calculate the actual popularity of each content.

In challenged networks, it is more difficult to estimate the popularity due to mobility of data mules. Multiple interaction of the same data mules create loops in the network. These loops can overestimate the popularity by adding the same distinct requests from multiple users. To accurately estimate, we present two unique solutions for popularity estimation in complete delay tolerant environments.

### 3.3.3 Reference Algorithm

A naïve solution to the aforementioned problem of decentralized popularity estimation of end user requests (with randomly moving data mules) would be to append to each end-user request a unique nonce (say randomly created and appended by each end user for each interest for a given name). In our reference algorithm, each data mule saves each distinct nonce embedded in the interest packet from all the queries of the end users and other data mules to have a transparent popularity ranking.
In CCNx software version 0.8 and NDN specification an interest packet has a nonce, which is helpful in detecting loops in well-connected networks.

```
Algorithm #1 Reference Algorithm
1: Receive an Interest Packet (name, face_in)
2: If Cache_Miss && PIT_Match then
3:       If Lookup (nonce_match) then
4:              Loop (Duplicate Packet);
5:       else
6:              add_nonce () \*Add nonce to hashtable*\
7:  else if PIT_MISMATCH then
8:       Insert_Prefix_PIT ();
9: end if
10: function get_popularity (name)
11:      count_all_nonces (name)
12:             return (count)
13: endfunction
```

Figure 21: Reference Algorithm.

As shown in algorithm 1, every new nonce for a particular prefix is added in the nonce hash table to estimate the correct number of requests for a particular name. A special function is used to count all the nonces for a name and return it to the desired function. This function could be used by the cache to retain the most popular values and delete less popular values in case there is a shortage of memory.

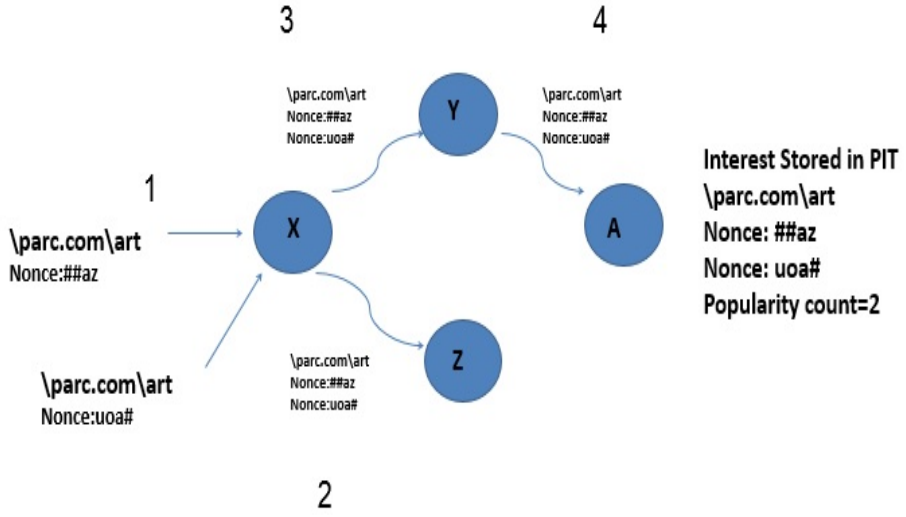


Figure 22: Popularity Estimation using Reference Algorithm

Figure 22 shows four nodes(X,Y,Z,A) moving among different communities to fetch and distribute data and requests. These nodes are not connected and meet each other in chronological order mentioned in the above shown figure. Node $X$ gets request for the same name by two distinct end users. Node $X$ meets node $Y$ and $Z$ at different time and forward the interest packets with their respective nonces. Eventually node

$Y$ then forwards the interests to node $A$.This figure also shows the state of pending interest table and the nonce hash table at each node.

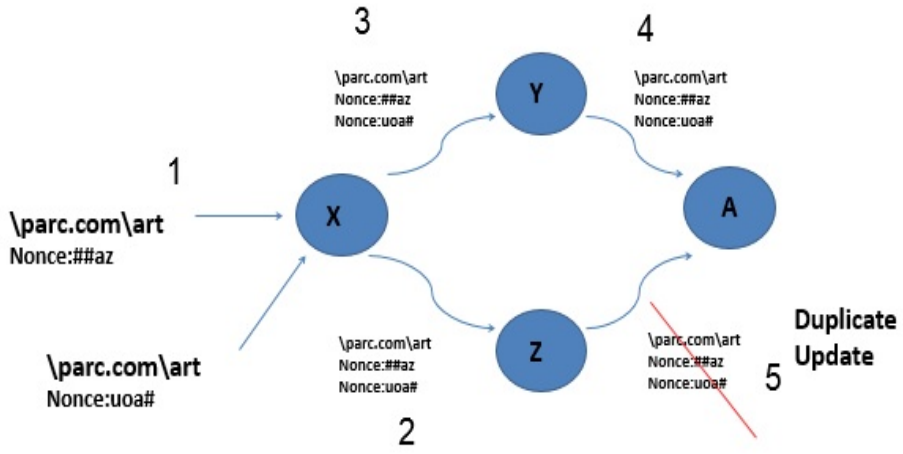


Figure 23: Loop detection in data mule network

Reference algorithm can detect the duplicate requests and can prevent the loops in the data mule network.It saves every distinct nonce to prevent the overestimation of popularity as shown in Figure 23.
Figure 24 is the next step of the previous figure, which shows node $Z$ gets request for the same name /parc.com/art by two new distinct end users. Node $Z$ add those nonces into the nonce hash table. Eventually if each node in the network meets every other node, the popularity of the prefix /parc.com/art will be 4 in each node. The popularity of a content can be known by counting all the all the nonces for that particular prefix.

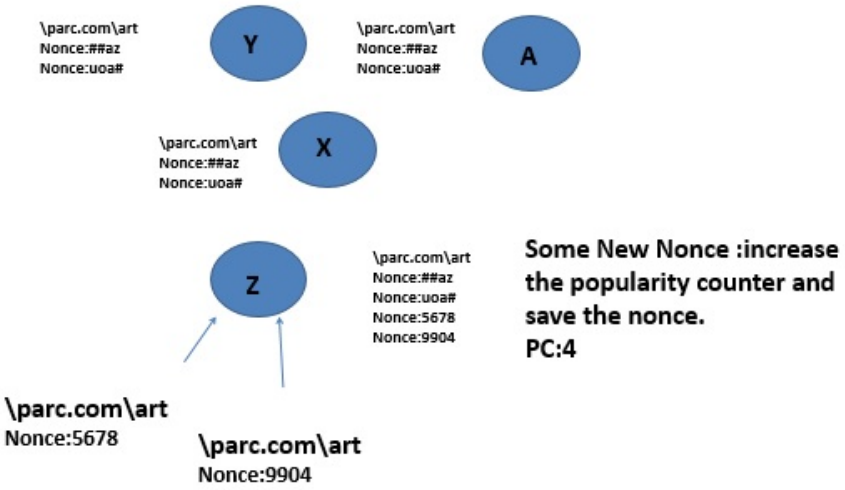


Figure 24: Reference Algorithm saves every distinct nonce

These three figures explain how interests are aggregated and popularity is estimated in a data mule network by forwarding all the nonces attached to the interest packet to the encountering node.Loops can be easily detected since nonce from every distinct user is saved in each node , once the information reaches it.

However, such a solution does not scale: if, say, a data mule has received 1000 interests for the same name, it would need to store 1000 nonces just for this name. Moreover, if two data mules meet that each have received 1000 interests so far, they would need to match and exchange their 1000 nonces with each other. This extra space for the nonce hash tables will reduce the memory for the actual content in the router. Maintaining large amount of nonces in this dynamic environment where there is a scarcity of resources will have a damping effect on resources.

### 3.3.4 Core Algorithm

The key idea is: while end-users assign each interest a unique nonce (as in the naïve solution explained above), when they forward such a request to a data mule and the data mule has a counter larger than 0 for the name of the interest already at that point in time, increments its counter by 1, and **assigns** its own *[nonce:counter]* tuple to the end-user for the given name. If the end-user encounters a different data mule in the future, it uses the [nonce:counter] tuple it has been assigned. Furthermore, when two data mules encounter each other and both have for a given name already a [nonce:counter] tuple, aggregation of nonces and counters is performed, as detailed below.

The following rules are applied on processing the new [nonce:counter] tuple which is appended to interests:

• If an end-user meets another data mule (after having previously being assigned a [nonce:counter] tuple from a different, previously met data mule), it will append the previously assigned [nonce:counter] tuple to the interest instead of its own generated nonce.

• When a data mule receives an interest from an end-user, two cases can occur

o *New end-user request*: If this is the first data mule encountered by the end-user, the end-user will send an interest with a freshly generated nonce. The data mule accepts this, and if it has already received one or more different interests for that name (i.e. its counter is equal to or larger than 1), assigns a new [nonce:counter] tuple and sends it to the end-user for use form now on.

o *End-user request with assigned [nonce:counter] tuple*: If the end-user has already been assigned a [nonce:counter] tuple (from a previously met data mule), when meeting a new data mule this is send along with the interest. If this is the first interest the data mule receives for this name, it stores the [nonce:counter] tuple. If

it has already a different [nonce:counter] tuple but with the same nonce, from that point in time the [nonce:counter] tuple with the larger counter prevails and will be used by the end-user and data mule from thereon (as a lower bound on end user requests and this popularity of the interest). If the nonce is different, the data mule assigns its previous nonce but with the counter being the sum of both previous counters.

• When two data mules meet, three cases can occur (for each given name these data mules have a pending interest for):

o *Same [nonce:counter] tuple*: If the [nonce:counter] tuple is the same at both data mules, nothing needs to be done, as both data mules have apparently the same aggregated counter value.

o *Same nonce, but different counter*: If the nonce is the same, but the counter is different, the counter cannot be added as – due to the random movement by data mules over time – it is not clear that both counters do not contain overlapping end-user requests. However, from that point in time both data mules can use the [nonce:counter] tuple with the larger counter (as a lower bound on end user requests and this popularity of the interest). One can summarize as: ***Nonce with originally largest counter prevails at both nodes, largest counter prevails at both nodes.***

o *Different nonce*: If the nonce is different, the nonce with the largest counter prevails at both nodes. The counter can be added and both data mules use from thereon one nonce with a new counter being the sum of both previous counters. One can summarize as: ***Nonce with originally largest counter prevails at both nodes with counter being sum of both previous counters.*** However, the following kinds of data mule encounter loops can occur as shown in Figure 25(as an example, similar loops are imaginable): data mule $A$ meets first data mule $B$, then meets data mule $C$, and then again data mule $B$. If in this case data mule $C$ had a larger counter and different nonce than $A$, $A$ will add the counter of $B$ twice (overestimating the actual content popularity).
To prevent such cases, each time a data mule encounters a data mule and updates its [nonce:counter] tuple for a given name during the encounter, the data mule adds the *ID* of the encountered data mule to a memory list of length k (where k is a configuration parameter each data mule can set individually). If the data mule is encountered and already on the list for a given name, only the nonce with originally largest counter prevails at both nodes, largest counter prevails at both node rule is being used (to prevent overestimation). Also, if the list has been filled up to maximum length k, only the nonce with originally largest counter prevails at both nodes, largest counter prevails at both nodes rule is being used from thereon. Data mules apply this same scheme to detect re-visits of end-users that have changed the nonce due to intermediate encounters of other data mules.

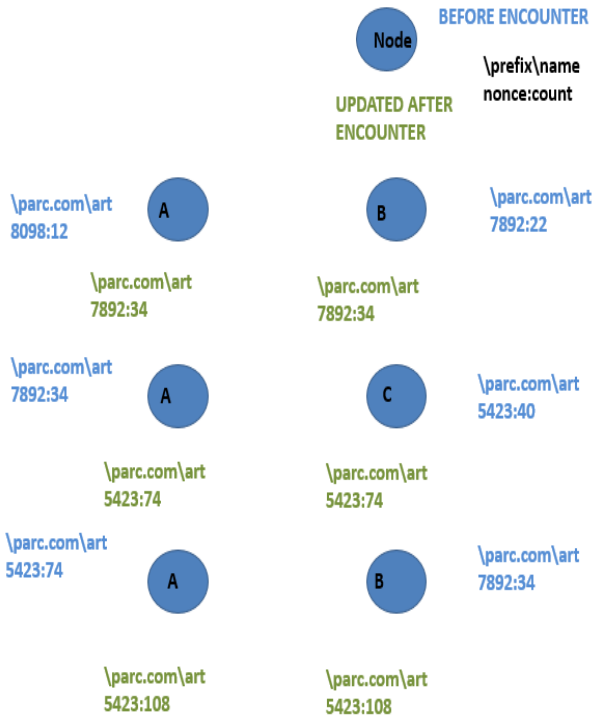


Figure 25: Overestimation of Popularity (/prefix nonce:popularitycount)

Note that the scheme is loop-free in the following sense: if there are loops in data mule encounters (as may very well be the case in a DTN scenario with random data mule movements), the counter for a given name does not overestimate interest popularity among end-users for that name. This is achieved by the memory list.

• *Gateway/Shelter router*: This router follows the same rules as for the data mule router mentioned earlier.However it just remains stationary with the fragmented community. The interaction with gateway routers is more time and energy efficient as compared with all the members of fragmented community separately.

**Alternative option for using memory list of length k (sliding window)** As an alternative to using the nonce with originally largest counter prevails at both nodes, largest counter prevails at both nodes rule when the memory list of length k is full for a given name at a data mule encounter, a sliding window approach can be used for the memory list as follows: if the list is full, the first encounter in the list gets removed and all other entries get shifted in a sliding window kind of fashion, so that the newly encountered data mule can be added to the last position in the list. Then, the nonce with originally largest counter prevails at both nodes with counter being sum of both previous counters rule is being used. In other words, once the list is full newly encountered data mules get added in a FIFO fashion where the oldest encountered node leaves the list, and adding of counters is applied at the risk of having encountered a node before but being of the memory list due to the sliding window. For this option, the length of the memory list needs to be selected

carefully such that loops of data mule encounters longer than k are highly unlikely; in this case, the mechanism provides a more accurate prediction than the nonce with originally largest counter prevails at both nodes, largest counter prevails at both node rule.

**COPA Operation** The basic working of the popularity counter – when data mules meet end-users and afterwards encounter each other - is explained with the diagrams displayed below, with the numbers showing the chronological order data mules encounter each other and the [name:nonce] tuples represent individual end-user requests. Each node has to remember only a a few nonces for the same name in the whole network. It is quite possible that data mules in the network have different nonce values and popularity counters for the same prefix at a given point in time. Note that this procedure only needs a minor change in the implementation of the end user.
Consider four nodes (X, Y, A, B) inside a data mule network. These mules move around randomly in the area to provide connectivity. For understanding we consider they meet in the chronological order as mentioned in the figure below. Figure 26 shows node $X$ receives two requests from the end users.

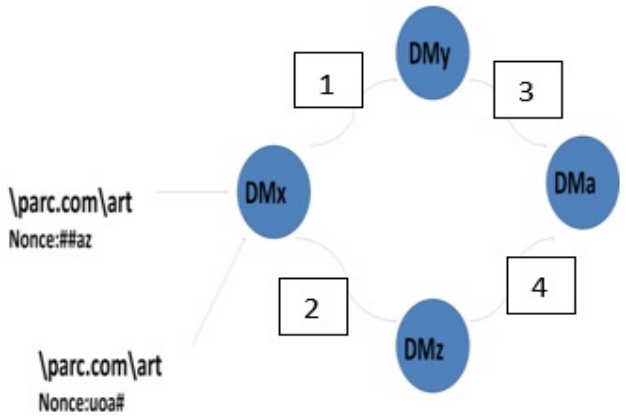


Figure 26: Two end users request the same name from DMx

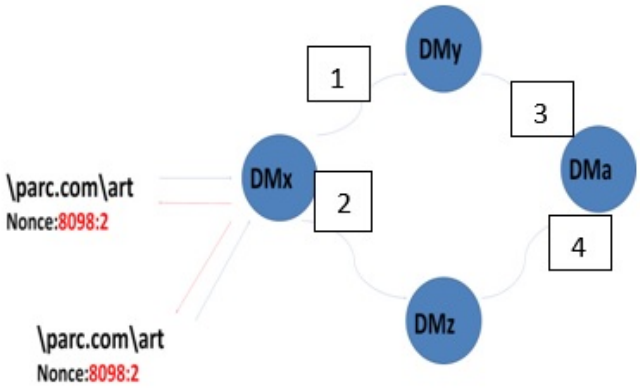


Figure 27: End users are assigned a new nonce with count 2 for the same prefix

Figure 27 shows the aggregation of requests from the data mule. The end users are assigned a nonce:counter tuple which so that next time , if they send the same request to the data mules, it doesn't gets overestimated.

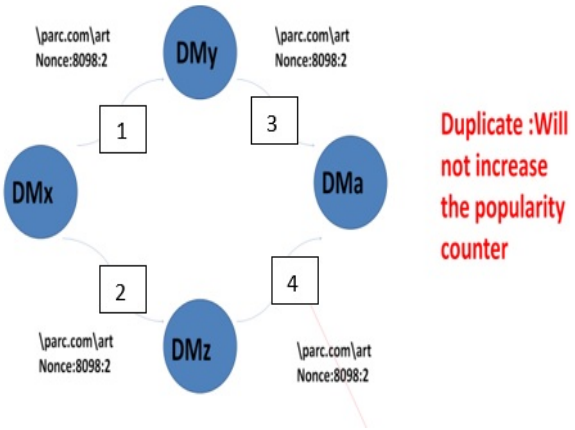


Figure 28: DMa receives duplicate query(loop) from the DMz, so drops the message (loop detection).

Figure 28 shows that the request is received by data mule $A$ , in a hop by hop manner through different ways. Node $A$ detects this loop and and rejects the update from the route (X-Z-A). So COPA also helps to detect duplicate updates to estimate the popularity accurately.

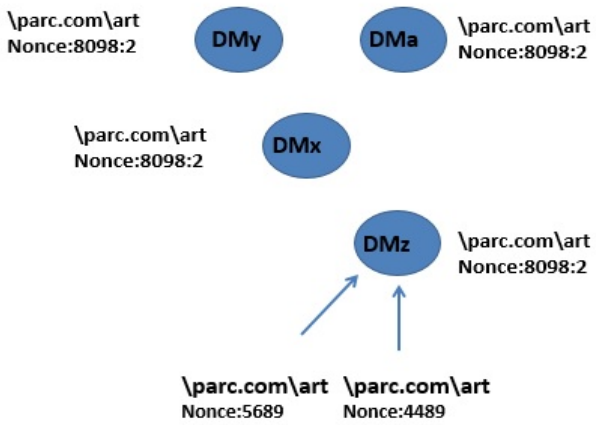


Figure 29: Two new users request the same prefix from DMz

Figure 29 shows that at this point of time, all of the data mules in the network have the same nonce:counter tuple(8098:2). Now two more end user request the same prefix from node $Z$ with distinct nonces.

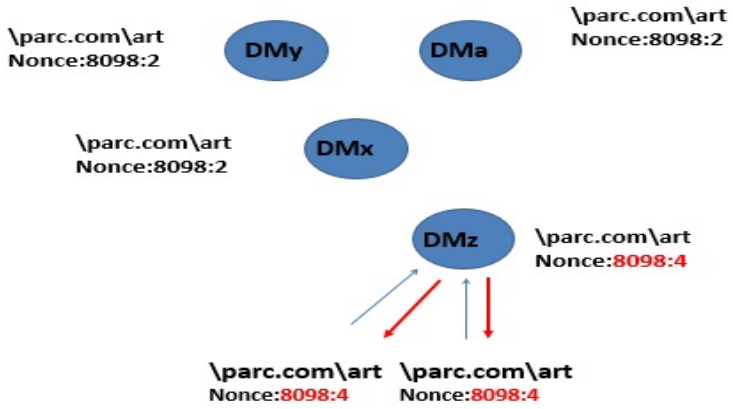


Figure 30: End users and node DMz updates the popularity count for prefix /parc.com/art

Node $Z$ aggregates the count and assigns a new nonce:counter to the end user as shown in Figure 30. Node $Z$ aggregates the count for prefix /parc.com/art and updates it popularity count to 4. Ultimately if node $Z$ meets every other node, this update gets stored in all of the data mule network.
This mechanism of aggregating interests can be implemented on any protocol which uses name based forwarding in a decentralized scenario. The internal working of the CCNx daemon is shown for this particular mechanism in the upcoming subsection .

### 3.3.5 COPA-CCN Integration

The three main data structures of the CCN abstraction were explained in the 2nd Chapter. Figure 31 below shows how CCN is extended to estimate the popularity in a DTN scenario.

This diagram explains the working of the protocol, when two data mules meet. The blocks in blue colour show how CCN protocol work, when it receives the request. The green colour blocks show the extensions added for the COPA for popularity estimation.
Once a CCN data mule receives an interest from an end user or another data mule, they go through the following process.
If the name is not found in the CS and found in the PIT, their nonce is matched with the nonce for that prefix available in the nonce hash table. If the nonce is different, the memory list for the encountered data mules is checked, if the data mule has not encountered the data mule before, the rules of addition apply as shown in the right most part of the diagram.
If the memory list is full or the data mules have met before, and if both data mules have same popularity of the prefix, the nonce with the higher nonce count prevail at both data mules. If the popularity is same, it is a packet which was received in the past, so it is discarded.

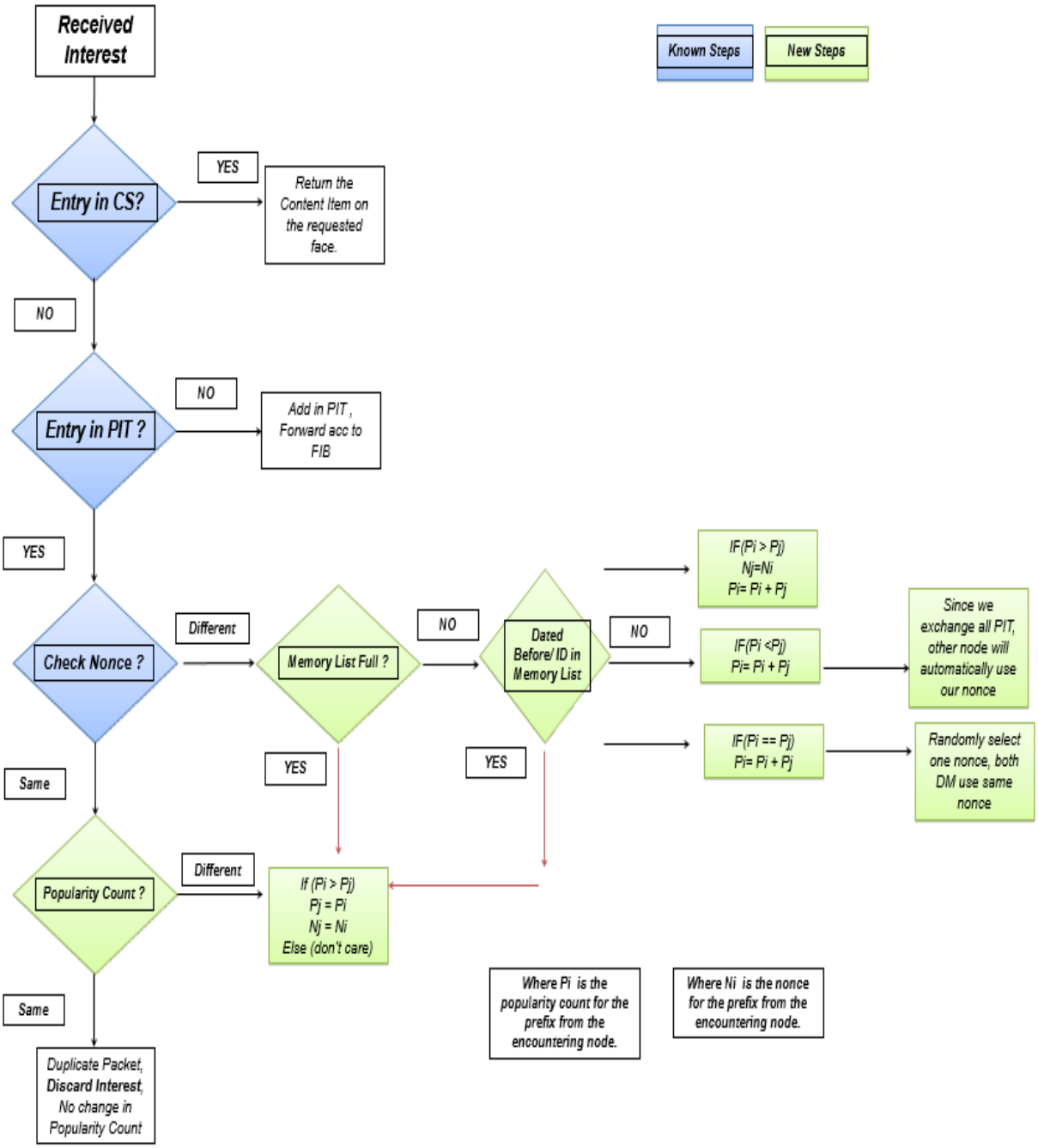


Figure 31: Flow Diagram of COPA for an incoming interest in a CCN data mule

### 3.3.6 Analytical Model for Memory Estimation of Nonces

Suppose there are $P$ prefixes in the pending interest table. For each prefix $Pi$ there are $Nj$ number of nonces which mirrors each distinct request from the end user. $m[Nji]$ denotes the memory occupied by a particular nonce of that prefix. The total memory occupied $M$ by the nonces for every prefix in a PIT of a content centric node for a reference algorithm can be calculated by Equation 1 .

$$M_{ref} = \sum_{0}^{i-1} \sum_{0}^{j-1} m[Nji] \tag{1}$$

COPA uses only one nonce $Ni$ and an additional popularity count $pi$ for each prefix as compare to the reference algorithm for estimating popularity. The total memory occupied by the nonces for COPA can be estimated by Equation 2

$$M_{copa} = \sum_{0}^{i-1} m[pi] + m[Ni] \tag{2}$$

Memory occupied in the whole intermittently connected network by the nonces for each prefix can be estimated by multiplying the above equations with number of data mules which we suppose is $K$.

$$\begin{aligned} M_{ref} &= K * (\sum_{0}^{i-1} \sum_{0}^{j-1} m[Nji]) \\ M_{copa} &= K * (\sum_{0}^{i-1} m[pi] + m[Ni]) \end{aligned} \tag{3}$$

Comparing Equation 1 and 2, one can depict that COPA would be quite memory efficient depending upon number of distinct nonces gathered for each prefix in the data mule network. For Zipf generated data , one can predict that COPA would take less memory in comparison to the reference algorithm.

## 3.4 Conclusion

In this chapter we explained two data centric mechanisms for dynamic environments which complement each other exceptionally well.
NBP is a flooding based content centric protocol which fully exploits the network awareness of the underline abstraction. This protocol forwards the queries to each inter-meeting node in the network to speed up the process of fetching required data. It prioritizes the encounters between the data mules based on the policies set on nodes. Furthermore this chapter presents a NBP API , which explains in detail the functions implemented on top of CCN abstraction.
Resources like storage, energy and time are very limited in these infrastructure-less environments. Popularity of content can help to make better forwarding and storing decisions at a node level. It can also help to convey critical information of desired popular information to the concerned authorities in disaster scenarios. To best of our knowledge we have not seen a popularity estimation mechanism for DTN scenarios in prior research. Estimating of popularity in a well connected CCN network is difficult as well due to the filtering effect. The main contribution of this thesis is two unique popularity estimation mechanisms proposed which can work in challenged wireless networks.

First this chapter introduces a unique reference algorithm to aggregate the interests from all the end users and inter meeting nodes. This algorithm needs to store each distinct nonce for each prefix in the hash table to have a popularity ranking. In contrast we also propose an intelligent popularity counting mechanism(COPA) which only saves one nonce per prefix. At the end we also provide an analytical model which validates the memory efficiency of COPA over the reference algorithm.
To sum up ,COPA provides an intelligent popularity metric which can be further used by NBP protocol to forward and store data more efficiently in an infrastructure-less environment.

# 4 Evaluation

This chapter introduces the prototype network built for the evaluation of COPA. This prototype network conforms to the rules of CCN protocol as mentioned in the previous chapters. The goal to build this prototype model is to evaluate the accuracy of the COPA and the unique reference algorithm. Once the popularity is estimated, COPA is compared with the reference algorithm with respect to accuracy and memory utilization etc.
Furthermore the chapter also discusses the results from the NBP protocol which uses COPA and important names with respect to the scenario and role to prioritize the exchange the data during the data mule encounters.

## 4.1 Evaluation Scenario

The scenario setup consists of two fragmented communities. These fragmented communities are cut off with their respective base stations. Only source of information retrieval is by data mules, in a DTN manner. End users request data to their respective data mules inside their fragmented communities when they come into the radio range.
There are in total six data mules who move around in both of the fragmented communities fetching interests and delivering data. We assume there is no disruption in the communication during the inter-meeting times of the data mules. The encounters between different data mules are totally random.
There are 100 distinct interests queried by the end users in each of the fragmented communities. The frequency of queries towards the data mules from the end users follow Zipf distribution, meaning that the popular interest will be queried the most during the meeting between end-users and data mules.
Each data mule has its own ID which is exchanged during the initial coordination as shown in Figure 32. There is a memory list in COPA which stores the ID of previous encounters with the other data mules. This memory list is empty when the experiment starts.

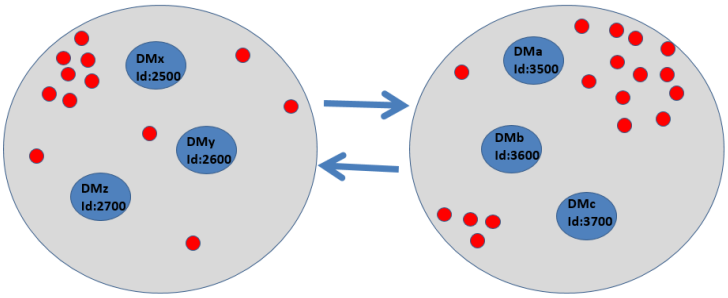


Figure 32: Test bed scenario for COPA and Reference Algorithm, where red circles represent end-users

## 4.2 Implementation

In order to reflect the operation of COPA and CCN, we have built a dedicated simulator. This simulator contains servers which emulate the working of data mules in a DTN scenario. The six servers are listening on dedicated sockets as seen in Figure 32, where the port number is their ID. There is one separate server which manages the movements and encounters of the data mules in the network. There is only one encounter at a particular time, where data mules exchange interests between each other. The interests and nonces are aggregated in the data mules depending upon the algorithm used.

### 4.2.1 Generating Interests wrt Zipf Distribution

An application was written for generating interests which follow Zipf distribution.

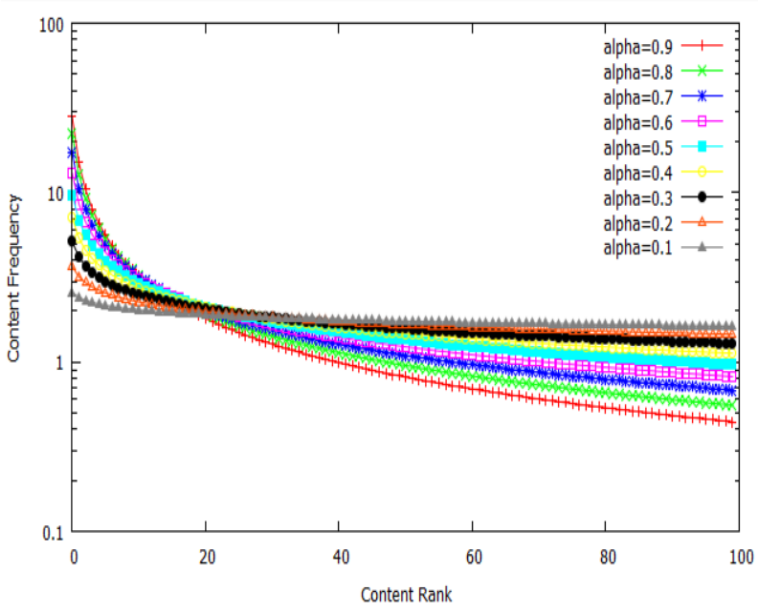


Figure 33: Zipf curve generated for different alpha parameters

Figure 33 verify the results generated from the application which follow zipf distribution. Y-axis uses a log scale while x-axis uses a normal scale. The graph shows the comparison between different zipf curves for different alpha values e.g 0.1<alpha<0.9 . For alpha=0.9 , the curve is the most sloppiest and has the largest difference in frequency between the highest ranked and lowest ranked content item. In contrast for alpha=0.1 , the zipf distribution follows a uniform distribution, as it is almost a straight line in the graph.

## 4.3 Testing COPA & Reference Algorithm

This section evaluates the results of popularity estimation algorithm with respect to different parameters. As mentioned earlier there is no prior art of calculating the

popularity of requests in a DTN scenario, so we compare our results of COPA with another reference algorithm which is also unique.

### 4.3.1 Reference Algorithm

The description of the reference algorithm is described in the previous chapter. Over here we evaluate the results. There are three lines in the graph. Zipf distributed data means the interests which were requested by the end users from the data mules. On the other hand it is compared with the popularity estimated by the reference algorithm by counting all the nonce for a particular ranked prefix in each of the data mules at the end of the experiment.
Once the experiment starts, there is a limit applied on the on the total number of the encounters inside the network. For this particular case it was tested with a total of 15 and 7 encounters. It is quite possible that the two data mules met twice or thrice during the whole experiment.

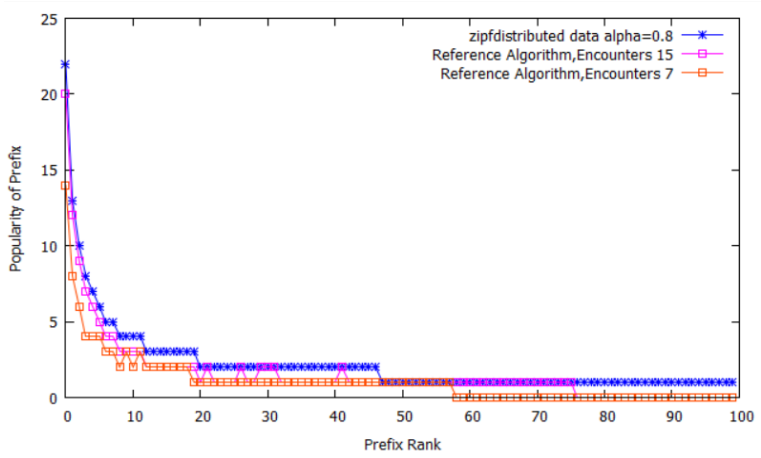


Figure 34: Accuracy of the Reference Algorithm vs Initially Poured Interests

Once the experiment stops, popularity count for each prefix from each of the data mules is added and divided by the number of data mules. These results were normalized by doing 10 experiments and each run took at least 4 minutes.
We can see from the Figure 34, with more no of contacts between the data mules, the popularity of the prefix is better estimated. For 15 number of contacts in all the data mule network, the popularity of prefix is almost same, as the originally requested data from end users.

### 4.3.2 COPA

For COPA, same kind of experiment was performed as mentioned previously with reference algorithm. Figure 35 shows that it estimates the popularity quite accurately. With seven encounters the popularity is not accurate but has the same linear decline as the original one.
The encounter list which saves the identity(ID) of all the encounters with the previous data mules, is empty in the start of the experiment.

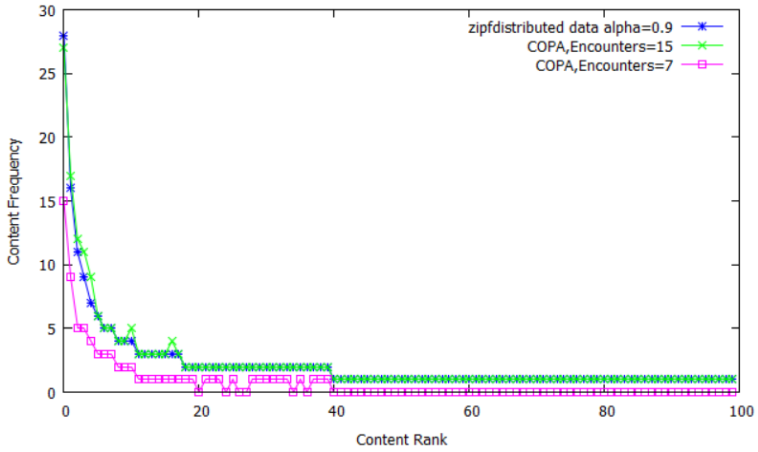


Figure 35: Show the accuracy of the COPA vs Initially Poured Interests

### 4.3.3 Analysis

We observe that, the estimation of popularity would be most accurate, if all the data mules have received update of each distinct request from the end users. This update can be received directly or transitively from the other data mules, end users and gateway routers.
From the current graphs, one can see that with fifteen encounters of six random data mules, COPA and reference algorithm estimate the popularity quite accurately for this data mule network. If we increase the number of data mules in the network, one would need a large number of contacts between the data mules to estimate popularity accurately.
Both the algorithms accurately measure the popularity of the prefix with appropriate number of encounters inside the data mule network. It is interesting to note that at some points in the graph, COPA provides higher popularity than the original demand of that prefix due to some loops in the network. One information is added twice by the data mule, which in result produces more popularity. However the data mule network evolves to have only one nonce for a certain prefix in all of the data mules.
On the other hand reference algorithm, do not have a single point in the graph which is more than the original demand of that certain prefix. Since it is saving each and every distinct prefix from the end users in its storage. Either way it does not matters for the forwarding and the caching decisions of a particular data mule, since the popularity linearly rises or declines in both of the algorithms. It is quite possible that at the end of experiment, there are more than one nonce for a particular prefix in data mule network , which is not aggregated in the above mentioned experiments.

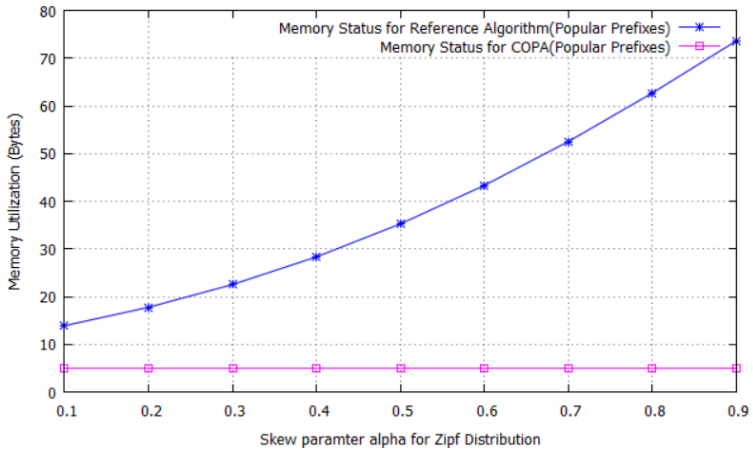


Figure 36: Memory Utilization Comparison between COPA and Reference Algorithm

**Memory Utilization** Figure 36 shows the memory utilization of nonce for the top 5 prefixes popularity wise, in all the data mules for both the algorithms for different values of the skew parameter of zipf distribution. CCNx abstraction utilizes one byte string to store one nonce in its hash table, so the y-axis represents the number of bytes for both algorithms. In the case of COPA, the memory usage is same for all the values, as it saves only one nonce. In comparison the reference algorithm saves each and every distinct nonce from the end users. That's why the line for the for reference algorithm increases with increasing value of the skew parameter.
Normally Zipf distribution with 0.8<alpha<0.9, mirrors the request rate of the current internet traffic with respect to the popularity of the content. With that interpretation in mind, COPA is 14 percent more memory efficient than the reference algorithm with 200 prefixes in this data mule network.


Figure 37: Encounter List of DMx with ID: 2500, during the previous experiment

**Encounter List Full** For this particular experiment, we maintain the list of the encounters in COPA from the previous experiment. Figure 37 shows encounter list of a particular data mule. Aggregation of popularity is totally different in COPA, once the memory list is full. From the previous chapter we know that if the memory list is full in COPA, the nonce and the count with the higher counter prevails unless and until the data mule is directly fetching from the end users.

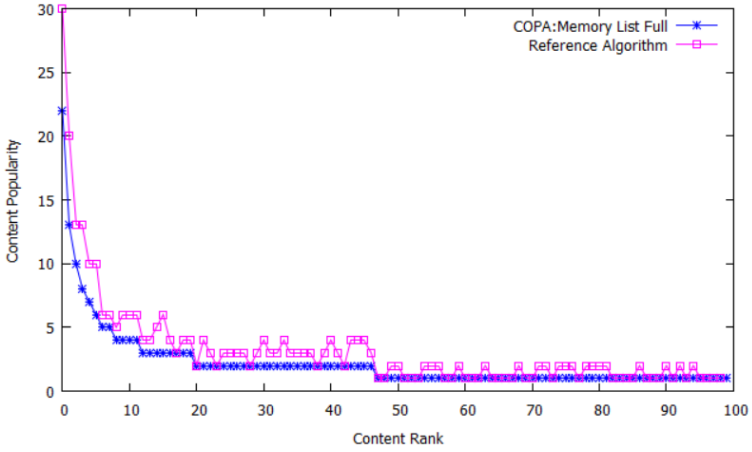


Figure 38: COPA (Encounter list full) vs Reference Algorithm

For the above mentioned experiment, more interests were added into the data mule network. We can see from Figure 38, COPA couldn't add more traffic, since in this particular case the nonce with the higher counter prevails in both of the data mules. If the popularity is aggregated from the end users itself, then COPA is able to add the popularity accurately.

## 4.4 Testing NBP Protocol

This section evaluates the NBP protocol which is implemented on the top of CCN layer. Some modifications were done to CCNx code base, to enable the communication with the encountered data mules. The data mules move in a random manner between different fragmented communities in the delay tolerant network, so one cannot predict the future movements. The initial coordination to exchange the list of pending interests and data items in Content Store is send by normal traditional sockets between two CCN nodes. The current CCNx works as an overlay abstraction on top of UDP or TCP as configured by the user.

### 4.4.1 Initial coordination in TCP

Suppose a scenario where two data mules come inside the radio range of each other. According to the NBP protocol both nodes want to exchange content stores entries from each other to fetch the data items in a prioritized manner.

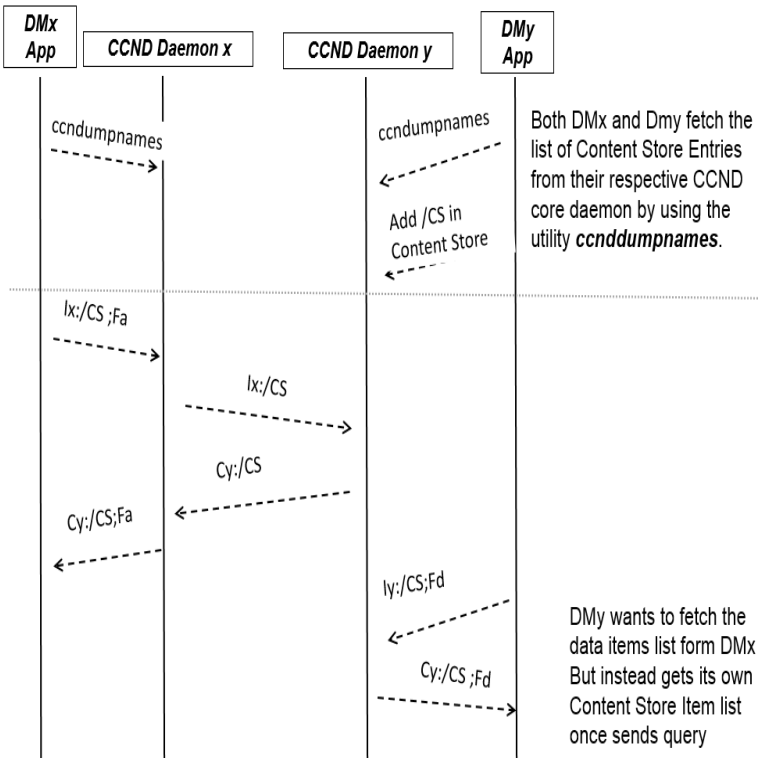


Figure 39: Problems with CCN protocol for initial coordination in NBP

The utility *CCNDUMPNAMES* is used to get all the content store entries from the core daemon in each of the data mules. In Figure 39 , at node $Y$ the list of the entries is saved inside the buffer memory by the name /CS . The application in $X$ queries /CS from the core daemon , which it forwards to $Y$ and gets the data item. After this procedure the /CS data item gets stored inside the content store of $Y$. So when node $Y$ queries an interest for prefix /CS, the core daemon send back the same stored data item rather than fetching it from the $X$.
This problem could be solved by removing the /CS prefix from the content store. But still need some synchronisation to decide, who should query the /CS prefix first. Another solution for this problem would be to decide initially between the two data mules to have different prefixes for the content store entries. This problem was identified, but the initial coordination was implemented on top of TCP to overcome the above mentioned problem.

### 4.4.2 Utilities in CCNx

For this implementation, the code from various applications implemented in CCNx code base was used in the name based prioritization application. Brief description about all the commands is given below.

CCNPEEK This utility generates an interest towards the CCND core daemon for a certain prefix, where the user can change the lifetime, scope and timeout of the interest packet.

CCNPOKE This command adds the content for a certain name in the CS.

CCNDC It is a routing utility which add or deletes FIB entries in the CCND core daemon.

CCNDUMPNAMES This utility prints all the names present in the buffer memory on the terminal.

### 4.4.3 Modification of CCNx

In both of the content centric mechanisms designed , interests are retransmitted whenever the node $X$ meets some other node $Y$. So there is always exchange of information once the environment changes. NBP protocol on top of CCN creates FIB entries for all the interests (names) we want to send to the next machine. This creation of FIB entries nullifies the filtering effect of the PIT. The life time of an interest sitting in a PIT, doesn't blocks the retransmission.
The life time of an interest packet is set to infinite or maximum, so that it could sit in the PIT until it finds the data against it. Since there is no hard limit on the PIT size in the CCNx implementation, we do not delete any pending entries. But the lookup time for the data can increase depending upon the PIT size of the data mule. An additional module for getting all PIT entries from the CCND core daemon to the application was added in the CCNx code base. This module was then used to add FIB entries to fetch the data from the encountered data mule. CCNDUMPNAMES utility was modified to send all the prefixes to the NBP application instead of printing it to the Linux terminal.

### 4.4.4 Test Setup

There are two different virtual machines used for this experiment. Each of them is running NBP protocol on top of CCNx. One of the CCNx core daemon is filled with 100 data items, and the other one is queried all the interests against all those data items. Then NBP protocol generates the FIB entries for all the interests it can get from the encountered data mule.

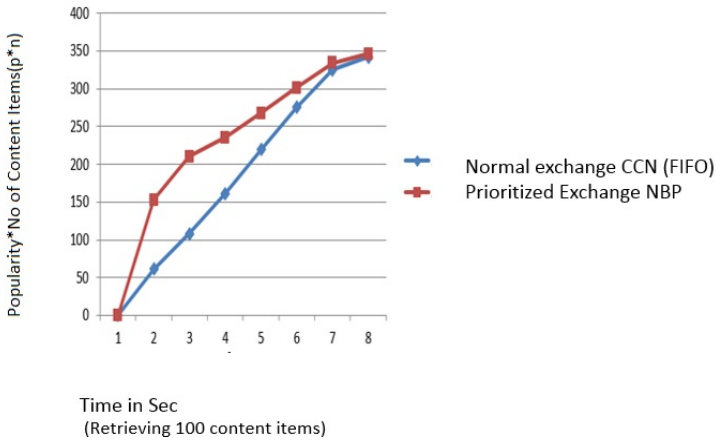


Figure 40: NBP protocol vs Modified CCN protocol

Here we compare NBP protocol with the CCN protocol for fetching data items during the encounter time. In both cases the CS entries are exchanged, so that one can forward only those queries which could be fetched in the least time. In Figure 40 we see a metric $p*m$ which is popularity multiplied by the number of content items received in that particular time. The popularity value for the popular prefixes and the most important prefixes is set to 10, while the value for the unpopular prefixes is set to 2.
CCN protocol send the interest messages in a First in First Out(FIFO) manner, in contrast NBP protocol forwards the interests in a prioritized manner. We can understand that from the graph that NBP protocol fetches the most important and popular preferentially, which is why the red line shoots up in the start of the exchange. It is also important to note that NBP exploits the situation in the best manner, if there is any disruption in during the inter-meeting times of the data mules. So suppose if there was a disruption at time =2sec, NBP would have fetched the most important and popular items in that time as compared to the normal exchange.

## 4.5 Conclusion

This chapter provides some measures to demonstrate the proficiency of two data centric mechanisms presented in the previous chapter. The evaluation scenario for the COPA and reference algorithm is explained.
It is essential to validate the popularity estimated by both mechanisms. To check this, both popularity estimation mechanisms were compared with the total number of queries poured into the network during the start of experiments. Both COPA and reference algorithm provide quite accurate popularity ranking per prefix with decent amount of encounters between all the data mules. Furthermore both popularity aggregation mechanisms are then tested for their memory utilization,where COPA excels in memory efficiency for Zipf distributed interests in the network.
This chapter also discusses the modifications needed and utilities used in the CCNx abstraction to integrate it with the NBP protocol. NBP protocol performance shows that it fetches the most important and popular interests in a prioritized manner. Our

results show that higher priority messages will be disseminated to more nodes in the network which might be of vital importance depending upon the critical situation.

# 5 Conclusion & Future Work

This chapter concludes the thesis and discusses the future directions about the thesis and ICN in general.

## 5.1 Conclusion

The master thesis was aimed to construct new mechanisms in the field of disruption tolerant networks using data centric functionalities. The research work included the development of two mechanisms which helped to make efficient decisions while forwarding the data in DTN scenarios. The thesis also identifies some future research areas for CCN protocol in intermittently connected networks.
The necessity to design a new abstraction arose when TCP/IP protocol was unable to with stand long delays. This problem persuaded researchers to design a new abstraction (bundle) which could work in DTN scenarios. This overlay abstraction was network agnostic and send packets from source to destination with no network optimization. However data centric abstraction (CCN) which was mainly built for the well-connected networks also provided an automatic support for DTN.
We choose to work with CCN protocol in this thesis, due to its network layer awareness, in network caching and name based routing. We developed two mechanisms on top of this data centric abstraction to efficiently distribute data in the after math of a disaster when energy and communication resources are very critical. However both of these protocols can work in any delay tolerant network scenario or in general wireless scenario.
NBP protocol is an application built on top of CCN protocol which prioritizes the exchange between two data mules. It manipulates the three main data structures implemented in CCNx software, to give precedence to the names which are popular or more important to the DTN scenario. To be very clear this protocol is an extension to a research work[29] which was simulated on one simulator[32] before, but not implemented on top of CCN abstraction.
We also described and implemented a unique popularity estimation algorithm (COPA) for DTN enviroment. This mechanism is explained in detail, which involves DM-DM and DM-end user interest aggregation. Once a data mule manages to build popularity during the mobility, it can make efficient forwarding and caching decisions.
COPA was evaluated and compared with a reference algorithm to understand the memory and network congestion constraints. The prioritization exchange protocol was compared with a normal CCN exchange (FIFO) to verify its significance in the time critical encounters of data mules.
Furthermore this thesis provides a brief understanding why ICN is really the future of internet. Its performance and efficient resource usage in the well-connected network and intermittently connected networks advances the case of adapting a different abstraction for the world internet users.

## 5.2 Future Work

Although it's been more than a decade since data centric architectures started appearing in the research conferences, but still there is roam of improvement. In the following section, first we will discuss the enhancements that could be made to this particular thesis. Second we will describe potential research areas in the field of ICN.

### 5.2.1 Permanent Storage in CCN

CCN abstraction provides permanent storage, where data saved in memory is not transient. This permanent storage was not exploited to disseminate data in DTN scenarios in this thesis. Since data gets replaced in the buffer memory depending upon the caching mechanism, authorities can save the disaster recovery announcements in the repository.
Data saved in the repositories can also be advertised to the peers in the data mule network for efficient dissemination of important information. These advertisements can be push based mechanisms, as compare to request-reply mechanism in CCN. However mechanisms need to be designed for deciding which data to be published from these repositories and for how many hops these advertisements should be propagated.

### 5.2.2 Acknowledgements and PIT Size

The solution presented in this thesis presents flooding mechanism, where each request is forwarded to every other data mule to retrieve data. This flooding mechanism can increase the PIT size drastically, in each of the mobile CCN routers. Increase in the PIT size will increase the lookup time for requests drastically which can degrade the performance. Requests and data items in our solution can follow completely different paths as compared to completely connected networks, so most of the time queries get stuck in the PIT for a considerable amount of time.
A mechanism could be developed to flood acknowledgements in the data mule network, so that the stagnant queries inside the PIT could be removed. Another structure called data delivery table (DET) could be incorporated inside CCNx or provided as a service on top of CCN abstraction for DTN scenarios as shown in Table 3.

Table 3: DET for saving acknowledgements.

| Deleivered Data Entries | Nonce |
|---|---|
| /aalto.fi/firstaid.png | N1,N2 |
| /aalto.fi/evacuationplan.png | N3,N4 |
| /bbc.com/heidelberg/weather/report | N5 |

### 5.2.3 Data Discovery in CCN

Each separate object has a different name in CCN. Name resolution and data routing are coupled together in CCN. Retrieving data in CCN can be very tedious, since

there could be billions of content descriptors in this kind of future internet.
If there is no FIB entry for a request in the access router, the request is broadcasted in the CCN network. The request is replicated in the network until the data is found in the network. Serving data against the unpopular requests might cause network congestion and unwanted forwarding in the entire network.
A solution to this problem could be to divide the network into autonomous areas, where each area can maintain a list of names which it can serve. In this manner, requests could be directly sent to the edge routers in different autonomous systems to find out the data.

### 5.2.4 Services over CCN

Most of the data centric approaches consider the static content as the main building block of their architecture. However the traditional internet not only provides content but a lot of customized services to different entities in the network like google maps, load balancing, ad advertisement, transcoding and streaming video. Some services are invoked by applications, to perform different set of operations. Current CCN architecture needs modifications to provide services to different entities in the network. These questions have given birth to a new paradigm called Service Centric Networking (SCN) [33].
Most of the approaches in SCN use CCN protocol to provide services. CCN protocol is a bit by bit protocol which has certain semantics, difficult to obey while serving the modern complex applications [34]. These complex applications and services require meta-information and application state at the server end. This extra information has no space in the current message (interest/data) structures of the CCN abstraction. There are some fundamental constraints which need to be addressed in a CCN jargon to integrate rest-ful services that drive the commercial web.

### 5.2.5 Data Advertisements by Intermediary Routers in CCN

Currently in CCN, data is only advertised by the authoritative servers who are responsible for the names for their organisation. These advertisements are floated into the network, where each intermediary node fill up their FIB entries.
There is no strict rule yet defined, that how many hops this advertisement (prefix announcements) would be floated in the network. So it would be interesting to see the results, if intermediary routers can use their repositories (permanent storage) to publish advertisements. It would be same as redundant CDN servers, which are placed in different parts of the world, to provide traffic engineering in the current traditional internet architecture.
Although off path caching has been exploited in other ICN architectures, but it's a new phenomenon in CCN.

### 5.2.6 Exploiting Multiple Interfaces in CCN

Current internet architecture need overlay solutions to provide complex multipath solutions. Most of the information centric architectures[2][7] are intrinsically able to

deal with multiple interfaces and fluctuating connectivity. This feature provides a lot of opportunities to exploit the network bandwidth from different technologies. However there is no significant work done in wireless scenario(4G,WIFI) , to use multiple interfaces to provide efficient ways of communication to best of our knowledge.

### 5.2.7 Information Centric Mechanisms for Traditional Internet

Researchers have been looking how information centric concepts (caching, name based forwarding) can be applied in the current internet architecture to provide flexibility, scalability and dynamicity. Recent studies have shown the potential of name based routing in SDN based network management [35]. We believe information centric functionalities can be incorporated in the current protocol stack for achieving better results in areas like data centres, software defined network and network virtualization.

# References

[1] G. Tyson, N. Sastry, R. Cuevas, I. Rimac, and A. Mauthe, "A survey of mobility in information-centric networks", *Commun. ACM*, vol. 56, no. 12, pp. 90–98, Dec. 2013, ISSN: 0001-0782. DOI: 10.1145/2500501. [Online]. Available: http://doi.acm.org/10.1145/2500501.

[2] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content", in *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT '09, Rome, Italy: ACM, 2009, pp. 1–12, ISBN: 978-1-60558-636-6. DOI: 10.1145/1658939.1658941. [Online]. Available: http://doi.acm.org/10.1145/1658939.1658941.

[3] (2013). Green ICN Project, [Online]. Available: http://www.greenicn.org (visited on 11/01/2014).

[4] (2011). CCNx Software, [Online]. Available: "Available:http://http://www.ccnx.org//" (visited on 07/19/2015).

[5] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture", in *Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '07, Kyoto, Japan: ACM, 2007, pp. 181–192, ISBN: 978-1-59593-713-1. DOI: 10.1145/1282380.1282402. [Online]. Available: http://doi.acm.org/10.1145/1282380.1282402.

[6] A. Anand, F. Dogar, D. Han, B. Li, H. Lim, M. Machado, W. Wu, A. Akella, D. G. Andersen, J. W. Byers, S. Seshan, and P. Steenkiste, "Xia: An architecture for an evolvable and trustworthy internet", in *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*, ser. HotNets-X, Cambridge, Massachusetts: ACM, 2011, 2:1–2:6, ISBN: 978-1-4503-1059-8. DOI: 10.1145/2070562.2070564. [Online]. Available: http://doi.acm.org/10.1145/2070562.2070564.

[7] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, k. claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named data networking", *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 3, pp. 66–73, Jul. 2014, ISSN: 0146-4833. DOI: 10.1145/2656877.2656887. [Online]. Available: http://doi.acm.org/10.1145/2656877.2656887.

[8] C. Dannewitz, D. Kutscher, B. Ohlman, S. Farrell, B. Ahlgren, and H. Karl, "Network of information (netinf) - an information-centric networking architecture", *Comput. Commun.*, vol. 36, no. 7, pp. 721–735, Apr. 2013, ISSN: 0140-3664. DOI: 10.1016/j.comcom.2013.01.009. [Online]. Available: http://dx.doi.org/10.1016/j.comcom.2013.01.009.

[9] (2008). FP7 PSIRP Project., [Online]. Available: Available:http://www.psirp.org/ (visited on 08/09/2015).

[10] (2010). NSF Mobility First Project., [Online]. Available: Available:http://mobilityfirst.winlab.rutgers.edu/ (visited on 08/01/2015).

[11] E. Nordström, D. Shue, P. Gopalan, R. Kiefer, M. Arye, S. Y. Ko, J. Rexford, and M. J. Freedman, "Serval: An end-host stack for service-centric networking", in *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, ser. NSDI'12, San Jose, CA: USENIX Association, 2012, pp. 7–7. [Online]. Available: http://dl.acm.org/citation.cfm?id=2228298.2228308.

[12] A. Ghodsi, T. Koponen, J. Rajahalme, P. Sarolahti, and S. Shenker, "Naming in content-oriented architectures", in *Proceedings of the ACM SIGCOMM Workshop on Information-centric Networking*, ser. ICN '11, Toronto, Ontario, Canada: ACM, 2011, pp. 1–6, ISBN: 978-1-4503-0801-4. DOI: 10.1145/2018584.2018586. [Online]. Available: http://doi.acm.org/10.1145/2018584.2018586.

[13] S. K. Fayazbakhsh, Y. Lin, A. Tootoonchian, A. Ghodsi, T. Koponen, B. Maggs, K. Ng, V. Sekar, and S. Shenker, "Less pain, most of the gain: Incrementally deployable icn", in *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, ser. SIGCOMM '13, Hong Kong, China: ACM, 2013, pp. 147–158, ISBN: 978-1-4503-2056-6. DOI: 10.1145/2486001.2486023. [Online]. Available: http://doi.acm.org/10.1145/2486001.2486023.

[14] CISCO, "Cisco visual networking index: Forecast and methodology: 2013–2018", Tech. Rep., June 2014. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/serviceprovider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.pdf.

[15] (2005). MobileIP, [Online]. Available: "http://searchunifiedcommunications.techtarget.com/feature/Mobile-IP-networks-An-overview" (visited on 07/19/2015).

[16] V. Jacobson. (2006). A new way to look at networking, [Online]. Available: http://www.i2c-bus.org/i2c-primer/typical-i2c-bus-setup/ (visited on 08/01/2015).

[17] S. Farrell and V. Cahill, *Delay- and Disruption-Tolerant Networking*. Norwood, MA, USA: Artech House, Inc., 2006, ISBN: 1596930632.

[18] M. Khabbaz, C. Assi, and W. Fawaz, "Disruption-tolerant networking: A comprehensive survey on recent developments and persisting challenges", *Communications Surveys Tutorials, IEEE*, vol. 14, no. 2, pp. 607–640, 2012, ISSN: 1553-877X. DOI: 10.1109/SURV.2011.041911.00093.

[19] L. Pelusi, A. Passarella, and M. Conti, "Opportunistic networking: Data forwarding in disconnected mobile ad hoc networks", *Communications Magazine, IEEE*, vol. 44, no. 11, pp. 134–141, 2006, ISSN: 0163-6804. DOI: 10.1109/MCOM.2006.248176.

[20] K. Fall, "A delay-tolerant network architecture for challenged internets", in *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '03, Karlsruhe, Germany: ACM, 2003, pp. 27–34, ISBN: 1-58113-735-4. DOI: 10.1145/863955.863960. [Online]. Available: http://doi.acm.org/10.1145/863955.863960.

[21] (2014). Interplanetary Networking Special Interest Group(IPNSIG)0, [Online]. Available: "Available:http://http://ipnsig.org//" (visited on 07/19/2015).

[22] S. Jain, K. Fall, and R. Patra, "Routing in a delay tolerant network", in *Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '04, Portland, Oregon, USA: ACM, 2004, pp. 145–158, ISBN: 1-58113-862-8. DOI: 10.1145/1015467.1015484. [Online]. Available: http://doi.acm.org/10.1145/1015467.1015484.

[23] B. Schneier, *Applied Cryptography (2Nd Ed.): Protocols, Algorithms, and Source Code in C*. New York, NY, USA: John Wiley & Sons, Inc., 1995, ISBN: 0-471-11709-9.

[24] J. Seedorf, B. Gill, D. Kutscher, B. Schiller, and D. Kohlweyer, "Demo overview: Fully decentralised authentication scheme for icn in disaster scenarios", in *Proceedings of the 1st International Conference on Information-centric Networking*, ser. ICN '14, Paris, France: ACM, 2014, pp. 191–192, ISBN: 978-1-4503-3206-4. DOI: 10.1145/2660129.2660130. [Online]. Available: http://doi.acm.org/10.1145/2660129.2660130.

[25] A. Vahdat and D. Becker, "Epidemic routing for partially-connected ad hoc networks", Tech. Rep., 2000.

[26] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and wait: An efficient routing scheme for intermittently connected mobile networks", in *Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking*, ser. WDTN '05, Philadelphia, Pennsylvania, USA: ACM, 2005, pp. 252–259, ISBN: 1-59593-026-4. DOI: 10.1145/1080139.1080143. [Online]. Available: http://doi.acm.org/10.1145/1080139.1080143.

[27] A. Balasubramanian, B. Levine, and A. Venkataramani, "Dtn routing as a resource allocation problem", in *Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '07, Kyoto, Japan: ACM, 2007, pp. 373–384, ISBN: 978-1-59593-713-1. DOI: 10.1145/1282380.1282422. [Online]. Available: http://doi.acm.org/10.1145/1282380.1282422.

[28] (2011). ICN vs DTN, [Online]. Available: "https://www.ietf.org/mail-archive/web/icnrg/current/pdfNzWEtKLi7A.pdf" (visited on 07/19/2015).

[29] I. Psaras, L. Saino, M. Arumaithurai, K. Ramakrishnan, and G. Pavlou, "Name-based replication priorities in disaster cases", in *Proceedings of the 2nd IEEE workshop on Name-Oriented Mobility*, ser. NOM'14, Toronto, Canada: IEEE, 2014.

[30] T. Wang and P. Hui, "Cooperative caching based on file popularity ranking in delay tolerant networks.", *ExtremeCom*, 2012.

[31] F. Neves dos Santos, B. Ertl, C. Barakat, T. Spyropoulos, and T. Turletti, "Cedo: Content-centric dissemination algorithm for delay-tolerant networks", in *Proceedings of the 16th ACM International Conference on Modeling, Analysis & Simulation of Wireless and Mobile Systems*, ser. MSWiM '13, Barcelona, Spain: ACM, 2013, pp. 377–386, ISBN: 978-1-4503-2353-6. DOI: 10.1145/2507924.2507931. [Online]. Available: http://doi.acm.org/10.1145/2507924.2507931.

[32] Teemu. (2006). The one simulator for dtn enviroment, [Online]. Available: "https://www.netlab.tkk.fi/tutkimus/dtn/theone/" (visited on 10/12/2015).

[33] T. Braun, A. Mauthe, and V. Siris, "Service-centric networking extensions", in *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, ser. SAC '13, Coimbra, Portugal: ACM, 2013, pp. 583–590, ISBN: 978-1-4503-1656-9. DOI: 10.1145/2480362.2480475. [Online]. Available: http://doi.acm.org/10.1145/2480362.2480475.

[34] I. Moiseenko, M. Stapp, and D. Oran, "Communication patterns for web interaction in named data networking", in *Proceedings of the 1st International Conference on Information-centric Networking*, ser. ICN '14, Paris, France: ACM, 2014, pp. 87–96, ISBN: 978-1-4503-3206-4. DOI: 10.1145/2660129.2660152. [Online]. Available: http://doi.acm.org/10.1145/2660129.2660152.

[35] M. Arumaithurai, J. Chen, E. Monticelli, X. Fu, and K. K. Ramakrishnan, "Exploiting icn for flexible management of software-defined networks", in *Proceedings of the 1st International Conference on Information-centric Networking*, ser. ICN '14, Paris, France: ACM, 2014, pp. 107–116, ISBN: 978-1-4503-3206-4. DOI: 10.1145/2660129.2660147. [Online]. Available: http://doi.acm.org/10.1145/2660129.2660147.

# A Appendix

This appendix present the snapshot of CCN protocol from the Wireshark to understand the working of the protocol.

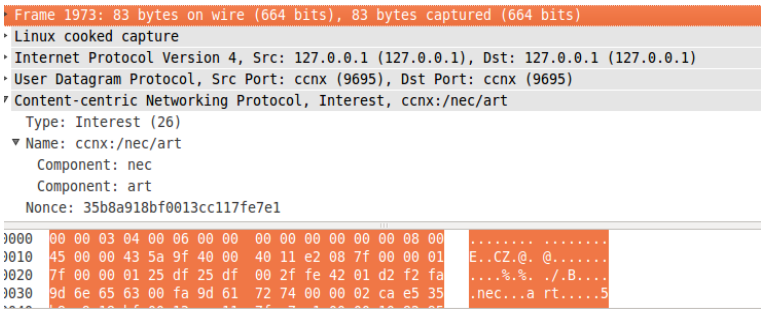


Figure A1: Interest Packet Wireshark

Figure A1 shows an interest packet named /nec/art caught by the Wireshark. It shows it has two components nec and art showing the hierarchy of names. It also shows the nonce attached to the name which is byte string.


Figure A2: Periodic Updates about the Neighbours

Figure A2 shows the updates about the connected machines are exchanged during the encounter of two data mules. These updates are exchanged using the interest and the data packets in the CCNx jargon.

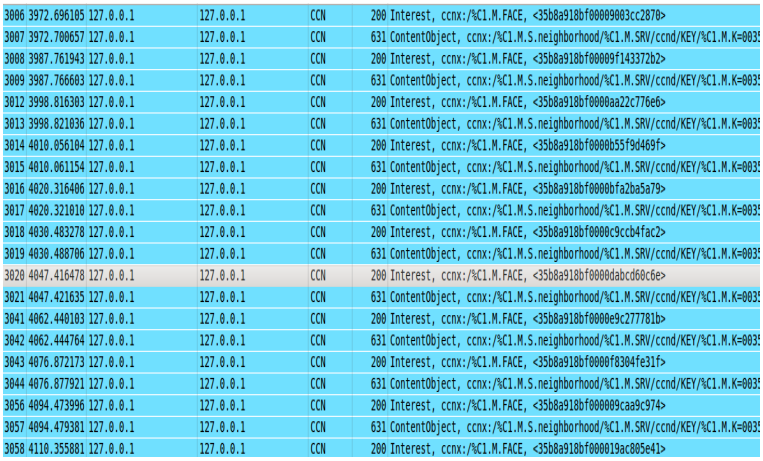


Figure A3: Snapshot from Wireshark, with CCN dissector

Figure A3 shows the interest and content packet exchanged during the Name based prioritization protocol.

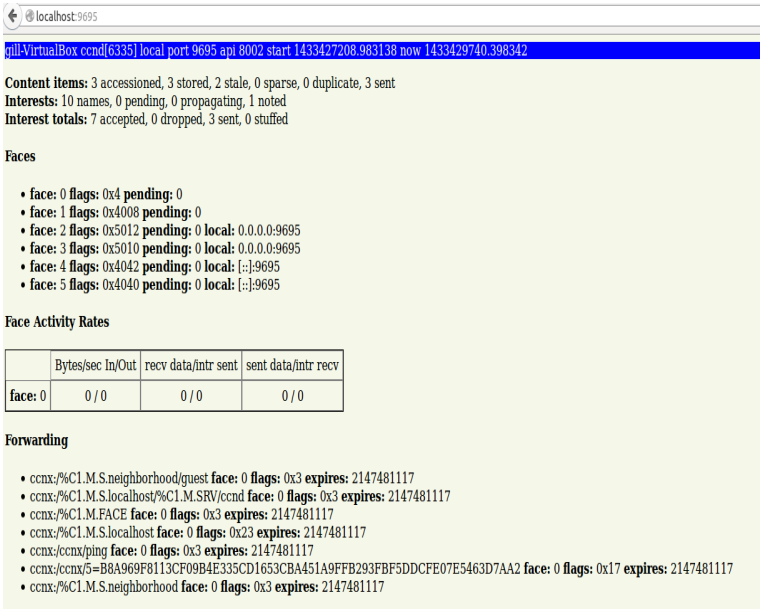


Figure A4: CCNDSTATUS utility

Figure A4 shows the status of the CCN protocol by using the ccndstatus utility. This status page shows the number of content items stored inside the buffer and the number of the pending interest table entries. It also shows the faces from the internal applications as well faces generated for the encountered machines.